Edited by Sebastian Engell

WILEY-VCH

Logistic Optimization of Chemical Production Processes

Logistic Optimization of Chemical Production Processes

Edited by
Sebastian Engell

WILEY-VCH Verlag GmbH & Co. KGaA

Logistic Optimization of Chemical Production Processes

Edited by
Sebastian Engell

Related Titles

F.J. Kel (Ed.)

Modeling of Process Intensification

2006
ISBN 978-3-527-31143-9

L. Puigjaner, G. Heyen (Eds.)

Computer Aided Process and Product Engineerin

2006
ISBN 978-3-527-30804-0

K. Sundmacher, A. Kienle, A. Seidel-Morgenstern (Eds.)

Integrated Chemical Processes

Synthesis, Operation, Analysis, and Control

2005
ISBN 978-3-527-30831-6

Wiley-VCH (Ed.)

Ullmann's Chemical Engineering and Plant Design
II Volumes

ISBN 978-3-527-31111-8

K. Sundmacher, A. Kienle (Eds.)

Reactive Distillation

Status and Future Directions

2003
ISBN 978-3-527-30579-7

Logistic Optimization of Chemical Production Processes

Edited by
Sebastian Engell

WILEY-VCH Verlag GmbH & Co. KGaA

Editor

Prof. Dr.-Ing. Sebastian Engell
Technische Universität Dortmund
Department of Biochemical and
Chemical Engineering
Process Dynamics and Operations
Emil-Figge-Strasse 70
44221 Dortmund
Germany

Library of Congress Card No.: applied for

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library.

Bibliographic information published by the Deutsche Nationalbibliothek
Die Deutsche Nationalbibliothek lists this publication in the Deutsche National-bibliografie; detailed bibliographic data are available in the Internet at http://dnb.d-nb.de

Printed in the Federal Republic of Germany
Printed on acid-free paper

Typesetting Aptara Inc., New Delhi, India
Printing betz-druck GmbH, Darmstadt
Bookbinding Litges & Dopf GmbH, Heppenheim

ISBN 978-3-527-30830-9

Contents

Preface

Lucky managers of chemical production units do not have to care much about logistics. Raw materials are always available at fixed low prices, the equipment produces the same (few) products day in day out, and the customers are eager to send more trucks, ships, etc. to be filled with products than can actually be loaded. The marketing department regularly makes auctions to determine which customer gets how much of the output of the plant. Such a situation may have existed in countries without competition and a free market, but while it makes the life of a plant manager easy on the one hand, we all know that there are most severe drawbacks for the economy as a whole, and as a plant manager you suffer from those on the other hand as well because, e.g., the vendors of the equipment that you need are in the same position and then you have to wait in line until you get a new valve, vessel, column, etc.

In a market-driven economy, demands are fluctuating, cost matters and customers want optimized products that are tailored to their specific needs. Most chemical plants can therefore produce a variety of different products or grades of products in parallel or sequentially, but the resources that are available in the production units are limited. Hence, a key question in the operation of a chemical plant is when to produce what and with which resources, and possibly also in which manner (according to which recipe). On a higher level, in a global company there are many different production sites that can deliver intermediates and products, so the production chain can be distributed over several sites of a production network. And there is the "make or buy decision", i.e. rather than producing intermediates in house they can also be bought from external sources. These questions have to be answered on different time scales, from the long-term planning of production capacities through setting yearly and monthly production targets to the daily decisions on product changeovers or the start of campaigns or individual batches and the immediate reaction to breakdowns, variability of yields and availability of personnel. All these operational decisions require logistic optimization that for the most part means to take the right *discrete* choices, i.e. choices among a finite set of alternatives. While such choices at first sight seem simpler than continuous optimization because in principle one can enumerate and grade all possible alternatives, as soon as there are several simultaneous options, the problem of the combinatorial explosion arises: the number of alternatives becomes too large to explore them all within

a reasonable period of time. This type of problems is addressed by this book. Its topic can thus be described as "tools to solve combinatorial optimization problems that arise in chemical production management".

Logistic optimization is not only of importance in the operation of chemical plants and networks or chains of plants, but also in the planning of new plants or extensions and modifications of existing ones. For all multiproduct or multigrade plants, the quality of a concept for a new plant, a new unit or an addition to a unit can only be evaluated if the diversity and the temporal fluctuation of the demands and the ability of the plant to satisfy them are taken into account. And this, in turn, requires logistic optimization problems of the sort described above for a set of possible scenarios to be solved and the optimal operation of the different possible designs to be compared. As the pressure for immediate decisions in planning is not as high as in plant operation and fewer details have to be considered, in this area rigorous optimization is more likely to be an option than in real-time plant operation.

As stated above, the focus of this book is on *tools* for logistic optimization, including but not limited to optimization algorithms in the rigorous sense of the word. As the first of its kind, this book addresses the logistic optimization of chemical production processes both from a practical and from an academic point of view. From the review of the main problems in supply-chain optimization through the description of methods and tools that are currently used in industry, logistic simulation, campaign planning under uncertainty, heuristics- and optimization-based production planning and scheduling, the twelve chapters span the scope to recently proposed advanced optimization algorithms and to the embedding of such optimization algorithms into Enterprise Resource Planning (ERP) systems. All the chapters discuss real-world applications or case studies that are derived from real industrial problems. The authors represent the industrial users of tools for logistic optimization, the developers and vendors of such tools and software systems and academic researchers in a balanced fashion. It was my intention as the editor to provide an up-to-date survey of the field and at the same time a reference book than can be taken as the basis of courses on the operation of chemical and biochemical plants.

In the first chapter, Mario Stobbe (Evonik Degussa) sets the stage by giving an introduction into supply chains, supply-chain modeling and supply-chain management in the chemical industry. Chapters 2 and 3 deal with logistic simulation as a tool for logistic optimization, probably the tool that is most widely used in industry for this purpose (here optimization is understood in the practical and not in the rigorous sense of the word). Markus Schulz (Evonik Degussa) gives an overview of the potential applications of simulation and the organization of simulation projects. Andreas Liefeldt (Universität Dortmund, now with ABB Corporate Research) describes a simulator for a type of plant that has even more operational flexibility than traditional multiproduct batch plants, pipeless plants. The simulator includes a heuristic scheduling algorithm and supports both the planning and the operation of such plants.

The third part of the book is devoted to industrial solutions for complex scheduling and supply-chain management problems. Christoph Plapp, Dirk Surholt and Dietmar Syring (Axxom AG) present a tool for the solution of large supply-chain

optimization problems that combines optimization and heuristics to successfully solve problems that are currently beyond the scope of rigorous optimization. Iiro Harjunkoski, Marco Fahl and Hans Werner Borchers (ABB Corporate Research) discuss the application of state-of-the art optimization technology to a copper production process that can serve as an example of how adaptation to the needs of the real problem and careful engineering can bridge the gap between academic algorithms and practical applications and give benefits in real industrial applications. Rudolf Metz (Bayer Technology Services) focuses on stochastic tools for handling the randomness of demands in the planning of production campaigns.

Part IV of the book deals with optimization methods and is intended to provide an introduction to the state-of-the-art in optimization technology from an academic point of view. The authors of this section have invested a lot of effort to make these chapters easier to follow and more pleasant to read than most journal papers on scheduling and discrete optimization. First, Guido Sand (Universität Dortmund, now with ABB Corporate Research) describes the engineering of mixed-integer programming (i.e. the rigorous solution of decision problems with real and discrete degrees of freedom as they arise in the solution of production planning and scheduling) for the solution of real batch-production problems. Carlos A. Mendes, Ignacio Grossmann, Iiro Harunkowski and Marco Fahl (Carnegie Mellon University and ABB Corporate Research) discuss the choice of linear mixed-integer optimization models for the same task, in particular the key issue of the modeling of time that has been the focus of scientific discussion for many years now. The contribution by Jochen Till, Guido Sand and Sebastian Engell (Universität Dortmund) addresses the issue of how to include uncertainty about the future evolution of demands, production capacities, etc. into the solution of scheduling problems and present an alternative algorithmic approach to the solution of scheduling problems, evolutionary algorithms. The last chapter in this section by Sebastian Panek, Olaf Stursberg and Sebastian Engell (Universität Dortmund) introduces a completely different approach to the modeling and solution of scheduling problems, based upon timed automata that were introduced in computer science in the past decade. The formulation of the models can be done in a modular, intuitive fashion, and the problems are solved using tools from computer science for reachability analysis.

In the last part of the book, the embedding of the solution of operational planning and scheduling problems into the mid- and long-term material and resource planning performed by ERP (enterprise resource planning) systems is discussed. Mathias Göbelt, Thomas Kasper and Christopher Sürie (SAP) describe a concept for the integration of short- and midterm scheduling and demonstrate it for the solution of a case study. Winfried Jaenicke and Robert Seeger (OR Soft) discuss the integration of scheduling algorithms with ERP systems and stress the role of humans and organizations in the planning and scheduling process.

In the collection of the contributions I tried to achieve a balance between the end users of tools and methods in the chemical industry, the tool developers, whose main concern is to develop and to market tools that are user friendly and efficient, possibly for a limited class of problems and without full regard for rigorous optimality, and academic researchers who have to venture into new areas,

to try new ideas and to be concerned with optimality in the rigorous sense of the word. In my view, the book demonstrates that the inevitable differences and occasional tensions between these views in this area have led to a productive "supply chain" from academia to industry, as demonstrated by several applications and case studies where state-of-the-art optimization methods have been brought to use for challenging industrial problems. My special thanks go the industrial contributors, because for them, in contrast to the academic authors, writing chapters of a book is a low-priority activity at least in the view of their superiors. It is my sad duty to mention that one of the authors, Thomas Kasper from SAP, passed away this year, caused by a severe illness. I hope that this book will contribute to keeping his memory alive among his colleagues and friends. For the collection of the chapters, the activity of the section Produktionslogistik (logistics of chemical production processes) within the German VDI-GVC Fachausschuss Prozess- und Anlagentechnik (Working Group on Process and Plant Technology, now ProcessNet Fachausschuss PAT) was very helpful and is gratefully acknowledged. Finally, it is my pleasure to thank the publisher, Wiley-VCH, and in particular Waltraud Wüst, for encouraging me in this endeavor and for their continuous support and the right combination of patience and pressure.

Wetter (Ruhr), January 2008 *Sebastian Engell*

List of Contributors

Hans Werner Borchers
Department of Industrial Software and Applications
ABB Corporate Research
Wallstadter Strasse 59
68526 Ladenburg
Germany

Sebastian Engell
Department of Biochemical and Chemical Engineering
Process Dynamics and Operations
Technische Universitat Dortmund
Emil-Figge-Strasse 70
44221 Dortmund
Germany

Marco Fahl
Department of Industrial Software and Applications
ABB Corporate Research
Wallstadter Strasse 59
68526 Ladenburg
Germany

Mathias Göbelt
SAP Deutschland AG & Co.KG
Hasse-Plattnei-Ring 7
69190 Walldorf
Germany

Ignacio E. Grossmann
Chemical Engineering Department
Carnegie Mellon University
5000 Forbes Ave.
Pittsburg, PA 15213
USA

Iiro Harjunkoski
Department of Industrial Software and Applications
ABB Corporate Research
Wallstadter Strasse 59
68526 Ladenburg
Germany

Winfried Jaenicke
OR Soft Jaenicke GmbH
FH, Geb. 104
Geusaer Strasse
06217 Merseburg
Germany

Thomas Kasper†
SAP
Dietmar-Hopp-Allee 16
69190 Walldorf
Germany

Andreas Liefeldt
Automation Engineering Group
ABB Corporate Research
Wallstadter Strasse 59
68526 Ladenburg
Germany

Carlos A. Méndez
Industrial Engineering Department
INTEC – (Universidad Nacional del Litoral – CONICET)
Güemes 3450
3000 Santa Fe
Argentina

Rudolf Metz
Bayer Technology Services GmbH
PMT-SCL-PPL
51368 Leverkusen
Germany

Sebastian Panek
Dr. Johannes Heidenhain GmbH
Dr.-Johannes-Heidenhain-Strasse 5
83301 Traunreut
Germany

Christoph Plapp
Axxom Software AG
Paul-Gerhardt-Allee 46
81245 München
Germany

Guido Sand
ABB Corporate Research
Wallstadter Strasse 59
68526 Ladenburg
Germany

Markus Schulz
Supply Chain Development
Evonik Degussa GmbH
Rodenbacher Chaussee 4
63457 Hanau-Wolfgang
Germany

Robert Seeger
OR Soft Jaenicke GmbH
FH, Geb. 104
Geusaer Strasse
06217 Merseburg
Germany

Sven Spieckermann
SimPlan AG
Edmund-Seng-Strasse 3–5
63477 Maintal
Germany

Mario Stobbe
Supply Chain Development
Evonik Degussa GmbH
Rodenbacher Chaussee 4
63457 Hanau-Wolfgang
Germany

Olaf Stursberg
Lehrstuhl für Steuerungs- und Regelungstechnik
Fakultät Elektrotechnik und Informationstechnik
Technische Universität München
80330 München
Germany

Dirk Surholt
Axxom Software AG
Paul-Gerhardt-Allee 46
81245 München
Germany

Christopher Sürie
SAP Deutschland AG & Co.KG
Hasso-Plattner-Ring 7
69190 Walldorf
Germany

Dietmar Syring
Axxom Software AG
Paul-Gerhardt-Allee 46
81245 München
Germany

Jochen Till
BASF Aktiengesellschaft
67056 Ludwigshafen
Germany

Part I
Introduction

1
Supply Chain and Supply Chain Management*

Mario Stobbe

1.1
Introduction

The design, planning and controlling of networks of business processes with multiple stages in order to improve competitiveness has been a theme of operations research since the 1950s [1]. In practice, international operating companies with large supply and distribution networks especially applied the research results. Since the 1980s, the interest in the theme of networks in general as a competitive means has increased for the following reasons:

- globalization of the markets for distributing and procuring materials;
- internationalization of site structures;
- emerging customer expectations regarding quality, time of delivery and price;
- significant improvements of information technology as a means of dealing with increasing complexity.

The increased interest led to new terms such as supply chain and supply chain management and – at least in the US – an abundance of new research. In this introductory article, we discuss the characteristics of a supply chain and supply chain management.

1.2
Terms and Definitions

1.2.1
Supply Chain

1.2.1.1 Structure

The word chain in supply chain is misleading as it implies a linear structure. However, the structure of a supply chain is usually a network structure and only in

* A list of abbreviations is given at the end of this chapter.

rather seldom cases a linear chain. The supply chain can be described in different levels of detail as will be outlined below when discussing the Supply Chain Organizations Reference Model (SCOR-model). For a first characterization, a supply chain will be considered here as a network of organizations exchanging materials, service and information in order to fulfill customers' demands. In a broad sense, the organizations are companies (legal entities). In a narrow sense, this definition applies to large companies with numerous sites in different countries providing a variety of materials and services as well. Some authors term the latter an intra-company supply chain and the former an inter-company supply chain. The setting of a complex intra-company supply chain is typical for large chemical companies.

The structure of a supply chain in the broader sense is comparable to a virtual corporation. A virtual corporation is a network of legally independent companies which cooperate for a limited time in order to achieve a given objective.

1.2.1.2 **Function**

Defining a supply chain solely by its structure and its components will be inadequate. From a functional point of view, the supply chain is comparable with logistics networks. A closer look at the characteristics of logistics networks and at supply chains will show some significant differences even when applying a modern characterization of logistics. In a classical sense, logistics only comprises storage and transportation of materials. Nowadays, logistics is treated as an enabling function including tasks such as procurement, production, distribution and disposal of materials. Both definitions have in common that logistics are seen from the point of view of a single company. A holistic definition of logistics includes suppliers and consumers as participants. Some authors equate this holistic concept with supply chain management as both concepts share some essential characteristics regarding organization and tasks. These characteristics are process orientation, co-ordination of information and material flow.

Process orientation means that the organizational structure corresponds to the key processes. This is in strong contrast to the functional organization where (parts of) processes are assigned to departments and, thus, processes are organized according to the structure of the departments. Besides the typical logistic processes, order acquisition, order processing and product development are typical key processes. These processes may cross the legal boundaries between companies in order to serve the needs of the customer which leads to the necessity of co-ordination of material and information flow.

However, the players in a logistics network are participants whereas in the supply chain they are (or should be) partners. This becomes apparent when looking at planning processes. Participants make decisions on their own trying to improve some variable – usually the profit – related to their own company. In contrast, partners in a supply chain make their decisions based on a collaborative and holistic consideration of effects along the supply chain in order to achieve a competitive advantage for the supply chain as a whole. The decisions include not only operational/short-term decisions but also tactical and strategic decisions regarding the design

of the supply chain. Eventually, the intended purpose of a supply chain is to fulfill the customers' demands in a most efficient manner and to outperform other supply chains.

1.2.2 Supply Chain Management

Based on the characterization of a supply chain, supply chain management (SCM) can be defined as "a process oriented approach to procuring, producing, and delivering end products and services to customers." It includes sub-suppliers, suppliers, internal operations, trade customers, retail customers and end users. It covers the management of materials, information, and fund flows [2].

A large variety of definitions of SCM exist which cannot be discussed in detail here. A look at the origins of the term will partly explain how different and sometimes misleading definitions evolved. The term SCM was established by consultants in 1982 [3]. They were the first to treat logistics as a top management concern. They argued that only the top management can balance the conflicting objectives of different functional units, e.g., long production runs (production) vs low inventories (finance). From this fact it becomes apparent that SCM is a management concept (!) and that it has evolved from practice. Theoretic considerations and interpretations followed some years later and often reflect the theoretical background of the author.

Managing the supply chain generally comprises three elements of activity:

- supply chain analysis
- supply chain planning
- supply chain execution

Before starting an improvement process, a clear picture of the supply chain has to be obtained. Therefore, Supply Chain Analysis is a critical success factor. Usually, this analysis will describe the "as-is" status and the desired "to-be" status. As a supply chain is built up of different companies for a limited time, it is essential that all partners speak the same "language" to describe and measure the as-is-status as well as to evaluate the to-be-status. For this purpose, usually a widely accepted model called the SCOR-model is used.

Supply Chain Planning (SCP) comprises all planning activities at the operational, tactical and strategic levels. Well known activities at the operational level are demand forecasting, network planning and scheduling. In order to ease these complex activities, so-called Advanced Planning Systems (APS) are used. At a strategic level, SCP includes supply chain design. Supply Chain Design comprises the selection of partners, the definition of the core business of each partner, selection of outsourcing strategies, supplier management and the selection of enabling technologies such as e-commerce and e-procurement.

Finally, Supply Chain Execution means putting agreed operational plans into practice with minimum effort.

1.3 Network Dynamics and Management of the Supply Chain

Although the term SCM first appeared in 1982, several effects connected with SCM were investigated long before then. From systems theory it is well known that the behavior of complex systems is more than the sum of its components and therefore cannot be understood solely by the analysis of its parts.

In 1958, Forrester started studies on an effect which is nowadays often referred to as the bullwhip effect. The bullwhip effect describes the amplification of temporal variations of the orders in a supply chain the more one moves away from the retail customer. Forrester showed that small changes in consumer demand result in large variations of orders placed upstream [4, 5]. It is interesting that this effect occurs even if the demand of final products is almost stable. For his studies, he assumed that some time delay exists between placing an order and the realization of this order (production). Furthermore, he assumed that each part of the supply chain plans its production and places its orders upstream taking into account only the information about the demands of its direct customer.

One may argue that Forrester investigated this effect theoretically; however, several authors were able to prove that this effect also occurs in reality [6–9]. This shows that an unmanaged supply chain is not inherently stable.

Nowadays, the bullwhip effect is best known from the so-called beer game. The Beer Game is a simulation developed at MIT in the 1960s to clarify the advantages of taking an integrated approach to managing the supply chain. A detailed description of the beer game and a playable version can be accessed via the internet (http://beergame.mit.edu/). In the beer game, the human players take the role of a part of a linear supply chain, e.g., a retailer, a wholesaler, a distributor or a manufacturer. The objective of the game is to minimize the total costs of the supply chain by maintaining low stocks but nevertheless managing to deliver all orders. There exists only one product called Lovers's Beer which is manufactured in units of one crate of beer. Two different costs have to be taken into account: inventory costs and backlog costs. Orders can be placed each week and it takes another two weeks before the supplier receives the order and two weeks before the orders reach the next part of the supply chain. If a part of the supply chain is unable to deliver in time, the orders are backlogged and the units have to be delivered the next week. The game is started in week one and each player has to decide how many units he wants to order from his supplier. The first round is finished by checking how many orders are delivered in time and how many orders are backlogged. The next round is started by placing the orders for the next week.

Usually, the game is started assuming that the only information a player gets are the orders of the player he supplies with beer. This is referred to as placing orders on local information. In this setting, human actors provided with local information usually tend to overact by an amplification of orders placed. Together with the inherent dynamics of the system, a slight variation of the end user demand in the beginning of the game is sufficient to introduce a persistent oscillation of demands

resulting in boosting stocks and number of orders and high costs for operating the supply chain.

In another setting, the human players are provided with global information about the system. This means that all players are informed about inventory levels and orders placed for each of the components of the supply chain. Furthermore, they are encouraged to work out co-operative strategies to deal with the dynamics of the system. Compared to the local information structure, this usually results in lower inventory levels and less out-of-stock-situations for all participants. Typically, the stocks and the number of orders in this setting are much lower, resulting in much lower costs for operating the supply chain and lower costs for each player as well.

The beer game demonstrates the value of sharing information across the various supply chain components. In practice, supply chains are usually more complex and much harder to manage. Current research has investigated that in practice the bullwhip effect is due to the following reasons:

- overreaction to backlogs;
- neglecting to order in an attempt to reduce inventory;
- no communication up and down the supply chain;
- no coordination up and down the supply chain;
- delay times for information and material flow;
- shortage gaming: customers order more than they need during a period of short supply, hoping that the partial shipments they receive will be sufficient;
- demand forecast inaccuracies: everybody in the chain adds a certain percentage to the demand estimates. The result is invisibility of true customer demand.

The identification of these reasons led to recommendations how to avoid the bullwhip effect. Some of these recommendations are:

- ordering decisions should be based on the demand of the ultimate customer instead of upstream forecast updates;
- eliminate gaming in shortage situations;
- stabilize prices in order to avoid large variations of demands;
- avoid order batching.

Many of these recommendations can be achieved using modern means of information technology. Standardized order procedures based on widely accepted information protocols will help to reduce the delay of information and current systems for advanced planning and scheduling (APS) provide means to support humans in decision making in complex networks. Building blocks of APS systems are:

- strategic planning
- forecasting
- global network planning
- distribution planning
- transportation planning
- production planning
- scheduling

Electronic data exchange is the enabler for these building blocks as manual data administration is error-prone, time-consuming and costly.

1.4 Design Criteria/Integration Concepts

Operating the supply chain has a major impact on its efficiency. Efficiency in this sense means that the operating expense in terms of time and money for a given design of the supply chain is as low as possible. However, the operating expenses are influenced by the design of the supply chain as well and the design varies according to the company's business and strategy. This is usually referred to as effectiveness. Effectiveness in this sense means that the design of the supply chain enables low operating expenses for a given business.

The design of the supply chain has different levels of interest. The driver of the supply chain design is the strategy the supply chain has agreed to follow. On this level, the partners agree on a strategy (e.g., prioritization of products and customers) and controlling issues (e.g., common performance indicators). These decisions are the drivers of the design of the other levels.

On the level of material flow, physical properties of the network are designed, i.e., decisions upon the existence of plant sites, warehouses and distribution centers, the transportation links between these components and their capacities are made. The decision upon the customer order decoupling point is a good example of how strategic decisions may influence the level of material flow.

The decoupling point is the boundary between the order-driven and the forecast-driven operations within a supply chain. Operations upstream of the decoupling point are forecast-driven, i.e., production for a certain time period is started before all customer orders are known. Operations downstream of the decoupling point are order-driven, i.e., production for a certain period of time starts after all customer orders are known. Furthermore, the decoupling point dictates the form in which inventory is held. Upstream, it is usually held as semi-finished goods while downstream it is held as finished goods. The semi-finished goods are generic in the sense that they allow for customization. Customization is related to the product (viscosity, color, water content, etc.) as well as to the choice of some other attributes such as packaging material, packaging size and pallet size. In order to gain flexibility, several authors recommend to design a supply chain such that it carries inventory in a generic form awaiting final processing or treatment so as to postpone product customization. Besides flexibility, postponement leads to lower inventories as it enables the production of materials according to customer orders and prevents building stocks resulting from inaccurate forecasts.

Many of the problems exhibited on the level of material flow are the result of the distortion of marketplace sales information as it is transferred upstream through the supply chain. Therefore, the design of the information flow is as important for the effectiveness and efficiency of supply chains as the design of the material flow. The information flow is obviously influenced by the level of material flow. However, the information flow is not necessarily dependent on the material flow.

Introducing new information links or improving existing ones may have no causes in material or process flow while having an impact on the efficiency of the supply chain, e.g., exchanging information regarding the sales planning between suppliers and distributors enhances planning quality enabling lower response times and lower storage costs.

The level of the information flow boils down to a purely technical level where the partners agree on common protocols for transferring data. For the chemical industry, an initiative to set uniform standards is CIDX. CIDX (http://www.cidx.org) is a trade association and standards body whose mission is to improve the ease, speed and cost of conducting business electronically between chemical companies and their trading partners. It provides the Chem eStandards, a collection of defined messages and related business process guidance that companies use to understand the requests and fulfill electronic business orders and related transactions.

At the process level, the flow of material and the flow of information are linked together by describing the transformation of information and material. Hence, it reflects the workflow of a supply chain.

1.5 SCOR: Modeling the Supply Chain

In this chapter, we want to switch from the components of a supply chain to its processes. For simplification, we focus on intra-company supply chains.

Companies have been creating processes and workflows for decades and these processes and workflows were subject to local optimization many times. As discussed above, these local optimizations usually do not lead to a global optimum. In the worst case, the objectives of local optimizations are inconsistent and contradictory. For example, operations usually aim at simplifying the product portfolio to achieve long production runs and low cleaning times while marketing in contrast assumes diversification as a means to gain some competitive advantage. Aiming at a global optimum means to weigh these different objectives according to the strategy of the company. This objective can be achieved best with a team made of expert members of all departments along the supply chain including marketing, sales, procurement and production which are able to draw the whole picture. The main problem of such a team are the different viewpoints and the different vocabularies which are used to describe the same processes. For such a team, a common language is urgently needed allowing for efficient communication and a common view of the supply chain. A widely accepted approach to provide such a common language is the SCOR-model.

1.5.1 The SCOR-Model

The Supply Chain Operations Reference-model [12] has been developed and endorsed by the Supply-Chain Council (SCC), an independent non-profit-making corporation, as the cross-industry standard for supply-chain management.

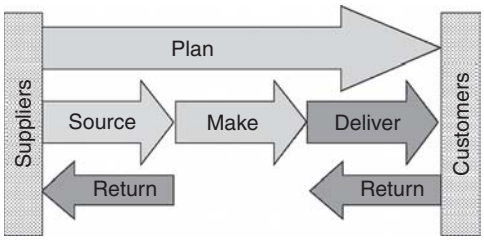

Fig. 1.1 The first level of the SCOR-model [12].

The SCOR-model is used to describe, measure and evaluate the configuration of a supply chain. It supports:

- Business Process Reengineering: capture the "as-is" state of a process and derive the desired "to-be" future status.
- Benchmarking: quantify the operational performance of similar companies and establish internal targets based on "best-in-class" results.
- Best Practice Analysis: characterize the management practices and software solutions that result in "best-in-class" performance.

The SCOR-model is a process reference model which is defined at different process levels. Besides the definition of the process, for each level indicators are proposed to allow to assess the performance of the process. In order to improve the process, best practices for the process elements are described which are based on the experience of the council members.

At the first level (Figure 1.1), the scope and the contents of the model are described using five types of management processes:

- Plan: processes that balance aggregate demand and supply to develop a course of action which best meets sourcing, production and delivery requirements.
- Source: processes that procure goods and services to meet planned or actual demand.
- Make: processes that transform product to a finished state to meet planned or actual demand.
- Deliver: processes that provide finished goods and services to meet planned or actual demand, typically including order management, transportation management, and distribution management.
- Return: processes associated with returning or receiving products returned for some reason.

At the second level (configuration level) (Figure 1.2), to each basic process a process type is assigned which is one of the following:

- Plan: a process that aligns expected resources to meet expected demand requirements.
- Execution: a process triggered by planned or actual demand that changes the state of material. The process types Source, Make and Deliver are detailed with regard

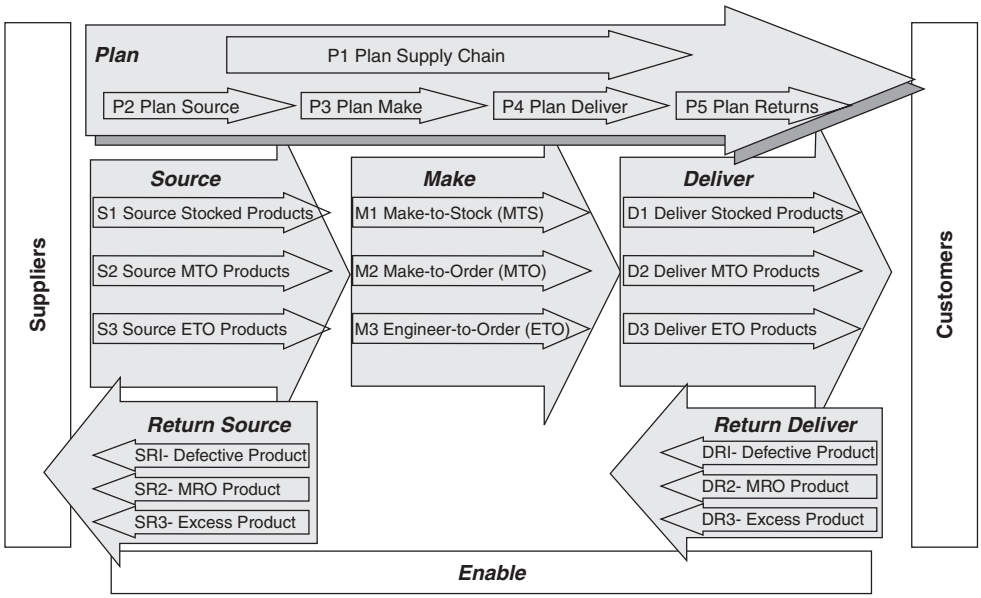

Fig. 1.2 The configuration level of the SCOR-model [12].

to the type of customer order. For the Make-process this may be make-to-order, make-to-stock or engineer-to-order.

- Enable: a process that prepares, maintains or manages information or relationships on which planning and execution processes rely. These process category comprises support processes of the Execute- and the Planning-processes which maintain information flow.

On level three, each process can be further detailed. In Figure 1.3, level 3 is depicted for the process configuration deliver-stocked-product.

Further levels can be added to take into account the supply chain management practice of the companies involved.

Beside the definitions, each level of the SCOR-model includes key performance indicators to measure performance and recommendations regarding best practice.

1.5.2 Quick Checks Using the SCOR-Model

Several projects applying the SCOR principles were already carried out within Evonik Degussa. Evonik Degussa, a wholly owned subsidary of Evonik Industries AG, is a multinational corporation consistently aligned to high-margin specialty chemistry. It is organized on a decentralized basis. Business operations are in the hand of twelve Business Units.

Up to now, several projects were accomplished which are termed Quick Checks but are better known within the company as SCOR-projects. The objective of a

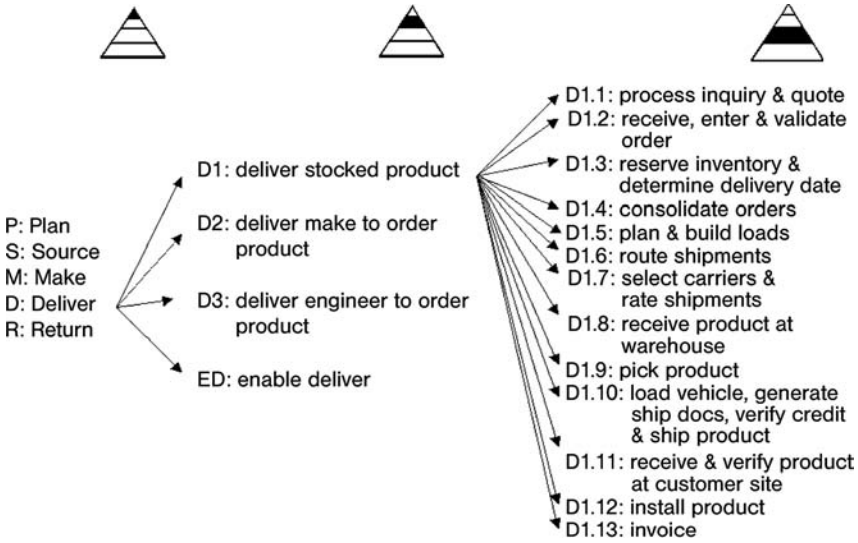

Fig. 1.3 Level 3 of the SCOR-model [12].

SCOR-project is to identify, evaluate and prioritize actions using shortcut methods. SCOR-projects are usually executed by teams made up of members of different departments including customer service, sales, controlling, operations, procurement and logistics. The teams are supported by internal consultants, who provide knowledge about the SCOR-model and preside over the team meetings.

1.5.2.1 **Describing the As-Is Status**

A SCOR-project comprises a sequence of workshops which last between one and three days. The sequence of workshops starts with teaching principles of supply chain management, the SCOR-terminology and key performance indicators in order to set up a common vocabulary and view. The next step is to describe the flow of materials running from the suppliers of raw materials to the main customers of the final products. In order to reduce complexity, products and customers are usually grouped according to substantial similarities. For customers, usually some geographic attribute is used. For products, similarity is decided on a case-to-case basis. In some cases, similarity is defined according to similar ways of production, e.g., a group of products are made by applying a make-to-stock-strategy and another by applying a make-to-order-strategy, in some other cases according to similar packaging, e.g., returnable and non-returnable containers. The depiction in a geographical map supports this process indicating suppliers, plant sites, distribution centers and final customers (Figure 1.4). Once the map is finished, it can be used to add some further details about the sourcing, making and distribution of the product groups, i.e., adding the process types of level 2 of the SCOR-model. In the geographical map, the process types are simply represented as letters and numbers, e.g., m1 for make-to-stock and m2 for make-to-order. From the geographical map,

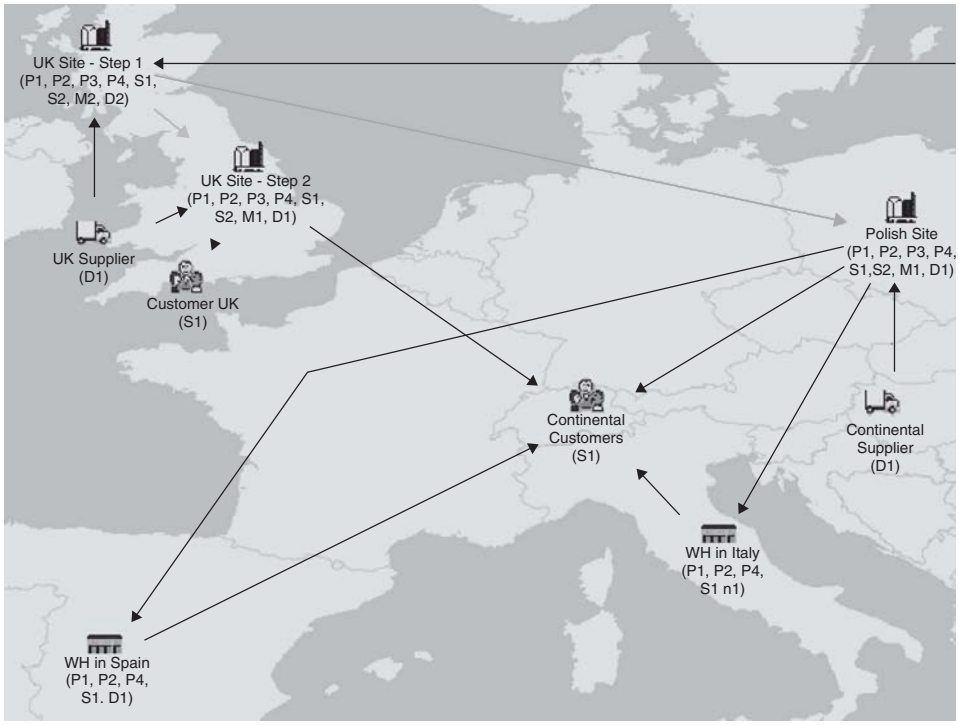

Fig. 1.4 Geographical map depicting suppliers, plant sites, distributors and customers.

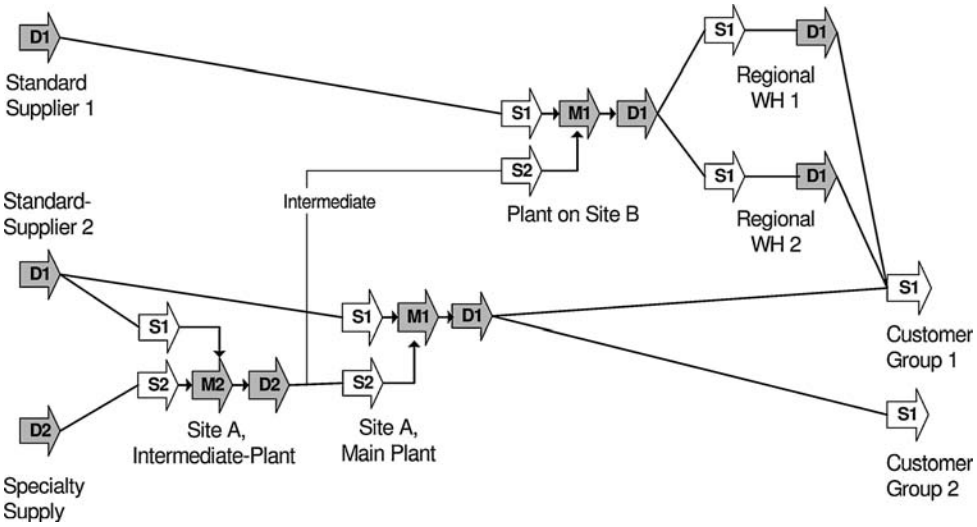

Fig. 1.5 Thread diagram with material flows and process types.

potential defects become apparent by reviewing the as-is-state of the supply chain and the corporate strategy, e.g., small trading units are filled at the plant site and delivered to the customers although a distributor is contracted to fulfill this task.

Accordingly, the process types are refined by using a so-called thread diagram which links the process types together. From the thread diagram it becomes apparent whether the process types fit together. For example, the only customer of a certain product places his orders such that it is possible to start production after he places the order while being sure to deliver in time. In this case, the thread diagram will show a process type for the deliver process which is deliver-make-to-order-product (Figure 1.5). On the other hand, operations produces this product as stock which will be depicted as the process make-to-stock. These process types do not fit as it is obviously not necessary to build up stock to satisfy the customer. The stock leads to additional net working capital and, thus, additional costs which either prevent additional profit or will lead to competitive disadvantages.

After the material flow, the information flow in the organization is described in a matrix where on the left side the functional units are represented. The process elements of level 3 are assigned to the functional units as depicted in Figure 1.6. From this figure it can be seen which departments are responsible for processes and where the responsibility is unclear.

The figures mentioned in this section are sufficient to describe the as-is state of the supply chain. Establishing and discussing these figures lead to first ideas of potential enhancements of the supply chain regarding the structure of the supply chain and the process flow.

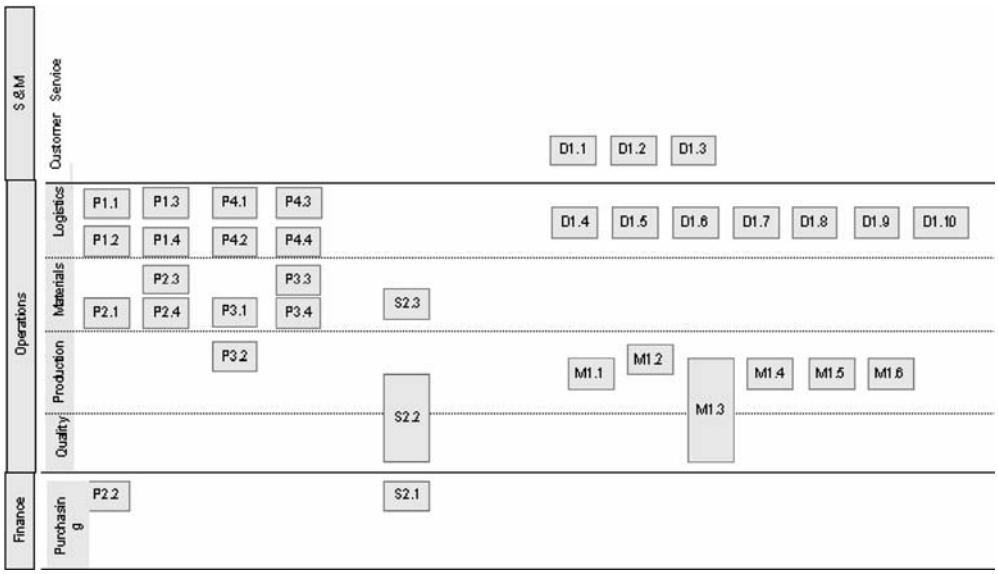

Fig. 1.6 Assignment of elements of level 3 to departments.

1.5.2.2 Performance Measurement

A perfect structure and process flow does not necessarily mean that the supply chain is performing well. For this purpose, key performance indicators and the drivers for the economic value added (EVA) are calculated.

The EVA [10, 11] combines information from the profit and loss statement (revenue, costs, earnings before interests and taxes (EBIT), etc.) and the financial sheet (net working capital (NWC), assets, etc.). The EVA is the interest calculation in absolute measurements and strongly related to the return on capital employed (ROCE) where the gained interest rate is calculated (Figure 1.7). In the long term, this interest rate should be above the capital costs of the company which is the interest rate the company has to pay for a credit on the capital market. Hence, a positive EVA means that the company has earned some money above the capital costs.

Furthermore, the SCOR-model defines five generic performance attributes to measure performance:

- reliability,
- responsiveness,
- flexibility,
- cost,
- assets.

There are different levels at which the performance can be measured. At the first level performance of the supply chain as a whole is measured. Further performance levels include level 2 processes and sub-processes within a level 2 process.

Using these performance indicators for benchmarking is usually very complex due to the diversity of production processes and business models. For example, processing special chemistry in batch processes usually leads to higher fixed costs

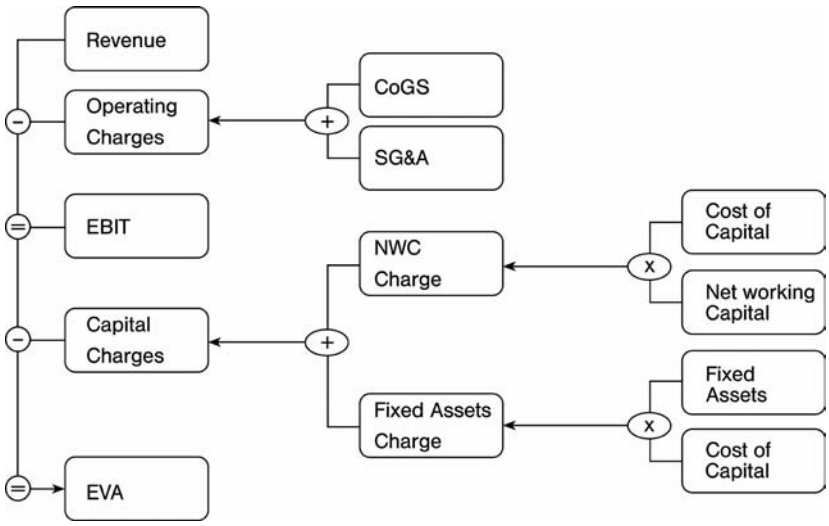

Fig. 1.7 Calculation of the Economic Value Added (EVA).

(e.g., for customizing the product) compared to continuously operated plants for processing bulk materials which are more dominated by variable costs. However, comparing the numbers and checking with the business strategy gives first indications regarding critical or costly processes.

These indications lead to detailed investigations of the processes, e.g., from a financial perspective this might be revenue, contribution margin and earnings before interests, taxes, depreciation and amortization (EBITDA) of certain product groups and customers, cost analysis for specific processes, overall equipment efficiency or research and development costs.

Based on the investigations, potential improvement actions are listed and evaluated regarding the financial impact on the EVA, costs and effort of implementation, and potential risk . This results in a prioritization of the improvement actions.

The prioritization of the improvement actions is the final results of the SCOR-projects which is not aiming at a detailed project description including cost effectiveness studies and project plans but at short cut evaluations and project proposals.

1.5.2.3 **Further Steps**

While the SCOR-project assumes the strategy and goals to be given, the SCOR methodology recommends as a next step to consider alternative targets for improvement and determine how they might improve the company's performance. Similarly, one can identify which changes would yield the highest return and priorities any improvement efforts.

The SCOR-model provides a number of tools to help redesigning a supply chain. It provides tools for identifying gaps between strategic considerations and operational practice and suggests best practices used by companies with superior supply chains. Once the design is complete, it has to be implemented by using software and human performance improvement techniques. After the implementation, sustainability of the changes and the improvement of the supply chain performance has to be ensured which addresses issues such as on-line performance measurement and supply chain controlling. A project approach usually ends here. Nevertheless, supply chain optimization should be a process rather than a project aiming at the continuous improvement of the supply chain.

1.6 Summary

Regarding the potential of supply chain management, several authors report impressing numbers regarding lowering of inventories, reduction of cycle times, higher degrees of service, etc. However, this does not mean that supply chain management is the solution to all the problems encountered in managing a supply chain. It should be pointed out that supply chain management potentially creates new problems.

Some of the potential new problems to be taken into account arise from organizational changes. Taking part in a supply chain means to give up – at least

partly – the control over a company's resources. Furthermore, long-term partnerships which are said to be essential to establish a supply chain may lessen the stress of competition which is said to be the driving power of progress.

Besides these problems, the impact on financial issues has to be discussed. There is no doubt that the introduction of supply chain management improves co-ordination, communication and control of the supply chain in general, leading to lower costs or some competitive advantage. Theoretically, this effect is due to the fact that the optimum of a complex system is not the same as the local optima of its components. However, the global optimum may result in sub-optimal results for some of its components. For an economic system this means that some companies may suffer from their participation in a supply chain. This leads to the problem how to distribute the economic value along the supply chain. The problem becomes more complex if we consider large companies taking part in several – maybe competing – supply chains.

Apart from these potential problems, there is huge potential by applying supply chain methods and processes to increase competitiveness and decrease supply chain cost. The skills and competencies to realize (the vision of) supply chain management are not widely understood or readily available. Therefore focused education and training is required based on the industry's specific requirements.

Abbreviations

CoGS	costs of goods sold
SG&A	selling, general and administrative expenses
NWC	net working capital

References

1 Simpson, K.F. (1958) In-process inventories. *Operations Research*, **6**, 863.

2 Metz, P.J. (1998) Demystifying supply chain management. *Supply Chain Manag Rev*, (*4*), 46–55.

3 Oliver, R.K. and Webber, M.D. (1992) Supply chain management: Logistic catches up with strategy, in M. Christopher (ed.), *Logistics* – the stratetic issue, Chapman Hall, London, 1992, p. 61.

4 Forrester, J.W. (1958) Industrial dynamics: a major breakthrough for decision makers. *Harvard Business Review*, (July/August), pp. 37–66.

5 Forrester, J.W. (1961) *Industrial Dynamics*, MIT Press, Cambridge MA.

6 Lee, H., Padmanabhan, V. and Whang, S. (1997) The bullwhip effect in supply chains. *MIT Sloan Manag Rev*, **38**(*3*), 93–102.

7 Lee, H., Padmanabhan, V. and Whang, S. (1997) Information distortion in the supply chain: the bullwhip effect. *Manag Sci*, **43**(*4*), 546–558.

8 Anderson, E., Fine, C. and Parker, G. (2000) Upstream volatility in the supply chain: the machine tool industry as a case study. *Prod Oper Manag*, **9***(3)*, 239–261.

9 Terwiesch, C., Ren, J., Ho, T.H. and Cohen, M. (2005) An empirical analysis of forecast sharing in the semicondutor equipment supply chain. *Manag Sci*, **51**(*2*), 208–220.

10 Roztocki, N. and Needy, K.L. (1998) An integrated activity-based costing and economic value added system as an engineering management tool for manufacturers. ASEM National Conference Proceedings, Virginia Beach, 1–3 October 1998, pp. 77–84.

11 Hostettler, S. (2002) *Economic Value Added (EVA)*, Paul Haupt, Bern.

12 Supply Chain Council (2006). Supply-Chain Operations Reference-model-SCOR overview, Supply Chain Council, Inc., Pittsburgh, PA. Available at http://www.supply-chain.org.

Part II
Simulation

2
Logistics Simulation in the Chemical Industry

Markus Schulz and Sven Spieckermann

2.1
Introduction

For several years, the manufacturing industry has had to deal with increasingly difficult conditions such as a growing number of product variants, smaller lots and reduced batch sizes. At the same time the development from a supplier to a buyer market, where (besides prices) product quality and delivery dates also play a major role in the sales process, puts increasing pressure on production logistics. In this context, the objective is to achieve an economic production process with a high level delivery service, low inventories, and short lead times. These classic challenges, at times with slightly varying target variables, can be resolved or at least simplified with computer-aided analysis of production and logistics processes. One widespread technology for the analysis of production processes is computer simulation. Simulation helps to increase the transparency of the structures and operating rules within a production system and it allows a quantitative assessment of the efficiency of material and information flows. Some typical areas of application are bottleneck analysis, balancing of production and buffer capacities, support of investment decisions, and issues around dispatching, scheduling and sequencing of production lots and orders.

2.2
Areas of Application for Logistics Simulation in the Process Industry

The methodology of logistics simulation was first used with success in the early 1970s in the discrete manufacturing industry – mainly in the machine and the automotive industry. The origins of simulation technology date back even further and many significant developments were initiated by military applications. Some historical information can be found in Nance [1]. In the machine industry and even more in the automotive industry, simulation has for many years become a well-established methodology and very few investment decisions are made without

it. On the basis of the success in the discrete manufacturing industry, several companies in the process industry began using material flow simulations in the late 1980s and early 1990s, initially for bottleneck analysis and decisions on plant design (see Watson [2]). However, it turned out that simulation software tools and the results from other industries could be used in the process industry only after some modifications. Günther and Yang [3] give several examples on the specific complexity of processes in the process industry: there are constraints to batch sizes, shared intermediates, changing proportions of input and output goods, production of by-products, limited predictability of processing times and yields, blending and mixing processes, use of multi-purpose resources, sequence and usage dependent cleaning operations, finite intermediate storage, product specific storage devices, cyclical material flows, usage of secondary resources such as energy or steam, complex packaging and filling operations, detailed quality controls. These and more restrictions and side factors distinguish production planning and hence modeling of production processes in process industry from discrete parts manufacturing as it is found, e.g., in the automotive industry.

Hence, it took the software industry longer to provide tools for logistic simulation that are capable of covering these extended requirements and could be used in the process industry environment. During the 1990s the process industry in Germany started several joint initiatives to enhance discrete-event simulation packages to their needs. Today, several large companies in the chemical industry make use of the benefits of simulation.

The areas of application for material flow simulation and the simulation activities of the users in the chemical industry do not only cover the various sections of the supply chain on different levels of detail but also the entire lifecycle of technical systems, including the organizational procedures. Simulation is used in the planning and engineering phase, during system ramp up, and in the operational phase, depending on the task and the objective of the investigation as indicated in Figure 2.1.

This chapter strives to give some insight into the use of simulation in the chemical industry. Accordingly, the remainder of this paper is organized as follows: Section 2.3 provides a short overview on the typical steps in a discrete-event simulation project. Section 2.4 presents three examples on the application of simulation in the process industry. The examples highlight the great scope of possible applications starting at the planning (or re-engineering) of the replenishment in a multi-site production network, supporting decision-making within a plant engineering process on production and tank capacities, and finally being part of the daily lot scheduling process in the context of material requirement planning (MRP) in collaboration with enterprise resource planning (ERP) systems such as SAP R/3.

Section 2.5 discusses some experiences with expenses and benefits of simulation use and Section 2.6 will provide some technical information on available simulation software. The concluding Section 2.7 summarizes some of the aspects and contains statements on possible future lines of development.

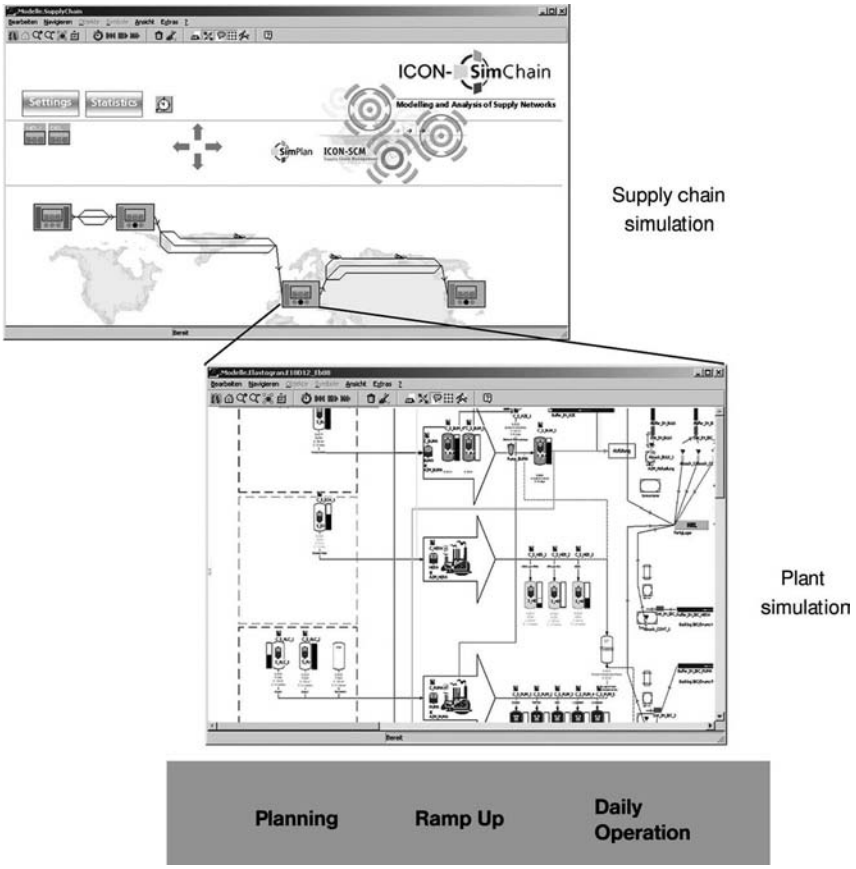

Fig. 2.1 Level of detail and timeline of simulation application.

2.3 The Simulation Process in Manufacturing and Logistics

In the simulation literature there are several process models for simulation studies, e.g., Sargent [4], Nance and Balci [5], or VDI [6]. Whereas these process models deviate in several details they all have four typical steps in common:

1. Problem analysis and definition of objectives.
2. Data acquisition.
3. Model design, implementation, and validation.
4. Application of the model.

Of course, these steps usually are not processed in a linear order. Rather there are loops and iterations. The acquired data might lead to new insights in the problem

and hence cause a shift in project objectives. The model implementation may cause additional data requirements and the application might show that there need to be model modifications. Depending on the scope and the objectives of the model the extent of the four steps may differ significantly. However, there are characteristics for each step of a simulation study which are briefly discussed in the following subsections.

2.3.1 Problem Analysis and Definition of the Objectives

Every logistic simulation needs to be preceded by the analysis of the investigated problem. The customer's needs and expectations are defined within one or several project meetings. One of the main issues to be answered is whether simulation is the appropriate methodology to tackle the specific problem. In this context, general advice is rather difficult, but some criteria for the use of simulation are a sensible cost-benefit ratio, a lack of alternative methods, e.g., analytical-mathematical models, stochastic influences with regard to resource availability or incoming orders, etc. Besides the assessment of simulation as the suitable method it is also crucial that the project objectives are stated as precisely as possible and that there is a clear understanding that these objectives may well be achieved by means of simulation. In this context it should be made clear that simulation is not a substitute for a sound planning process. Simulation does not *develop* concepts but it is a good means to assess them.

2.3.2 Acquisition of Required Data

After clarification of objectives and methodology the relevant data to create and run a simulation model is defined and compiled. In general, the required data can be divided into technical data, organizational data, and system load data. The technical data includes information about the system topology and layout (e.g., the number of tanks, batch processing units and pipes for the simulation of a chemical plant), material flow data (e.g., transportation via pipe, bulk, container, etc.), performance data (e.g., the input and output information for the processes), and usage times of equipment and production facilities, storage capacities (e.g., the capacity of the tanks) and availabilities (including cleaning or set-up times of tanks or processors). Organizational data includes production strategies (e.g., the campaigns to run), rules for product manufacturing (e.g., the dispatching of orders in a specified process step) and information about staffing of processes and the working time models. Information about production orders, quantities, and deadlines as well as product data (e.g., formulas) are described as system load data. The period of time considered within the model (e.g., the production schedule of one year), the level of detail, and the quality of the data to be recorded depend mainly on the complexity and requirements of the task at hand.

Detailed acquisition of data before the actual model is created leads to increased transparency of the procedures and thus usually has its own intrinsic value. However, the effort to collect and prepare data for a simulation study should not be underestimated. As a rule of thumb, the data collection sums up to one third of a simulation project's time budget. In supply chain studies, where data of several production sites may be needed, the expenses for data collection may even be higher.

2.3.3 Model Design, Implementation, and Validation

Modeling in the context of this article means the implementation of an actual or a planned production or logistics system in a computer model. In this respect, modeling has some similarities to a software engineering project and as within a software project it is good practice to specify or design an application before the implementation starts. Hence, a (good and experienced) simulation analyst creates a conceptual model and a formal model before actually implementing a computer model. The conceptual model describes in common language the scope of the model. It contains decisions on the elements, structures, rules and stochastic influences in the actual or planned system which have to be considered important for the project objectives and therefore need to be part of the model. Typical decisions during the process of conceptual modeling are for example which sites or product lines may or may not be included in a supply chain model. Decisions on the level of detail of a plant model are explained (e.g., whether the maintenance staff is explicitly modeled or not). In general, it can be said that a system should not be modeled as exactly as possible but as exactly as necessary to tackle the respective task – and the root for this decisions is conceptual modeling.

While the conceptual model is still on a non-simulation-expert level and understandable for the simulation expert as well as any project engineer, the formal model is steps further towards an expert level. Here, data structures and algorithms may be designed in detail before, in the final modeling step, the formal model is transformed into a computer model.

The implemented model must be tested with regard to correctness and completeness. Therefore, i.e., to validate the model and ensure the credibility of the simulation results, suitable scenarios with a broad spectrum of different events are reproduced with the model and compared to reality (or to expectations on reality). A model validated successfully can then be used for several systematic experiments (or as part of other applications, e.g., as part of a MES).

Modeling and validation require the close cooperation of all parties involved in the project. Further success factors in simulation modeling include adequate planning experience, special experience with simulation tools, and the ability to think in abstract structures.

2.3.4 Application of the Model

The way of applying a simulation model depends on the purpose it has been created for. A model for supply chain or plant design usually is created to support engineering decisions. Hence, the application of such a model means to conduct several series of experiments where design parameters of the considered system are modified. Design parameters may be production or warehousing capacity of a site, the allocation of products to sites and warehouses, or lead times for products in supply chain studies. In a study to support plant design, the capacity of tanks or of production processes may be the subject of the experiments. The results of the model experiments are presented using appropriate key figures. Common key figures are throughput times, output per time unit, tank and resource utilization over time, service levels, etc. Most simulation software tools present the results in tables or graphs such as line graphs, bar diagrams, pie charts, or Sankey diagrams. Other important information about the behavior of the simulated systems is provided by process animation. This visualization of the procedures during a simulation experiment provides additional transparency and reliability for planners and simulation experts when they are evaluating the model behavior and the results.

If a simulation model is used as part of a MES to evaluate production schedules and support daily operation the presentation of simulation results quite often is integrated in the MES environment. The planner might not even see or know the simulation model itself. There might be a feature such as "assess order schedule" within the MES, which starts a simulation experiment. Details on this and on the other ways of application will be illustrated by the examples in the next section.

2.4 Case Studies

As sketched out initially, simulation can be used at different points in time of a production system lifecycle and with a different scope (see Figure 2.1). Considering the lifecycle and the scope, the three case studies described in this section may be classified differently, from supply chain to plant level and from planning to daily operation.

2.4.1 An Example of Simulation in Supply Chain Design

In general, supply chain simulation can be used along the lifecycle of supply chains, that is from supply chain design to support of supply chain operation. In supply chain design, simulation models can support supply network design (decisions on the location, ramp up or shutdown of production or warehousing sites) as well as the adjustment of control parameters such as safety stock at different

nodes, production lot sizes, transport options, etc. An overview on the potential of supply chain simulation based on a survey of 80 articles can be found in Terzi and Cavalieri [7].

The case considered in the following is about a small study which was carried out using a discrete-event simulation tool and a specific add on to model supply chains efficiently. Even though it is a rather small example it highlights the level of modeling and decisions under consideration in a supply chain simulation study. The objective of the study was to assess different replenishment strategies for a product processed in Europe and refined at three different sites in China. The transportation is carried out by cargo vessels. The analysts had to evaluate three alternative strategies:

- direct replenishment of the tanks at the three Chinese production sites based on safety stock and forecasts on customer demand;
- using a concept of floating stock, i.e., the product is shipped in Europe without already having a request for replenishment from China. Instead, the anonymous "floating" stock is assigned to an order while it is on the passage somewhere between Europe and China;
- employing an additional intermediate tank system in Malaysia.

Due to stochastic demand in China, stochastic production yields in Europe and some stochastic variations in transport times between the two it was decided to support the decision between these alternatives by means of simulation. The structure of the simulation model is shown in Figure 2.2.

The resulting stock at the different locations of the supply chain is displayed in charts as in Figure 2.3. Additionally, the total stock, the different transportation costs and the costs for using the resources in Malaysia were taken into account and finally led to a recommendation for the floating stock concept together with a specific combination of decision variables.

2.4.2 An Example of Simulation in Plant Design and Engineering

Whereas the model objects in supply chain simulations are whole plants or transport relations between plants, in plant engineering the simulation objects are on a far more detailed level. Typical elements of the modeling process on this level are process units such as reactors, tanks, pumps, filling stations and discrete transport units such as bulk containers, bigbags, etc. In the presented case study, the objective was to investigate whether a tank farm for a given product had the right capacity to ensure the continuous supply of downstream processes on site (captive use) as well as the satisfaction of external customer orders. Figure 2.4 gives an overview over the modeled structure. On the left hand side there are two processors continuously producing a product. While there is on average a constant production rate, the effective daily yield is significantly fluctuating due to tolerances within

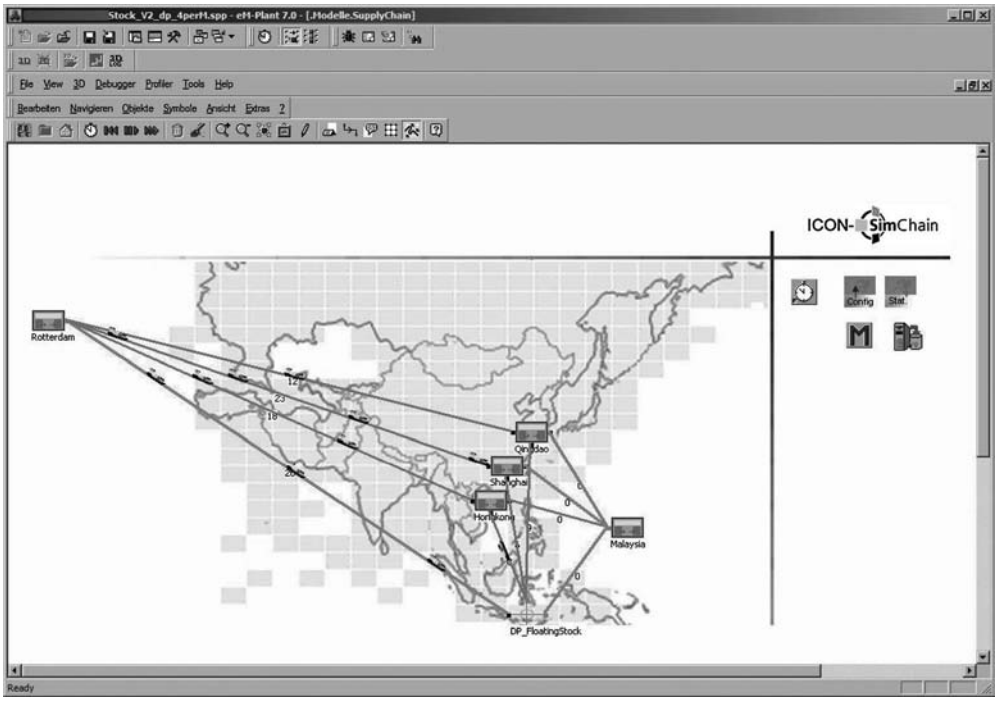

Fig. 2.2 Example of a highly aggregated supply chain model.

the process, maintenance activities, etc. Figure 2.5 indicates those variances in production output for the first 15 days of the simulation period. The amount requested by customers is known several days in advance but is also subject to substantial variations as can be seen on the lower part of Figure 2.5.

The two processors are delivering the product via pipes into one of several tanks in a tank farm. Each tank has parameters such as capacity or cleaning time. From the tanks the product is either delivered to processes on site (captive use) or to customers based on given customer orders. Additionally, it may be stored in two external buffers. In that case there are transportation moves from the tank farm to one of the external buffers (in case of over-production) or from the external buffer to the tank farm (in case of shortages in production) induced.

Even though this model does not cover several production sites such as the preceding example it still requires a significant amount of input data:

- capacity of upstream and downstream production facilities including averages and deviations in day yield, maintenance policies, shift models and availability;
- customer orders per day and per transport medium (IBC, bigbags);
- number and capacity of tanks including rules for cleaning, re-filling, etc.

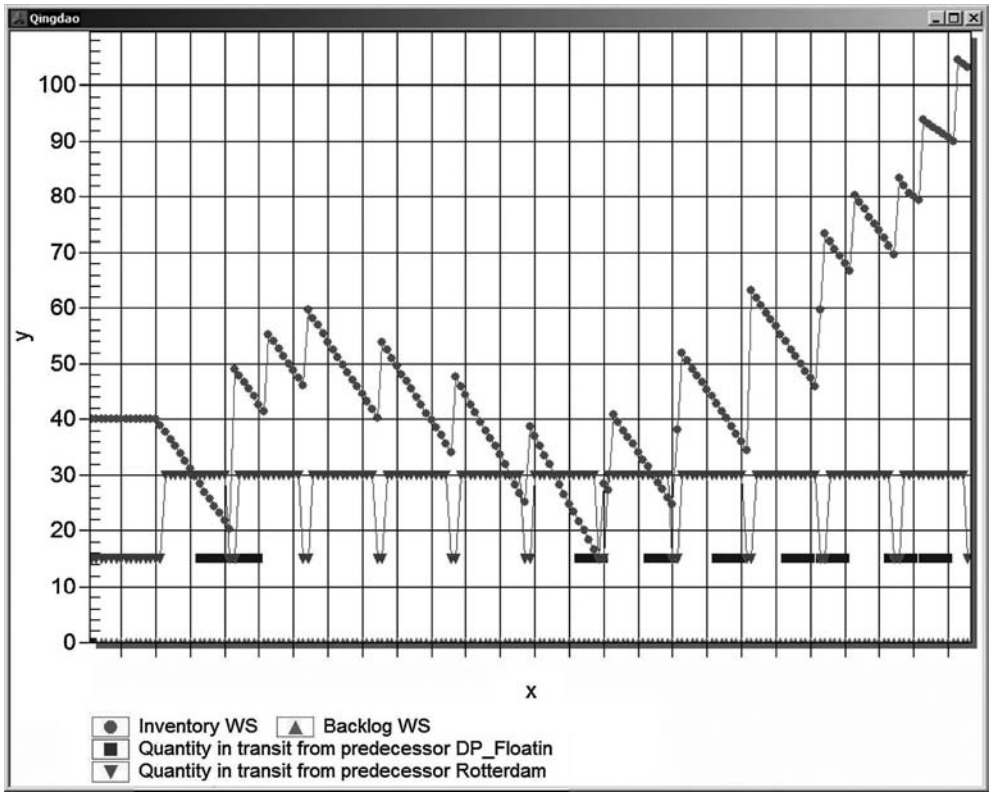

Fig. 2.3 Inventory, Backlog, and Transit Quantity during simulated period.

Based on these and other inputs the simulation model provides several outputs. Main results are:

- a chart for each tank showing the quantity held over time (see Figure 2.6);
- the service level in terms of customer orders fulfilled on time;
- the number of transports classified by transport type (transport of customer ordered material or transports from and to external buffers).

In the considered case study the simulation results had significant impact on investment decisions within the tank farm and on the agreements negotiated with the service partners responsible for the external buffers.

In particular results on tank capacity are a typical output of simulations on the plant engineering level. It could be argued that these results may be obtained without simulation as well and this is true as long as the stochastic impact on supply and demand is within certain boundaries. As soon as the facility needs to be able to handle stochastic supply and demand with significant variations static calculations reach their limits. These limitations become even more critical if a multi-product process is analyzed as quite often is the case.

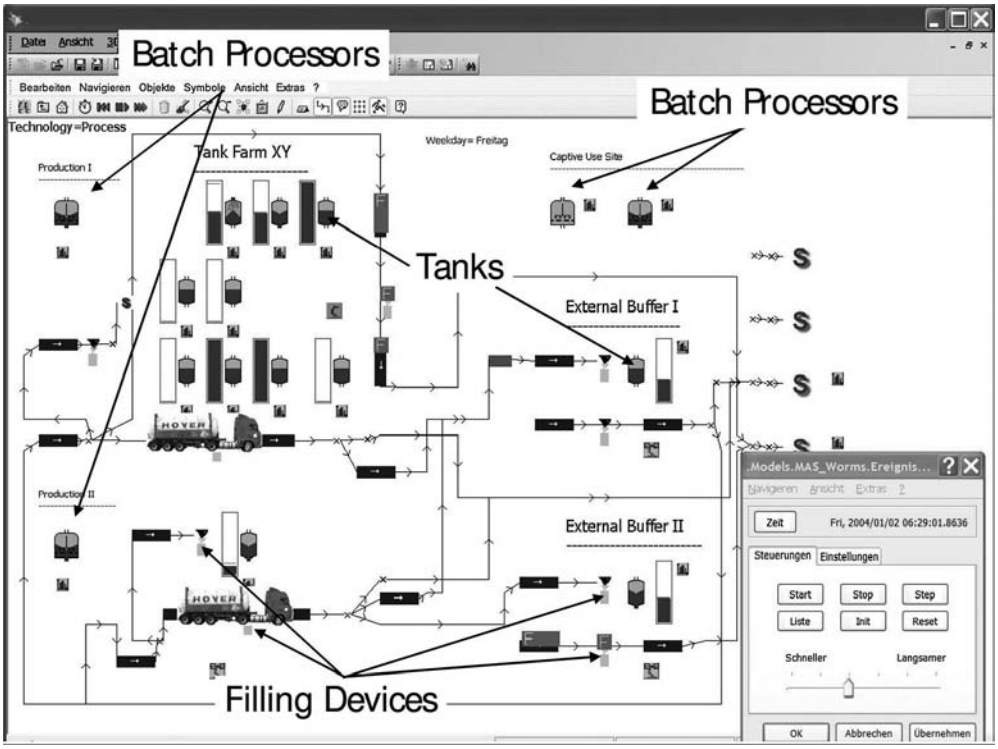

Fig. 2.4 Screenshot of a combined supply chain and plant simulation model.

2.4.3 An Example of Simulation in Plant Operation

The main difference between the third and the other two case studies is that the simulation model here is applied as a tool for daily production planning. In that sense the model may well be considered as part of the MES for the production. Its main purpose is to assess order schedules (calculated by optimization algorithms either within the model or handed over by an ERP system). Whereas the structure of the model and the data requirements may be very similar to a simulation model used for the support of the engineering process, the *way* the data gets into the model needs to be completely different. During the engineering process it usually is sufficient (and quite often the only way) to collect and compile the required data manually in spreadsheets or even to enter it directly into the simulation model. If the model is to be used for daily order dispatching in a production environment it needs to have several interfaces to other software tools: orders need to be supplied from the ERP system (e.g., from SAP R/3), the current status of the production needs to be fed into the model (coming from a shop floor control or monitoring system), and there needs to be a user interface that is easy to handle for the staff

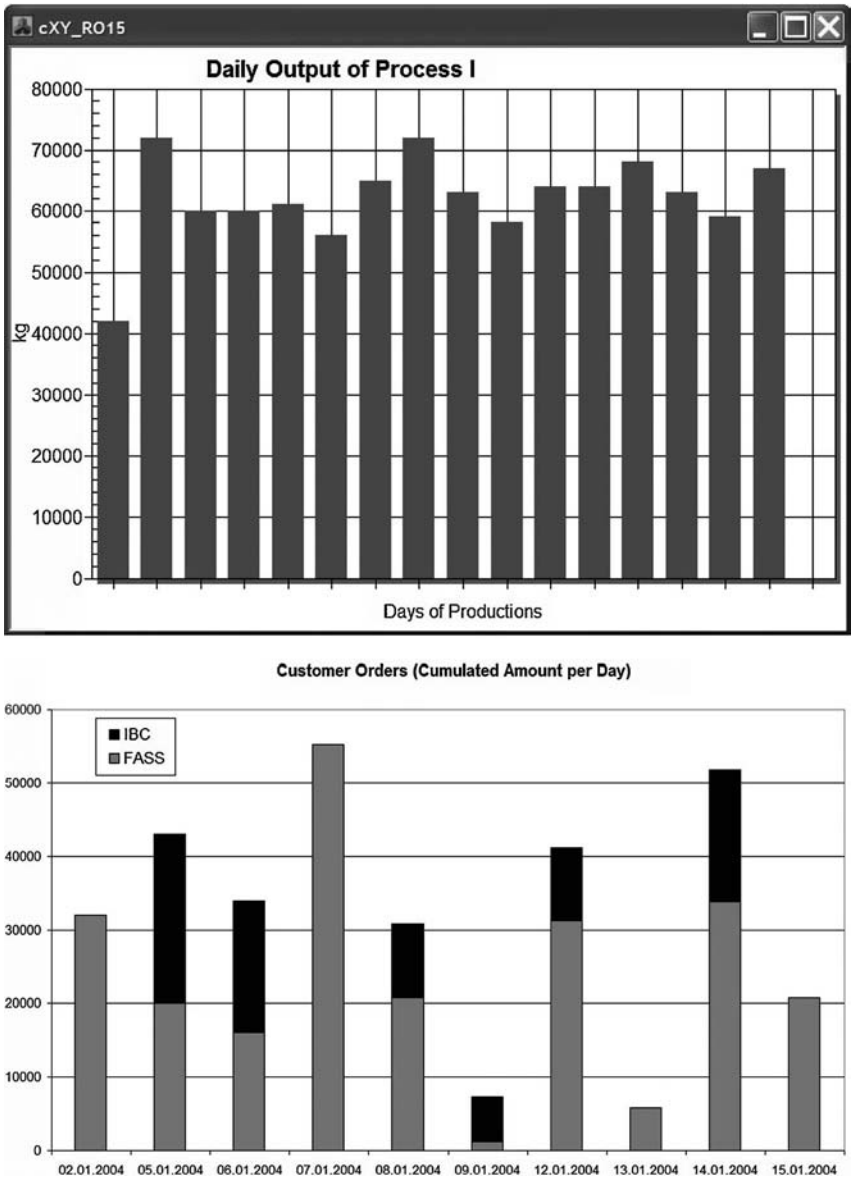

Fig. 2.5 Daily production output of simulated process and customer orders.

in charge of production control. A generic framework for the architecture of such a system is shown in Figure 2.7.

Of course there are many planning solutions offered by vendors of MES or APS (advanced planning systems) (see for example Stadtler and Kilger [8]). The great advantage of a simulation based solution is the almost unlimited flexibility

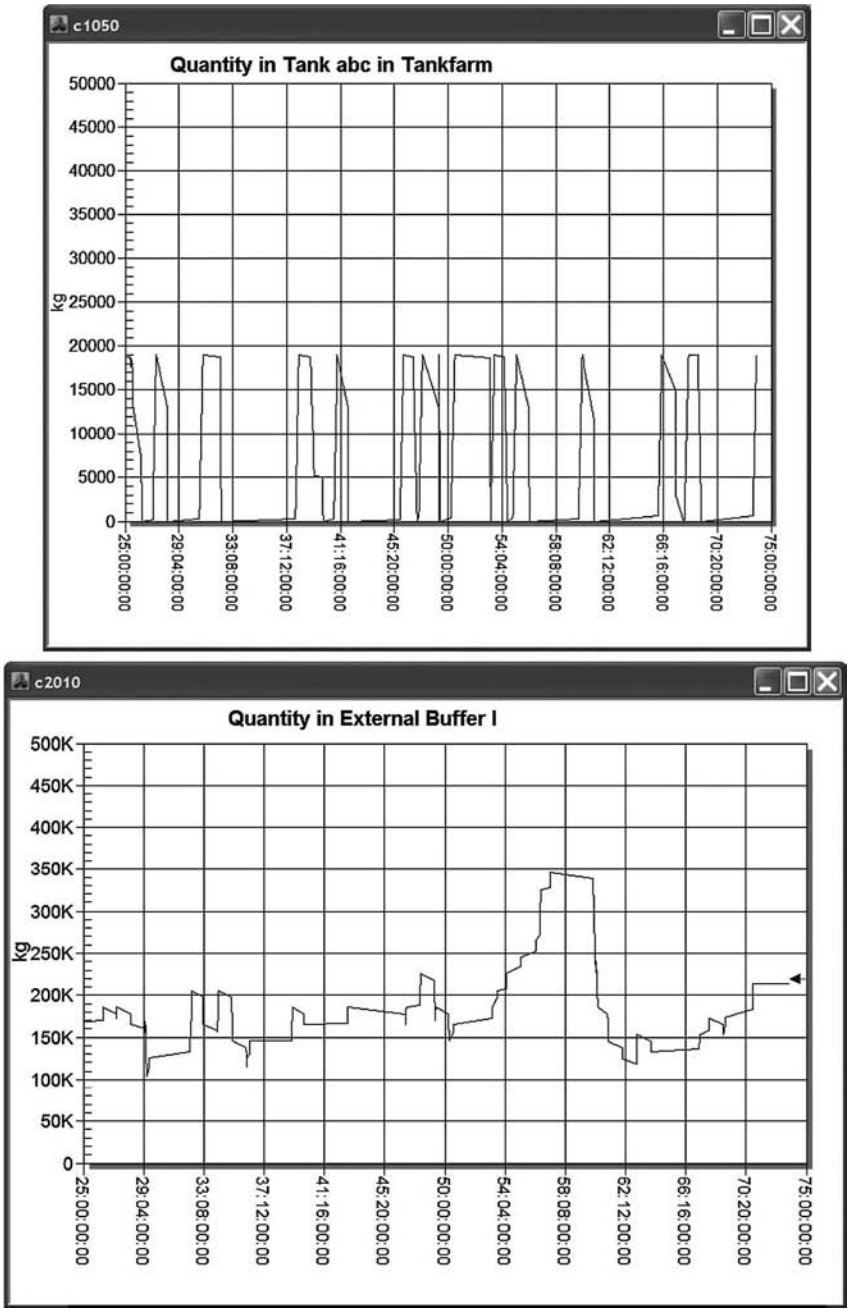

Fig. 2.6 Simulated quantity in selected tanks.

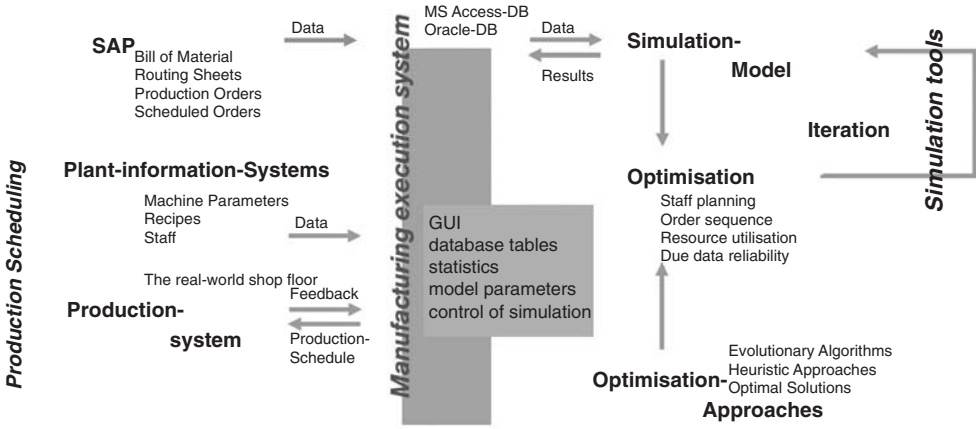

Fig. 2.7 Scheme of a MES integrated simulation application.

in modeling the production processes. Thus, even schedules for very complex production environments can be assessed. However, the main expenses in these projects quite often do not stem from the modeling process itself but rather from adjusting the user interface, implementing the interfaces to other applications, establishing a stable data transfer and the roll-out and test of the system. Hence, the scale of a MES integrated simulation project tends to be five to ten times larger than the typical engineering support project.

2.5 Benefits and Expenses of Simulation Projects

The expenses of simulation projects vary considerably depending on the type and the area of application. Apparently, this is the case if one looks into such different applications as engineering support and MES functionality. However, even simulation studies for engineering support can comprise several days or several months. The proportions for the different phases of a simulation in the overall project expense can be defined as follows: definition of the objectives: 10–20 %; data gathering and preparation: 25–40 %; modeling: 30–50 %; validation: 10–25 %; experiments and analysis of the results: 20–30 %. Such as in other fields of manufacturing or software engineering there is no standard procedure for evaluating the cost-benefit ratio for logistic simulation. Some general advantages are an improved understanding of the processes and the possibility of taking specific, effective actions. Incorrect planning can be identified at an early stage and the planning risk is minimized. However, the benefits can only be quantified for specific projects. Nevertheless, a cost-benefit ratio of 1:4 to 1:6 appears to be realistic. In investment projects for new plants, a cost-benefit ratio of 1:20 and more could quite easily be achieved.

In spite of the potential benefit, the possibilities of logistics simulation are often not fully exploited. Some reasons are a lack of knowledge about the basic methodology and the available simulation tools, the fact that simulation models are rarely deployed more than once, and simulation investigations are often integrated into the planning process too late.

2.6 How a Simulator Works

There are many different simulation techniques for modeling different types of complex systems (e.g., process simulation, finite element methods, system dynamics). Essentially, discrete-event simulation has become established for use in logistic issues. Specific software tools (simulators) are needed for the implementation of simulation studies. The corresponding market has grown over the years and now offers a wide variety of programs, e.g., the simulation tools eM-Plant, Witness, or AutoMod. An overview of commercial simulation packages is provided in a biannual series by Swain [9]. There are significant differences in the level of maturity of, both, the tools and the software vendors, and in the technology that is used for modeling and optimization, the possibilities of integration with other software systems, and the costs for license and software maintenance.

The discrete-event simulation tools of the leading software vendors usually can be used to cover all the fields of applications discussed in this article. Some may have a focus more on detailed material handling, while others offer better support on modeling supply chains. Also, there are significant differences as to what extent interfaces are supported, a feature which is becoming important if a simulation model is going to be integrated with other applications. An analyst in the process industry who is planning to use simulation and who is looking for the right software should specifically pay attention to the following issues:

- Since almost all of the discrete-event simulation tools have their roots in modeling discrete manufacturing processes, some of them still do not offer the objects an analyst in the process industry needs. Every simulation package will offer typical objects of discrete manufacturing plants: machines, buffers, parts, vehicles, etc. But to model chemical processes objects such as tanks, pipes, pumps, reactors, etc., should be provided by the simulation tool. Similarly, if the simulation tool mainly is used for supply chain studies, it should offer objects suitable to model on a higher level. A pipe or a packaging machine are not quite the appropriate objects to start the modeling of a worldwide supply chain. Here, it rather takes objects such as site, transport relation, depot, etc.
- A discrete-event simulation tool considers – *nomen est omen* – discrete events at discrete points in time. Typically, in a discrete-event simulator items such as parts are moving through the modeled system changing their state, e.g., when they enter or leave a machine. A reactor in the process industry continuously produces a certain output. This is something a discrete-event simulator is not really made

for. One approach to model continuous output with a discrete simulator is by "slicing" the output in discrete portions, e.g., by using a part to model a certain amount of the reactors yield. This can be very costly in terms of computational performance if the slices are too small. If the slices are too large the simulation may loose accuracy. Another approach for continuous modeling is not to use moveable objects such as parts but rather model changes of reactor processes by state variables and counters. This may be a bit less intuitive but it usually leads to increased speed of simulation experiments in combination with sufficient accuracy. However, an analyst should be aware of both approaches and carefully look into performance limitations of the considered simulation package.

- The integration of a simulation application into a MES or ERP environment requires several interfaces. The simulation package needs to be "open" in the sense that it can easily connect to other IT systems and that other IT systems easily can connect to it. Possible interfaces are TCP/IP or ActiveX for telegram exchange, ODBC as database interface, or an interface to a programming language such as C++ or JAVA. These interfaces are also a prerequisite for a sensible way of exchanging data with an ERP system as SAP R/3 because usually the data exchange with such a system is implemented via one of those means. The degree of support for the different interfaces can be very different from simulation package to simulation package.

2.7 Developments in the Field of Logistics Simulation

At present, the trend is moving from the analysis of individual production and logistics systems towards the optimization of entire production networks, that is, the optimization of the distribution to different production locations taking account of the procurement and distribution chain as indicated by the remarks on supply chain simulation earlier in this chapter. However, in particular for these applications there are some challenges in the basic work of providing consistent and coherent data describing the processes at different sites which most likely are located in different countries.

Another perspective for production simulation is automatic capacity utilization optimization of multi-product systems. As discussed, this task may be very difficult because of the many different variables and boundary conditions. In an environment integrating optimization and simulation, the optimizer systematically varies the important decision variables in an external loop while the simulation model carries out production planning with the specified variables in the internal loop (see Günther and Yang [3]). The target function, for example total costs or lead times, can be selected as required. The result of optimization is a detailed proposal for the sequence of the placed orders.

It finally may be stated that the use of discrete-event simulation on different decision levels even though state-of-the-art is still slightly underrepresented in the process industry. However, since the technology has proven itself in an

increasing number of cases in the past couple of years, there seems to be some promising potential for further successful applications.

References

1 Nance, R.E. (1993) *A History of Discrete-Event Simulation Programming Languages*. Proceedings of the 2nd ACM SIGPLAN History of Programming Languages Conference, *ACM SIGPLAN Notices*, **28** (3), 149–175.

2 Watson, E.F. (1997) An application of discrete-event simulation for batch-process chemical plant design. *Interfaces*, **27** (6), 35–50.

3 Günther, H.O. and Yang, G. (2004) *Integration of Simulation and Optimization for Production Scheduling in the Chemical Industry*. Proceedings of the 2nd International Simulation Conference 2004, Malaga, Spain, pp. 205–209.

4 Sargent, R.G. (1982) Verification and validation of simulation models, in *Progress in Modeling and Simulation* (ed. Cellier), Academic, London, pp. 159–169.

5 Nance, R. and Balci, O. (1987) Simulation model management requirement, in *Systems and Control Encyclopedia: Theory, Technology, Applications* (ed. Singh), Pergamon, Oxford, pp. 4328–4333.

6 VDI (2000) *VDI-Richtlinie 3633 Blatt 1 Simulation von Logistik-, Materialfluss- und Produktionssystemen, Draft*. Beuth, Berlin.

7 Terzi, S. and Cavalieri, S. (2004) Simulation in the supply chain context: a survey. *Computers in Industry*, **53** (1), 3–16.

8 Stadtler, H. and Kilger, C. (2005) *Supply Chain Management and Advanced Planning*, 3rd edn, Springer, Berlin.

9 Swain, J.J. (2005) Software survey gaming reality. *OR/MS Today*, **32** (6), 44–55.

3
Logistic Simulation of Pipeless Plants

Andreas Liefeldt

3.1
Pipeless Batch Plants

In the production of life-science products, adhesives, coatings, fine chemicals, etc., the trend towards small production volumes of application-specific high value products continues, and the time to market and the time in market of the products decrease. Therefore in the production of these chemicals flexibility and cost efficient and timely production of small amounts of material are key factors for economic success.

Pipeless plants are an alternative to the traditional recipe-driven multipurpose batch plants with fixed piping between the units. In this production concept, the batches of material are moved around between stationary processing stations in mobile vessels. The processing steps are performed at different single purpose or multipurpose stationary units but the material remains in the same vessel throughout the production process. The transportation of the mobile vessels can be realized by a transportation system that is fixed to the vessels or by automated guided vehicles (AGV) that pick up the vessels only to perform a transfer order [1].

In comparison to traditional recipe-driven multipurpose batch plants, pipeless plants provide a significant increase of flexibility.

The transport of products or intermediates in mobile vessels enables a quick change of the priorities among the orders and the bypassing of stations that are temporarily unavailable by parking or redirecting of vessels.

The reduced amount of fixed piping reduces the effort for cleaning and sterilization before product changeovers significantly. The necessary cleaning of vessels can be carried out in separate cleaning stations and the processing stations are cleaned in place (CIP). Due to the possibility of cleaning parts of the equipment separately, it is not necessary to shut down a number of coupled units of the plant during cleaning cycles, resulting in a higher utilization of the plant [2, 3].

Another advantage is the higher scalability of the processes, because vessels of different sizes can be used in parallel.

On the other hand, in comparison to traditional recipe-driven multipurpose batch plants, new technical requirements arise from the use of mobile units. An important pre-requisite for a safe and automatic production are reliable docking systems that provide failure-free connections between the mobile vessels and the stationary processing stations. In the docking process of the mobile vessels the connection of pipes, of electric power and of signal processing equipment is necessary. The vessels therefore must be placed accurately. If vessels of different size are used, the connections must be flexible enough to cope with these.

Because of the high standards for the avoidance of leakage and reliability, up to now the application of the pipeless plant concept is limited to plants with moderate processing conditions (low pressures and temperatures, in most cases atmospheric pressure) [2].

During the plant layout as well as during the operation of pipeless plants a large number of degrees of freedom including many discrete decision variables arise from the increased modularity and flexibility and these decisions strongly influence the profitability of the plant.

During the planning process decisions have to be made on:

- the plant layout and the positioning of the stations;
- the number, the size and the processing equipment of the vessels and the stations;
- the type of transportation system and the number of AGVs if automatic transport is included.

During plant operation decisions have to be made on:

- the scheduling of the production orders;
- the assignment of orders to vessels, stations and, possibly, AGVs;
- the routing of the mobile vessels and the timing of the transport operations.

In regard of the resulting combinatorial complexity it is useful to support the planning stage as well as the operation of pipeless plants by appropriate simulation and optimization tools. As a large part of the manual work, e.g., related to the cleaning of piping, vessels and stations, can be automated, the workforce in pipeless plants is usually small. This favors the application of a simulation- or optimization-based production planning and control system to support the operators of pipeless plants.

PPSiM is a graphical modeling and simulation environment that has been designed for the specific demands of pipeless plants and offers decision support in all areas mentioned above. The user is enabled to model a pipeless plant fast and in an intuitive manner and to perform simulation studies to compare design alternatives. As scheduling and routing algorithms are integrated in the simulation, dynamic simulation can be used to support the scheduling of plant operations if feedback from the real plant is used to update the state of the simulator.

3.2
PPSiM – Pipeless Plant Simulation

The modeling of the plant, the chemical processes, and the production plan can be performed entirely in a graphical manner using three editors, the recipe editor, the plant editor and the production plan editor. A fourth editor is available for the definition and update of a physical properties library. The physical properties library is required to parameterize the models of the plant and of the recipes for the calculation of state changes during the processing steps in simulation.

The editors and the underlying models were designed following the guideline of the standard ISA SP88.01. The main directive of the standard is the conceptual separation between the description of the plant and of the process. The main advantages of the standard are a clear and well structured representation of the possibly large and complex models of batch plants and a high reusability of model components. Additional information about the plant layout, the positioning of the equipment and the transportation system, which is not considered in the standard, is stored in the plant model as an undirected graph [4].

3.2.1
Modeling of the Production Processes

The chemical processes performed in batch plants are usually described by general recipes that do not reference specific units of the plant. The recipes are modelled in *PPSiM* by Sequential Function Charts (SFC) using the recipe editor shown in Figure 3.1. SFCs consist of recipe steps that describe the processing steps, and transitions that define the conditions that have to be fulfilled to move to the next step(s). As a consequence, the steps and the transitions in a recipe always have to be arranged in an alternating fashion. The recipe editor enables the modeling of any sequential, parallel or interleaving configuration of processing steps. The conditions of the transitions may specify thresholds for the overall weight, the temperature, and concentrations or the duration of a processing step. During the modeling process, a syntax checker prevents the modeller from generating syntactically wrong recipes.

To minimize the effort for modeling of the recipes, the possibility of an automatic scaling of recipes is implemented in the discrete process simulator. The recipes are assumed to be scaled to 100 %. The actual weight scaling of the referenced recipe is specified in the production plan on the control-recipe level. During the simulation, weight dependent transitions are scaled according to the specified weight percentage. Transition conditions which limit the duration of a processing step by a fixed execution period are assumed to remain valid.

3.2.2
Modeling of the Plant

The model of the plant describes the plant layout and the numbers and properties of the equipment and defines the operations that can be performed on

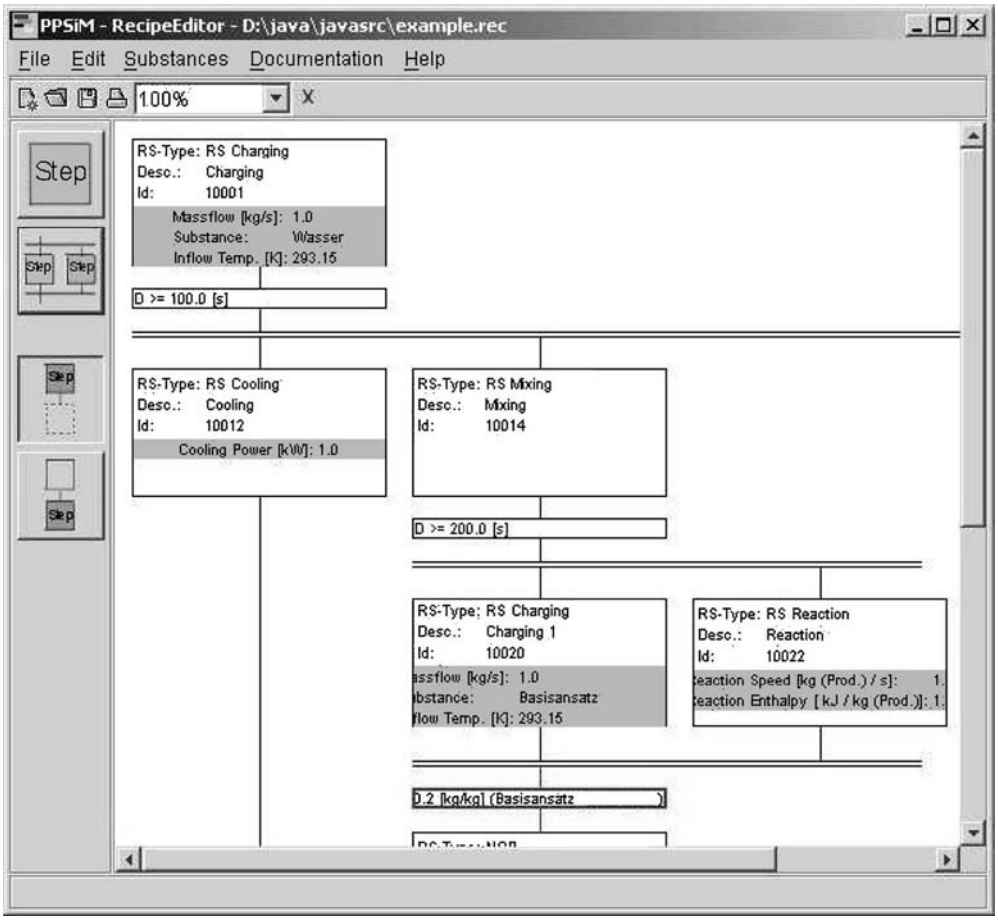

Fig. 3.1 Screenshot of the recipe editor.

the respective unit of the plant. A screenshot of the plant editor is shown in Figure 3.5.

Initially the overall plant area and the biggest moving object (assumed to be square) have to be specified. Resulting from these dimensions, a gridded map of the plant area is generated on which the equipment can be placed. The chosen positions of the vessels and of the AGVs are regarded as initial positions in a simulation run. Additionally the areas available for transport and the blocked areas (light and dark gray) have to be specified. Each square which can be used for the transport of a vessel is internally represented as a node of an undirected graph.

To specify the operations that can be performed on a piece of equipment *technical functions* have to be assigned and must be parameterized afterwards. Technical functions are elementary operations, for example charging, discharging, mixing, etc.

3.2.3 Modeling of the Production Plan

The third editor of the modeling and simulation environment is a production plan editor, which connects the two parts of the model and generates an executable simulation model. The production plan is hierarchically structured into orders, batches and control-recipes. Orders and batches are structuring elements containing only information about the desired starting times and the cumulated amount of raw materials and products of the underlying batches or control-recipes respectively. The connection of the two parts of the model, the recipes and the plant model, is done on the control-recipe layer. A control-recipe is generated from a general recipe and the desired weight scaling of the recipe. Then each basic operation of the recipe is assigned to a technical function of a vessel and a station. In addition, an AGV has to be assigned to this triple to carry out a transportation job of a vessel to a station if necessary.

This assignment of basic operations to the technical functions of vessels and stations and to AGVs can be computed automatically by a scheduling algorithm during a simulation run.

3.2.4 Simulation

The simulation of a production run of a pipeless plant is realized by a 2-level algorithm which incorporates a scheduling module, a routing module and a simulation module.

On the upper level, the scheduling module allocates the equipment which is required for processing the next batch. In these decisions, the current equipment allocation, the equipment properties and a cost-function are taken into account. The allocation is performed in a sequence according to increasing starting times of the batches.

The routing module and the discrete event process simulator are triggered by the scheduling module.

3.2.4.1 The Scheduling Module

The scheduling module maintains a master list of all batches and control-recipes and sorts the basic operations of the chronologically next control-recipe by their position in the recipe. In the case of identical starting times a priority which can be specified on the batch level determines the sorting. During the sorting procedure parallel steps in the recipe with all associated basic operations are regarded as one unit.

In the first step all suitable vessels are determined which meet the technical requirements of all basic operations of the control-recipe and which are able to hold the maximum load determined by the (scaled) recipe. Additionally all stations are selected which are suitable to fulfil at least the requirements of the next recipe step. In the case of parallel steps in the recipe, all basic operations occurring in this

step have to be performed by the chosen station. It is assumed that any AGV can transport any vessel, so no AGV pre-selection is necessary.

In the next step, all feasible equipment triples consisting of a vessel, a station and an AGV are taken into consideration. The overall execution times, including necessary waiting times, docking times, transfer times and processing times, are calculated for each equipment triple.

The transfer times and the durations of the processing step(s) are calculated by the underlying routing- and simulation-module. In case a suitable vessel is already docked to a suitable station, the routing-module is bypassed and the transfer time is set to zero.

After sorting the overall execution times, the availability of the equipment triples is checked. The first combination that is available during the required period of time is allocated, the starting times of subsequent basic operations are updated and the next basic operation(s) of the master list are processed.

If the resources were assigned manually to a control-recipe, the automatic assignment is bypassed and only the feasibility of the chosen equipment triple is checked.

3.2.4.2 **The Routing Module**

The routing module which is called by the scheduling-module calculates the fastest conflict-free path of a transportation job between two positions of the plant.

The topography of the plant is represented by an undirected graph and consists of nodes and undirectional edges. The paths through the plant are represented by node lists, which are chronologically sorted by the point of time an AGV reaches the center of a node. Additionally the nodes include attributes which specify the points of time an AGV enters and leaves a node and the period of time which the AGV waits at the center of a node. These additional attributes are needed during the conflict resolution to model waiting, acceleration and deceleration of an AGV.

For the calculation of the fastest conflict-free way through the plant, a 4-stage algorithm, which is illustrated in Figure 3.2, is used.

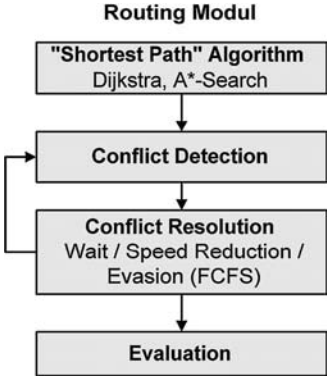

Fig. 3.2. Routing algorithm.

In the first step a Shortest Path-Algorithm (Dijkstra or A*) calculates one or more shortest paths between two given nodes of the plant graph [5].

The allowed AGV movement can be set to "only orthogonal" or "orthogonal and diagonal". During this step, it is assumed that the AGVs moves at maximum speed and no waiting is necessary. The algorithm generates node lists where the waiting time is set to zero and the node entry and exit times are calculated depending on the respective size of the nodes and the maximum speed of an AGV. In the case where an AGV has to pick up a vessel first and then has to move to a station, two paths are generated.

The second step searches for collisions with already scheduled trajectories. This is done by comparing the time intervals of the nodes of the current path with scheduled paths which are active at the same time.

If a time slot of the current path overlaps with a time slot of an existing path, the node positions are analyzed to identify the collision type. The detection procedure distinguishes between three possible collision types (head-to-tail, head-to-head and side collision). This information is required for an efficient conflict resolution in the next step.

If no conflict free shortest path can be found, the conflict resolution tries to solve the conflict by waiting, speed reduction or evasion of the currently considered AGV. The applied priority rule is a "First-Come-First-Serve" heuristic. By using a FCFS-priority rule, paths that were once scheduled stay conflict free in the future [6, 7]. If the detected conflict can be solved, the resulting path is again analyzed for conflicts because new conflicts can emerge from the resolution of an earlier conflict.

3.2.4.3 **The Simulation Module**

The simulation module simulates the basic operation(s) which are processed by a combination of a vessel and a station using a discrete event simulator. All necessary data (basic operation(s), equipment parameters, recipe scaling percentage, etc.) is provided by the scheduling-module. The simulator calculates the processing times and the state changes of the contents of the vessels (mass, temperature, concentrations, etc.) that are relevant for logistic considerations.

The simulation model is generated from the parameters of the basic operations and the parameters of the vessels and the stations (heating power, cooling power, mass flows, etc.) as a set of analytically solvable differential equations which are included in the simulation module. In case of a scaled recipe all weight depending transitions are scaled to the desired percentage. Time transitions, for example the duration of a mixing step, are assumed to stay constant. From the structure of the steps of the recipe (sequential, parallel or interleaving) the active basic operations and the transition conditions are determined. The execution of the recipe is simulated to determine the durations of the active steps. The transition conditions are used to calculate the point of time when the first active transition fires. Subsequently all active state variables of the basic operations are evaluated and the next set of active basic operations is determined [8, 9]. State information is stored in the data model and the overall duration of the executed recipe step(s) is returned to the scheduling-module.

3.3 Industrial Case Study

In co-operation with a German chemical company, a recipe-driven batch process was modelled and simulated using *PPSiM*. In the simulation study, different pipeless plant scenarios were tested and evaluated. The plant under consideration produces a set of consumer care products.

The motivation for the investigation of a pipeless plant as an alternative production concept was the increasing product diversification and the decrease of the individual production volumes. It is expected that the capacity and the flexibility of the existing standard multipurpose plant that consists of several batch mixers will be not sufficient to meet the market demands and to stay economically competitive in the future.

The objective of the simulation study therefore was a comparison of the profitability and the flexibility between an existing standard multipurpose plant and different conceivable pipeless plant scenarios. Based on the production data of the existing plant, an optimal pipeless plant setup was developed and representative production plans were simulated and evaluated.

The degrees of freedom during the plant design were:

- the numbers of stations and vessels;
- the allocation of the technical functions to the stations and vessels;
- the positioning of the stations and the AGV parking zones on the plant area.

As plant surface area the layout of the existing production building was used. In addition, three existing bottling plants were incorporated in the simulation model.

3.3.1 Process Description

The considered batch process produces approximately 750 different products which can be divided into 9 product groups. Within the 9 product groups, the recipes only differ by the additives used. Figure 3.3 shows the average fractions of the raw materials for the 9 product groups.

The underlying chemical process consists mainly of the unit operations *charging*, *mixing*, *heating*, *cooling* and *emulsifying*. All unit operations are processed at atmospheric pressure. The production recipes define 12 steps on the average and mostly have linear structures. Differences between the recipes arise from the sequence of the steps, the types and the quantities of the raw materials used, and the durations of the heating, cooling and emulsifying steps. After the production process, all products are stored for 24 h in separate storage vessels and afterwards the product quality is determined by a laboratory analysis. Finally the products are discharged at the three bottling stations and the vessels are cleaned. The reactors are cleaned after each production step.

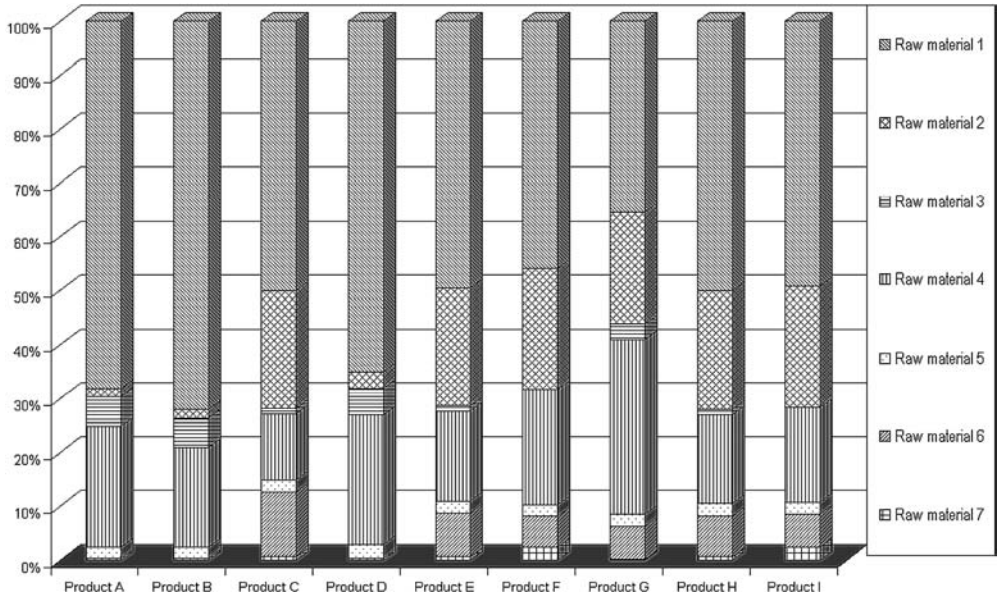

Fig. 3.3 Distribution of the raw materials within the nine product groups.

Typical batch sizes are 200, 500 and 1000 kg, where within this study the largest batch size was not considered, since these products are produced in a regular fashion. The production is run in a two-shift operation, where two workers are responsible for the smaller batch sizes (200 and 500 kg). The bottling plants operate in continuous three-shift operation. On the weekend the production is shut down. Up to now, the scheduling of the production process is done manually with a planning horizon of one week.

3.3.2 Modeling Using *PPSiM*

The modeling of the recipes, the plant(s) and the production plans was done graphically using *PPSiM*. The individual steps of the modeling process are summarized in the following sections.

3.3.2.1 Basic Recipes

The chemical process reflected by the basic recipes was modelled using the *PPSiM*-recipe editor. Each of the nine product groups was represented by one recipe for a nominal size of 100 kg. The outputs of the recipes were adjusted to 200 and 500 kg during production planning by the scaling factors of the control recipes.

The structure of the recipes and the parameters of the unit operations (heating power, cooling power, mass flows, etc.) were taken from the process description and had to be slightly adapted during the first simulation runs.

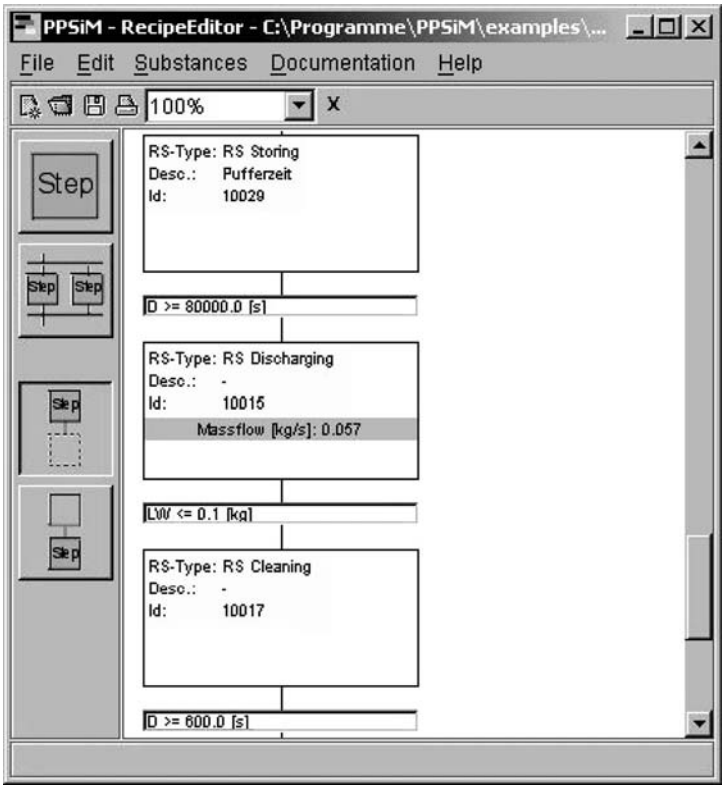

Fig. 3.4 Sequence of the steps *storage*, *discharging* and *cleaning*.

All recipes are terminated by the three process steps *intermediate storage*, *discharging* and *vessel cleaning*. The duration of all intermediate storage procedures was specified as 24 h, the duration of the cleaning of the vessels at the cleaning stations was estimated as 10 min. Figure 3.4 shows a screenshot of the recipe editor with the sequence of steps described above.

3.3.2.2 **Plant Model**

The modeling of the plant took into account the sketch of the existing production facility. The surface area has a size of 18 m × 12 m and the grid size was set to 1 m. Figure 3.5 shows one of the plant setups that was evaluated in the simulation runs.

In each plant setup, the production area was placed in the top left corner. The production area consists of stations equipped with the technical functions *charging*, *heating*, *cooling*, *mixing* and *emulsifying*. The CIP times of the stations of the production area was estimated as 10 min.

An intermediate storage area, realized by 18 parking slots, was added in the lower part of the plant to offer space for empty (clean) and product-loaded vessels. In addition these parking slots serve as starting positions of the vessels.

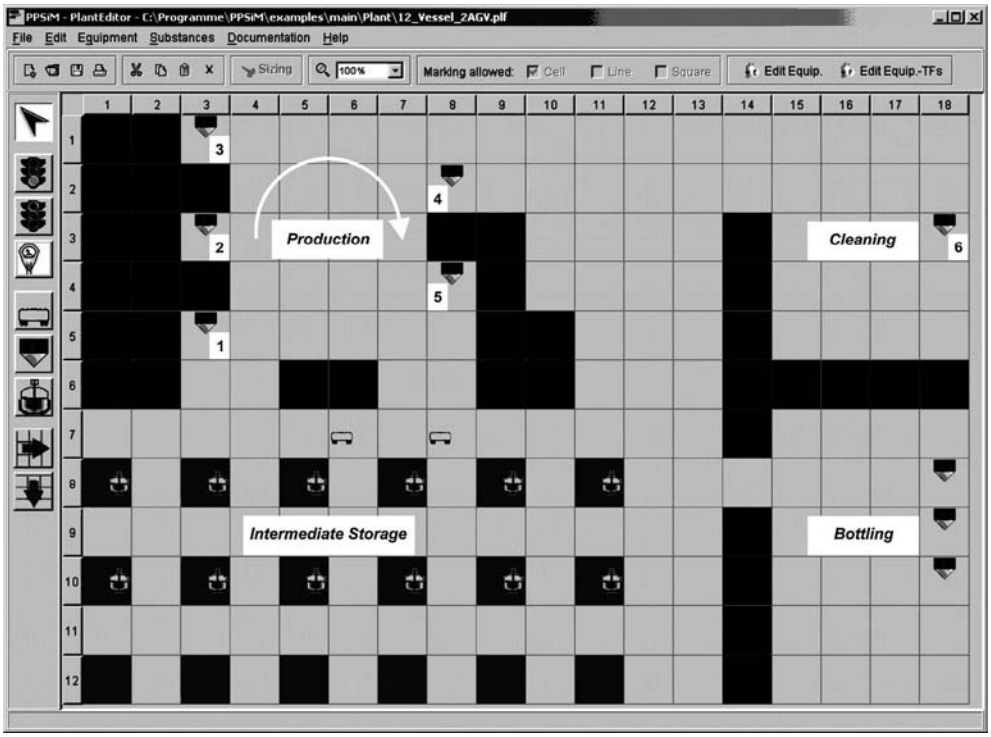

Fig. 3.5 Screenshot of the plant editor – basic configuration.

The stations for the cleaning of the vessels after discharging the product at the bottling stations (bottom right) were placed in the top right corner of the plant.

The number of stations and of AGVs and the distribution of the technical function among the stations were changed iteratively based upon the results of the simulation runs.

3.3.2.3 **Production Plans**

The production scenarios considered here were derived from a representative daily production and uniformly contained 12 batches in order to be able to compare the different plant setups. Each production plan demanded a production of 6 batches of 200 kg and 6 batches of 500 kg.

The start time of all batches was set to 7:00 AM. For all scenarios evaluated, the sequencing and the resource allocation were done automatically by the scheduling algorithm.

3.3.2.4 **Simulation Results**

The simulations evaluated different pipeless plant setups starting from a *basic configuration*. The scenarios differed by the numbers of stations and of AGVs. As the evaluation criterion the overall production time of the production plans was used.

Table 3.1 Assignment of technical functions to the stations for the basic configuration

Station	Charging	Heating/cooling, homogenization
1	Raw material 1	
2	Raw material 3	Heating
3	Raw material 2, Raw material 5	Cooling
4	Raw material 4, Raw material 6	
5	Raw material 7	Cooling, Homogenization
6	Cleaning	

After a simulation-based design of an efficient pipeless plant had been performed, the existing standard multipurpose plant was modelled by a reference model and compared to the pipeless plant setup by determining the overall production time for different production plans.

Basic Configuration Due to the batch sizes of the production plan, six vessels with a capacity of 200 kg and 6 vessels with a capacity of 500 kg were used. The technical functions were assigned to the vessels such that every product can be produced in every vessel.

An initial number of stations was determined by an analysis of the recipes. Within the recipes, sub-sequences of unit operations were identified which must be processed without waiting time or in parallel. The remaining unit operations were distributed on existing or new stations, so that the utilization of the stations was approximately evenly distributed and subsequent unit operations could be processed at one station. By this allocation the number of vessel transfers was minimized. An overview on the allocation of technical functions to the stations in the basic configuration is listed in Table 3.1. The numbers of the stations correspond to the labelling of the stations in Figure 3.5.

The stations were arranged in a clockwise order on the plant area in order to minimize the probability of collisions or interlockings during the execution of the recipes. For example station 1 (charging raw material 1), which is used for the processing of the first step of all recipes, was placed at the bottom left in the production area and station 4 (cooling and homogenization) was placed at the bottom right.

In the basic configuration one cleaning station (station 6) and two AGVs were used. The speed of the AGVs was set to 0.025 m s^{-1}.

Alternative Plant Configurations An overview on the different plant setups that were considered in the simulation study is listed in Table 3.2. The table contains a short description of the configuration and the corresponding values of the production times and the processing times. The processing time is the duration of the processing of all 12 batches (makespan) without the time needed for intermediate storage, discharging and cleaning. This time was used to decide about the feasibility

Table 3.2 Overview of the evaluated plant setups

#	Configuration	Production time [hh:mm]	Processing time [hh:mm]
1	Basic configuration: six stations, two AGVs	47:45	20:45
2	Doubling of station 1	48:08	20:55
2a	Doubling of station 1 and 3 AGVs	47:17	20:06
3	Doubling of station 2	42:43	15:19*
4	Doubling of station 3	46:41	19:27
5	Doubling of station 4	47:56	20:42
6	Doubling of station 5	47:18	20:07
7	Doubling of station 6	47:58	20:45
8	Doubling of stations 2 and 3 (Basic configuration 2)	41:50	14:26*
9	Doubling of stations 2 and 5	42:09	14:43*
10	Doubling of stations 2, 3, 5	41:23	13:26*

*Feasible production plan for a two-shift operation with 16 h available production time.

of the production plan because only for a processing time smaller than 16 h the production plan can be executed by a two-shift operation. Feasible production plans are marked by a*. The production time includes the processing time, the intermediate storage, the discharging of the products and the cleaning of the vessels.

Table 3.3 shows in the upper half the relative degrees of allocation and of utilization of the stations of the basic configuration.

Table 3.3 Degrees of allocation and of utilization of the stations

Station	Allocation [%]	Utilization [%]	Delta [%]
	Basic configuration		
2	37.08	33.18	3.90
5	36.62	21.95	14.67
3	36.41	21.31	15.10
1	30.16	14.70	15.46
6	35.46	8.34	27.12
4	34.88	8.10	26.78
	Basic configuration 2		
5	26.57	24.22	2.35
1	21.36	16.85	4.51
2.2	22.10	17.26	4.84
2.1	25.70	20.78	4.92
3.1	26.52	16.73	9.79
3.2	20.88	9.05	11.83
4	24.73	9.29	15.44
6	27.34	9.56	17.78

The relative degrees of allocation and of utilization are defined as follows:

$$\text{relative degree of allocation } [\%] = \frac{\text{total time of equipment allocation}}{\text{total time of production plan}}$$

$$\text{relative degree of utilization } [\%] = \frac{\text{total time of equipment utilization}}{\text{total time of production plan}}$$

Allocation defines the time span between the first and the last usage of a station including unproductive waiting times. Utilization sums the times when a station is used for processing. The small difference (delta) between the degrees of allocation and utilization of station 2 shows that this station is working almost to full capacity. In contrast, the other stations have a significantly lower utilization because the vessels have to wait for station 2, which means that station 2 turns out to be a bottleneck.

Starting from the basic configuration (#1) each station was doubled (configurations 2–7) in order to compute the influence of this addition of processing capacity on the production time. Table 3.2 shows that only the doubling of the stations 2, 3 and 5 reduces the production time significantly. A doubling of station 4 or of the cleaning station (station 6) has no influence. In configuration 2, the doubling of station 1 even leads to an increase of the processing time, since now 2 vessels can always be charged in parallel with raw material 1 but then they have to wait before station 2. As a consequence, the two AGVs have to drive several times across the plant in order to accomplish the transportation between station 1 and station 2. Therefore other transportation jobs have to wait for available AGVs. Scenario 2a confirms this statement. In addition to the doubling of station 1, a third AGV was added. In this scenario the production time is only slightly reduced compared to the basic configuration since station 2 still acts the major bottleneck. Only configuration 3 (doubling of station 2) leads to a feasible production plan. For all other configurations, the 16 h time-window is exceeded.

For a further reduction of the production time, the most promising candidate stations were doubled in parallel (configurations 8–10). This doubling of stations did not reduce the production time as efficiently as the doubling of station 2. Since configuration 10 where three stations were doubled only slightly reduced the production time in comparison to scenario 8 and 9, this configuration was rejected for cost reasons. Configuration 8 (doubling of stations 2 and 3), which is called *basic configuration 2* in the sequel, could be identified as the best plant setup and was analyzed further.

Table 3.3 (bottom) shows the utilization of the stations of the basic configuration 2. In comparison to the basic configuration the production time can be reduced by 13 % and the processing time can be reduced by 30 %. The utilization of all other stations increased significantly.

Further simulation runs analyzed the influence of the number of AGVs on the production times. When using only one AGV, the production time significantly

increased as expected. The use of a third AGV only slightly decreased the production times, which does not justify the installation of an additional AGV.

Comparison of the Plant Concepts To be able to compare the pipeless plant concept with the existing multipurpose batch plant, a reference plant was modelled using *PPSiM*. In the existing plant three conventional batch mixers work in a shifted parallel fashion. The three batch mixers were modelled by three stations and equipped with all technical functions necessary for the production of all recipes. Therefore each batch could be processed at one of the stations and the vessel transfers were limited to the transportation of empty or loaded vessels. All the other parameters of the model, e.g., charging mass flows, the durations of vessel cleanings and the recipes remained unchanged.

In case of the standard multipurpose batch plant the downtime of the reactors for the product transfer into temporary storage vessels, the transportation of the vessels to the storage and the cleaning of the reactors before product changeover have to be included. This was modelled by an adjustment of the CIP times of the three stations to 60 min.

In order to be able to compare the production times of the reference plant with the pipeless plant (in basic configuration 2), the following production plans were simulated:

- 12 batches of 500 kg each;
- 6 batches of 200 kg and 6 batches of 500 kg each (original production plan);
- 12 batches of 200 kg each;
- 12 batches of 100 kg each.

Table 3.4 lists the production times and the processing times of the evaluated scenarios. Figure 3.6 compares the processing times of both plant concepts. Scenario G2 in Table 3.4 is identical to configuration 8 in Table 3.2.

Table 3.4 Comparison of a standard multiproduct plant and a pipeless plant

#	Scenario	Production time [hh:mm]	Processing time [hh:mm]
G1	Basic configuration 2, 12 × 500 kg	43:41	20:10
G2	Basic configuration 2, 6 × 200 kg, 6 × 500 kg	41:50	14:26
G3	Basic configuration 2, 12 × 200 kg	37:20	11:38
G4	Basic configuration 2, 12 × 100 kg	36:22	11:03
R1	Reference plant, 12 × 500 kg	46:33	20:10
R2	Reference plant, 6 × 200 kg, 6 × 500 kg	43:27	17:06
R3	Reference plant, 12 × 200 kg	37:26	12:34
R4	Reference plant, 12 × 100 kg	34:27	10:02

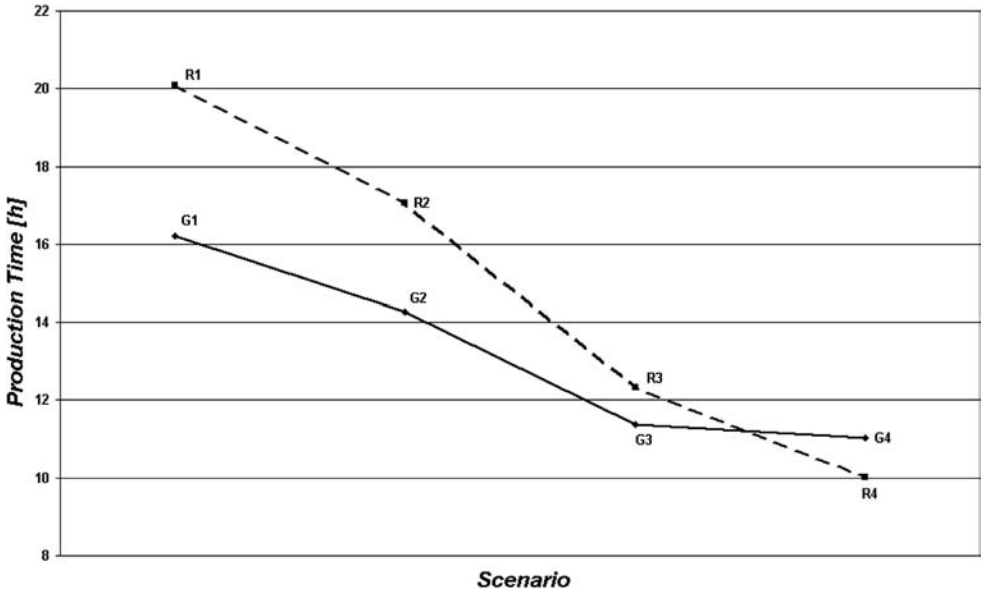

Fig. 3.6 Comparison of the processing times of both plant concepts.

The comparison of the two plant concepts shows that the pipeless plant concept leads to smaller production and processing times at batch sizes of 200 and 500 kg. This advantage decreases towards smaller batch sizes.

At batch sizes of 100 kg the standard multipurpose plant becomes superior to the pipeless plant with respect to the processing time. The reason for this is that the CIP times of the stations and the transfer times of the vessels remain constant in both cases and therefore the portion of the duration of the 'productive' operations becomes smaller in comparison to the processing time. The advantage of the production of several batches in parallel is therefore reduced by the increasing portion of the cleaning times of the stations.

In the basic configuration 2, a maximum of seven batches can be produced in parallel (see Gantt Chart in Figure 3.7). During the recipe execution each batch uses on the average 5 stations. In this case the 'unproductive' time sums up to 70 min (5 × 10 min for station cleaning and transportation). In case of the standard multipurpose batch plant, which produces a maximum of three batches in parallel, the cleaning requires only 60 min (see Gantt Chart in Figure 3.8). This comparison illustrates that the CIP times of the stations and the speed of the AGVs have a significant influence on the processing times of the batches and thus on the economic efficiency of a pipeless plant.

Assessment of the Plant Concepts The economic comparison of the pipeless plant developed in this study with the existing multipurpose batch plant resulted only

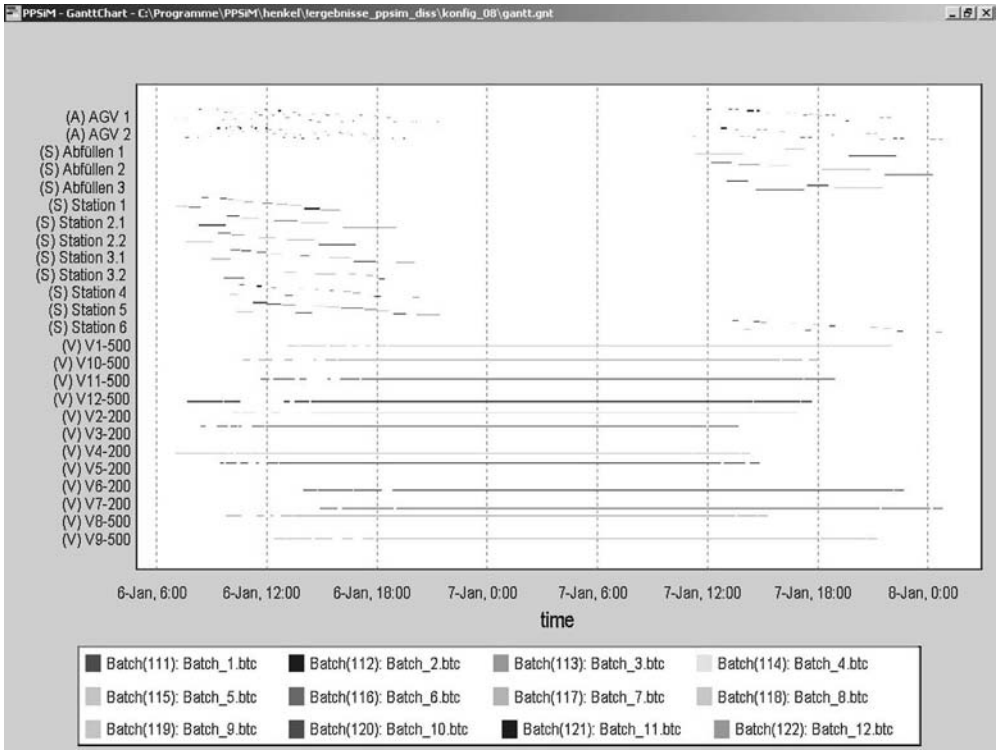

Fig. 3.7 *PPSiM* Gantt Chart of the basic configuration 2 (scenario G2).

in small differences regarding purchase costs. The detailed calculation was performed by the industrial partner. In the case study considered here, the higher utilization of the stations leads to a smaller number of technical functions which have to be installed. On the other hand this benefit is compensated by the necessary installation of special and expensive equipment, in particular the couplings and the transportation system.

The pipeless plant concept leads to 20 % shorter processing times for batch sizes between 200 and 500 kg, what clearly reduces the manufacturing costs of the products. By the distribution of the technical functions on several stations, the transport of the intermediate products in mobile vessels and the cleaning of the vessels in separate cleaning stations, the utilization of the stations rises and in parallel the productivity of the plant is increased.

Further advantages result from the immanent flexibility of the pipeless plant concept. By the possibility of storage of intermediate products during the production, urgent orders with a higher priority can be preferred or inserted into the production plan.

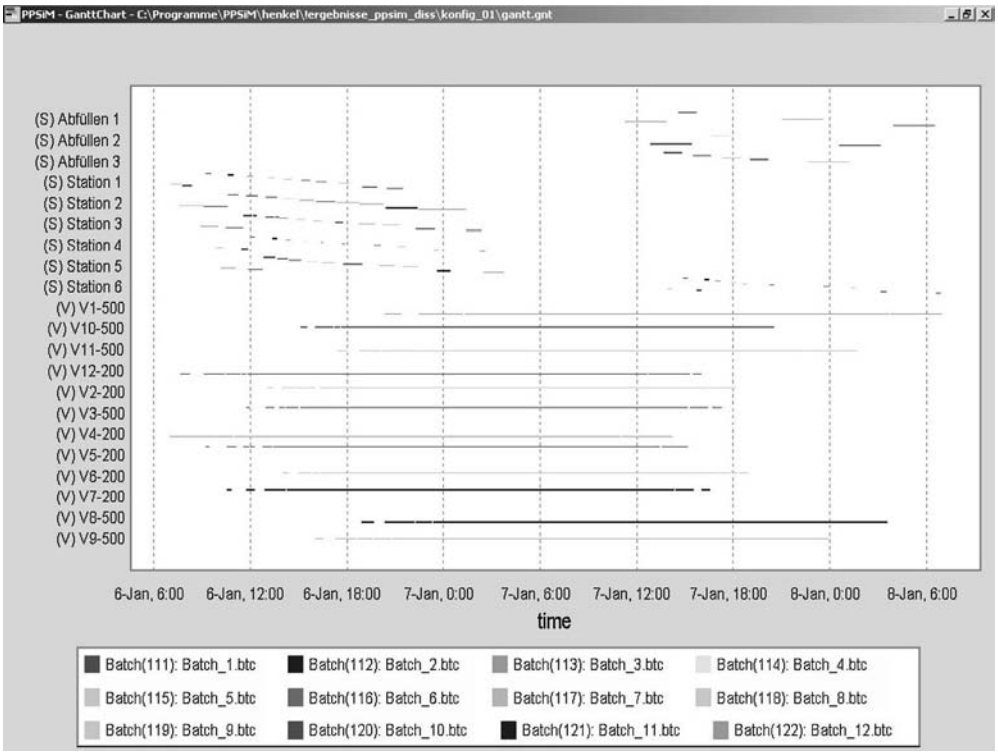

Fig. 3.8 *PPSiM* Gantt Chart of the reference plant (scenario R2).

3.4 Conclusions

The graphical modeling and simulation environment presented here enables a detailed and realistic representation of pipeless plants and can be used to support their design as well as their operation. A scheduling algorithm has been implemented that automatically assigns resources on the basis of the current equipment allocation and equipment properties. The routing of the mobile components is also done automatically taking into account the space and the traffic inside the plant. The simulation of the processing steps is carried out using a discrete event simulator.

For the investigation of logistic questions or for the design of pipeless plants, like in the presented use-case, *PPSiM* offers a suitable solution. The comparison between the two plant concepts showed that the developed pipeless plant configuration leads to 20 % shorter processing times for similar investments, due to the higher utilization of the plant equipment.

References

1 Hiransoog, C., Parkin, R.M. and Chung, P.W. (1997) Towards the pipeless process plant. Proceedings of the Workshop on Recent Advances in Mobile Robots, Leicester, UK, pp. 82–87.

2 Rauch, J. (ed.) (1998) *Mehrproduktanlagen*, Wiley-VCH Verlag GmbH, Weinheim.

3 Schiffelers, R.R.H., van Beek, D.A., Meuldijk, J. and Rooda, J.E. (2002) Hybrid modeling and simulation of pipeless batch plants, in *Computer-Aided Chemical Engineering* (eds J. Grievink and J. van Schijndel), vol. 10.

4 Liefeldt, A. and Engell, S. (2004) Modellierung, Simulation und Produktionsplanung vonrohrlosen Anlagen. *at-Automatisierungstechnik*, **52**, 342–350.

5 Qiu, L. and Hsu, W. (1999) Scheduling and routing algorithms for AGVs: a Survey. Technical Report: CAIS-TR-99-26, Center for Advanced Information Systems, Nanyang Technological University, China.

6 Liefeldt, A. and Engell, S. (2003) A modeling and simulation environment for pipeless plants. Proceedings of 2003 International Symposium on Process Systems Engineering (PSE 2003), Kunming, China, pp. 956–961.

7 Liu, C.I., Jula, H. and Ioannou, P.A. (2001) Design and simulation of automated container terminal using AGVs. Proceedings of 2001 European Control Conference, Porto, Portugal, pp. 295–300.

8 Fritz, M. (1999) Eine Softwarearchitektur zur Modellierung und Simulation vonrezepturgesteuerten Mehrproduktanlagen. Dissertation, Fachbereich Chemietechnik, Universität Dortmund and Shaker, Aachen.

9 Fritz, M., Liefeldt, A. and Engell, S. (1999) Recipe-driven batch processes: Event handling in hybrid system simulation. Proceedings of 1999 IEEE International Symposium on Computer Aided Control System Design (CACSD '99), Hawaii, USA, pp. 138–143.

Part III
Industrial Solutions

4
Planning Large Supply Chain Scenarios with "Quant-based Combinatorial Optimization"

Christoph Plapp, Dirk Surholt, and Dietmar Syring

4.1
Introduction

An increasing integration along the value chain, vertically and horizontally, cost as well as time pressures and high quality standards are just a few of the reasons why planning problems in practice increase and need to be considered on a high level of detail. However, despite rapid progresses in the field of solution algorithms, most of the existing approaches still cannot solve the complexity of the prevailing planning scenarios.

This is especially true for the pharmaceutical and the chemical industry where a great range of details have to be taken into account. To here successfully plan and optimize large supply chain scenarios it needs a specifically sophisticated combination of algorithms and heuristics [15, 16, 29, 30].

This chapter focuses on a new approach that allows for the comprehensive planning and optimization of multi-stage production processes – the quant-based combinatorial optimization. First, a distinction is drawn between classical approaches such as Linear Programming (LP) and the quant-based combinatorial approach. Before going into the special characteristics and requirements of the process industry the one model approach with quant-based combinatorial optimization is introduced. Then we will give two examples of how this new approach is applied to real life problems.

4.2
The Limits of Traditional LP

The optimization of value-added processes is a subject that scientists all over the world have been dealing with for more than 70 years. The first basic algorithms for so-called Linear Programming (LP) were developed at American and European universities already in the 1930s, for the first time allowing the planning and simulation of simple business processes. LP soon became the base of the first software systems and even today almost all Supply Chain Management (SCM) or

alternative planning tools are based on further developments of LP. Though in the United States LP is very successful up to the present day, in Europe it was not until the late 1980s that the companies' requirements could not be fully addressed anymore by the means of LP solvers. Too many constraints had to be met, requiring the modeling of so-called mixed integer linear programming (MILP) solutions which could solve much more complex planning scenarios. Integer decisions, high variant numbers, basic set-up and cleaning times, fully-fledged production and inventory layouts, environmental regulations and work time models, to name just a few, are all examples of such constraints [46].

Most commercial standard software systems in the field of SCM cannot address the complexity that lies behind such multi-stage and entangled planning processes as they are based on three sequential planning steps: First, dependent requirements are derived from primary requirements by BOM explosion (Bill Of Materials/requirement explosion). Then the calculated quantities are allocated to the resources in the context of capacity planning. In a third step the concrete sequence planning of the processes is done. Since the steps are carried out sequentially in order to generate a plan, the solution area is limited, meaning that also many potentially better solutions can be cut off [20, 21, 33]. As a result, a capacity plan may be created to which no valid sequence plan exists, requiring a complete new plan generation. Such an approach consequently only enables a better match by introducing a problem-oriented combination of exact methods and heuristics. These, however, have to be developed and implemented individually, not leaving enough flexibility needed in most planning processes [1, 2, 5, 13].

An example taken from the process industry may clarify the complexity of today's planning problems and the limits of classical sequence-based, one-dimensional approaches. When, for example, looking at the lacquer production of the automotive industry the different lacquers are manufactured in batch production, that is, in several sequential production steps on different machines. The different production steps include pre-dispersion, dispersion, mixing and filling, with the time for mixing being product dependent. In such a production process a number of constraints come into play, most of which are typical for the chemical industry. Mixers, for example, are blocked until filling is started, some machines are restricted to the production of certain lacquer types, sequence-dependent cleaning times have to be adhered to, shift models have to be coordinated with production times and certain process steps can only be interrupted for a certain amount of time.

To fully address such a planning scenario with all its constraints, an integrated standard model is needed which simultaneously reflects and matches *all* requirements at the same time.

A benchmark done in a European research project showed that in the context of the above described case study from the lacquer industry realistic model sizes could neither be solved by the timed automata model nor by the classic MILP approach. It affirmed though a new methodical approach which allows the modeling of diverse, relevant problem scenarios on a high level of detail in just one integrated model. This new approach is based on the so-called quant-based combinatorial

optimization. It is a model that allows the simultaneous consideration of all business relevant constraints.

4.3 Quant-based Combinatorial Optimization

The quant-based combinatorial optimization accounts for the entire value chain and its associated processes, that is for all of its inputs, outputs and system parameters. Based on these data an integrated and complete model of a company's supply chain is derived, detecting in advance what impact a decision or an external influence will have on the total system.

Initially – using the physical term "quant" which is used to describe the smallest energy package – the term "business quant" has been introduced. Let's give a definition of quant-based combinatorial optimization. First: what is a quant?

- A quant is a meaningful logistic disposition unit, which can be cost tracked in the value chain. If, for example, a batch problem exists, a quant corresponds exactly to the batch size. If continuous production exists, a quant corresponds to a meaningful small rounding unit and in the case of a filling process to the respective bundling units. Quants can be individually defined.
- A quant is a discrete quantity of any material (e.g. final product, raw material). This discrete quantity is called the quant size. The quant size may be chosen as needed.
- A quant has a unique identification. Any quant in a model can be identified by this identification. The identification is called the quant-ID.
- Any quant is linked with other quants. The quant gets its supplies from them or delivers to them. Along the link travels a certain quantity of material. Additionally a link describes the dependencies of the two quants, for example minimum or maximum offset time. Quants and links form the quant network. This link type is called a quant link. Quant links may be added or deleted at any time, depending on the process of satisfying the given constraints.
- Quants may be linked with anonymous objects, for example demands or stocks as well. This type of link is called an anonymous link. Anonymous links may be added or deleted at any time, depending on the process of satisfying the given constraints.
- A quant has an associated cost function. Depending on the "place" of the quant in the solution space the costs are calculated.
- A quant requires some capacity of some resources, thus resulting in a start and end of a quant. Very often a quant will require just one resource but an infinite number of resources is possible as well.
- Quants may have setup time and costs, depending on for example resources or preceding quants.
- Quants can be dynamically generated, split, combined or deleted. This process is called quant generation.

There are numerous definitions of combinatorial optimization. We will use this definition: "Combinatorial optimization means algorithms which generate quants and assign them to resources such that the costs summarized over all quants are minimized and all constraints are met."

As there is no simple algorithm which can generate a solution just in one step several algorithms have to be combined [7–10]. Those algorithms which may improve the solution are called operators. The overall combinatorial solution algorithm consists of many operators. Any operator may work in any combination on the solution. Some examples are:

- Sequence optimization: create a good sequence of quants on a resource.
- Partial enumeration: perform a full enumeration of parts of the solution.
- Local search: make local changes to the solution and see whether there is an improvement of costs. Local changes might be: changes of resource assignments, split or combine quants, add or remove links.
- Rest cost estimation: give a hint what would be the ideal costs for a part of the solution. Rest cost estimation [6] could be done for example by best search, LP or cost scaling algorithm (CSA) [3, 23, 25–27].

The idea of the new approach is to, first, define only one model that includes *all* constraints – be it shift models, personnel, batch sizes, maximum perishabilities or "soft" constraints such as costs and values, characteristic numbers or feasibility and optimum of the plan. The case studies will show that the model follows an "intuitive" representation of the relevant items. Second, the operator based approach makes the overall solution procedure extensible. If one operator gets stuck in a local optimum another operator may help out and could be added at any time. Third, the whole approach works on "understandable" objects such that at any time during the solution procedure an easy check can be made what happens.

The quant-based combinatorial optimization differs from classical approaches in a number of ways. First, it is the strength of the quant-based approach to account for all constraints that prevail in a real-life planning scenario. Even though they usually come into effect in complex, interrelated and varying ways.

Second, it includes a significant real-world constraint in its calculation – whole units. In contrast to traditional solutions based on LP/MILP the new approach calculates based on logical planning units such as full containers or full pallets. These so-called "quants" form the basis for the realistic modeling of planning problems (e.g. beer crates).

Third, the new approach allows the simultaneous planning and optimization of production processes. Where LP or MILP alone breaks the planning problem into disconnected models and solves each independently, the quant-based method works simultaneously, identifying an optimal distribution of capacity while taking into account optimal production sequences and respecting all key constraints in the system. Target objectives are flexible and can be delivery liability, lowest cost production and so forth.

Last, but not least, the new method optimizes plans in an intelligent, combinatorial way. As in reality any planning problem consists of many sub-problems, there will be no single solver algorithm that can be applied to all of them. In fact, to deduce the overall optimal plan it is necessary to solve each problem individually and find the best combination of all solutions. The combinatorial approach has the ability to intelligently apply and combine many optimization techniques and algorithms to derive the best possible solution. Among these are branch and bound, decision tree, genetic search [11, 28, 32, 42], LP, cost scaling [25–27] or approaches such as heuristics [31, 35, 36, 38], taboo-search strategies [14, 17–19] and so forth. By combining a wide range of algorithms ("operators") and techniques and applying them to the planning scenario in an iterative way the solution space is systematically narrowed down until the best possible plan is derived [10, 39].

4.4 Typical Planning Scenarios in the Process Industry

Companies who to date have decided in favor of the quant-based combinatorial optimization approach, are collectively characterized by a complex and multi-step supply chain whose scheduling requires the consideration of multiple constraints. By nature of their production processes, many of these companies come from the chemical or pharmaceutical industry.

To get a better idea of the complexity of a real application scenario in these industries it makes sense to, once, exemplarily depict the planning processes in a typical production of active pharmaceutical ingredients (API production). Most pharmaceutical companies are looking at planning scenarios in which several hundred individual resources or facilities have to be accounted for, with demands and orders for some thousand final products. The planning horizon is often set to 2–5 years. Next to single equipment, there are facility pools, with one pool consisting of several individual units.

Such a supply chain network easily adds up to tens of thousands of nodes and edges with which the product relations are described, whereby a node can represent raw material, an intermediary product or a final product. An edge represents the relationship between two products. As there are usually predecessor/successor relations, the relation network can be interpreted as a directed graph. The material flow is modelled in form of an edge, material factors and offset times are stored as attributes [3, 10, 23, 25, 33].

In a typical planning scenario of an API production the demands of the master plan usually break down into more than 100,000 quants. Each quant corresponds to the specific production of a production stage. Afterwards, the quants have to be assigned to the available resources within the planning process in such a way that demand dates can be met. At the same time, delay, production, inventory, transportation and change cost should be minimal.

4.5 Constraints

In addition to dimensions of size an API production model also has to map specific constraints in order to generate a feasible plan. Throughput times for example – from the first starting quantity to the final product – are typically very long, often taking up to one-and-a-half years. In addition, there is an extensive production depth, meaning that the production process of a final product sometimes has a chain of up to 40 preliminary steps that all need to be represented in the planning process. All of this necessitates an appropriately long planning horizon to cover and schedule the complete process chain. In addition, relatively detailed mapping is needed to ensure feasibility of realization [24, 34, 40].

Details of the planning model also should include co- and by-products. These may result from the production of the primary product and can later be used somewhere else in the production cycle. Even these cycles have to be reflected in the planning scenario.

Due to the long-range term of the plan any possible change in the production methods must be accounted for. As most of the equipment used in an API production is suitable for manufacturing different products, the same manufacturing processes may exist on different resources. It may also happen that the yield in a production stage is increased due to planned technical improvements. Or, as is well known, that there is a learning effect with new processes and that the yield improves with the course of time [41, 43, 45].

All these examples require that a variety of production methods are mapped with different validities in the model. The differentiation may refer to the use of different resources, dynamic material factors, modified offset times or various batch sizes.

Next to batch production, where exact quantities of a product have to be produced many times over, it is minimum lot sizes that deserve particular consideration. This is usually the case when products are manufactured in large campaigns. Here, products are usually manufactured in a continuous process, with different production processes being combined within the framework of the process chain. When linking the individual production stages to each other it is important to also consider offset times. These can for example include the transportation and analysis time between two production stages.

Another constraint that needs to be mentioned is minimum transfer quantities, meaning that a subsequent quant can only be started if a certain quantity of the previous material is available. When products are used in subsequent process steps it is necessary to also account for their durability time limits. If this is not the case, a new production has to be initiated. Last but not least, the re-procurement times for raw materials should not be neglected.

Besides all these quant-related constraints there are numerous others such as down times for production equipment when maintenance or rebuilding activities have to be carried out or fixed production orders. Finally, there are constraints such as varying shift models, and so forth.

The diversity of the stated constraints clearly shows that feasible solutions can only be achieved where constraints and their interdependency is mapped in their entirety in the production plan.

4.6 Additional Modeling Elements of the Quant-based Combinatorial Optimization

In production and logistics we find some typical objects: Products, processes, BOMs, work flows, resources, shift models, lockups, departments/business units, locations, demands (anonymous, orders). Most of these objects are discrete in nature, for example orders and batches are typical discrete objects. All these objects can be used to group and attach information. Objects which are not discrete may be approximated by discrete quants.

A *product* can be a real product that is producible on a set of resources. A description for a process is stored as a product, because processes often produce a product. As an abstraction a product also represents a production step/step in a work flow in the optimization model. Important data are, for example:

- stock costs,
- delay costs,
- production costs,
- value,
- minimum and maximum lot sizes,
- batch size,
- safety stocks,
- maximum stock level.

The *product flow* connects the products to the *product network*. Here BOMs (bills of material), work flows and additional information how to control the generation of quants (see Figure 4.1) and how to enumerate all necessary alternative BOMs are represented in this net. Important data are:

- predecessor and successor products,
- type of relation,
- the relation between the time frames (e.g., start-start, end-start),
- transport time,
- maximum offset between the linked times (maximum perishability or buffer),
- material factor.

Resources are used to model assets or on a more general level to model every unit on which time consuming or quantity producing operations can be performed. Quants are assigned to resources and have executing times and costs on resources.

A *shift model* and also *lockup times* can be assigned to a resource.

Departments are containers to collect products, resources, quants.

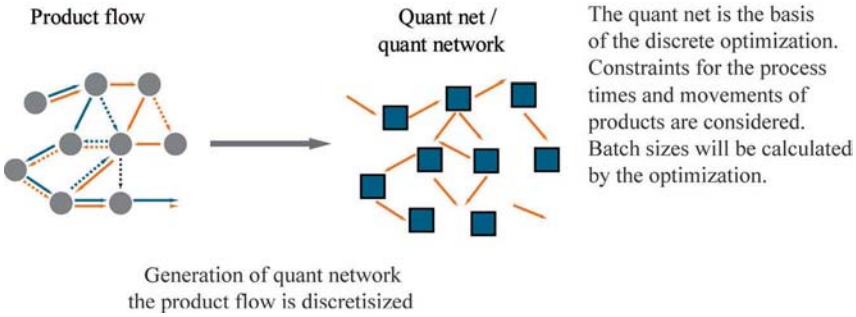

Fig. 4.1 The principle of the quant-based combinatorial optimization.

The *forecasts or anonymous demands* are represented as an ordered list of none intersecting time intervals. Any product may have forecasts.

Similar to the forecasts *orders* can be added.

4.7 The Solution Approach

The structure of the quant generation can be visualized with the help of a block flow diagram (see Figure 4.2). The combinatorial optimization here is integrated as one step within the whole process flow. In general, it is possible to change the sequence of optimization algorithms arbitrarily. A general combination of algorithms is to let them call each other recursively. This enables every algorithm to include the specialties of each other algorithm and thereby its functionality. This can be done using optimization scripts based on XML or other languages that are readable by existing parsers.

In many cases a start solution (as shown in Figure 4.3) already exists or is easy to obtain. The most common target is the due date where we distinguish between the

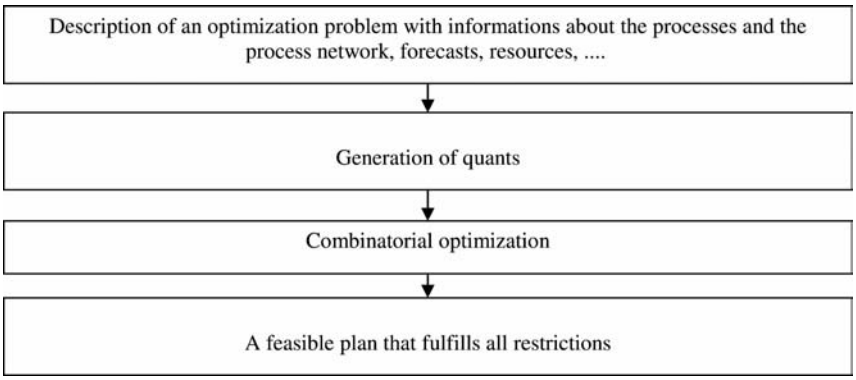

Fig. 4.2 An often used sequence for solving optimization algorithms.

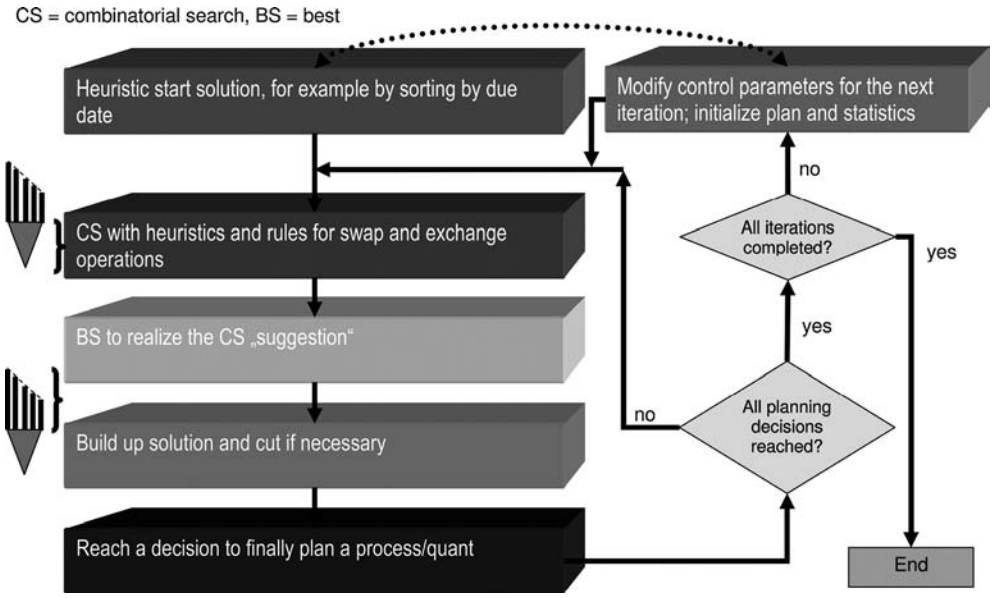

Fig. 4.3 Sketch of an often used branch and bound algorithm and corresponds to the box "combinatorial optimization" in Figure 4.2.

final due dates of the orders that are relevant to deliver the end products in time and the intermediate due dates that are calculated as latest production ends possible to be just in time for all successors. Breaking down the demands over all necessary productions, the calculation of the due dates and the assignment of productions greedily to the resources with intersections already gives an overview of less critical resources and possible bottlenecks.

Let's assume a start solution is available. Then optimization methods and operators are applied to this solution. Any kind of operator – even operators that destroy the solution and try to rebuild it and improve it – at the same time by iteration: From the appearance of an existing solution with respect to the sequences on the resources information, about how to sort the processes and which resources are optimal for a process to building an optimal solution are derived. Counters like the conflict matrix are used for this purpose. Operators that do not destroy the solution, but try to improve the solution step by step use branch and bound as shown in Figure 4.3. The dense lower part of the search tree enumerates interesting planning alternatives. The next step tries to build a best solution by choosing sequences and resources according to already existing solutions. When all planning decisions, i.e., all processes are finally assigned to a sequence and a resource, the control parameters can be changed, due dates can be recalculated, priorities can be set for certain critical orders or even quants. To do so, a critical path analysis can be applied.

With these changed and hopefully improved parameters the next iteration tries to find a better solution. If so, the best solution is replaced and finally given back to the user in the end.

4.8
Special Requirements and Advanced Modeling Features for the Chemical Industry

The two following examples are different with respect to the demands, constraints, objective functions and the whole master data and transaction data. They were constructed as reference examples out of real life requirements and real life systems.

4.8.1
Example 1: Lacquer Production

Number of resources: 14
Number of products: 22
Number of quants: 7000
Planning horizon: 5 years

The production consists of 5 steps.

Steps 1 and 2: Pre-dispersion and dispersion: solid and liquid basic materials as well as solvents are prepared for the actual production and mixed.

Step 3: After the pre-dispersion or dispersion the previously prepared materials are filled into mixing vessels with the help of dose spinners. Each dose spinner contains 200 valves, which emit predefined quantities of those basic materials which are required for the lacquer production into the vessels below. This is all done automatically. When the quantity required for a certain lacquer production has been obtained the mixing vessel is moved to a different place in the hall – and there it begins to mix the components during a defined period of time in order to obtain the end product. For production around five of these mixing vessels are available. Thereby you should keep in mind that the mixing vessels can be of different size and therefore of different filling quantity (volume capacity). Therefore the assignment of mixers is dependent on the order quantity. When the mixing procedure has finished the lacquer quality is checked.

Step 4: If the quality does not meet the requirements the mixing vessel is once again moved under the dose spinner and the dosing procedure is repeated. After this the gained color value is once again checked. The checking times can in some single cases last up to three weeks (in the model you can presume that a re-dosing is always needed).

Step 5: If the lacquer is of the desired quality then the mixing vessel is moved to a filling station, starts with docking and is then emptied. The mixing vessels are treated as occupied until the mixing procedure has finished and the lacquers are considered as having the desired quality.

The filling stations are also, as well as the dose spinners, critical resources, where bottlenecks can frequently occur.

The model contains three different end products with their specific production sequences. In our model for each quant that is part of the cycle an individual

product is created. In the following chapters the process flow for each of the three products is explained.

Sequence for Uni Lacquers: The production of uni lacquers starts with the predispersion and dispersion of the basic materials. Within four hours after the predispersion the dispersion has to take place. At the same time, when the dispersion starts, the usage of the mixing vessels starts. Six hours after the start of the dispersion the dose spinner will be allocated.

When the processes on the dose spinners have finished the first checking procedure takes place in the laboratory. This has to be done directly after the procedure on the dose spinner has finished or a maximum of four hours later. This maximum waiting time between the finishing of a process and the beginning of the successor in the material flow is modeled with a maximum offset that is a characteristic of the link between the two linked quants. After the first checking procedure a correcting procedure is now started. When this has been finished a second checking procedure has to be performed. When this checking procedure has once again finished the lacquers can be filled at the filling stations. There is no time constraint with these procedures. It can be performed immediately after the preceding procedures but also hours or days later. (You should however try to perform these procedures if possible without long waiting times as this can influence the length of use of the mixing vessels). If the lacquers have been fully filled then the assignment of the mixers is over. However, due to cleaning procedures the assignment of the mixing vessels has to be two to four hours longer as the filling of the lacquers would take time.

In Figures 4.4 and 4.5 you can find a graphic representation of the production flow and an example for the resource assignment when producing uni lacquers.

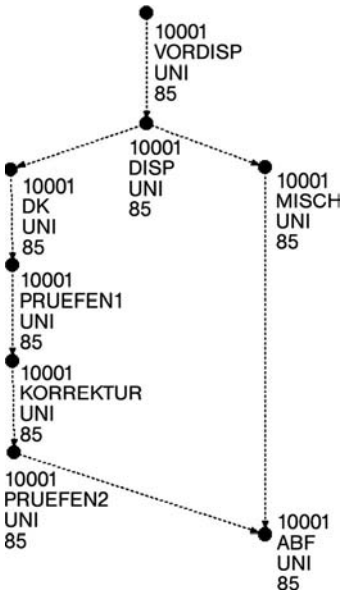

Fig. 4.4 Quant sequence for uni lacquers as bill of material.

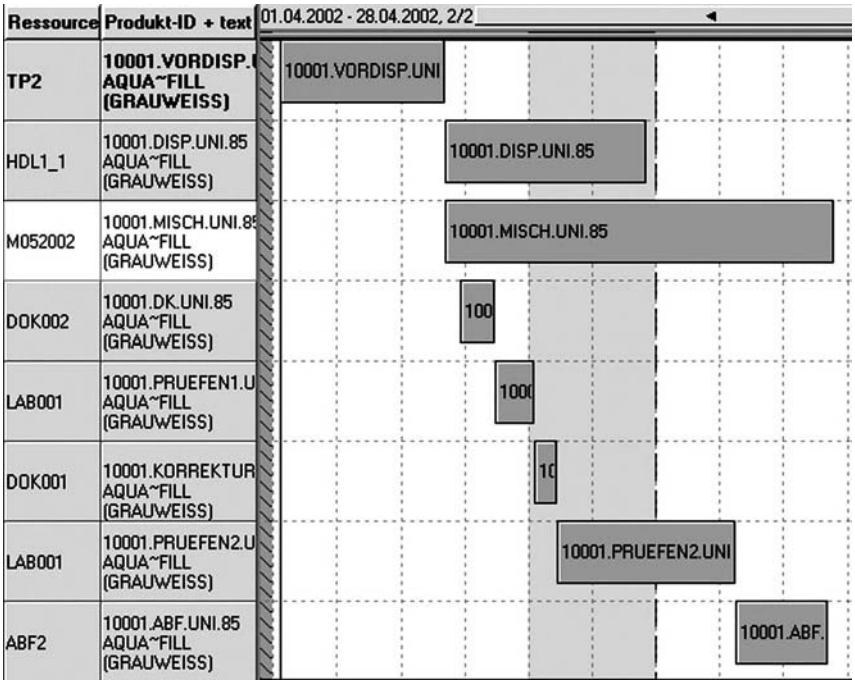

Fig. 4.5 Example for a resource assignment when producing uni lacquers.

Sequence for Metallic Lacquers The sequence when producing normal metallic lacquers differs from the sequence when producing uni lacquers to the extent that in this case no pre-dispersion or dispersion has to take place. The production sequence is the same as the production sequence when producing uni lacquers beginning with the dose spinners until the filling procedure (see Figures 4.6 and 4.7).

Sequence for *Special* Metallic Lacquers For some special metallic lacquers it is necessary to perform a pre-dispersion of the basic materials. For this process an additional dose distributor is required for the pre-dispersion resources. The assignment of the dose distributor starts at the same time as the assignment of the pre-dispersion resource. When the pre-dispersion process has finished the procedures on the dose spinner and on the mixer begin. The following production steps are equal to those of the standard production sequence (see Figures 4.8 and 4.9).

4.8.1.1 Products/Product Characteristics

For each end product or pre-product the density, the product value and the delay costs have to be defined. The density is necessary in order to convert the order quantity (kg) into order volume (l) (order quantity divided by the density of the corresponding product equals the order volume).

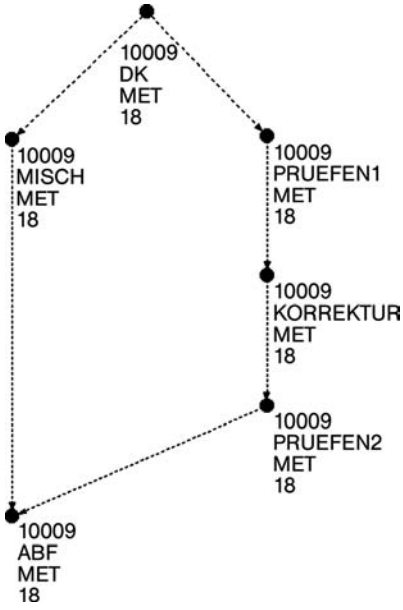

Fig. 4.6 Processes for metallic lacquers as bill of material.

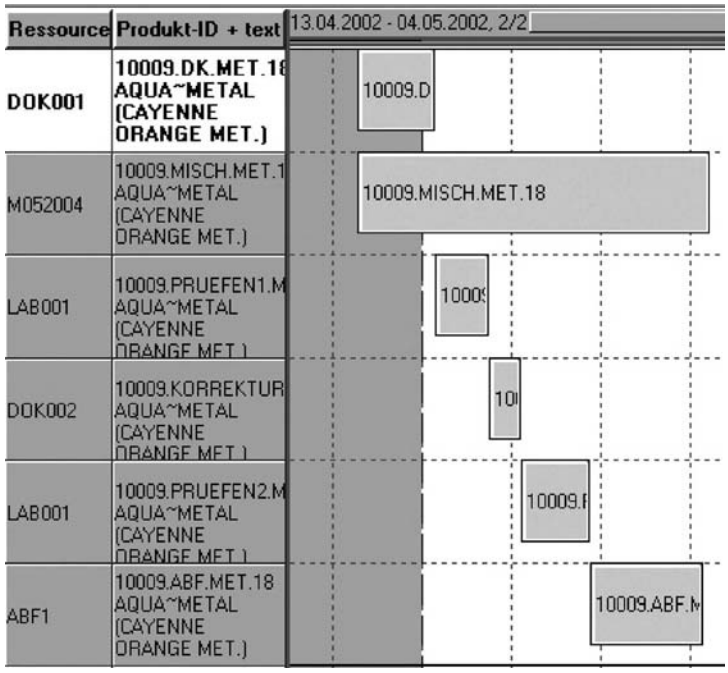

Fig. 4.7 Example for a resource assignment when producing metallic lacquers.

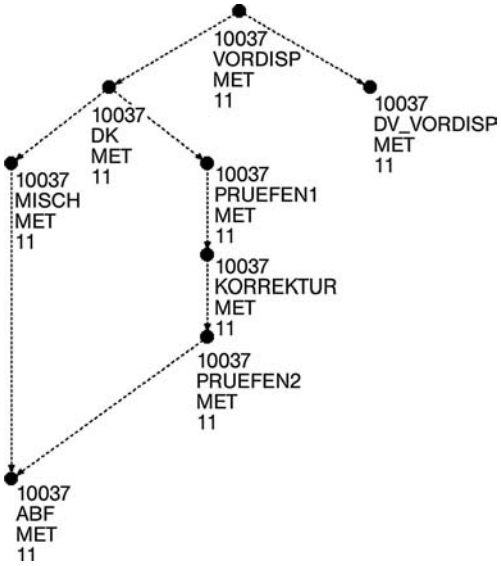

Fig. 4.8 Process sequence for special metallic lacquers as bill of material.

Delay costs arise in case the product cannot be produced to meet a certain deadline. In order to determine these the value specified for each product is multiplied with the amount of days which the product has been produced late (example: delay costs = 100 Euro, production ends four days after the due date → 400 Euro delay costs arise).

4.8.1.2 **Product Flow**

In addition to the material factor we can define the constraints concerning the material flow with the so-called product flow. These tags in the product flow will result in special link types for the links which will be created to connect the generated quants.

Link type 13, for example, states that the start of production of the succeeding product may only be set after the production end of the pre-product. The minimum offset time mirrors a minimum transport time so that in the earliest case possible the production of the successor can start at the time of the production end of the predecessor plus minimum offset. The maximum offset equals a maximum transport time so that the production of the succeeding product can start in the latest case possible at the time of the production end of the predecessor plus maximum offset. The offset times therefore limit the time window for the production start of the successor.

Link type 15, for example, determines the time window for the production start of the successor on the basis of the production *start* of the predecessor and not as with link type 13 on the basis of the production *end* of the predecessor.

Product relations with link type 98 state that the end of the successor determines the end of the predecessor. This is for example necessary for the calculation of

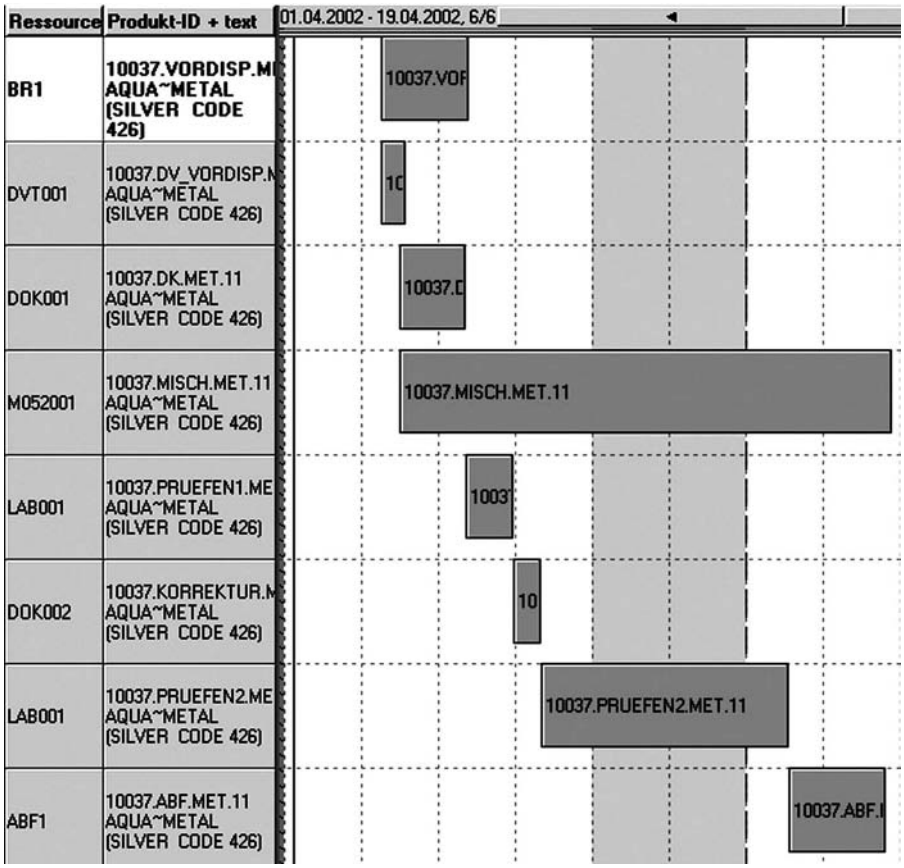

Fig. 4.9 Example of a resource assignment when producing special metallic lacquers.

the assignment time for the mixing vessels. For example the assignment of the mixing vessel is finished when the filling has taken place. However the filling is a succeeding production step if you consider the material flow logic. The assignment of the mixer can therefore in the earliest case possible be finished when the filling has stopped and the offset time has passed. In the latest case possible it has to be finished when the filling has stopped and the maximum offset time has passed.

4.8.1.3 **Resource Layout**

The minimum and the maximum filling quantity are stated in liters. Therefore only orders can be produced on the single resources which have an order quantity or order volume that is within these filling quantity limits.

A matrix will be used to determine which resource can be used for the production of a product. The cost statement is some kind of penalty term. (Example: the production of a product on a certain resource is more expensive than on another resource due to higher energy consumption.)

4.8.1.4 Order Data

Each of the orders is marked by an order quantity, an earliest start and a deadline. The earliest start states at what time an order can be produced in the earliest case possible (production start of the first quant). The deadline is the desired delivery date (production end of the latest quant) of the end product.

The intermediate products (process steps) which are necessary for fulfilling an order can be derived from the existing quant network.

The deadlines of the quants of an order differ as a backward termination is performed. This means that the quant for the filling process or the quant for the mixing process have the latest date as they equal the end in the production chain. The quants for the pre-dispersion or the dispersion have the earliest date as they equal the beginning in the production chain.

4.8.1.5 Objective Function

A first simple objective function is based on the following terms:

Delay costs per day + operational costs per kg as given in the matrix

A more complex objective function can be based on setup/ cleaning, storage, raw materials and transportation.

4.8.1.6 Introduction of a Maximum Perishability

Often products in the chemical production have a maximum perishability. This has two consequences. The process may not be interrupted longer than the maximum perishability. For example the weekend may cause a long break and as a consequence the quant must start the next week instead of the current week.

A more complex constraint to meet is the maximum perishability as the maximum time frame within the material produced by a process must be processed by the successor processes. This maximum perishability may even be dependent of the successor process and the predecessor process. Using different storage or tanks the perishability may change dependent on the chosen storage location.

4.8.1.7 Example of an Advanced Operator: Reducing Throughput Time

In this example it is obvious that the optimal use of the capacity of the mixers is critical. The mixers are longer in use if the total throughput time is increased and are less in use if the throughput time is decreased. As the mixers are expensive they are one of the bottlenecks.

There is a general operator available to reduce throughput times. This is done by adding artificial limits of perishability. Because of these limits some links in the quant network are not necessary any more (see Figure 4.10). So this operator works in three steps:

1. Add a virtual constraint "perishability".
2. Simplify network.
3. Re-schedule.

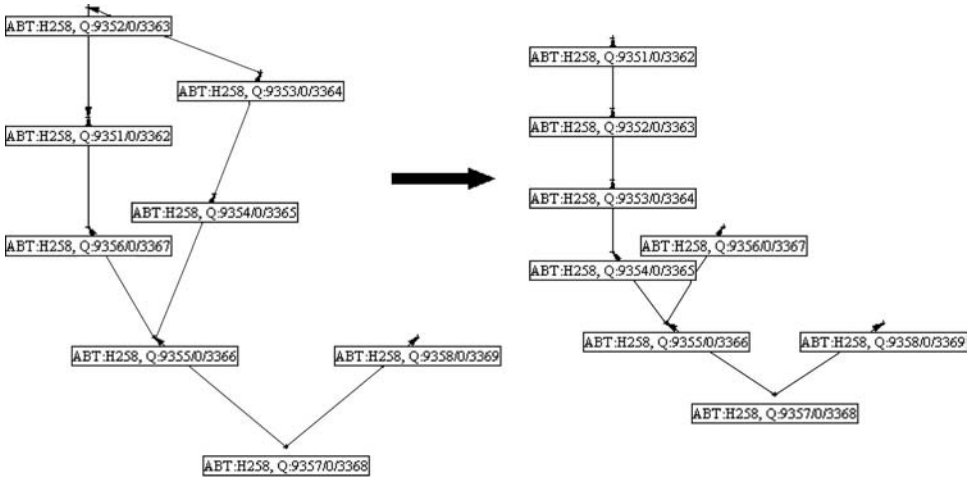

Fig. 4.10 Step 2: networks of processes like the *left* one are transformed into trees shown on the *right hand side*. Thereby the need of synchronization at the node H258, Q 9351/0/3362 is not necessary any more, because this synchronization is an inherent property of the process network now and the functions realizing this constraint are not search algorithms/optimization methods, but functions that realize the planning restrictions.

The positive effect on the planning results after the re-scheduling is obvious, see Figures 4.11 to 4.13.

The above example was used as a reference example in the European Research Project AMETIST [46]. A comparison with other solution approaches clearly showed the advantage of the quant-based combinatorial optimization.

The benchmark with other software tools with respect to the considerations of modeling visualized the following: Some operations cannot be interrupted (non-preemptive scheduling) and the material remains in the mixing vessels during some of the steps, as the quality check in the laboratory, or possible additional mixing. This leads to a situation where the operation times of the vessels are variable because they result from the scheduling of the operations for this batch. This modeling is possible in the Axxom standard, but leads to an increase of integer variables in MILP models and therefore to an immense increase of calculation time. Because of this results were only computed for at most 29 operations [8, 10, 16] within 412–2610 s and within 830–6420 s for an extended model with setup times. Axxom, in comparison, calculated feasible solutions for more than 1000 jobs within 6000 s (for 29 jobs 11 s).

The same problems occur using state task network (STN) formulations that use binary variables. There, possible events may occur for each instant expressed by a binary variable and the models become very large. No calculation was carried out for this formulation.

The problem of assigning the jobs (consisting of the aforementioned sequence of operations to obtain a lacquer) to the 14 resources belongs to the class of job

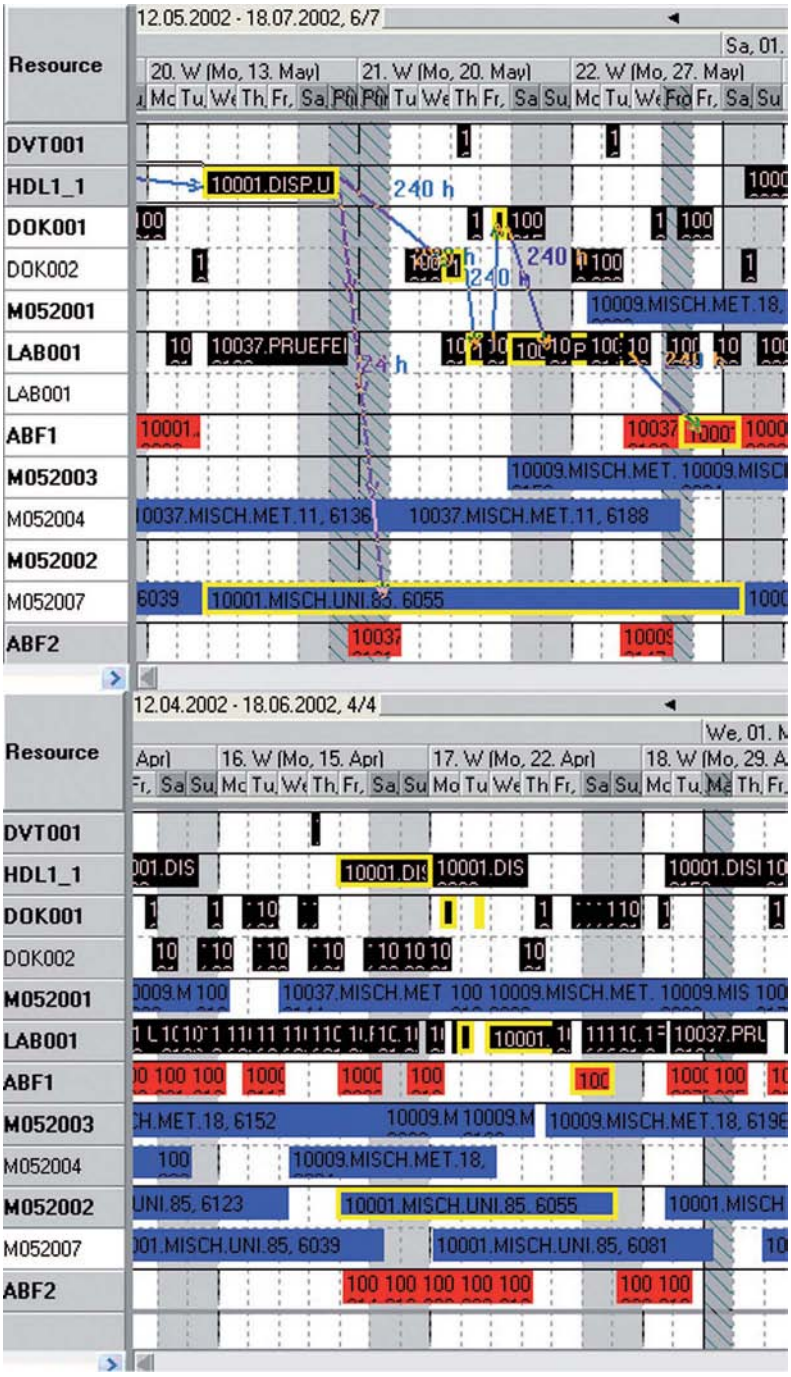

Fig. 4.11 (legend see p. 77)

shop problems, with the additional complication that mixing vessels are required as a second resource to perform the operations on some machines. With Axxom's software this is done in separated small algorithms which are embedded into the main solver algorithms, solving the resource assignment problem as a sub-problem. These algorithms use the knowledge of the planners or certain special properties as the reservation of resources. In generalized formulations this specialization can not be done because the language used in this formulations is too general, i.e. only constraints in form of variables and equations are allowed. So the main solver itself has to calculate everything and also the problems that are not the main focus of the end optimization goal.

Another property of the problem, which is not standard in job-shop problems, but often found in scheduling problems of the chemical industries, is the presence of constraints on the storage times. These constraints are expressed as bounds on the differences between the starting and the ending times of two operations of a job. Three types of such constraints occur: start-start, end-start, and end-end constraints. With the Axxom software for the coupling of processes in a workflow there exist objects called "links." To express the connection of two processes and the properties of this connection with respect to time and for example material flow a link has the appropriate data fields. Other models again may use variables and equations to express this quite simple and direct constraint. Again, the solver itself has to calculate or even optimize the restriction whereas Axxom leaves this work to the "link"-algorithm that is efficient and fast. The time horizon of the entire problem defined by the earliest release date and the latest deadline of all jobs comprises approximately seven weeks, and the processing times on the machines range from few hours up to three days for laboratory testing. The objective function of the scheduling problem combines costs for the delayed finishing of jobs, operational costs per amount of product, and storage costs (i.e., early termination of jobs is penalized). When minimizing this cost function, it has to be considered that the three different lacquers incur different production cost.

The cost function is used at Axxom and also in other models. It simply gives a possibility to compare differences in calculated solutions and decide which one probably meets the requirements of the user the best. Axxom may use the structure

Fig. 4.11 The execution and reservation time on the mixer resources was reduced between the two views in the Gantt chart. Additionally the resource was changed from Mixer M052007 to mixer M052002. The mixer resources are in the state "reserved" until the filling on the ABF1 resource is done. With the tree networks it is easier for the optimization algorithms to reduce the through put time. In general, one can see that the resource utilization in the *lower* Gantt chart is higher than in the *upper* Gantt chart. The chain objects used in the upper calculation represent processes to be planned together. If one process of the chain is shifted, the other processes consequently will be shifted, too. The number of chains is arbitrary in general; but in practical cases two to five simultaneously gives good and fast results. Chains are synchronized with respect to reservations to avoid dead lock situations. The color blue in the Gantt charts indicates a start of a reservation, the red color ends a reservation started before.

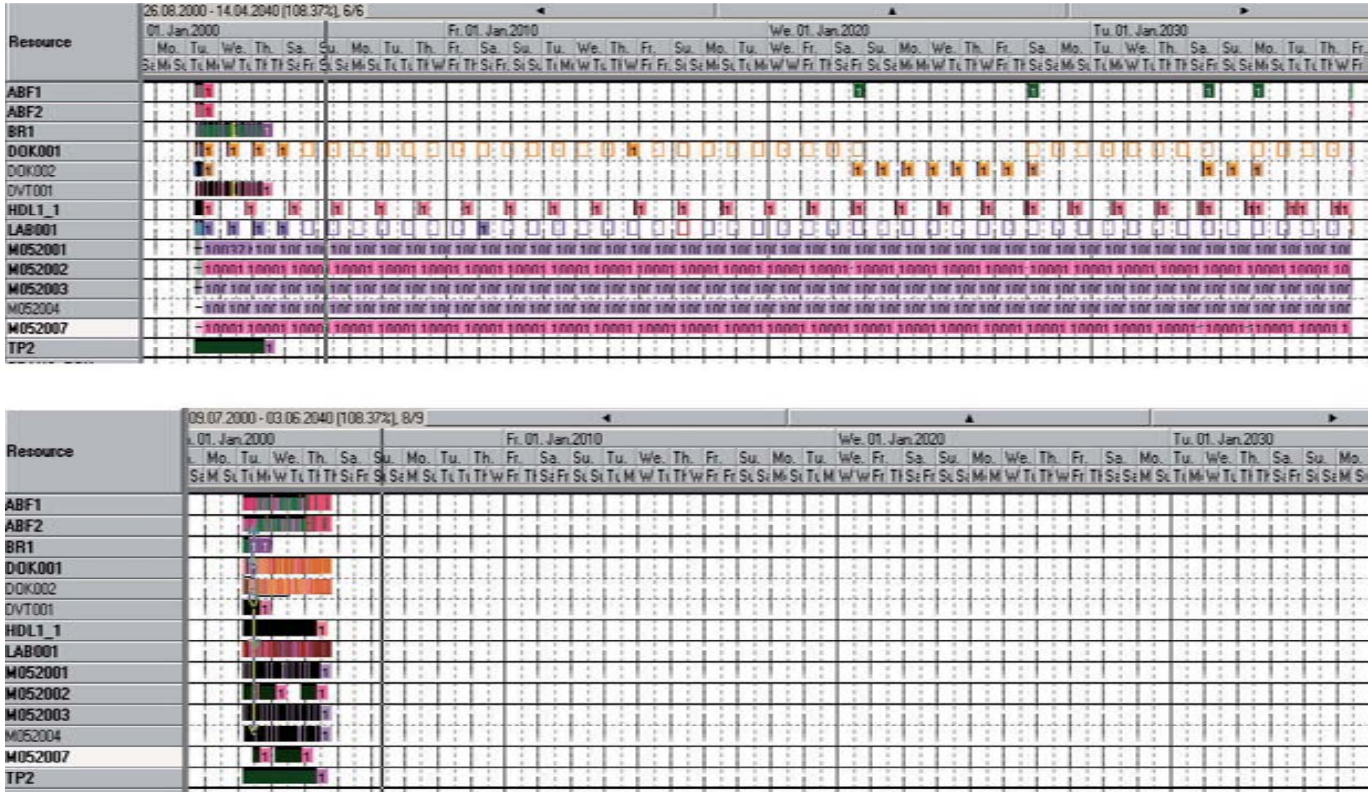

Fig. 4.12 The difference before and after the change of the network. The first scheduling result shows a start solution before the change of the network, the second after the change. The planning horizon necessary to take all processes is reduced from about 20 years to 2.5 years.

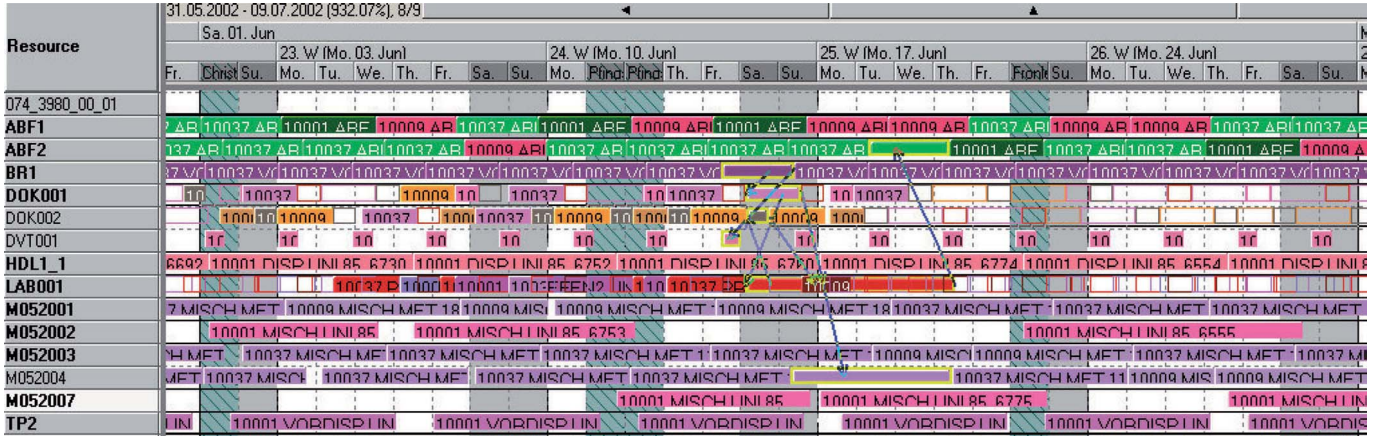

Fig. 4.13 A zoom into a certain planning period (June 2002) shows once more the detailed improved flow.

of the cost function to derive where the minimum may be found, but any other user defined calculable criterion that helps to find the optimal solution may be used. LP-solvers only use the objective function to find the minimum. All other criteria must be expressed in terms of the objective function. This increases modeling expenses and leads to abstract, less intuitive formulations. Because of this solvers like UPPAAL use also heuristics [6]. Using this software, solutions for 29 orders can be obtained after 1s. For models with up to 2000 processes special simplifications have been applied such as limiting the number of clocks by the non-overtaking heuristic. But explicit working hours, i.e., shift models and lock up times, still introduce additional clocks and are a field of further investigations. Axxom, too, has non-overtaking rules in the algorithm. These are for example expressed by sorting functions determining a "best" pre-ordering of the processes, a most common used sorting function sorts by due dates. Shift models are simply again small algorithms to calculate the times of a process when it is planned on a certain resource.

4.8.2 Example 2: API Manufacturing in Pharmaceutical Production

Number of resources: 200
Number of different staff qualifications: 10
Number of products (including intermediates and raw material): >10 000
Number of quants: >30 000

Planning horizon: three years with stepwise less constraints on capacities when moving to the future.
The production consists of more than 30 steps.

4.8.2.1 Constraints

Some special requirements and constraints of this example are:

- optimal lot sizing with minimum and maximum lot sizes and target stock levels,
- shift models,
- limits on staff capacity,
- quants with feedback information,
- BOMs with intervals of validity and alternative routings,
- limited perishability of intermediates,
- cycles and co-products,
- sequence dependent setup and cleaning,
- moving bottlenecks.

For some quants staff is required with different qualifications and fractional numbers (multiple resources). One staff qualification for the setup and one qualification for the production for each quant are taken into account. For certain resources a shift model and also lockup intervals were defined. These constraints may force quants to be enlarged or lengthened over a shift break or lockup interval. This means that within the setup and production intervals of a quant there can

be waiting times. Some of the quants produce intermediate products that have a maximum perishability. The maximum idle times within the production intervals have to be kept shorter than the maximum perishability.

Every link in the product flow network can have a validity interval. Validity intervals may be automatically extended to add missing validity intervals that are necessary to cover the whole planning horizon.

4.8.2.2 Additional Modeling Features for this Example

Products: Special products may accumulate a group of products to model the fact that the collected products are the same, but produced with different recipes for example if there are alternative BOMs.

An important group of products are the secondary products or co-products. These are important for modeling the reuse of a co-product and hence cycles in the production process. There can be forward and backward cycles. Sometimes cycles need an initial quantity to start with. Some cycles should stay in a stationary state, others are only used to run until a material in stock is processed and then change to an alternative production.

Additional data in the *product flow* is used:

- In some cases there are specific batch sizes dependent on the two linked products. The start batch size describes how much quantity has to be produced before the succeeding quant can start.
- Some BOMs have validities: start and end of validity interval and the reference time used to determine if the product flow is valid or not: earliest start, due date, start or end of production.

More data of *resources* is used: The batch mode defines how to round (up or down) to the next batch. The formula is

$$\begin{aligned} T(p) = {} & \text{Quantity/rate of production} \\ & \times \text{batch size/performance factor of the resource} \end{aligned}$$

A constant time dependency on the resource and the product is added to model none quantity dependent execution times, for example re-purchase times.

Shift models and *lockup times*: Every shift may have a performance factor f that changes an execution time T of a quant to T/f. The shift model may change after a few weeks or months. Therefore after this time the shift model can be simplified to the availability factor that expresses how much percent of the time can be used for production. It is possible to specify if a quant is interruptible by a shift pause or a lockup. This may depend on the product that is produced by the quant. Using the maximum allowed break for a quant is also a technique to model multiple single resources as one multiple resource.

4.8.2.3 Material Balance

There is an alternative number to group alternative product flows to alternative groups: A set of product flows f1, . . ., fn that ends in one certain product P2 links

the predecessor products P1(f1), . . ., P1(fn) with the successor product P2. The product flows f1, . . ., fn are grouped by their alternative numbers.

For the successors of a quant Q1 with quantity m given by all quant links to successors l1, . . ., lm the sum of out flowing quantities is the sum over l1, . . ., lm material factor × covered quantity. This must be smaller or equal to m.

4.8.2.4 Calculation of Delay Costs and Stock Costs

The first way to calculate the delay and stock costs of a quant is to multiply a delay cost factor/stock cost factor with the difference between the due date and the end of production or any time of the quant. In general this is a rough approximation for an easy and fast calculation.

The due date calculated by backwards termination is a minimum over all latest ends of production for every successor quant. Therefore it is exact for quants with at most one successor. If there are two or more successors the delay costs calculated with the due date may be higher than the real delay costs because the due date is the smallest end necessary to supply the earliest successor.

4.8.2.5 Dynamic Balancing of Material

This more detailed model is necessary for target stock calculations where productions may overlap, the lengths of quants differ significantly and quants have multiple predecessors and successors (see Figure 4.14). Calculating lot sizes with for example the formula of Andler yields completely different results and the sum of changeover costs and stock costs are much higher.

4.8.2.6 Objective Function

The demands are given as orders which are partially movable or have a fixed assignment to a resource with clearly defined setup, production and cleaning times. There are also anonymous demands that were calculated from forecasts. The target inventory is a soft constraint that is used to model dynamic safety stocks. Most quants must fulfill integer batch sizes and often minimum lot sizes.

Changeover times are between zero and three weeks corresponding to costs of some thousand Euros. Depending on the objective the optimizing algorithms organize the production in campaigns by grouping quantities on different levels: demand level, static lot sizing, scheduling.

The planner can include the stock costs in addition to the other costs, but as a consequence delay costs are balanced with stock costs. Here this effect is not wanted: at first high demand satisfaction is required. Keeping this to the maximum level, stock costs are lowered around this optimum. The trade off between production for future demands and inventory costs can be maximized by the so called shift operator.

The objective function of this scenario contains changeover costs, delay costs, stock costs and production costs. The delay costs and stock costs can be calculated in different ways dependent on the situation in which they are needed.

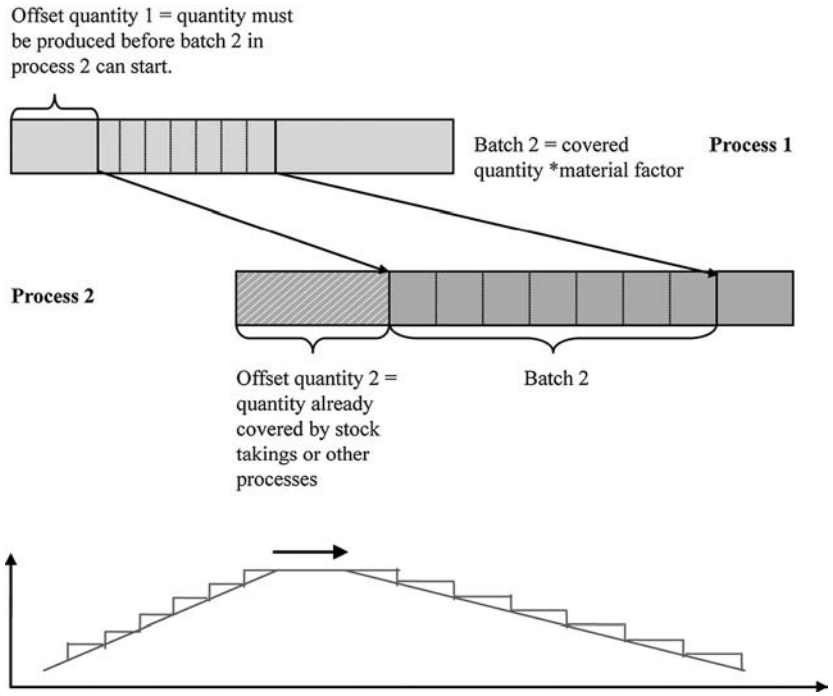

Fig. 4.14 The first campaign sends its quantity in steps of batches to the stock. From there the second campaign takes the needed material for production. The transport time can be interpreted as a "moving warehouse." The first stock taking action is then the start of batch 2 or if the transport time passes before the material can be used by batch 2 then the first stock taking is the start of batch 2 minus the transport time.

4.8.2.7 General Optimization

A user can give an arbitrary amount of time to complete an optimization. A result can be extracted by letting the actual best solution be written or skip the optimization, view the result and restart again with this last result after having eventually modified/fixed/moved some quants. The longer the optimization runs the better the end result will be.

In real life data is usually not fully correct. Therefore – before any optimization will be successful – a thorough check of the input data is necessary. The first check of data correctness can be done within approximately 2 min. This step has to be done by the user at least one time and then until the predefined checks show no errors any more.

4.8.2.8 The First Generation of the Quant Network

In principal the generation of the quant network is done by decomposing the overall decision problem into smaller sub-problems by looping around nested recursive functions that are used to divide the search tree into the parts that are useful to

enumerate. This can be done in the kind of a simple rule based backwards explosion. The algorithm already considers at this point the complete set of constraints and also the complete objective function. Of course a good combination of both methodologies can use the advantages of both methods: simple backwards explosion with rules is an easy to use planning procedure with the drawbacks that for example resource capacities can only be considered approximately. The consideration of all constraints every time leads to long runtimes with no equivalent effect on the quality of the solution. Therefore if no decisions have to be taken or the decisions only depend on local information (e.g. material factors) fast methods are applied. Especially for checking alternatives or lot sizing decisions the full complexity of all constraints and the objective function are taken into account.

4.8.2.9 **The Outer Optimization Loops**

In the beginning there is a general loop to decide if more lot sizing procedures should be applied to the existing quant network to meet the constraint of the minimum batch sizes of products. Then the quant network is examined, free usable stocks and free quantities of quants are made available. The material balances of any quant are calculated and decisions are taken whether quants require further explosions of their BOM. Structures for a fast cycle checking, sorting of existing quants and quant links and forecast intervals are built up. A recalculation of the due dates for all quants – also the ones of orders – can be done if specified by the user.

At first the demands caused by target inventories are not considered. Because of the given minimum lot sizes or batch sizes the target inventories may already be met. After that additional quants may be generated if the target inventories are not yet met.

Typically, orders are simpler to explode than anonymous demands, because for an anonymous demand it might be required to define a distribution of the forecast quantities over the forecast interval: only one quant, quants with equal quantities, due dates at the beginning or end or equally distributed. At this time it can be defined for each product if lot sizes, batch sizes, minimum quantities, maximum quantities of quants should be considered. The quantities are broken into predefined equal parts and then assembled until they meet the mentioned constraints.

If the object is a forecast interval the desired balance level is considered by estimating the balance resulting from the actual existing objects (quants, stocks, forecasts) and if necessary generating only the difference to the desired balance level. Quantities can be:

1. left as they are;
2. rounded up or down;
3. rounded up to the minimum lot size;
4. increased to meet the minimum lot size by taking quantities from following forecast intervals within a predefined range.

4.8.2.10 **Enumeration and Branch & Bound**

The central recursion traverses the product flow net. At arbitrary points of the explosion additional search trees to examine alternative explosions can be built up

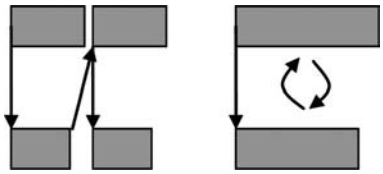

Fig. 4.15 Cycles in campaigns can be removed by aggregation.

in scaleable detail levels. The detail level can be high if the results are required to be exact.

At these solution steps the user can adjust own small rules and algorithms to find local optima. This could be, for example, a formula to calculate the quantity to explode for a forward cycle: When producing a primary quant of product A with quantity, for example, 1 t a co-product of product B with 0.5 t is created automatically. Using another quant to transform B back into A (material factor for simplicity reasons is assumed to be 1) it is sufficient for getting 1 t of A to generate a quant for A of $1t = xt + 0.5xt \leftrightarrow x = 2/3t$. The calculations including batch sizes, material factors and cycles in general networks again involve network traversing algorithms in combination with the backwards explosion algorithms for the product flow net.

In example 2 a simpler approach is used to correctly handle backward cycles (co-products). The difference to the forward cycle is that the co-product quant B created by a quant A cannot be used as predecessor of A, because cycles in the quant network are not allowed (violates the cause effect principle). A model can avoid this cycles using aggregation in such a way that cycles are "within" these quants A and B (see Figure 4.15).

The user is allowed to define a cost model for controlling the behavior of the optimizers. Rules can be derived from costs, times, resource utilization or lead times. Also parts of the quant network can be deleted again if they are identified as not optimal by applying user defined evaluation procedures.

In this example the validity is crucial for the choice of alternative BOMs. Validity intervals are none intersecting. The most left and right interval (may be the same if only one exists) are extended to cover the parts of the planning horizon where no specific recipes are defined.

4.8.2.11 **Static Lot Sizing**

The static lot sizing has several positive aspects. Big quant networks are transformed into smaller ones thus making faster calculations possible still fulfilling all constraints. The objective function can be reduced to contain only the changeover costs and the stock costs with the aim to find a trade off between these two costs (see Figure 4.17).

4.8.2.12 **Some Views on the Solution Process**

There is a campaign building operator with the objective of combining quants by shifting quants of equal products so that there is no gap between them, removing the changeover between the quants (see Figure 4.16).

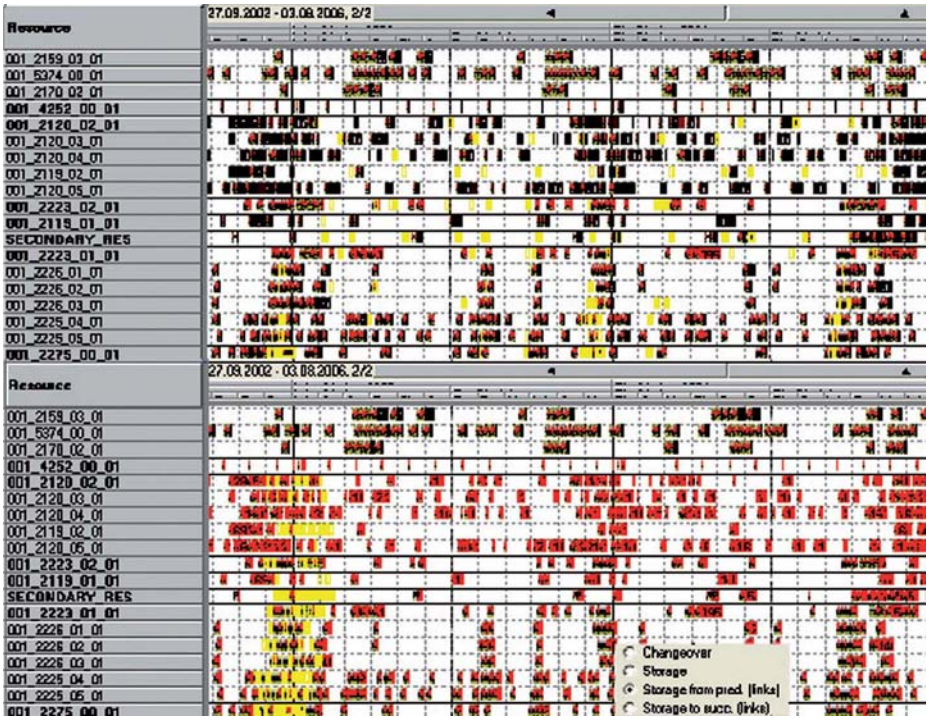

Fig. 4.16 The examination of combining 236 of a total of 28770 quants (253 quants are visible). Change over costs are Euro 11.2 m. in the *upper part*, the stock costs are 235,000 Euro. When combining the selected quants the stock and delay costs are reduced to 45,000 Euro + Euro 4 m. for changeover.

4.8.2.13 **Propagation of Information Between Operators**

At the end of an optimization operator a feasible solution has been built up. Further application of algorithms should improve this plan. Therefore the parts with the most promising potential for improvement must be found. Costs are mostly not a good criterion because they change in a non continuous way: quants change from delay to stock costs, changeover costs are zero or non-zero. Also a shift model introduces a lot of volatility.

An approach to some problems is the bottleneck analysis:

- The conflict number of a quant P on one of its alternative resources A is the number of other quants Q that were shifted or pushed away by P when assigning P on A.

The simplest case in which this characteristic number works is a set of quants produced on a bottleneck that have alternative resources which are not bottlenecks and where these quants can be assigned to just in time. A following optimization

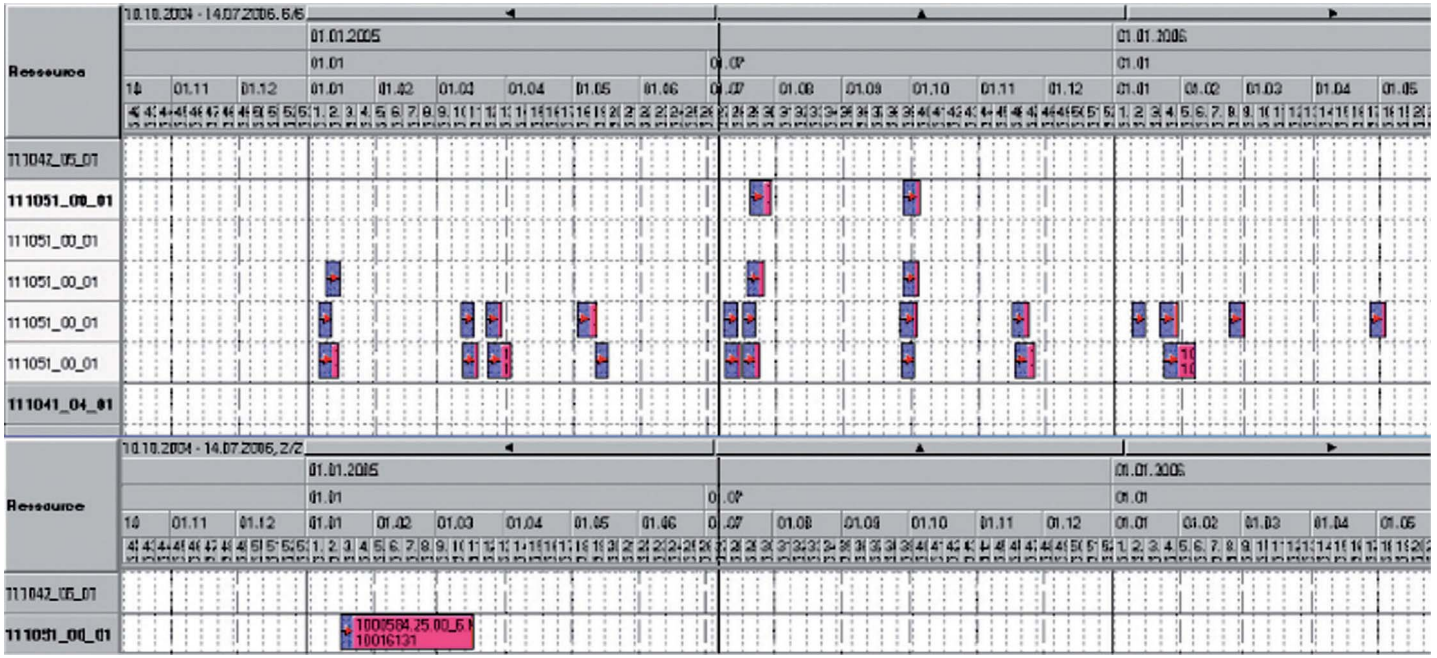

Fig. 4.17 Static lot sizing for a batch product. The plan is a just-in-time production. According to the due date every batch is planned with minimum costs. Setups are indicated with a *red arrow to the right*. The stock costs of 45,216 Euro are a result of the different batch sizes of the product itself and its successor product. Therefore not every batch can be planned just-in-time. The changeover costs are 4,182,935 Euro. The sum of both costs is 4,228,151 Euro. The lower part of Figure 4.17 shows the result of the same branch & bound algorithm. A campaign was built by combining the batches until the optimal trade off between changeover and stock is reached. This campaign must be planned earlier to deliver the first sub quantity. Stock costs are now 334,000 Euro, but the changeover costs were reduced to 160,884 Euro and the total costs to 494,884 Euro thus saving 90% of the total costs.

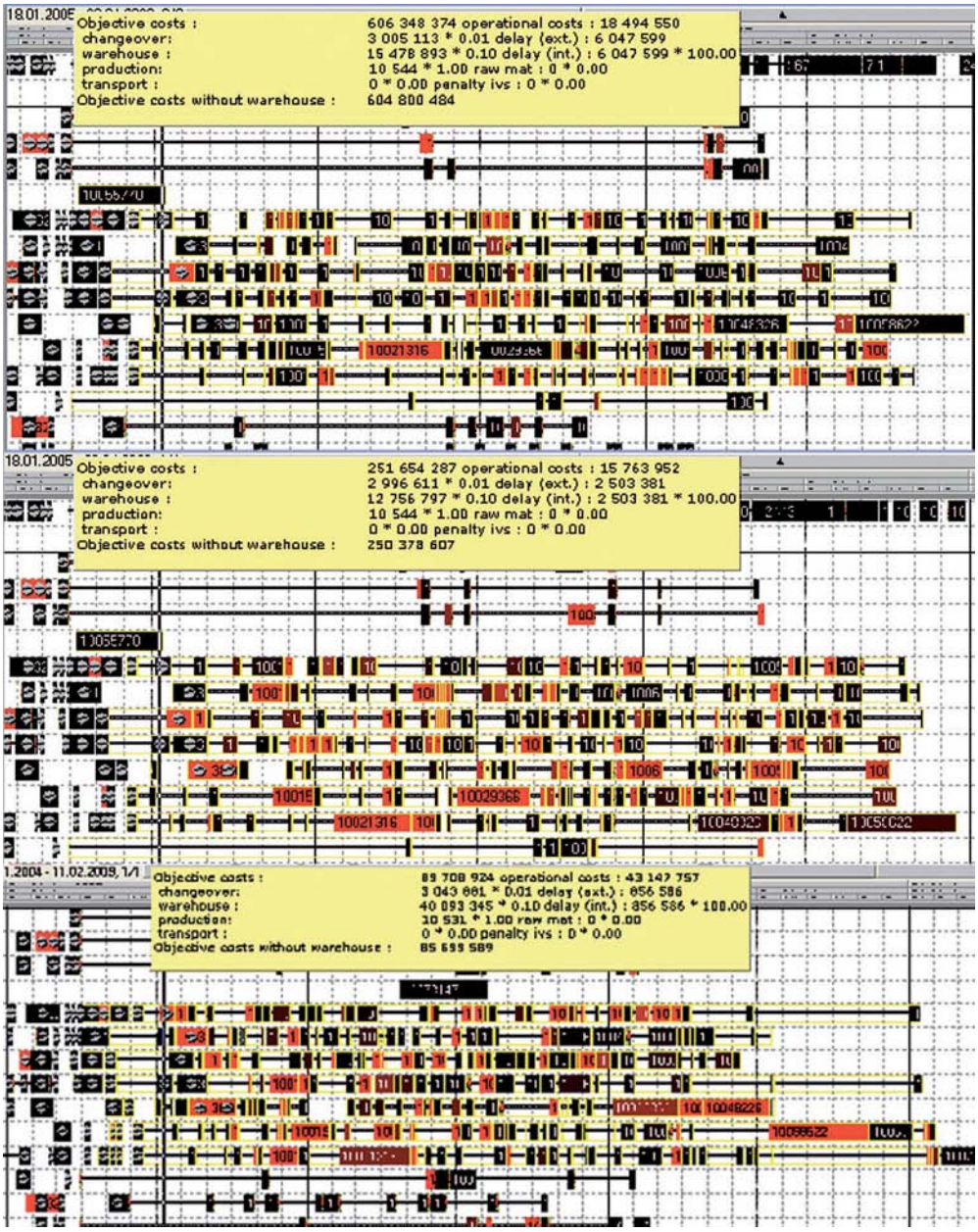

Fig. 4.18 The denser the quants planned on the resources the more bottlenecks over time arise and the conflict numbers rise. An evolution is shown of three optimizations with permanent improvements. In the *upper Gantt chart* the conflicts are mostly in the middle time horizon of the Gantt chart. In the *middle Gantt chart* the conflicts are distributed over the whole time horizon.

run can use the conflict numbers to modify the selection of the alternative resources: the resource with the smallest conflict number is chosen within a tolerance interval that allows increasing costs temporarily, for example the alternative resource is slower and has higher costs. High conflict numbers are often a good indicator for an optimum (see Figure 4.18).

4.9 Summary

The quant-based combinatorial optimization is a new approach. It supports the simultaneous planning and optimization of complex production problems. The solution procedure is built with operators which may be applied in any sequence allowing integrated BOM explosion and scheduling. The overall solution procedure may be extended at any time by simply adding new operators [20, 21].

The algorithms have to handle large numbers of quants. This requires very efficient data structures and algorithms.

Comparisons show that the quant-based combinatorial optimization is a leading approach when it comes to big production networks with many constraints which are typical for the chemical or pharmaceutical industry.

Quant-based combinatorial optimization is not a theory. It is working in real life action at many companies.

References

1 Aboudi, R., Hallefjord, Å. and Jørnsten, K. (1991) A facet generation and relaxation technique applied to an assignment problem with side constraints. *Eur J Oper Res*, **0** (3), 335–344.

2 Aboudi, R. and Hallefjord, K. (1990) Resource constrained assignment problems. *Discr Appl Math*, **26** (2/3), 175–191.

3 Ahuja, R., Magnanti, T. and Orlin, J. (1993) *Network Flows – Theory, Algorithms, and Applications*, Prentice-Hall, Englewood Cliffs, NJ.

4 Behrmann, G., Hendriks, M., Mader, M. and Brinksma, E. (2005) Axxom scheduling with uppaal. Technical report, AMETIST.

5 Bellman, R. (1958) On a routing problem. *Q Appl Math*, **16** (1), 87–90.

6 Bertsekas, D. and Castañon, D. (1991) Parallel synchronous and asynchronous implementation of the auction algorithm. *Parallel Comput*, **17** (6/7), 707–732.

7 Bertsekas, D., Castañon, D. and Tsaknakis, H. (1993) Reverse auction and the solution of inequality constrained assignment problems. *SIAM J Optim*, **3**, 268–299.

8 Bertsekas, D. (1990) *The Auction Algorithm for Assignment and Other Network Flow Problems: A Tutorial*, **20** (4), 133–149.

9 Bertsekas, D. (1994) Mathematical equivalence of the auction algorithm for assignment and the ε-relaxation (preflow-push) method for min cost flow, in *Large Scale Optimization. State-of-the-Art* (ed. W.W. Hager), Papers Presented at the Conference, held 15–17 February 1993 at the University of Florida, Gainesville, FL, pp. 20–44.

10 Blazewicz, J., Ecker, K., Pesch, E. *et al.* (1996) *Scheduling Computer and Manufacturing Processes*, Springer, Berlin.

11 Bierwirth, C. and Mattfeld, D. (1999) Production scheduling and rescheduling

with genetic algorithms. *Evol Comput*, **7**, 1–17.

12 Bohnenkamp, H.C., Hermanns, H., Klaren, R. *et al.* (2004, September) Synthesis and stochastic assessment of schedules for lacquer production. Proceedings of QEST'04, LNCS.

13 Brucker, P. (1995) *Scheduling Algorithms*, Springer, New York.

14 Catoni, O. (1998) Solving scheduling problems by simulated annealing. *SIAM J Contr Optim*, **36**, 1539–1575.

15 Cavalieri, S. and Gaiardelli, P. (1998) Hybrid genetic algorithms for a multiple-objective scheduling problem. *J Intell Manuf*, **9**, 361–367.

16 Chakrabarti, P. (1999) Partial precedence constrained scheduling. *IEEE Trans Comput*, **48**, 1127–1131.

17 Chandrachoodan, N., Bhattacharyya, S. and Liu, K. (2001) Adaptive negative cycle detection in dynamic graphs. *Proc Intl Symp Circ Syst* (ISCAS 2001), V, 163–166.

18 Cherkassky, B. and Goldberg, A. (1996) Negative-cycle detection algorithms. *Eur Symp Algorithm*, 349–363.

19 Cherkassky, B., Goldberg, A. and Radzik, T. (1994) Shortest paths algorithms: Theory and experimental evaluation. SODA: ACM-SIAM Symposium on Discrete Algorithms (A Conference on Theoretical and Experimental Analysis of Discrete Algorithms).

20 Cook, W., Cunningham, W., Pulleyblank, W. and Schrijver, A. (1998) *Combinatorial Optimization*. Wiley-Interscience Series in Discrete Mathematics and Optimization, Wiley, New York.

21 Cormen, T., Leiserson, C. and Rivest, R. (1990) *Introduction to Algorithms*, McGraw-Hill, New York.

22 Engell, S. and Panek, S. (2003, May) Mathematical model formulation for the Axxom case study. Technical report, University of Dortmund.

23 Ford, L. and Fulkerson, D. (1962) *Flows in Networks*, Princeton University Press, Princeton, NJ.

24 Graham, R., Lawler, E., Lenstra, J. and Kann, A. (1979) Optimization and approximation in deterministic sequencing and scheduling: a survey. *Ann Discrete Math*, 287–326.

25 Goldberg, A., Oldham, J., Plotkin, S. and Stein, C. (1997) *An Implementation of a Combinatorial Approximation Algorithm for Minimum-Cost Multicommodity Flow*, Springer, London.

26 Goldberg, A. and Kennedy, R. (1995) An efficient cost scaling algorithm for the assignment problem. *Math Prog*, **71**, 153–178.

27 Goldberg, A. (1997) An efficient implementation of a scaling minimum-cost algorithm. *J Algorithms*, **22**, 1–29.

28 Heistermann, J. (1994) *Genetische Algorithmen – Theorie u. Praxis evolutionärer Optimierung*.

29 Goldberg, A. and Radzik, T. (1993) A heuristic improvement of the bellman-ford algorithm. *AMLETS: Appl Math Lett*, **6**, 3–6.

30 Herroelen, W. and Demeulemeester, E. (1998) Resource-constrained project scheduling: a survey of recent developments. *Comput Ops Res*, **25**, 279–302.

31 Holland, J. (1975) *Adaption in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, MI.

32 Kinnebrock, W. (1994) *Optimierung mit genetischen und selektiven Algorithmen*.

33 Kernler, H. (1995) *PPS der 3. Generation: Grundlagen, Methoden, Anregungen*.

34 Ku, D. and De Micheli, G. (1992) Relative scheduling under timing constraints: Algorithms for highlevel synthesis of digital circuits. *IEEE Trans Comput Aided Des*, **11** (6), 696–718.

35 Moore, E. (1959) The shortest path through a maze. Proceedings of the International Symposium on the Theory of Switching, pp. 285–292.

36 Pallottino, S. (1994) Shortest-path methods: Complexity, interrelations and new propositions. *Networks*, **14**, 257–267.

37 Panek, S., Engell, S. and Lessner, C. (2005) Scheduling of a pipeless multiproduct batch plant using mixed-integer programming combined with heuristics. Proceedings of European Symposium on Computer Aided Process Engineering, ESCAPE 15.

38 Pape, U. (1974) Implementation and efficiency of moore-algorithms for the shortest route problem. *Math Progr*, **7**, 212–222.
39 Pratt, V. (1977) Two easy theories whose combination is hard. Technical report.
40 Ramalingam, G., Song, J., Joscovicz, L. and Miller, R. (1995) *Solving Difference Constraints Incrementally*.
41 Späth, M. (2001) *Heuristische und kombinatorische Verfahrenselemente für Suchalgorithmen in der Prozeßoptimierung*.
42 Morad, N. and Zalzala, A. (1999) Genetic algorithms in integrated process planning and scheduling. *J Intell Manuf*, **10**, 169–181.
43 Rechenberg, I. (1973) *Evolutionsstrategie*.
44 Shue, L. and Zamani, R. (1999) An intelligent search method for project scheduling problems. *J Intell Manuf*, **10**, 279–288.
45 Tinhofer, G. (1999) *Kombinatorische Optimierung*, Vorlesungsskriptum.
46 AMETIST (2005) Case Study 4: Value Chain Optimization – Final Report, available at: http://ametist.cs.utwente.nl/INTERNAL/PUBLICATIONS/DELIVERABLES/del3.4.2.pdf.

5
Scheduling and Optimization of a Copper Production Process*

Iiro Harjunkoski, Marco Fahl, and Hans Werner Borchers

5.1
Introduction

The production of copper, mankind's oldest metal, dates back more than 10 000 years and copper still plays an important role in everyday life. Today, the *red gold* is used in a wide area of applications, such as construction, electronic products, transportation equipment, consumer and general products, and a number of industrial machinery and equipment. Copper is commonly found in products from automotive, marine, piping and telecommunication industries and is therefore present almost everywhere. The importance of copper is still increasing, not only because of its wide applicability in the rapidly growing industries but also owing to its unique material properties and practically 100% recyclability: copper does not degrade, neither in quality nor value, during its reprocessing. It can be expected that the major copper-consuming industries will continue to grow, which also will keep the demand high. This predicts the need of higher overall production volumes and consequently a drive towards more efficient processes in existing copper plants.

Copper production is quite a complex process to plan and to schedule due to the many process interdependencies (shared continuous casters and cranes, emission level restrictions, limited material availability, to name a few). This makes it very difficult to foresee the overall consequences of a local decision. The variability of the raw material has alone a significant impact on the process, various disturbances and equipment breakdowns are common, daily maintenance operations are needed and material bottlenecks occur from time to time. The solution that is presented here considers simultaneously, and in a rigorous and optimal way, the above mentioned aspects that affect the copper production process. As a consequence, this scheduling solution supports reducing the impact of various disturbance factors. It enables a more efficient production, better overall coordination and visualization of the process, faster recovery from disturbances and supports optimal

* A list of nomenclature is given at the end of this chapter.

maintenance planning. This translates directly into increased plant throughput and revenues.

Very few contributions on copper plant scheduling problems can be found in the literature. Pradenas et al. [1] developed a heuristic algorithm to maximize the production at the Chuquicamata Copper Smelter in Chile. In that work, some basic requirements are considered, some of which concern parallel processing. First all combinations of three possible production cycles are generated, and then assigned to the equipment and thereafter the resulting schedule is tested against the best existing reference solutions. The objective is to maximize copper production for one day of operation, i.e., to assign as many cycles as possible. The main constraints are operational, metallurgical, mass balance related, environmental or related to the timing of loading and unloading operations.

No purely mathematical programming-based solutions that would be able to handle the requirements described here are known to the authors.

5.2 Copper Production Process

The principle of the considered copper production process with sulfured copper ores is very simple: The main task is to remove all extra elements from the copper ore (mostly sulfur and iron). As for most metals, this is done through a smelting process at extreme temperatures and thus requires special equipment. The processing equipment may contain more than 350 tons of material at a time. A large copper plant often has parallel processing lines at least for the bottleneck stages. The material is commonly transported in ladles carried by cranes. Depending on the plant layout, the process must be synchronized such that parallel activities do neither overload nor block the cranes, which could cause expensive process delays. This adds complexity to the process logistics and requires a complete overview on the plant activities when doing the detailed production scheduling. Many interdependencies between equipment status, material amounts, process timings, and parallel process events may easily lead to schedules that are far from optimal.

The main production steps are:

- Sulfured copper concentrate (25–35% Cu) is processed in a primary furnace where the copper level is enriched to around 65%.
- The melt copper (matte) is further processed in a converter, where the remaining sulfur and iron is removed by underbath injection of oxygen-enriched air. The resulting so-called blister copper has a copper content of 98%.
- Various materials are added for maintaining an optimal temperature, melt viscosity, reaction conditions and internal and external recycling ratio of copper.
- Excess oxygen is removed from the melt in an anode furnace by blowing natural gas into the melt.
- After reaching the target copper purity (99.6%) the copper is cast into copper anodes, which are cooled down and taken into further processing.

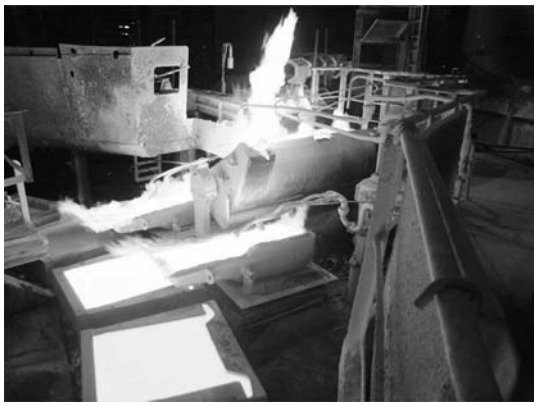

Fig. 5.1 Continuous casting.

A picture of the last step, the casting, can be seen in Figure 5.1. The processing steps are done under strictly controlled temperatures (up to roughly 1220 °C). During these, several chemical reactions take place and these enable the separation of copper from other materials. A typical reaction during the conversion is to remove the ferrous sulfides (FeS) by oxidization to iron oxides (FeO), which react with silica and can be thereafter removed as slag that is collected on the top of a ladle. As a side product, sulfur dioxide is formed and it is quite common to reuse it in a combined sulfuric acid plant. These two reactions are shown in (A) and (B):

$$2FeS + 3O_2 \rightarrow 2SO_2 + 2FeO \quad (A)$$

$$FeO + SiO_2 \rightarrow FeSiO_3 \quad (B)$$

Another major reaction takes place where the copper sulfides are oxidized, resulting in blister copper and sulfur dioxide:

$$Cu_2S + O_2 \rightarrow 2Cu + SO_2 \quad (C)$$

These and many other reactions must be taken into account when calculating the mass and energy balances, reaction times and other information that are needed for a detailed scheduling solution. The overall production process is illustrated in Figure 5.2, where the processing steps are displayed by rectangles and the corresponding copper content is shown by the curve. The processing times in each processing unit depend to a high degree on the input material quantities, and batch sizes and are therefore not shown in detail in the figure. Typically, it takes between 14–18 hours for one batch to go through the process. These slow dynamics also limit the possibilities to change or to fine-tune batches that are already partly in process.

The solution discussed in the following considers the main process steps, making it possible to optimally relate input material properties (amount, quality) to corresponding processing times and thus also enables an integrated planning and

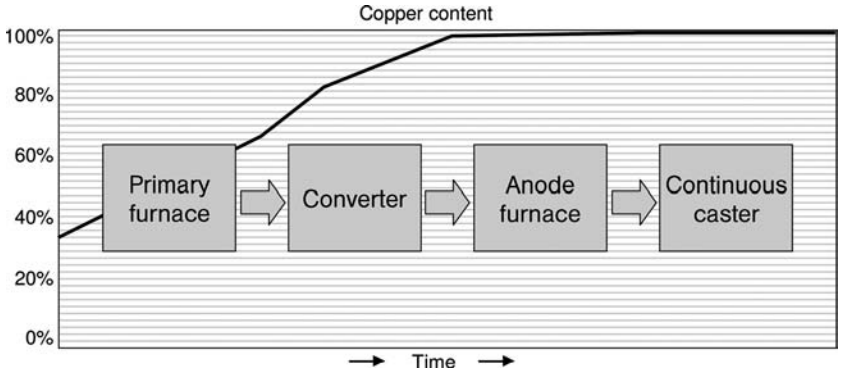

Fig. 5.2 Copper production process.

scheduling approach that simultaneously creates the corresponding production recipe for each batch. This ensures that the schedule obtained is feasible from the process point of view. The solution approach also provides the required flexibility to account for frequent quality variations in the available input material.

5.3 Scheduling Problem

Generally speaking, the main task of a typical scheduling solution is to perform the assignment and sequencing decisions for jobs on given resources or pieces of equipment. Also, it usually provides the detailed timing of each task. This is necessary for optimal and efficient sharing of limited resource capacities. In a copper production process with parallel production lines, a typical requirement is to synchronize the lines such that they serve the caster in an alternating way with anode copper. Furthermore, it is important to synchronize the lines also from the production logistics perspective. For the application at hand, it had to be ensured that no conflicting resource requirements (e.g., cranes) overlap. An example of such a conflict is the simultaneous charging and discharging of equipment in the parallel production lines. In our example, this is done by shared cranes that serve both lines. For instance, scheduling other crane activities to happen while the crane is busy with filling or emptying a converter (the most labor-intensive crane jobs that block a large part of the plant floor) would slow down the process considerably – delays of more than an hour are possible which also increases the probability of further production disturbances.

Another special aspect of the production process considered here is that there is only one unique end product – copper anodes with a final copper content of 99.6%. This changes the problem focus compared to other typical scheduling problems, where different properties of various products have to be taken into account in determining a production sequence, as well as cleaning requirements and product-equipment compatibility, to name a few. Here we do not have, e.g.,

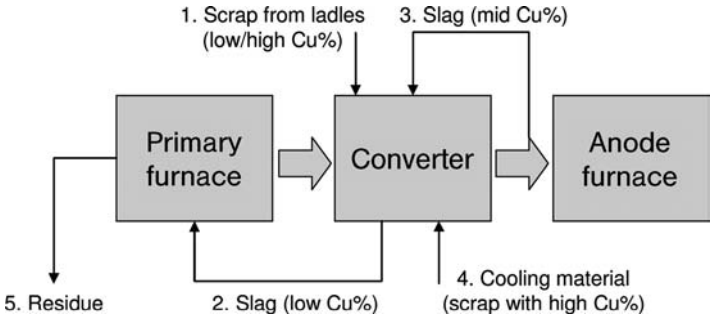

Fig. 5.3 Recycling during the process.

sequence-dependent changeovers but the major challenge comes from the highly varying raw material properties, which affect the optimal production recipe. An example of these variations is the copper content of the matte, which can vary roughly between 60% and 70%, depending on the chemical composition of the raw material (copper concentrate) used. A high copper amount significantly reduces the blowing time needed in the converter and furthermore also results in less slag that is collected along the blowing phases. Low copper content, on the contrary, leads to longer blowing times, large slag amounts and a need of more cooling material, which has two main purposes: (1) to cool the processing temperature during an intensive and oxygen-rich blowing phase and (2) to enrich the copper concentration, as the scrap used for cooling often has a very high copper content (recycling material).

According to the German Copper Institute [2], around 45% of the copper need in Germany can be satisfied by recycling material. As mentioned above, copper remelting is uncomplicated and involves neither material nor quality losses. Furthermore, through electrolysis it is possible to separate pure copper from various contaminants, which lowers the purity criteria of recycling goods.

The main recycling principle is shown in Figure 5.3. Practically all intermediate goods that occur during the process are fed back to the process. At the beginning of the slag blowing phase, scrap (1) that has, for instance, been collected from cooled ladles is added to the converter. The slag that is formed during the blowing (2) is returned to the primary furnace. At a later stage, a richer slag (3) that has been collected at the end of copper blowing of a previous batch is fed back into the converter. During the copper blowing, the temperature is partly controlled by adding recycling material (4) with a very high copper content. In fact, the only place where slag is really taken out of the system is in the primary furnace. The slag enters an electric cleaning furnace, it leaves (5) the cleaning furnace with a copper content of 0.7–0.8%. This residue has practically no copper and can be used for instance as construction material. Thus, all copper is efficiently used and no waste appears.

The raw material quality variations have a high impact on the schedule itself, as also the time for logistics (e.g., crane movements) has to be taken into account and

one of the main goals of an optimized scheduling approach is to ensure a well-synchronized process. For this reason, the scheduling model must also comprise the main mass balances, the most important dependencies between the amounts and the purities of the materials, the processing times (reaction kinetics), as well as the major crane transportation requirements. Consequently, in order to achieve an optimal and realistic production schedule that can be executed on the plant floor, the resulting scheduling problem also needs to encapsulate a simultaneous recipe optimization.

Thus, from a problem classification perspective the scheduling problem boils down to a single-product multistage batch-scheduling problem with parallel processing equipment and the requirement of simultaneous recipe optimization. The main goal of the solution is to generate a detailed and optimized schedule for the copper production process that explicitly considers two parallel converters and anode furnaces, as well as, a continuous caster that is directly connected to the anode furnaces and the throughput of which is maximized. Scheduling of the primary furnace is not considered in detail. What is furthermore important, the computing time for generating a valid optimized schedule should not exceed a few seconds. Therefore the scheduling model is not allowed to be too complex but should only capture the essentials, i.e., major issues that affect the production timing. Due to the fact that the extreme operating conditions create regular maintenance needs and cause unexpected equipment breakdowns, it is also important to include the scheduling of maintenance tasks in the optimization model. Optimal maintenance scheduling is needed in order to minimize the impact of maintenance actions and corresponding process interruptions with respect to throughput reduction.

Summarizing the above, the main decisions to be made are:

- timings of the processing steps (parallel lines, maintenance),
- material amounts (melt copper, scrap material, slag),
- production cycles (converter and anode furnace),
- utilization of limited resources (timing critical),
- reserving sufficient time for necessary logistics (e.g., cranes).

Since we are dealing with a single-product problem it is important to mention that the sequence of batches is insignificant and therefore assumed as given *a priori*. However, this refers only to the batch numbers. Of course, the recipe optimization will be applied to each batch individually but since that is done as an integrated part of the scheduling problem, the given sequence is left unchanged. The assignment problem is somewhat limited as well, since normally the predefined batch number also defines the converter to be used. Furthermore, it is often an advantage to couple a certain converter with a given anode furnace in order to minimize the crane traffic complexity. However, all these are not strict requirements and therefore depend on the actual status in the plant with respect to disturbances and the maintenance needed. Therefore, the batch routes are not strictly predefined but may vary as

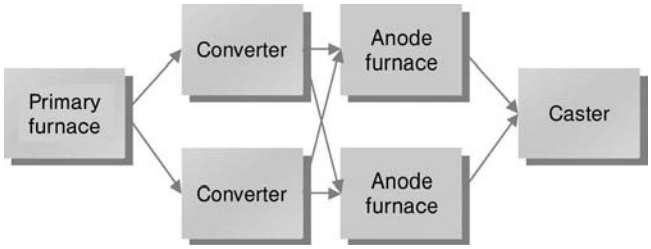

Fig. 5.4 Parallel processing equipment.

indicated in Figure 5.4. The frequency and likelihood for the optimization to suggest, e.g., crossing batches, as in Figure 5.4, can be controlled through setting proper objective function coefficients. This is important since crossing batch routes can occur at disturbance situations and, e.g., due to the more complex logistics, the sooner the process can return to *normal operation*, the better.

5.4 Solution Approach

The copper production process is difficult to plan in advance due to the lack of sufficient and exact measurement (or forecasting) data, logistical complexities, high raw material specification variability, frequently occurring disturbances and additional tasks, e.g., maintenance operations that heavily affect the process cycles. Normally, the production planning and scheduling is therefore done manually. Since in such a case a *global* overview of the process is often missing, typically each unit ends up running at *full speed*, i.e., trying to produce as much as possible (local optimum from an individual equipment perspective). This results in productivity losses since the overall process efficiency may be far from optimal as many batches end up unnecessarily waiting for equipment in the next production stage.

The solution developed (see Figure 5.5) considers simultaneously, and in an optimal way, the most important aspects affecting the copper production. In order to cover the process itself and the necessary information and decision flow, the solution builds on a valid and robust process model that captures the main chemical reactions and is able to link the variable material amounts with predicted processing times. The main input data comprises:

- batch numbers and data (fixed material amounts),
- copper content of various material inputs (laboratory results and prediction),
- current status of the equipment (with estimated end-times),
- maintenance jobs to be planned (exact time or a given time window),
- start-time of the schedule horizon (automatic or manually entered).

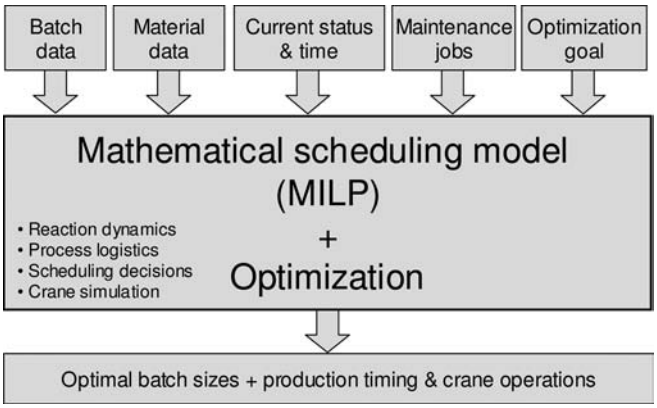

Fig. 5.5 Solution overview.

A mathematical programming solution approach has been selected to ensure that an optimal solution can be obtained. This poses two main challenges:

- The resulting process model is intractable in its pure form, mainly due to the underlying chemical reaction kinetics.
- Logic decisions that need to be made in scheduling (commonly precedence and equipment assignment) must be represented by discrete or binary variables.

In order to make the problem solvable, a linearized process model has been derived. This enables the use of standard Mixed Integer Linear Programming (MILP) techniques, for which robust solvers are commercially available. In order to ensure the validity of the linearization approach, the process model was verified with a significant amount of real data, collected from production databases and production (shift) reports.

A standard continuous-time job-shop scheduling formulation [3] can be used to model the basic aspects of the production decisions, such as sequencing and assignment of jobs. Here, the key of the mathematical solution is to capture the durations of each processing step and to relate it to the amounts of material. Therefore, only a top-down approach will be presented to illustrate some main principles of the model.

First, some constraints related to the converter operations are defined. In the following, we assume typically four major converter operation stages: charging, slag blowing, copper blowing and discharging. The index s is used to denote a certain stage and p refers to a batch. The corresponding upper case letters refer to the respective sets, e.g., S^C refers to the stages that are valid for the converting. Two different timings must be included. The time when a batch actually is in the converter is defined by the start and end-times, t_p^{CB} and t_p^{CE}. These are relevant for keeping track on the equipment availability. This time is then divided into more detailed times, which define the start and the end-time of each converter stage,

$t^{CSB}_{p,s}$ and $t^{CSE}_{p,s}$. These in turn are necessary for the synchronization of parallel tasks. Here, we consider mostly the converting stages.

$$t^{CSE}_{p,s} = t^{CSB}_{p,s} + T_{p,s} \qquad \forall p \in P,\ \forall s \in S^C \tag{5.1}$$

$$t^{CSB}_{p,s+1} = t^{CSE}_{p,s} \qquad \forall p \in P,\ \forall s \in S^C \,|\, s \leq 2 \tag{5.2}$$

$$t^{CSB}_{p,s+1} \geq t^{CSE}_{p,s} + \Delta_{p,s} \qquad \forall p \in P,\ \forall s \in S^C \mid 2<s<|S^C| \tag{5.3}$$

$$t^{CB}_{p} = t^{CSB}_{p,s} \qquad \forall p \in P,\ s = 1 \tag{5.4}$$

$$t^{CE}_{p} = t^{CSE}_{p,s} \qquad \forall p \in P,\ s = |S^C| \tag{5.5}$$

Constraint (5.1) specifies the relation between the start-times and the end-times of a stage, where $T_{p,s}$ is the processing time needed. Constraint (5.2) specifies that the two first stages must be immediately followed by the next one. Some flexibility is allowed for the rest of the stages, as defined by the delta-variable in Eq. (5.3). The relationship between first and last stages and the converter beginning and ending times are defined in Eqs. (5.4) and (5.5). They specify exactly the time when a converter is assigned to each batch.

Similar relations are given for each stage of all equipment operations. Below, an example is given where a stage combines two pieces of equipment. This is, e.g., the case when a converter is emptied to an anode furnace (requires that both units are available simultaneously) or during the continuous casting, which actually is done by discharging an anode furnace. The former case is illustrated by Eqs. (5.6) and (5.7):

$$t^{CSB}_{p,s} = t^{ASB}_{p,s'} \qquad \forall p \in P,\ s = |S^C|,\ s' \in \min\{S^A\} \tag{5.6}$$

$$T_{p,s} = T_{p,s'} \qquad \forall p \in P,\ s = |S^C|,\ s' \in \min\{S^A\} \tag{5.7}$$

The start-time of the last converter stage (emptying) should be equal to the start-time of the first anode furnace stage (filling). This is expressed by Eq. (5.6), where *s'* denotes the first stage of the anode furnace. Similarly, the durations are equal, as shown in Eq. (5.7). A minor phase shift, due to the crane movements and the initial filling of the converter, could be included but since this is typically on the order of a few minutes, it has been omitted here.

Another important issue is to avoid overlapping of the crane operations. In the middle of Figure 5.6 one can clearly observe the relation between the anode furnace (AF) filling and the converter (C1) emptying as described by Eqs. (5.6) and (5.7). For simplicity, the second anode furnace is not shown. The figure also roughly indicates the need to avoid simultaneous filling/emptying operations; the corresponding constraints are shown below. We assume that batch p is immediately followed by p+1. Here, also the converter stages are numbered from 1–4 according to Figure 5.6.

$$t^{CSB}_{p+1,1} \geq t^{CSE}_{p,2} + \Delta_{cr} \qquad \forall p \in P | p<|P| \tag{5.8}$$

$$t^{CSB}_{p,4} \geq t^{CSE}_{p+1,2} + \Delta_{cr} - \delta_{cr} \qquad \forall p \in P | p<|P| \tag{5.9}$$

These are two constraints for almost the same purpose. However, one of them is strict and the other one can be relaxed if needed, i.e., an overlap of emptying the

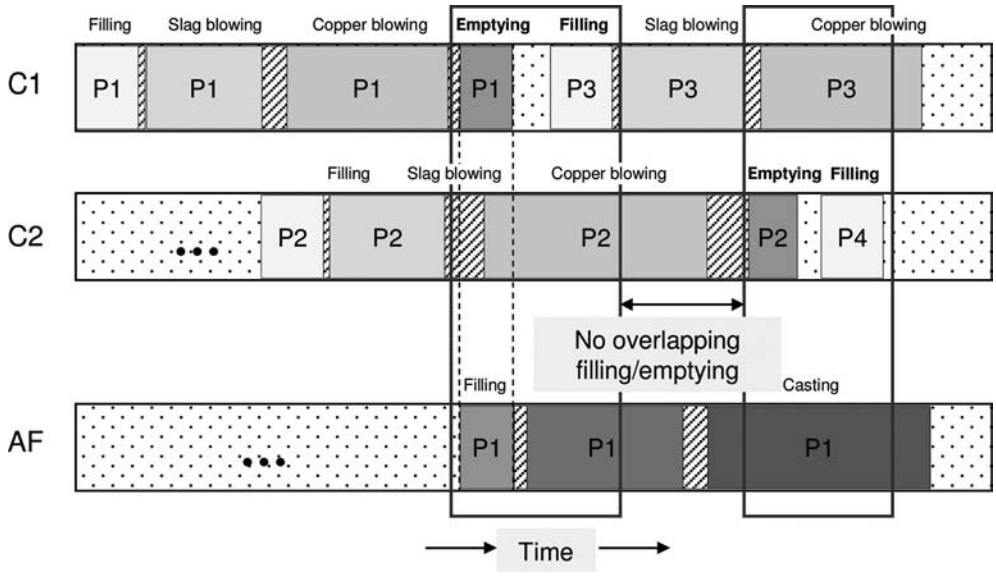

Fig. 5.6 Overlapping of crane operations.

previous batch and slag blowing can sometimes be allowed, as the cranes are not continuously needed during the slag blowing. Equation (5.8) states that the next batch should not be filled before the slag blowing has ended. On the other hand, according to Eq. (5.9) the previous batch should not be emptied before the next batch has ended its slag blowing. These cross-relationships between the batches make it nontrivial to find feasible sequences and also limit the flexibility to add new requirements into the model. Not least because of this, the integration of recipe optimization plays an essential role in finding an optimal production schedule.

Material balances are also crucial for a successful recipe optimization. It is important to ensure that the total amount of material does not exceed the converter capacity. Also, calculating the copper balance of the components is used to calculate the amounts of slag and a higher degree slag type that is recycled between the batches. A copper mass balance of the slag blowing is given in Eq. (5.10):

$$\begin{aligned}&\left(\eta_{Cu,f}-\eta_{Cu,1}\right)\cdot M_1+\left(\eta_{Cu,f}-\eta_{Cu,2}\right)\cdot M_2+\left(\eta_{Cu,f}-\eta_{Cu,3}\right)\cdot M_3\\&+\left(\eta_{Cu,f}-\eta_{Cu,4}\right)\cdot M_4=\left(\eta_{Cu,f}-\eta_{Cu,5}\right)\cdot M_5+\eta_{Cu,f}\cdot M_6\end{aligned}\tag{5.10}$$

In Eq. (5.10) the incoming components are just numbered for simplicity and their respective copper contents are given by η_{Cu}. The final copper content of the slag blowing stage is denoted as $\eta_{Cu,f}$. The inputs on the left-hand side are the matte and solid scrap material and the rich slag. In the slag blowing phase, the only outputs are the slag and sulfur in the form of sulfur dioxide. The mass of each component is expressed by M. A number of additional equations are needed to determine the individual masses. The slag with higher copper content is taken out

during the final converter steps and thereafter fed back to the next batch. This can be mathematically expressed as:

$$M_{p,4} = M_{p+2,4} \quad \forall p \in P \mid p < |P| - 1 \tag{5.11}$$

where component 4 is assumed to represent the slag type. The processing times of the various production steps ($T_{p,s}$) are calculated using equations that are, e.g., based on reaction kinetics. Generally speaking, these are considerably different for each stage. The simplest ones are the filling and emptying stages, where only an average crane operation time per ladle is used in the scheduling model for describing the relevant processing time. Also, we assume a given target caster speed for calculating the casting time. More complex approaches have to be applied for the calculation of slag and copper blowing times which are described by nonlinear functions. For our scheduling purposes these are linearized around a reference point, which is assumed to be in the middle of the normal operating region. A large amount of data from a production database was used to specify the correct coefficients of the linearized formulas based on parameter identification techniques. The total time for one of these specific processing steps is the sum of the blowing time and the time that is needed for adding other material.

For obtaining a realistic process model, available data from a production database was used, for instance, in order to estimate the correlation between the oxygen feed and SO_2 formation during the slag blowing. This was needed for estimating the processing times of the converter and the results of the data analysis were transferred to the model through coefficients. Figure 5.7 shows a clear trend between the two flows and the best-fit linear approximation.

Having discussed the calculation of the individual processing timings, and some material balances, the last type of constraint that is needed is a standard constraint that relates two batches and prevents possible resource conflicts.

$$t^{CB}_{p+1} \geq t^{CE}_{p} - M \cdot \left(2 - X_{p,e} - X_{p+1,e}\right) \quad \forall p \in P \mid p < |P|, \forall e \in E \tag{5.12}$$

Equation (5.12) prevents two batches from occupying the same resources simultaneously. The constraint can also be expressed for each substage but here we restrict ourselves to using the variables defined previously. The constraint of Eq. (5.12) must, of course, be complemented by an additional constraint that enforces the use of one of the alternative resources.

The choice of a good and appropriate objective function is very important for solving real-world scheduling problems. In this case, there are a number of contributions to the objective function, such as the various slack variables and their penalty factors for violating the respective constraints, as well as a term for describing the caster throughput. Since it is very difficult to directly maximize the throughput, the final batch sizes that will be cast are maximized together with a minimization of the makespan:

$$\max\left\{\mathit{throughput}\right\} = \max\left\{c_1 \cdot \mathit{final\ batch\ size} - c_2 \cdot \mathit{makespan}\right\} \tag{5.13}$$

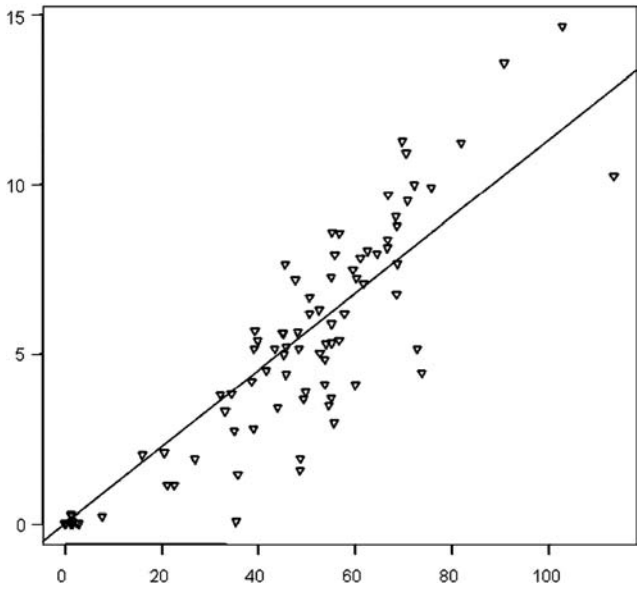

Fig. 5.7 Correlation between airflow and sulphur dioxide.

With a proper balancing of c_1 and c_2 this turned out to be a valid strategy, which of course is very sensitive to changes in the weighing factors of other terms in the objective function components. The makespan can be simply obtained by introducing an additional inequality that describes that the makespan should be greater than the end-time of casting for the last – or all – batches.

During normal operation of the copper plant, there are a number of regular maintenance jobs that need to be planned. They are included in the scheduling problem as additional jobs that have given release dates and due dates. These maintenance jobs can mostly be performed only when a unit is empty and not in use. The optimization approach finds the best location for each maintenance job with the least impact on production throughput and, furthermore, modifies the batch recipes such that there will be a suitable break in the operation for the equipment that must be maintained.

Because of the large number of aspects that must be covered by the model, it is important to have it well structured for further maintenance of the decision support system. A simplified representation is shown in Figure 5.8. The modeling was done such that the sections defining the general logistics were clearly separated from those closely related to the process. Also, the parts that refer to input/output to the user interfaces were strictly kept apart from the model itself to allow a smoother development.

The resulting optimization problem is solved using ILOG CPLEX [4], which generates a schedule for all major process steps, as well as the main material requirements for the production (optimal recipe definition for each batch). The schedule obtained is furthermore passed on to a crane movement simulation module, which

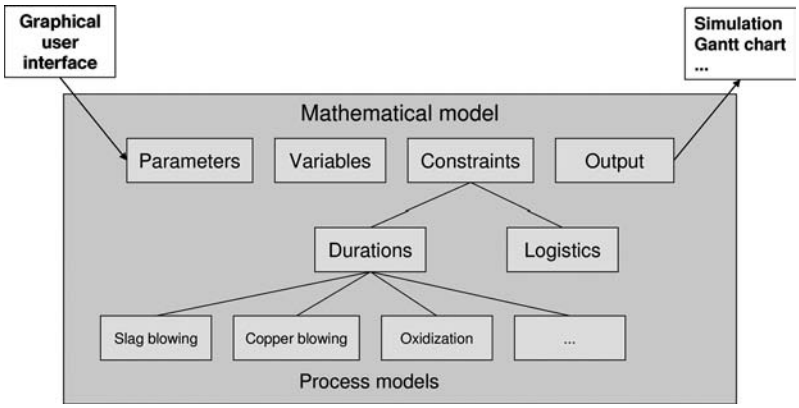

Fig. 5.8 Resulting mathematical model structure.

generates a description of the detailed crane movements needed, in order to check that the production schedule is also realizable at the plant floor from a logistics perspective. A graphical user interface (GUI) provides access to all components, including screens for the configuration of the model and the optimization. The output of a scheduling run is presented as a text-based report, as well as a graphical visualization of the optimal schedule. All planning results are available throughout the plant via HTML pages, which can be displayed using standard technology. Thus, everyone at the plant can view the most recent planning results and receives an update notification after each change.

The crane movement simulation covers the simulation of all required transportation actions during the processing and computes an optimized crane assignment to each job. Based on the plant layout and on information on the converter and anode furnace cycles, different classes of transportation jobs (emptying first converter, filling second anode furnace, waiting, etc.) are defined. These are connected to detailed information on source and target as well as duration information for a transportation job. Event-based simulation is then used to validate the feasibility of the crane schedule. An example of a resulting crane plot, showing the position of both cranes at each time is shown in Figure 5.9. In the figure, for example the filling of converter 2 (C2), performed by both cranes, takes place between 17:15 and 18:00.

There are several reasons for separating the crane planning problem from the process scheduling. The first and maybe the most important reason is that a robust mathematical programming approach can ensure the best solution quality that tightly combines the recipe optimization with production scheduling: this is where the impact of optimization is the highest. For providing the maximum flexibility to the decision making, a continuous-time approach has shown to be the most beneficial with regard to speed versus quality. On the other hand, an MILP model for a crane planning problem with around 300 jobs, each job containing on average 3 stop-points and the movements in between, turned out to be intractable. The fact that the cranes cannot cross but must always keep a minimum distance would

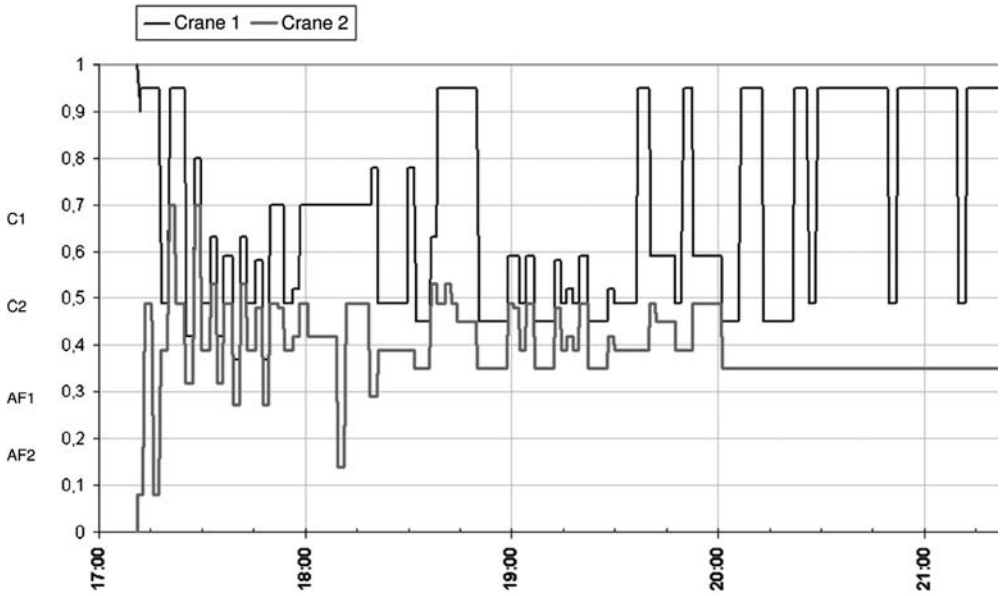

Fig. 5.9 Crane plot diagram.

advocate a discrete-time approach, with a time discretization of (max.) 1 minute. Such a model would involve hundreds of thousands of binary variables, which at least today is not solvable within the given time frame.

Instead, the chosen approach allows the user to work with approximations of duration times needed for each crane job, which are then taken into consideration in the scheduling model. Since most of the jobs are related to a specific production step, sufficient time is reserved for the crane operations, which creates a natural link to the crane planning problem. Using average times in the MILP scheduling model is practical, since the detailed movements and the effect of having two cranes interacting cannot be exactly measured without a detailed model. If it turns out that the crane jobs cannot be finished within the required time frame, which causes a delay in the production schedule, the user can increase the approximate timings for the crane jobs and do a reschedule until the two subproblems match. This parameter tuning must be done both for the simulation and the optimization part to ensure that the complete solution also corresponds to the underlying process.

5.5 Results

The presented solution approach is unique, since it performs a full schedule optimization, simultaneously taking into account equipment availability, process sequencing, material amounts (recipes) based on the underlying chemical reactions,

Table 5.1 Problem complexity.

Constraints	Variables	0–1 Variables	CPU time
984	750	84	< 5 s

major recycling streams and maintenance actions needed. It also reserves sufficient time for the corresponding crane actions.

A typical scheduling run is done for six batches, covering the next 20–35 hours of production. A longer planning period would not make sense as it becomes almost impossible to predict the copper content, e.g., in matte. Including a good estimate of these parameters is very important for the recipe generation. In Table 5.1, a typical problem size and the resulting optimization time is shown.

The crane simulation is also performed within a few seconds. The size and the complexity depend on the problem instance. Since the number of batches is always fixed, the main factor affecting the number of binary variables is the number of maintenance jobs. It should be mentioned that each schedule optimization run also considers two previous batches that are already in production from a resource availability perspective. The initial situation based on the previous batches, defines the complexity. If the production is far from the ideal production cycle, the flexibility may be very low and the main task of the optimization is to increase the total throughput as fast as possible. At this point, the schedule is very sensitive to additional disturbances, which may directly affect the throughput. However, when an optimal production cycle has been reached, a rescheduling optimization may use the existing flexibility (for instance time buffers between the most critical steps) to minimize or to eliminate the throughput decrease caused by disturbances.

The result of the optimization can also be represented as a table. In Table 5.2, an illustrative example with batch-related information is shown (artificial data).

The solution developed is able to solve the scheduling problem very efficiently, resulting in good and realistic schedules. Of course, the solution quality depends to a great deal on how well the parameter estimation matches with the production process. More illustrations on the solution can be found in [5].

5.6 Conclusions

The end customer's main benefits after commissioning of the solution described can be summarized as follows: optimal production schedules with optimal batch recipes are available on demand within seconds, better overall process coordination and visibility of the process and faster recovery from disturbances through efficient scheduling. This leads to a significant increase in plant throughput and revenues.

The production planning and scheduling solution presented here showed that systematic decision support tools not only provide an increase of production or

Table 5.2 Example of results

Batch 1	
Copper content (matte)	67.8%
Optimal recipe	
Copper amount	257 t
Initial scrap	17,5 t
Scrap from ladles	24 t
Slag low	98 t
Slag high out	40 t
Cooling material and scrap	110 t
Solid sulfur out	55 t
SO_2 out	34800 Nm3
Detailed timings	
Charging	17:15–18:02
Slag blowing	18:02–19:40
Copper blowing	19:40–23:13
Emptying	23:13–23:52
Anode furnace	23:52–01:52
Casting time	01:52–04:57
Batch 2	
Copper content (matte)	68.5%
...	

profitability, they also enable a more efficient internal communication between different production units and provide a standardized framework for the planning activities, which can have a huge impact on the productivity. The novel solution concept allows its users to work more efficiently and makes the planning activities more concrete and controllable.

Acknowledgements

The authors would like to acknowledge Dr. Pousga Kaboré's contribution in performing the fundamental work in the chemical process modeling. They also want to thank Mr. Markus Zuber at Norddeutsche Affinerie, Hamburg, for sharing his expertise and providing deep insights in the copper production process.

Nomenclature

Sets, corresponding index

S^C, s	Production stages for converting
S^A, s	Production stages for anode furnace operations
P, p	Batches
E, e	Processing equipment

Parameters

$\Delta_{p,s}$	Time needed between two production steps
Δ_{cr}	Time buffer reserved between crane operations
$\eta_{cu,f}$	Targeted copper concentration after slag blowing
$\eta_{cu,x}$	Copper content of component x
M	Large *big-M* number used to relax some constraints

Variables

$t_{p,s}^{CSB}$	Start-time of a converter production step
$t_{p,s}^{CSE}$	End-time of a converter production step
$t_{p,s}^{ASB}$	Start-time of an anode furnace production step
t_{p}^{CB}	Start-time of the converting
t_{p}^{CE}	End-time of the converting
$T_{p,s}$	Duration of a production step
δ_{cr}	Slack variable for relaxing a crane overlapping constraint
M_x	Mass of component x
$X_{p,e}$	Allocation of a batch p on equipment e

References

1 Pradenas, L., Fernández, R., Parada, V. *et al.* (2003) A solution to the copper smelter scheduling problem, in *The Hermann Schwarze Symposium on Copper Pyrometallurgy*, vol. IV, Pyrometallurgy of Copper (eds C. Diaz, J. Kapusta and C. Newman), The Canadian Institute of Mining, Metallurgy and Petroleum, Montreal, Quebec, Canada, pp. 351–357.

2 Deutsches Kupferinstitut (1997) Kupfer: Vorkommen, Gewinnung, Eigenschaften, Verarbeitung, Verwendung, available at: http://www.kupfer-institut.de.

3 Pinedo, M. (2001) *Scheduling: Theory, Algorithms, and Systems*, Prentice Hall, New York.

4 ILOG Inc. (2004) *ILOG CPLEX 9.0 User's Manual*, ILOG Inc., Mountain View, CA, USA.

5 Harjunkoski, I., Beykirch, G., Zuber, M. and Weidemann, H.J. (2005) The process "copper": copper plant scheduling and optimization. *ABB Rev.*, **4**, 51–54.

6
Stochastic Tools in Supply Chain Management

Rudolf Metz

6.1
Introduction

Logistic decision making often faces random demand. Random demand causes a queue of open orders if the demand is higher than the production or a queue of products in the warehouse if it is lower than the production. Quoting from Kulkarni [9]: "Queues ('or waiting lines') are an unavoidable component of modern life. ... Modern manufacturing systems maintain queues (called inventories) ... throughout the manufacturing process. 'Supply Chain Management' is nothing but the management of these queues!"

This article gives a short introduction to methods and tools based upon stochastic models that are applicable in supply chain management in order to give the reader a flavor of the potential of such methods. Typical terms we will deal with are service level, lot size, and production capacity.

The stochastic tools used here differ considerably from those used in other fields of application, e.g., the investigation of measurements of physical data. For example, in this article normal distributions do not appear. On the other hand random sums, invented in actuary theory, are important. In the first theoretical part we start with random demand and end with conditional random service which is the basic quantity that should be used to decide how much of a product one should produce in a given period of time.

Especially in the process industries various stochastic methods can be applied to cope with random demand. In many cases, random demands can be described by probability distributions, the parameters of which may be estimated from history. This is not always possible, the car industry is an example. No two cars are exactly the same and after a few years there is always a new model which may change the demand pattern significantly.

Section 6.2 introduces the most important concepts in the description of random demand. Section 6.3 deals with random service, which results from random demand and production planning. Section 6.4 discusses the optimal planning of the production given specified random demands and known stock levels. Section 6.5 illustrates the theory by a few examples of the results applied to case studies.

Section 6.6 outlines the implementation of the optimization in the software-tool BayAPS-PP and its integration in the software systems doing the daily business.

This article is only an overview. Therefore the references contain Brüll and Metz [8] and Metz [10] for an overview about supply chains and related planning systems in the chemical industry.

The books of Bauer [1], Fisz [3] and Gnedenko [4] are textbooks about stochastic. Beichelt [2] and Gross and Harris [5] cover especially stochastic processes including queuing processes.

Kulkarni [9] and Tempelmeier [11] contain applications of stochastic methods in supply chain management. Rinne [6] is always helpful for quick information about the many common used density functions and the relations between them.

6.2 Random Demand

In this section we present the basic definitions and facts about random demand. In order to keep this paper self-contained we start with elementary definitions and facts.

Three concepts are important: random demand, random sums and conditional random demand.

6.2.1 Densities and Distributions

In applications it is often a matter of convenience if random demands are modeled by a continuous or by a discrete random variable. In both cases we require that a random demand can not be negative. In this section we assume continuous demands and assure the reader that all formulas have a straightforward extension to the discrete case. Therefore we usually skip the adjective *continuous*.

6.2.1.1 Definition: Continuous Density Function

A *demand density* is by definition a continuous density function defined over the set of all nonnegative numbers, thus demands are always positive or zero.

Generally the variance or even the mean of a demand density function may not exist. However in practical applications one usually assumes that they do exist.

6.2.1.2 Definition: Mean and Standard Deviation

The *mean* μ, the *variance* σ^2 and the *standard deviation* σ *of a demand density* δ are, assuming their existence, defined as

$$\mu(\delta) := \int_0^\infty t\delta(t)dt$$

$$\sigma^2(\delta) := \int_0^\infty (t-\mu(\delta))^2\delta(t)dt = \int_0^\infty t^2\delta(t)dt - \mu(\delta)^2$$

and

$$\sigma(\delta) := \sqrt{\sigma^2(\delta)}$$

The definitions above fit roughly with the often used formulation "the demand is $\mu \pm \sigma$."

6.2.1.3 Mean and Variance

The following formula is used in proofs and calculations. The integral on the left hand side of the equation is often called the second moment of δ

$$\int_0^\infty t^2 \delta(t)\, dt = \mu^2(\delta) + \sigma^2(t)\, dt$$

6.2.1.4 Convolution

The density function of the sum of two independent continuous random variables is computed by the convolution of the two probability densities. Loosely speaking, two random numbers are independent, if they do not influence each other. Unfortunately, convolutions are obviously important but not convenient to calculate.

Let δ, ϕ be demand densities. Then the *convolution of* δ *and* ϕ is

$$\delta * \phi(x) := \int_0^x \delta(t)\phi(x-t)dt$$

The integral above sums up the probabilities for all cases in which the outcome of the random number associated with $\delta * \phi$ is x and that is exactly when the outcome of δ is t and the outcome of ϕ is $x - t$ for some time t.

6.2.1.5 Mean and Standard Deviation of Convolutions

The following statement is the basis of risk management, because it says that the combined standard deviation grows slower than the sum of random numbers.

Let ϕ_1, ϕ_2 be densities with means μ_1, μ_2 and standard deviations σ_1, σ_2. Then the convolution $\phi_1 * \phi_2$ has mean $\mu_1 + \mu_2$ and standard deviation $\sqrt{\sigma_1^2 + \sigma_2^2}$.

The standard deviation can be used as a measure of risk. For example, if we pool the *risk* of random demands by several customers in one warehouse, the above formula indicates that the risk increases slower than the mean demand. It is therefore generally a good idea to pool risks.

6.2.1.6 Gamma Density Function

Gamma densities are often a suitable *a priori* choice to describe random demands, see, e.g., Tempelmeier [11]. The reason for this is that they are not defined for negative numbers (demands are always positive) and that they are uniquely determined by their (positive) mean values and standard deviations. A further important property is that gamma densities are infinitely divisible, which means, for example, that there is a straightforward calculation of the daily demand from the weekly random demand.

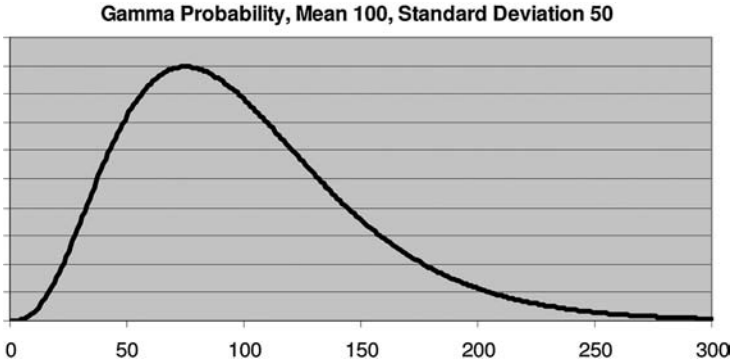

Fig. 6.1 Gamma probability density.

The *gamma density with parameters* $\lambda > 0$ *and* $c > 0$ is defined as $g(x) := \frac{1}{\Gamma(c)} \lambda (\lambda x)^{c-1} e^{-\lambda x}$ for $x > 0$.

Γ is the so-called gamma function, a smooth extension of the function $n \to (n-1)!$ to all positive real numbers.

Figure 6.1 shows the gamma density with mean 100 and standard deviation 50. It shows that gamma densities are skewed, i.e., the *typical* outcome is different from the mean.

The documentation of some software products contain the formulas for gamma densities and distributions, but do not give the connection between the parameters and mean and standard deviation. This connection is made clear by the formula below.

6.2.1.7 Calculation of the Parameters of a Gamma Distribution

The gamma density with parameters λ and c has mean $\mu = \lambda^{-1}c$ and variance $\sigma^2 = \lambda^{-2}c$. Conversely, $\lambda = \frac{\mu}{\sigma^2}$ and $c = \frac{\mu^2}{\sigma^2}$.

6.2.2 Demand on the Time Line

It is important to keep in mind that demands occur on the time line. If a reasonable minimal time interval is chosen, it is in many cases justified to consider the demands in these intervals as independent and identically distributed random variables. This means for example that the demand per week is the iterated convolution of the daily demand. A large customer base is a good indicator of independent random demand in different time intervals.

One advantage of gamma densities is their consistency with the assumptions above.

6.2.2.1 Convolutions of Gamma Densities

The convolution of two gamma densities with parameters λ, c_1 and λ, c_2 is a gamma density with parameters $\lambda, c_1 + c_2$.

If, for example, a product has independent daily random demand given by the gamma density with parameters λ, c then the weekly demand is described by the gamma density with parameters λ, $7c$ or λ, $5c$ depending on the behavior of demand at weekends. In many cases gamma densities are a good approximation of random demand. If, however, for a product the amounts vary significantly from order to order, it may be necessary to consider the demand as a random sum.

6.2.3 Demand as a Random Sum

Compound densities, also called random sums, are typically applied in modeling random demands or random claim sums in actuarial theory. The reason for this is simple. Assume that customers randomly order different quantities of a product. Then the total quantity ordered is the random sum of a random number of orders. The conversion to the actuarial variant is obvious: The total claim is made from the individual claims of a random number of damage events. Zero quantities usually are neither ordered nor claimed. This leads to the following definition.

6.2.3.1 Definition: Compound Demand Density Function

An *order density* is a demand density δ with $\delta(0) = 0$. The *number* of orders per interval can be described by a *discrete* density function η with discrete probabilities defined for nonnegative integers 0, 1, 2, 3, ... The resulting (η, δ)-*compound density function* is constructed as follows: A random number of random orders constitute the random demand. The random number of orders is η-distributed. The random orders are independent from the number of orders, and are independent and identically δ-distributed.

Let us now look into two examples to get an impression of a compound demand. We will look at compound Poisson distributions. Poisson distributions describe the random number of independent events per period, for example the number of customers with nonzero demands in a certain week from a large customer base.

In the *first example* a customer orders 1 unit with 70% probability and 5 units with 30% probability. The number of orders per period is Poisson distributed with mean 4. Figure 6.2 shows the resulting (discrete) compound Poisson density and the cumulated distribution and their gamma approximations.

It can be seen that the gamma function approximation of the compound density function is not accurate. However, as seen in Figure 6.3, the gamma approximation of the distribution function is sufficiently good, especially in the interval between 90 and 100% probability which provides the inventory needed to be able to serve at least 90% of the demand.

In the *second example*, each customer orders 1 unit with 60% probability, 2 units with 30% probability and 5 units with 10% probability. The number of orders per period is again Poisson distributed but now with mean 5. This results in the distribution shown in Figure 6.4.

In the second example the gamma approximation is accurate enough for practical purposes. The reason for this is that the order pattern is less extreme and that the order frequency is higher.

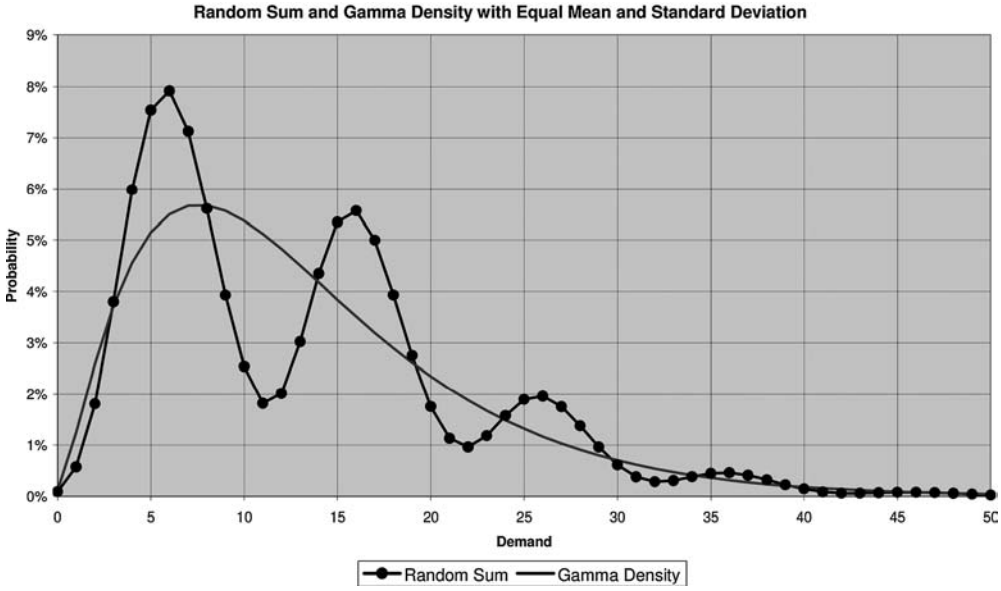

Fig. 6.2 Compound and gamma demand I (density).

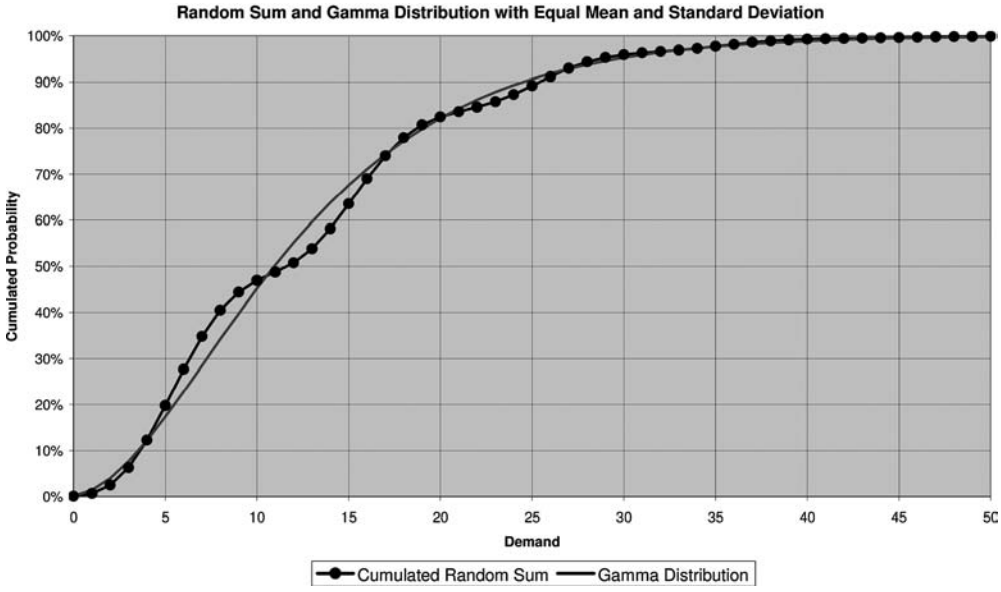

Fig. 6.3 Compound and gamma demand I (distribution).

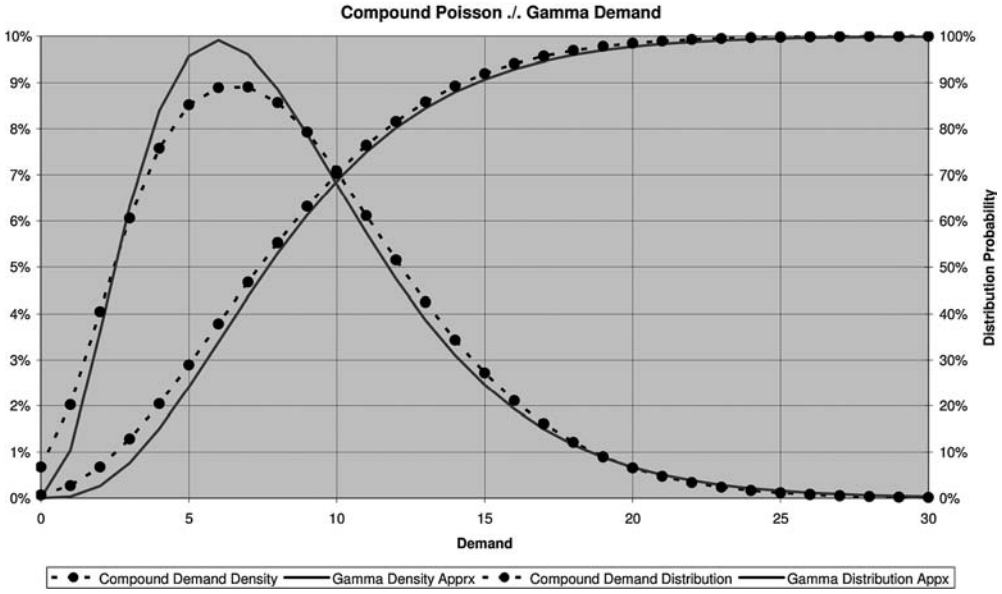

Fig. 6.4 Compound and gamma demand II.

It is therefore a good idea to look into the historic demand data at the beginning of a project and to decide whether a gamma approximation is sufficient or whether the more elaborate compound approximations should be used.

6.2.3.2 **Mean and Variance of Compounds**

There is a formula by which the mean and the variance of a compound density can be calculated from its constituents.

With the same notation as in Section 6.2.3.1, the mean $\mu_{\oplus}$ *and variance* $\sigma^2_{\oplus}$ *of a* (η, δ)*-random sum are*

$$\mu_{\oplus} = \mu(\eta)\,\mu(\delta) \text{ and } \sigma^2_{\oplus} = \mu(\eta)\,\sigma^2(\phi) + \sigma^2(\eta)\,\mu^2(\delta)$$

Note that the mean $\mu(\delta)$ has a proportional influence on the standard deviation of the total ordered amount. The tendency is intuitive: The formula means that only a few large orders require more safety stock to cover random demand variations than many small orders. It is therefore a good idea to measure the mean and the variance of demands twice. First in the usual way as mean and variance of a sequence of, say weekly, figures and then by analyzing the orders of a historic period and applying Eq. (6.6). The comparison of the two results obtained often provides insight in the independence and the randomness of the historic demand. If the deviation of the two mean values and/or two variances is large then the demand can not be considered as a random sum. A reason could be, for example, that the demand

results from a combination of the demand of a large customer base and the large but sporadic demand of one special customer.

6.2.4 Sporadic Demand

In this overview article we only briefly discuss sporadic demand. We also do not deal with leaking demand or the like. However, the concepts and methods presented in the following sections for regular (i.e., plain) random demand can be extended to sporadic, seasonal and other sorts of demand. These extensions are natural but require much more difficult calculations.

A random demand is *not* sporadic with respect to a period length if we can expect that the outcome for a period is almost surely greater than zero. In other words $\delta(0) = 0$ which does not mean that the outcome 0 is impossible: if you throw a dice with infinitely many faces then any outcome has probability 0.

Definition: Regular and Sporadic Demand
A continuous demand density δ is called *regular* if $\delta(0) = 0$. Otherwise it is called *sporadic*.

A frequently used model of sporadic demand is its description by two densities, a demand density δ which describes the demand in the periods where it is not zero, and a discrete density τ which gives the integer distance between periods with positive demand.

6.2.5 Conditional Demand

The term conditional demand is derived from the concept of conditional density and simply means the following: Let a random variable be described by its density. What do we know about the outcome of the random variable, if we know that the outcome will be not less than a certain value? The answer is, roughly speaking, that this additional knowledge in typical cases increases the mean value and reduces the standard deviation. This effect can be calculated precisely.

Translated to a logistic application this means that orders that were already received increase the mean demand and reduce its uncertainty, i.e., its standard deviation.

Figure 6.5 below shows the construction of the conditional demand density from the original demand density, assuming that 50 units have already been ordered.

The left section of the density function in Figure 6.5 is cut because the demand is below the already ordered amount with probability 0. The remaining area beneath the density function is normalized to the value of 1. We will describe the calculation and give examples in this section.

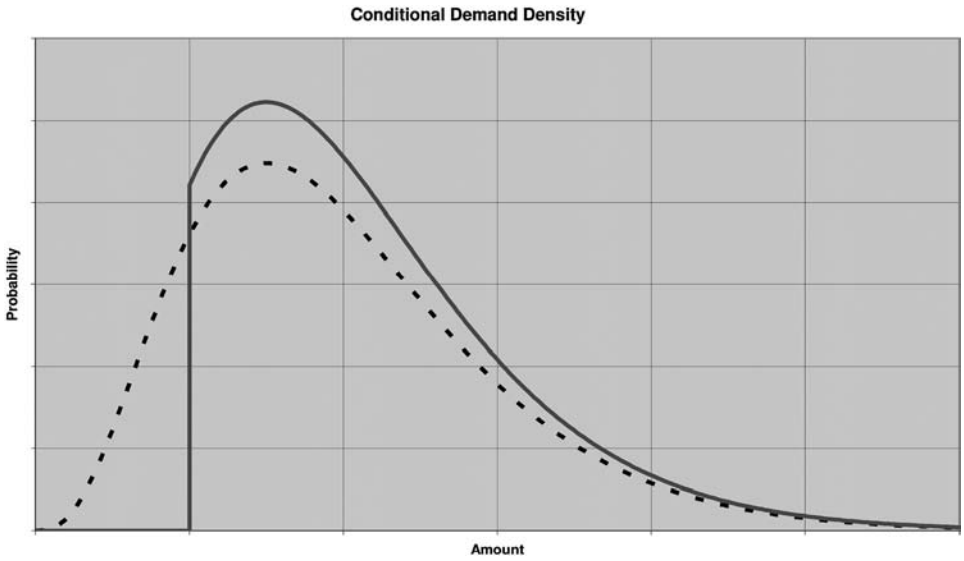

Fig. 6.5 Conditional demand.

But one has to act with caution. If we look at one product only, the lead times are important, because a number of last minute orders are the last orders and will not increase the expectation to even more orders.

Especially in an industrial environment conditional demand is an important concept. This is because orders usually allow for some delivery time. This delivery time is in many cases long enough to be taken into account in production scheduling. As the deadline for orders for a certain date of delivery comes closer one can compare the original forecast with the orders that were already received. It is intuitively clear that if many orders were already received this implies a somewhat increasing forecast with less uncertainty. Sometimes orders that were already received can be used to automate the forecast to a certain extent because one knows that *usually* 25% are ordered four weeks in advance, 50% are ordered two weeks in advance and the like. We assume here that there is a valid latest forecast at the point in time where a decision on the next production volume is necessary and the orders that were already received are taken into account at that fixed point in time.

Of course, when the deadline of delivery has been reached, the demand is no longer uncertain or random, but it is from then on exactly the amount that was already ordered.

By intuition we know the following: Orders that were already received usually reduce the standard deviation of the original forecast significantly and increase the mean somewhat as long as the orders that were already received are below the mean of the original forecast. *Usually* means, as we will see, if the standard deviation is below the mean value.

We only present the continuous version of the precise formulas for conditional demand, which makes this intuitive reasoning precise.

6.2.5.1 Conditional Demand Density

Assume that δ is a continuous demand density with corresponding distribution Δ. Assume further that there are already orders of amount r. Then the resulting conditional density $\delta_{r|}$ is given by

$$\delta_{r|}(x) = 0 \text{ if } x < r \text{ and } \delta_{r|}(x) = \frac{1}{1-\Delta(r)}\delta(x)$$

6.2.5.2 Mean Conditional Demand

With notation as in Section 6.2.5.1, we have $\mu\left(\delta_{r|}\right) = \frac{1}{1-\Delta(r)}\left(\mu(\delta) - \int_0^r t\delta(t)\,dt\right)$.

The mean conditional demand is not smaller than the unconditional mean demand.

6.2.5.3 Standard Deviation of the Conditional Demand

With notation as in Section 6.2.5.1, we have $\sigma^2\left(\delta_{r|}\right) = \frac{1}{1-\Delta(r)}\int_r^\infty t^2\delta(t)\,dt - \mu\left(\delta_{r|}\right)^2$.

Note that $\int_r^\infty t^2\delta(t)\,dt = \int_0^\infty t^2\delta(t)\,dt - \int_0^r t^2\delta(t)\,dt = \sigma^2 + \mu^2 - \int_0^r t^2\delta(t)\,dt$.

Therefore, to apply the theory one needs a software library which contains functions to compute $r \to \int_0^r t\delta(t)\,dt$ and $r \to \int_0^r t^2\delta(t)\,dt$ to calculate conditional random demand.

6.3 Random Service (and Shortage)

In the first section we discussed random demand. Then we calculated the conditional demand and now finally we define conditional random service and conditional random shortage. These concepts are very important for optimization of service levels under capacity constraints.

Random service levels result when random demands meet available inventories. Throughout this chapter we assume that the available inventory is not random but has a known value. This is justified because in many cases the production process is almost deterministic when compared with the varying demand.

In the process industries, service is often measured by beta service, which reflects that a customer will accept a partial delivery if the full ordered amount is not available in time. Due to the fact that we always mean beta service level in this chapter, we omit the word beta from further descriptions.

6.3.1 Basics

The following definitions result from the simple observation that the minimum of demand and available stock will be always delivered and therefore can be taken to measure the service level.

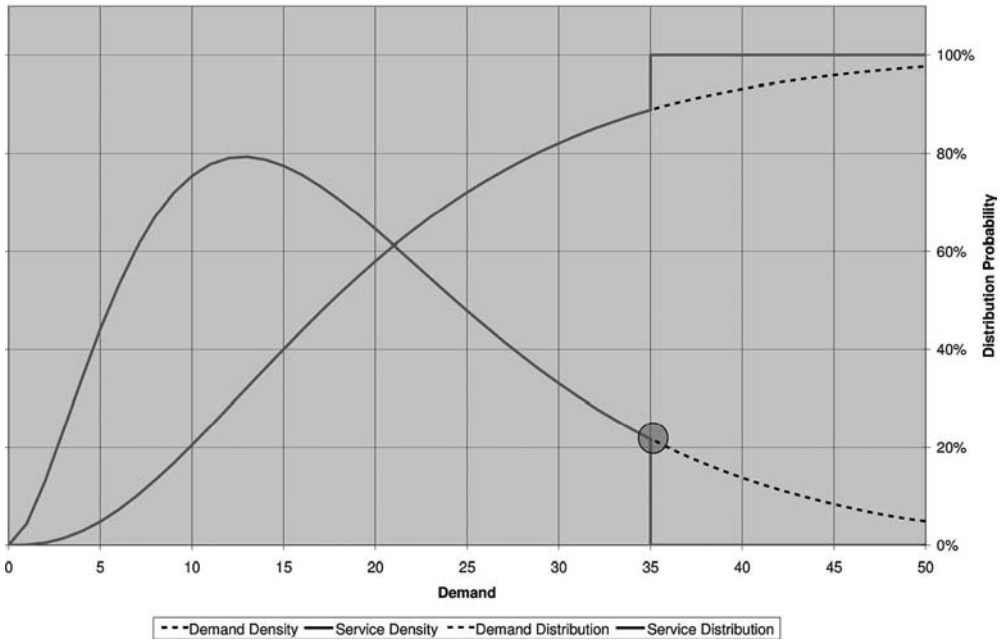

Fig. 6.6 Demand and service.

6.3.1.1 **Definition: Service Density**

Assume that δ is a demand density with distribution Δ and s is the available inventory. Then the corresponding *service density* is $\delta_s(x) := \delta(x)$ if $x \langle s$ and $\delta_s(s) := 1 - \Delta(s-1)$ for discrete δ and $\delta_s(s) := 1 - \Delta(s)$ for continuous δ otherwise.

The service density defined above and illustrated in Figure 6.6 is a real variable that describes the distribution of the corresponding random variable. The density as a function is not continuous because it has a point mass at $s = 35$, the available inventory in the example, because the service is always exactly s if the demand is at least s. As a result, the service level distribution jumps to the value 100% at 35 because with 100% probability the service is 35 or less.

6.3.1.2 **Definition: Shortage Density**

Assume that δ is a demand density and s is the available inventory. Then the corresponding *shortage density* is $\delta_s^-(x) := \delta(s+x)$ if $x > 0$ and $\delta_s^-(0) := \Delta(s)$.

This simply means that there is no shortage if the demand is below or equal to the inventory. Similar to the demand density, a shortage density has as well-defined mean and standard deviation. These values however depend on an additional parameter, the available inventory (see Figure 6.7).

Of course random service and shortage are not independent as the following simple observation shows.

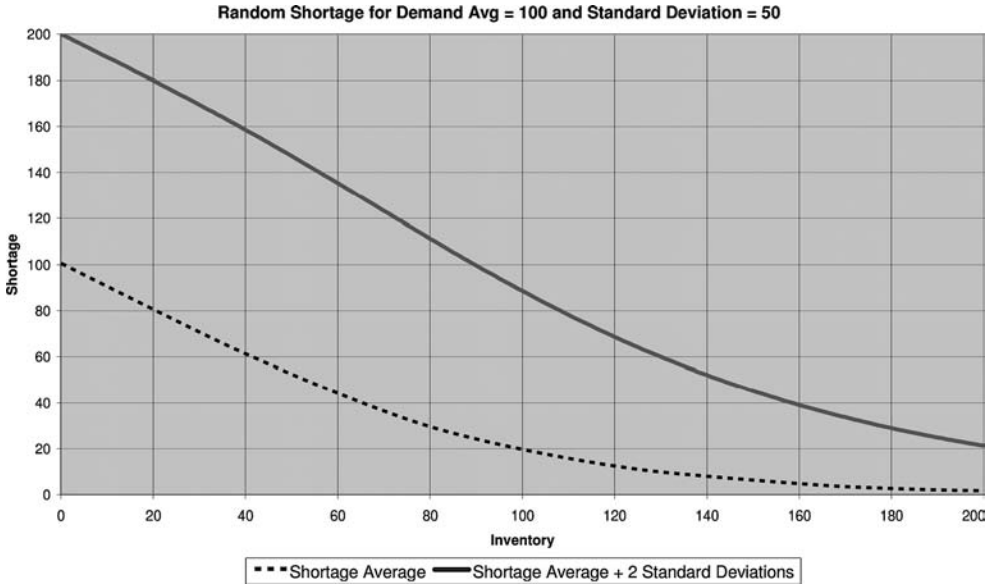

Fig. 6.7 Demand and shortage.

6.3.1.3 **Service and Shortage**

If the shortage is greater than zero, the service is equal to the inventory. If the shortage is zero then the conditional demand is at most the inventory.

Mathematically speaking, we only look at the two limiting distributions of a two-dimensional distribution that describes the service and the shortage.

6.3.2 Conditional Random Service and Shortage

This section relates random services and random shortages to conditional demands as defined in Section 6.2.5. Conditional random service is the crucial quantity that has to be calculated when a safety stock level has to be determined. Conditional random service results from three quantities: a demand density δ, an already ordered quantity r and an available inventory s. From these parameters we obtain two new densities, the conditional service density $\delta_{+,r,s}$ and the conditional shortage density $\delta_{-,r,s}$.

6.3.2.1 **Conditional Service Density**

Assume that δ is a continuous demand density, r units are already ordered, and s is the available inventory. Then the corresponding conditional service density $\delta_{+,r,s}$ is given by case 1, $r \geq s$:

$$\delta_{+,r,s}(s) = 1;\ \delta_{+,r,s}(x) = 0 \quad \text{otherwise}$$

case 2, $r < s$:

$$\delta_{+,r,s}(x) = 0 \text{ if } x < r;\ \delta_{+,r,s}(x) = \frac{\delta(x)}{1-\Delta(r)} \text{ if } r \le x < s$$

$$\delta_{+,r,s}(s) = \frac{1-\Delta(s)}{1-\Delta(r)};\ \delta(x) = 0 \text{ if } x > s$$

The conditional service density derived from a continuous demand density is continuous almost everywhere but has one exceptional point which carries a discrete probability mass: The probability that the whole inventory s goes to service is the integral of the conditional demand density from s to ∞. In other words, the service is s if the demand is s or above.

The conditional shortage x and the demand $x + s$ have the same density ($x > 0$). The conditional shortage is zero with the same probability for the conditional demand to be at most s.

6.3.2.2 Conditional Shortage Density

Assume that δ is a demand density, r units are already ordered, and s is the available inventory. Then the corresponding conditional shortage density $\delta_{-,r,s}$ is given by $\delta_{-,r,s}(x) = 0$ if $x \le s$; $\delta_{-,r,s}(x) = \delta_{|r}(x + s)$.

As already mentioned one needs the two integral functions $s \to \int_0^s t\delta(t)dt$, $s \to \int_0^s t^2\delta(t)dt$ to deal with conditional random service and shortage. Because the formulas for conditional service or shortage become a bit lengthy, we only mention the formula for the conditional service mean.

6.3.2.3 Conditional Service Mean

For a demand density δ with distribution Δ, already ordered amount r and inventory s the mean service is $\mu(\delta_{+,r,s}) = s$ if $s \le r$ and $\mu(\delta_{+,r,s}) = \frac{1}{1-\Delta(r)}\left[\int_0^s t\delta(t)dt - \int_0^r t\delta(t)dt + s(1-\Delta(s))\right]$.

What does this mean? If the inventory s is at most the already ordered quantity the whole inventory s is delivered. Now assume $s > r$.

$$\mu\left(\delta_{+,r,s}\right) = \frac{1}{1-\Delta(r)}\left[\int_0^s t\delta(t)\,dt - \int_0^r t\delta(t)\,dt + s(1-\Delta(s))\right]$$

$$= \frac{1}{1-\Delta(r)}\int_r^s t\delta(t)\,dt - s\frac{1-\Delta(s)}{1-\Delta(r)}$$

The first term describes the situation when the available inventory is at least the conditional demand. The last term follows from these observations: With total probability $1 - \Delta(s)$ the demand is greater than s. With total probability $\frac{1-\Delta(s)}{1-\Delta(r)}$ the conditional demand is greater than s. In these cases there is a shortage and we

deliver only s. The term $\frac{1-\Delta(s)}{1-\Delta(r)}$ is the probability that a shortage occurs, irrespective of its size.

6.3.2.4 Service Level

The mean service resulting from a random demand and an inventory is often divided by the mean demand and then called the service level. It is always within 0 and 1 and is usually written as a percentage value. Because the usage of this indicator implicitly assumes that partial deliveries are allowed, it is known as β-service level in order to distinguish it from the α-service level, which denotes the proportion of completely serviced orders.

6.4 Optimization of Service

This section deals with production lines for more than one product. In the process industries it is often a problem to assign the capacity of one production line to several products, all or some of which have uncertain demand. We want to optimize the overall service level for such a production line.

6.4.1 First Example

Even if only one product is considered, there is some need for the optimization of the production strategy, as the following simple example shows. In this example we compare two products that only differ in their order pattern.

Consider two products A and B with the same demand of 20 units per time period, the same buffer size of 60 units, and the same production speed and set up time. The only difference is that product A is only sold in single units, but product B has 80% of orders of 1 unit and 20% of orders of 10 units. It is intuitively clear that product B will have a lower service level because is has a larger variance of the demand. It is not immediately clear that both products have different optimal lot sizes. The optimal lot size, i.e., the lot size resulting in maximal service, is 20 units for product A which results in a β-service level of 98.6% (Figure 6.8). The optimal lot size for product B is 15 units which results in a not particularly good β-service level of 90.6%. In order to achieve a β-service level of 98.6% one would need a buffer of size 165 units, with a corresponding optimal lot size of 33.

In Figure 6.9, a fixed buffer size for each product is assumed (x-axis) and the resulting lot sizes that give maximum β-service levels and the resulting β-service levels are shown on the left and right y-axes. The discontinuous graph of the optimal lot sizes results from the fact that these are integers, This small example shows that the detailed stochastic description of the distribution of demand has a strong impact on a suitable production environment (i.e., buffer sizes, lot sizes, production capacities) and on the best production strategy, even if the mean values describing the production process and the demand are the same for two different products.

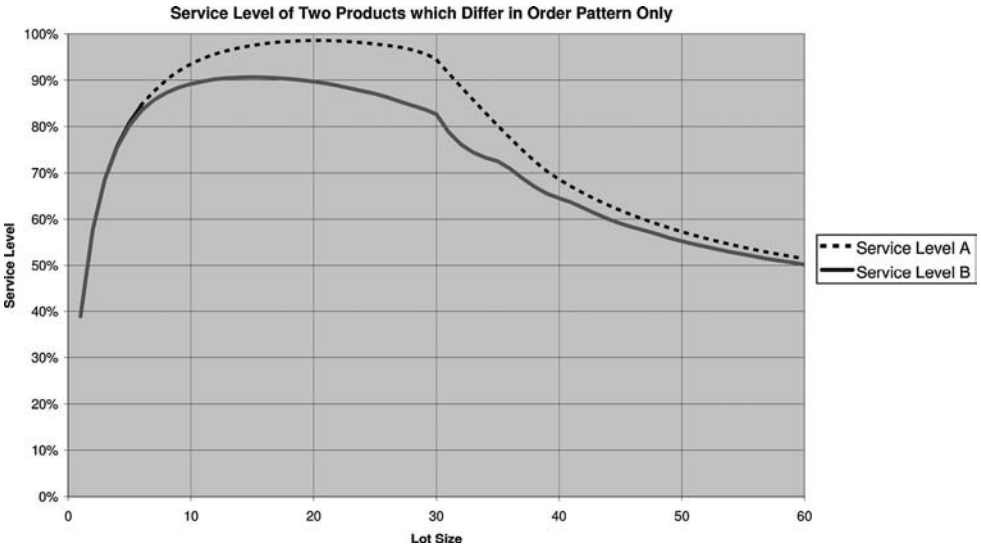

Fig. 6.8 Service level and order pattern.

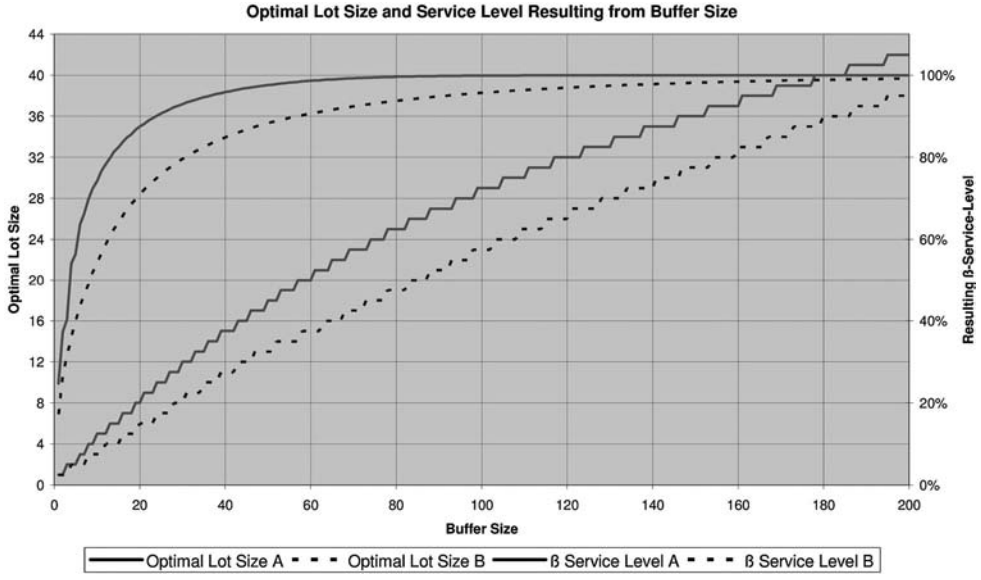

Fig. 6.9 Optimal lot size and order pattern.

6.4.2 Capacity Assignment

An important step in the optimization of production processes is to assign the production capacity in a certain period to the several products that can be produced on a production line. For simplicity, we ignore changeover time here. In this situation, the optimal capacity assignment is defined as follows.

Optimal Assignment
Assume that there are n products with demand densities $\delta_1, \dots \delta_n$, already received orders $r_1, \dots r_n$, and available stock $s_1, \dots s_n$. Assume that $t_1, \dots t_n$ time units are needed to produce one unit of product on a given production train and that $p_1, \dots p_n$ are the marginal profits achieved per additional unit of each product. The optimal capacity assignment is the vector $x_1, \dots x_n$ which maximizes the function $\sum_{i=1}^{n} \mu\left(\delta_{+,r_i,s_i+x_i}\right) p_i$ subject to the constraints $x_i \geq 0$ and $\sum_{i=1}^{n} x_i t_i \leq t$, t being the length of the period.

The sum $\sum_{i=1}^{n} \mu\left(\delta_{+,r_i,s_i+x_i}\right) p_i$ adds the mean marginal profits for all products, each summand resulting from the demand density δ_i, the already received orders r_i and the available stock $s_i + x_i$, where s_i is the product already in stock and x_i is the additional amount that is produced. The first constraint is obvious, the second requests that the production is feasible within a given time period.

6.4.3 The Optimization Problem

As the service level depends on the production capacity and the production strategy, it is obvious that both should be optimized. This means first to choose the optimal capacities for the production lines and the *buffers* (warehouses, tanks, silos) when planning a new factory or an extension.

Once a production system is given, the optimal operational production strategy defines in which sequence the products should be produced next and in which quantities.

The choice of the next product is affected by the changeover rules and costs. Sometimes potential risks to the achievement of the specified quality have to be taken into account. If that is the case one calculates with an average yield of production or the yield is treated as a further random number, which is beyond the scope of the topics presented here.

The problem of splitting the total production into lots or campaigns can be solved by a multiproduct extension of the old and well known Andler or Harris formula [7], which is for one product only. This extension copes with several products on the same production train and various side conditions, and minimizes the lot sizes. It can be shown that lot sizes should be proportional to the square-root of demand, as far as side conditions allow. The demand as well as the lot size can be measured in weight or value. It is important to realize that unnecessarily high lot sizes not only generate inventory cost but also negatively affect the production of more urgently needed products.

The sequencing decisions and the decisions on production volumes should be answered at the latest possible moment, because that is the point in time when the demand forecast is replaced by actual orders to the maximum extent.

In the process industries different products on the same production line are often only variants of a base product with almost the same raw materials needed, e.g., defined by different viscosities or chain lengths of a polymer or products defined by different additives to a base product. This means that planning the procurement of raw materials is (almost) independent from the exact assignment of the production line capacity to the single products. Therefore the decision on the distribution of the capacity among the different products can be made fairly late.

It may however well be that the sequence of products influences the capacity of the production line. A typical class of examples is as follows: The product variants are defined by a small content of an additive. If the exact content increases, the time for product changeover is negligible, however decreasing the content of the additive requires some or even extensive cleaning. In this situation products are often arranged in a cyclic order going from zero to high content and back to zero again, because only this step back to zero needs an extensive cleaning.

6.5 Solution Technique

6.5.1 The Algorithm

The tool BayAPS PP from Bayer Technology Services (BTS) contains algorithms that optimize production scheduling, as described in Section 6.4, for which a patent application has been submitted (file no. 102006040568.4). In particular, the assignment problem defined in Section 6.4.2.1 is solved by a fast algorithm which finds the optimum in a direct approximation without any sort of searching as for example in genetic algorithms. Current data about demand and inventories can be imported from SAP R/3 or any other ERP-system via an interface.

The core algorithm assigns the production capacity to the competing products of the same production line, such that the overall expected sales are maximal. If the capacity is critical this basically means that production capacity is designed to the more likely parts of the uncertain demand. If there is plenty of production capacity, safety stock is allocated reflecting the product-specific uncertainty of demand. When looking at all products, usually the situation is between these extremes and the algorithm provides an optimal compromise.

The key mathematical tool for the solution is the expected marginal service at a given stock s. If $\delta_{r|}$ as in Section 6.2.5.1 denotes the conditional demand density and $\Delta_{r|}$ the corresponding conditional service distribution, then the expected marginal service is simply $1 - \Delta_{r|}$. Indeed, if the stock s is 0 then $1 - \Delta_{r|}(s) = 1 - \Delta_{r|}(0) = 1 - 0 = 1$. In other words, if we add a little bit (mathematically speaking, an infinitesimal small amount) to our stock 0, this little bit is sold

with probability 1. Therefore, at stock 0 the expected marginal service is 1. If however the stock is infinite, the expected marginal sale becomes $1 - \Delta_{r|}(\infty) = 1 - 1 = 0$. Thus the expected marginal service at stock ∞ is zero.

For numerical purposes we the change the meaning of "a little bit" at this point. From now on it is 1 unit defined by whatever is appropriate, it may be kilograms for fine chemicals or time for polymers. The algorithm now investigates kilogram by kilogram or ton by ton of, say, the yearly capacity. Now one can calculate for each product the expected marginal service and one decides to produce one little bit of the product which has that maximal service. This one little bit is added to the stock of that product and the time needed to produce it is subtracted from the available time.

This step is repeated until the available time is totally consumed. At this point one has found the optimal total amount to produce for each product within the planning horizon. Of course the procedure above does not match the actual sequence of production and does not recommend the split of the total amount into several lots. This is done by further steps which are somewhat more technical and not described here.

The actual algorithms refine the procedure described above by multiplying the marginal expected service with the product-specific marginal income and dividing that figure by the time needed to produce the little bit. Therefore, one can optimize the money earned in the given time frame which is equal to the overall service level measured in money rather than weight or units.

6.5.2 Illustrative Examples

The example discussed in the first sequel consists of two pairs of products with inventories, mean demand with standard deviation, open orders, marginal profit, and production times. The basic data for a case without inventories and with equal demands and the solution are given in Table 6.1. The solution to the assignment optimization problem in Section 6.4.2.1 is given in the column Production. In total, 168 (equal to the hours per week) units are produced with equal amounts for equal products, as one would expect.

Table 6.1 Two pairs of equal products

Product	Inventory [1]	Mean demand [1]	Standard-deviation [1]	Open orders [1]	Margin [€ /1]	Production time [h/1]	Production [1]	Total available [1]
A01	0	100	30%	0	1	1	53	53
A02	0	100	30%	0	1	1	53	53
B01	0	100	50%	0	1	1	31	31
B02	0	100	50%	0	1	1	31	31

Table 6.2 Inventory varied

Product	Inventory [1]	Mean demand [1]	Standard-deviation [1]	Open orders [1]	Margin [€ /1]	Production time [h/1]	Production [1]	Total available [1]
A01	20	100	30%	0	1	1	38	58
A02	0	100	30%	0	1	1	58	58
B01	0	100	50%	0	1	1	36	36
B02	0	100	50%	0	1	1	36	36

Table 6.3 Open orders varied

Product	Inventory [1]	Mean demand [1]	Standard-deviation [1]	Open orders [1]	Margin [€ /1]	Production time [h/1]	Production [1]	Total available [1]
A01	20	100	30%	0	1	1	32	52
A02	0	100	30%	0	1	1	52	52
B01	0	100	50%	50	1	1	54	54
B02	0	100	50%	0	1	1	30	30

Now assume we have some inventory of product A01. The result of the optimization is given in Table 6.2.

The algorithm reacts to the situation by producing more of all products. Note that for products A01 and A02 the production is different but the available inventory after the termination of the production process, displayed in the last column, is equal. For B01 and B02 all parameters are still equal.

Now assume that there are already known orders for product B01. The result is shown in Table 6.3.

The algorithm assigns more capacity to product B01 than to B02. Why? The capacity is critical, because only 20 + 168 units of product are available, but the total demand is 400. Under these circumstances the optimal portion of production capacity is diverted from the other products to B01, the demand for which is more certain because of the orders that were already received. For A01 and A02 the same number of available units of product after production is still the same.

Now we assume that with product B02 the producer makes a mere 10% higher profit. We get the results shown in Table 6.4.

As one would expect, now capacity is shifted to B02 from the other products. Now we assume that the profit and the production time of A01 are also increased by 10%.

The result shown in Table 6.5 is that overall less product than before can be produced because production on average over all products became slower. However,

Table 6.4 Margin varied

Product	Inventory [1]	Mean demand [1]	Standard-deviation [1]	Open orders [1]	Margin [€ /1]	Production time [h/1]	Production [1]	Total available [1]
A01	20	100	30%	0	1	1	26	46
A02	0	100	30%	0	1	1	46	46
B01	0	100	50%	50	1	1	52	52
B02	0	100	50%	0	1.1	1	44	44

Table 6.5 Margin and production time changed accordingly

Product	Inventory [1]	Mean demand [1]	Standard-deviation [1]	Open orders [1]	Margin [€ /1]	Production time [h/1]	Production [1]	Total available [1]
A01	20	100	30%	0	1.1	1.1	25	45
A02	0	100	30%	0	1	1	45	45
B01	0	100	50%	50	1	1	51	51
B02	0	100	50%	0	1.1	1	44	44

at the end for A01 and A02 the available inventories after production were the same. That is because the specific profit per unit of time and the random demand were equal.

6.6 Implementation in BayAPS PP

6.6.1 Data Flow and Functionality

Figure 6.10 shows the data flow of the software tool BayAPS PP for optimal capacity assignment for given stochastic demands. Transaction data about demand and inventories is typically imported from SAP R/3 as indicated, production capacity master data and side conditions are stored in the software tool. Forecasts can be taken from a forecast tool or from SAP R/3. The output of the tool is a list of priorities of products and their lot sizes, which are optimal based on the presently available information. Only the next production orders are realized before the computation is repeated, and the subsequently scheduled production is only a prediction.

First of all the *resulting forecast* is calculated, which is initially the conditional random demand resulting from the forecast and the orders that were already received, as discussed in Section 6.2.5. It is then balanced with the current inventory resulting in a forecast of the production demand. There may be periods when

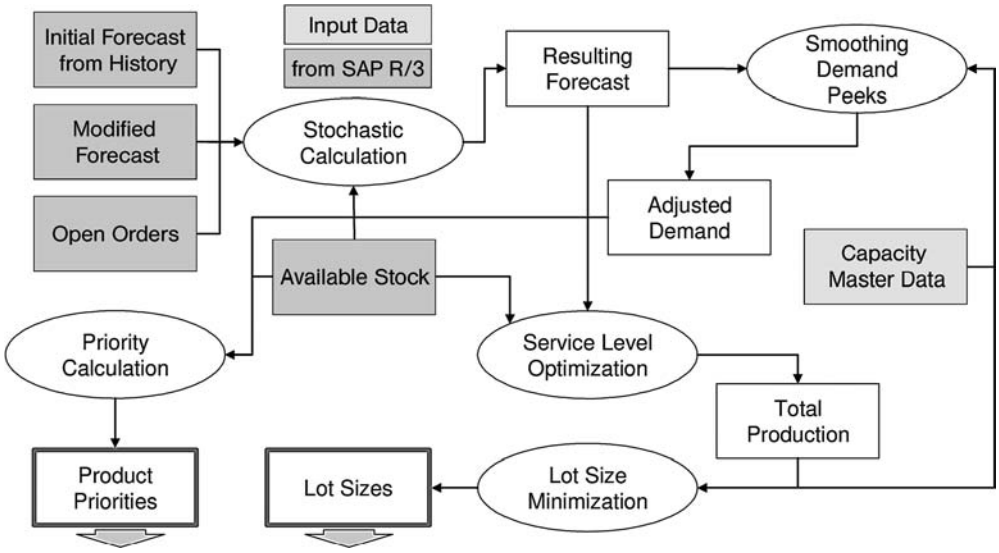

Fig. 6.10 BayAPS PP data flow diagram.

the total demand exceeds the total capacity. Then the software tries to shift the production forward to earlier periods, the result is called the adjusted demand. There are several possible ways to perform this step. BayAPS PP spreads the advanced production over all products with nonzero demand. As a result, the impact on the future production is reduced if a demand changes, because all products have a comparatively small additional adjusted demand.

Then the optimization is performed based on the adjusted demand and the available inventory. The algorithm assigns the total production capacity to the different products such that the total mean conditional service as defined in Section 6.3.2.3 is maximized. The service usually is measured in € of marginal income. This capacity assignment obeys various technical side conditions, which can be specified in the master data as, e.g., product changeover times, product specific production rates of the lines, minimal lot sizes, etc. There are two important rules for product changeovers to choose from. The first rule is flexible changes, which means that for planning purposes all product changeovers require the same amount of time. The second rule is to follow a predefined production cycle as described in Section 6.4.3.

After the total amounts of production for each product have been determined, these amounts are split in the second step into an optimal number of production campaigns for each product according to the relevant constraints and cost.

Finally the algorithm decides what to produce next. If product changes are flexible, this is the product with the shortest coverage, measured by the mean or by the safe coverage. If a product cycle has to be realized then the next product is the next one in the cycle. However, the software recommends skipping this product if the inventory covers the demand until this product appears in the next cycle.

6.6.2 Project Example

In the sections above we explained the stochastic theory behind BayAPS PP and the main influencing parameters. Real life projects always involve additional complications and additional essential constraints have to be met. Without going into the details of the solution, we roughly describe a project example to give the reader a flavor of such additional requirements.

Bayer Technology Services has introduced BayAPS PP in a factory producing about 40 pigment dispersions in 80 different packings on several production lines with a common packing station.

The shift calendar is complicated: Packing is usually done in 8 hours, Mondays to Fridays only, the production running continuously, but product changeovers are only possibly during those 8 hours from Monday to Friday. The reason is that packing and product changes have special personal requirements. Times for planned maintenance can be specified separately for each production line and for the packing station.

All lot sizes have to be divisible by the content of a single batch size in the production.

Frozen zones are *subplans* for actions in the near future which must not be changed in subsequent runs of the planning algorithm. This on the one hand causes planning stability. On the other hand reactions on changing demand are delayed. Because the delivery periods of the raw materials differ, individual frozen zones can be specified by the user. This means that the specified production orders will not change with respect to the lot sizes and the due dates in subsequent new planning runs. A so-called semifrozen sequence can also be specified. This means that due dates may be postponed and lot sizes may be reduced, due to fact that such changes will not cause serious problems with critical raw materials. However the sequence of products must not be changed for frozen or semifrozen production orders in order to reinforce planning stability with respect to product changeovers.

BayAPS PP has been extended to cover the requirements that were described above. Last but not least the easy to use Excel interface of BayAPS PP led to an interesting idea of a user: The end-user can create a report writer based on simple Excel formulas. These Excel formulas such as VLOOKUP reference and combine information from several BayAPS PP sheets and present it to the user as a list in the *traditional* layout known to the personal in the production plant.

6.6.3 Benefits and Outlook

The software tool performs an optimal calculation of lot sizes incorporating uncertain demand from forecasts or history as well as up-to-date inventory and open order data. The effort for the regular user is negligible because of the interface to SAP R/3. Various technical constraints can be included. Specific training to use the software is not necessary because it looks like the familiar Excel format to the

user. These are three of the reasons why in all projects so far the *decisions* proposed by the tool were accepted, despite the fact that they differed by about 10% from those used in manual production scheduling. A second benefit is that many typical *what if* questions can be answered almost instantly. For example: "What happens, if demand increases by *x*%?" or "What is the effect, if our product mix changes in a certain way?"

Currently Bayer Technology Services considers extending the software BayAPS PP to compute the optimal split of a product between several production lines or factories that can produce it. This split is influenced by uncertain demand with different characteristics in different regions, different cost of transport and different production cost in the factories. This means that different marginal incomes for the same product occur depending on the place of production and/or the customer group which receives it. The mathematical formulation of the optimization criterion again is to maximize the expected service. This has already been solved for several types of constraints.

References

General Stochastic

1 Bauer, H. (1991) *Wahrscheinlichkeitstheorie*, 4. Auflage. de Gruyter, Berlin.
2 Beichelt, F. (1997) Stochastische Prozesse für Ingenieure. Teubner, Wiesbaden.
3 Fisz, M. (1970) *Wahrscheinlichkeitsrechnung und mathematische Statistik*, VEB Deutscher Verlag der Wissenschaften, Berlin.
4 Gnedenko, B. (1997) *Lehrbuch der Wahrscheinlichkeitstheorie*, Harri Deutsch, Frankfurt.
5 Gross, D., Harris, C.M. (1998) *Fundamentals of Queueing Theory*, 3rd edn, Wiley, New York.
6 Rinne, H. (1997) *Taschenbuch der Statistik, 2*. Auflage. Harri Deutsch, Frankfurt.

Applications

7 Andler, K. 1929, *Rationalisierung der Fabrikation und optimale Losgrösse*. Oldenbourg, Munich.
8 Brüll, L., Metz, R. 2005, Supply Chain Management in der chemischen Industrie. *Automatisierungtechnische Praxis* 47 (8), 42–46.
9 Kulkarni, V.G. 1999. *Modeling, Analysis, Design and Control of Stochastic Systems*. Springer, Berlin.
10 Metz, R. 2005, Planungssysteme in der chemischen Industrie. *Automatisierungtechnische Praxis* 48 (1), 56–61.
11 Tempelmeier, H. 2005. *Bestandsmanagement in Supply Chains*. Books on Demand, Norderstedt.

Part IV
Optimization Methods

7
Engineered Mixed-Integer Programming in Chemical Batch Scheduling*

Guido Sand

7.1
Introduction

After more than two decades of academic research on mixed-integer programming in chemical batch scheduling, the relevant literature exhibits a variety of modeling frameworks, which claim to be "general" or "rather general". A review and comparison of related modeling concepts can be found in the chapter "MILP Optimization Models for Short-Term Scheduling of Batch Processes". However, the diversity of batch scheduling problems makes it impossible to include all potential problem characteristics in a unified model. Moreover, from a practical point of view this may even be undesirable as general, unspecific models typically suffer from their high computational effort. Nevertheless, the general modeling frameworks serve as an indispensable means to convey and to compare basic modeling concepts and techniques.

An alternative to mixed-integer programming based on general modeling frameworks is engineered mixed-integer programming based on tailored modeling and solution techniques [1]. In this chapter, a real-world case study is used to demonstrate how to develop and to solve a specific short-term scheduling problem. It will be shown that:

- The case study does not fit into the general modeling frameworks.
- The scheduling problem can be decomposed into a core problem and a subproblem.
- The specific problem characteristics are modeled most appropriately by a combination of concepts from various general modeling frameworks leading to a mixed-integer *nonlinear* programming (MINLP) model.
- A mixed-integer *linear* programming approximation can be derived following a problem specific approach.

* A list of symbols is given at the end of this chapter.

This chapter is organized as follows. First, the case study, the short-term scheduling of the production of ten kinds of polymer in a multiproduct plant, is presented (Section 7.2). In Section 7.3, the engineered approach is first motivated, the core problem is then worked out, and the modeling approach is finally sketched. The engineered MINLP-model with its binary and continuous variables, its nonlinear and linear constraints and its objective is developed and discussed in Section 7.4. In Section 7.5, a problem specific linearization approach is presented and applied, leading to a simplified mixed-integer linear programming (MILP) model. The solution of the MINLP-model and the MILP-model by various standard solvers is compared with respect to the solution quality and the computational effort (Section 7.6). In Section 7.7, some general conclusions on the application of engineered mixed-integer programming in chemical batch process scheduling are drawn.

7.2 The Case Study

The real-word case study considered here is the production of expandable polystyrene (EPS). Ten types of EPS are produced according to ten different recipes on a multiproduct plant which is essentially operated in batch mode. In this section, the multiproduct plant, the production process and the scheduling problem are presented.

7.2.1 Plant

The topology of the plant can be taken from Figure 7.1. It consists of a preparation stage for the production of two dispersion agents D1 and D2 and an organic phase OP, a polymerization stage and a finishing stage with two lines. The supply of the raw materials F1, F2 and F3 and the storage of the final products A1. . .A5, B1. . .B5 is assumed to be virtually unlimited. The preparation stage and the polymerization

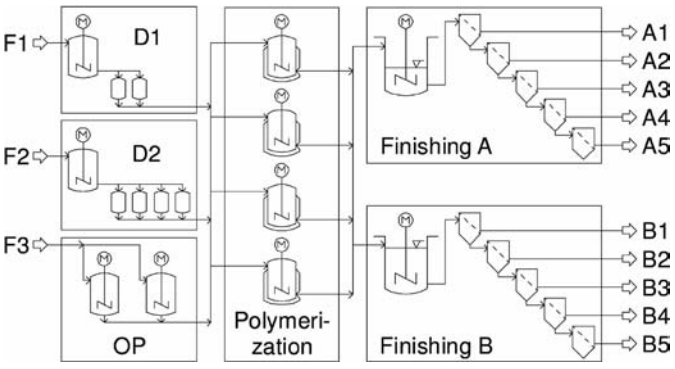

Fig. 7.1 Flowchart of the EPS-plant.

stage are operated in batch mode, whereas the finishing stage is operated in continuous mode.

The dispersion agents are produced in two stirred tank reactors with a capacity of two (D1) and four (D2) batches along with two (D1) and four (D2) storage tanks with a capacity of one batch each. The organic phase is produced in one out of two stirred tank reactors with a capacity of one organic phase batch each; no intermediate storage is provided for the organic phase.

The polymerization stage comprises four identical stirred tank reactors along with a common safety ventilation system designed for one runaway reaction (not shown in Figure 7.1). In the polymerization stage no intermediate storage is provided. The preprocessing stage, the polymerization stage and the finishing stage are fully networked by dedicated piping such that several batches can be transferred simultaneously.

Each finishing line consists of a mixing vessel and a separation unit. The mixing process is assumed to be ideal such that the following relation holds:

$$\frac{\text{mass of a product in the mixing vessel}}{\text{total mass in the mixing vessel}} = \frac{\text{feed rate of a product}}{\text{total feed rate}}$$

The mixing vessels serve as buffers between the polymerization stage operated in batch mode and the separation units operated continuously. The capacity of each mixing vessel is three polymerization batches and the minimal hold-up is 0.1 polymerization batches to ensure a sufficient mixing effect.

7.2.2 The Production Process

The plant is used to produce two chemically different EPS-types A and B in five grain size fractions each from raw materials F1, F2, F3. The polymerization reactions exhibit a selectivity of less than 100% with respect to the grain size fractions: Besides one main fraction, they yield significant amounts of the other four fractions as by-products. The production processes are defined by recipes which specify the EPS-type (A or B) and the grain size distribution. For each EPS-type, five recipes are available with the grain size distributions shown in Figure 7.2 (bottom). The recipes exhibit the same structure as shown in Figure 7.2 (top) in state-task-network-representation (states in circles, tasks in squares). They differ in the parameters, e.g., the amounts of raw materials, and in the temperature profiles of the polymerization reactions.

The composition of the dispersion agents, which are produced in dedicated reactors, is the same for all recipes. The batch sizes may be one or two polymerization batch units for D1 and one, two, three or four polymerization batch units for D2. The processing times are ten hours for D1 and two hours for D2, and they do not depend on the batch sizes. The dispersion agents in their final states are unstable in the reactors and stable for limited periods of time in the storage

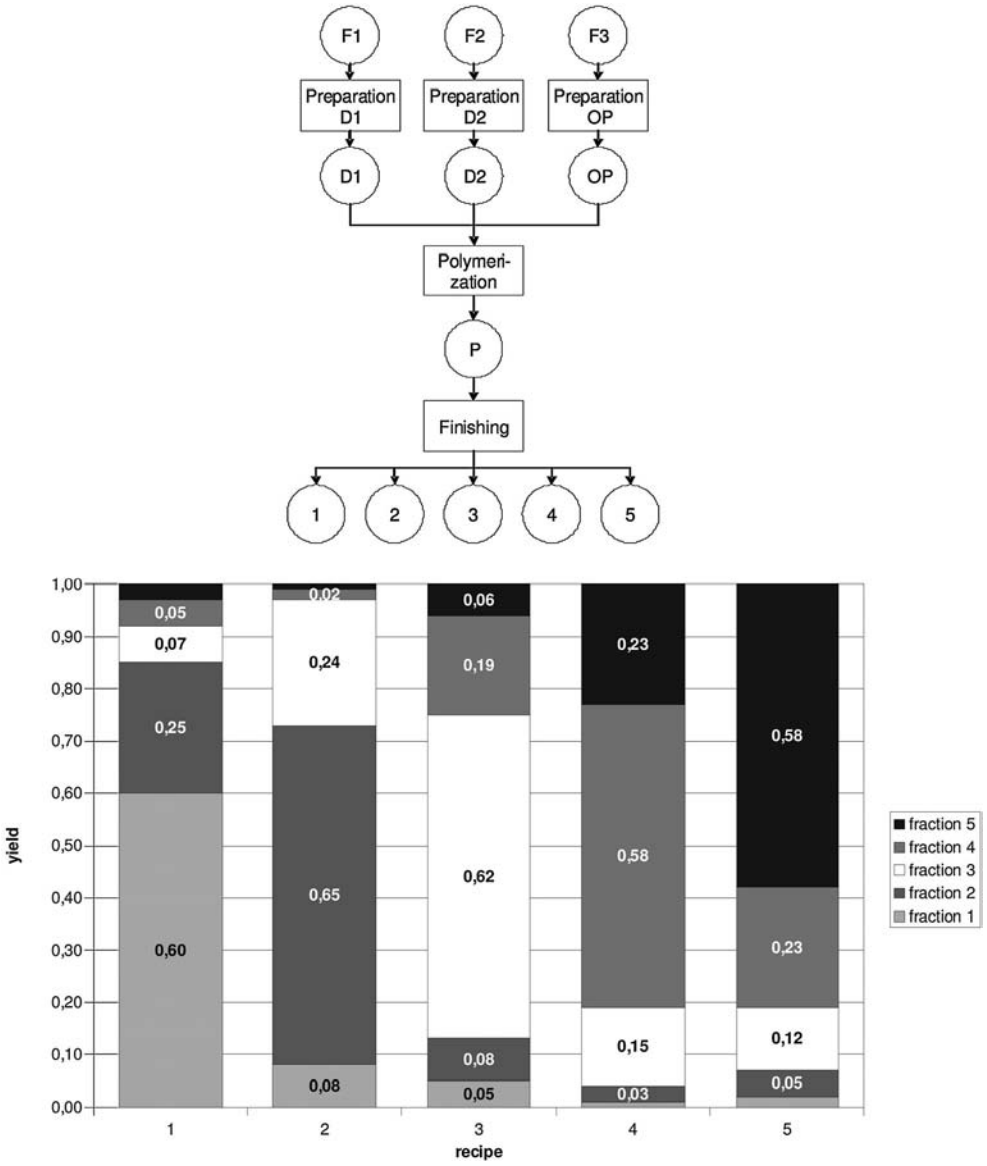

Fig. 7.2 Recipes (top) and grain size distributions (bottom).

tanks (for 24 h in case of D1 and for 36 h in case of D2). The composition of the organic phase, which is produced in one out of two dedicated reactors, depends on the chosen recipe. The batch-size is not scalable, and the processing time is 1 h. The organic phase in its final state is stable for an unlimited period of time.

Each polymerization batch is produced from one unit of each dispersion agent and one batch of the organic phase. Each polymerization batch is processed in one out of the four dedicated reactors, and the product properties depend on the recipes (they differ in the EPS-type and the grain size distributions, see above). The batch-size is fixed, and the processing time is 17 h, where for the first phase of four hours there is a risk of a run-away reaction. Because of the limited capacity of the safety ventilation system only one reactor may perform the first stage of the polymerization process at a given point in time. A polymerization batch is unstable in its finite state and has to be transferred into the corresponding mixing vessel immediately.

Each of the two finishing lines is dedicated to one EPS-type A or B. The separation units have to be provided with a permanent feed with a rate between 0.10 and 0.25 polymerization batches per hour. The residence time in a separation unit is 24 h regardless of the feed rates. A start-up as well as a shut-down of a separation unit requires a set-up time of 24 h.

7.2.3 Scheduling Problem

The EPS-production is driven by customer demands. The scheduling problem exhibits the following degrees of freedom, which may be discrete or continuous in nature:

- number of polymerizations/organic phase batches (discrete),
- number and size of dispersion agent batches (discrete),
- timings of batches in the preparation and the polymerization stages (continuous),
- assignment of recipes to polymerizations/organic phase batches (discrete),
- start-up and shut-down times of the finishing lines (continuous),
- hold-up profiles (or feed flow rates) of the mixing vessels (continuous).

The decisions should be taken in an optimal fashion subject to the plant topology and the processing constraints with the objective to maximize the profit, given as the difference of revenues for products and costs for the production. The demands are specified by their amounts and their due dates, where the revenues decrease with increasing lateness of the demand satisfaction. The production costs consist of fixed costs for each batch and for the start-up- and shut-down-procedures of the finishing lines, and variable costs for the product inventory.

The scheduling problem is complicated by the fact that the coupled production of grain size fractions and the mixing in the finishing lines prohibit a fixed assignment of recipes to products. Furthermore, there is neither a fixed assignment of storage tanks nor of polymerization reactors to batch processes.

7.3 An Engineered Approach to Optimal Scheduling

7.3.1 Motivation

A specific feature of the EPS-production is the coupling of stages that are operated in batch mode with stages that are operated continuously. This hybrid character prohibits the complete classification of the EPS-scheduling problem according to general schemes for scheduling problems of batch plants as, e.g., the roadmap presented in the chapter "MILP Optimization Models for Short-Term Scheduling of Batch Processes". As discussed below, this roadmap provides a suitable classification for the preparation stage and for the polymerization stage, but the finishing stage and the objective call for customized approaches.

1. Process topology: The EPS-process exhibits a sequential topology with multiple steps which are executed in a multiproduct plant. However, classical definitions as "flow-shop scheduling problem" or "job-shop scheduling problem" do not apply as the batches can no longer be identified in the finishing stage.
2. Options for assignment of the equipment: The assignment of the units to the processing steps is fixed with respect to the stages of the plant but variable with respect to particular units within the stages (e.g., the reactors of the polymerization stage).
3. Connectivity: The connectivity of the plant can be considered as full with respect to the given recipes as it does not constitute additional constraints.
4. Storage policies: In the preparation stage, finite intermediate storage (FIS) is provided by dedicated units; in the polymerization stage no intermediate storage (NIS) is allowed. The mixing vessels may be considered as special types of dedicated units which provide finite intermediate storage; the specialty here is the simultaneous execution of a mixing process.
5. Material transfer: Instantaneous material transfer can be assumed for the preparation stage and for the polymerization stage. The organic phase is stable in its final state and the dispersion agents are stable for limited periods of time such that unlimited wait (UW) and finite wait (FW) material transfer strategies apply in the preparation stage. The polymerization batches are unstable in their final states such that a zero wait (ZW) material transfer strategy applies in the polymerization stage. As the mixing vessels provide a permanent feed to the separation units, these types of material transfer strategies do not apply here.
6. Batch size: The batch sizes in the preparation stage and in the polymerization stage are variable as batches may be split in the preparation stage and mixed in the polymerization stage. However, the concept of batches does not apply in the finishing stage.
7. Batch processing times: The batch processing times in the preparation stage and in the polymerization stage are fixed and unit independent. Again, the concept of batches does not apply in the finishing stage.

8. Demand patterns: Multiple product demands appear which are specified by their amounts and their due dates.
9. Changeovers: No changeovers appear in the preparation stage and in the polymerization stage. The start-ups and shut-downs of the finishing lines are changeovers with certain set-up times which cause costs (see below).
10. Resource constraints: In none of the stages resource constraints (other then equipment constraints) apply.
11. Time constraints: In none of the stages time constraints apply.
12. Costs: Fixed costs are caused by the use of equipment for reactions and by changeovers, and variable costs are incurred for inventory.
13. Degree of certainty: For the short-term scheduling problem (studied in detail in this chapter) the data is assumed to be deterministic. However, in the long term uncertainties in the demands and the capacity of the plant become relevant. The chapter "Stochastic Integer Programming in Uncertainty Conscious Scheduling" deals with a modeling and solution approach for scheduling problems with uncertain data.

7.3.2 Analysis of the Problem

After having motivated that the EPS-scheduling problem calls for a customized model, the interdependence of the scheduling decisions is analyzed with respect to feasibility and optimality. This analysis is performed for arbitrary demand profiles. It exhibits that the scheduling decisions of the preparation stage are decoupled from the remainder such that they can be made based on rules once the decisions of the remaining core problem are fixed. Accordingly, the scheduling problem decomposes into a core problem and a subproblem.

For the dispersion agents finite intermediate storage is available, and the stored batches are stable for limited periods of time. Thus, a dispersion agent batch should be started when its storage runs empty such that it is finished just in time. With respect to the fixed production costs per batch, the batch size should be as large as possible; it is given by the number of polymerizations which are to be started within the period of stability. The processing times for the dispersion agents (10 h for D1, 2 h for D2) are smaller than the smallest interval between the starts of four polymerizations in case of D1 and two polymerizations in case of D2 which can be fed from the four (D1) and two (D2) storage tanks, namely 17 h in case of four polymerizations and 4 h in case of two polymerizations. Consequently, neither two batches of dispersion agents interact with each other nor are constraints on the polymerization stage imposed. The processing time of the organic phase is only one hour such that a just-in-time-strategy can be applied without constraining other decisions.

The decisions of the core problem may interact with each other over an infinite horizon. The number of polymerization reactors is the long-term bottleneck of the production process, whereas the capacity of the safety ventilation system imposes only short-term constraints. The run-away phase (four hours) of a polymerization

(with a batch time of 17 h) constrains the next polymerization in each of the other reactors ($4 \cdot 4\,\text{h} < 17\,\text{h}$). The timing of a polymerization batch imposes lower bounds on the timings of all subsequent polymerizations in the same reactor. Second, due to the coupled production, one polymerization batch corresponds to at least five demands with possibly very different due dates. That is why a very long horizon may have to be considered to assign the recipes optimally. Third, a finishing line which is out of operation or operated at its lower capacitive bound imposes constraints on the recipes and on the timings of the polymerizations. The duration of this effect is unbounded in principle such that the start-up and the shut-down times and the feed-rates may interact with infinitely many polymerization stage decisions.

In order to cover a long horizon on one hand and to provide feasible schedules on the other, the core problem is decomposed hierarchically into an aggregated scheduling problem and a detailed scheduling problem. Reasonable horizons are in the order of weeks and days, respectively. On the aggregated layer, the problem is modeled with a reduced temporal precision and solved to maximize the profit as was specified in Section 7.2.3. The long-term information is mapped onto the detailed scheduling horizon by means of demand profiles, the number of polymerization batches and start-up and shut-down times for the finishing lines. On the detailed scheduling layer, the timings of the polymerization batches, the assignments of recipes to polymerization batches and the holdup-profiles of the mixing vessels are optimized with respect to a simplified cost function. It is assumed that the profit is mainly determined on the aggregated scheduling layer such that the detailed scheduling problem aims at matching the demand profiles optimally. Its objective is to minimize a weighted sum of over- and underproduction of the demanded products at the respective due dates. In the following, the focus is on the detailed scheduling layer, for the aggregated layer the reader is referred to the chapter "Stochastic Integer Programming in Uncertainty Conscious Scheduling" (see also [2–4]).

7.4 Nonlinear Short-Term Scheduling Model

7.4.1 Modeling Concept

The short-term scheduling model is based on a continuous representation of time. Three distinct time axes are established corresponding to the polymerization stage, the finishing stage and the product storage units (see Figure 7.3). The events are represented following two different batch oriented concepts which both assume the number of batches to be given and their types (i.e., the recipes), sequencing and timings to be degrees of freedom. The assignment of recipes to batches is modeled on the polymerization stage by binary variables. For sequencing and timing, different concepts apply to the distinct stages.

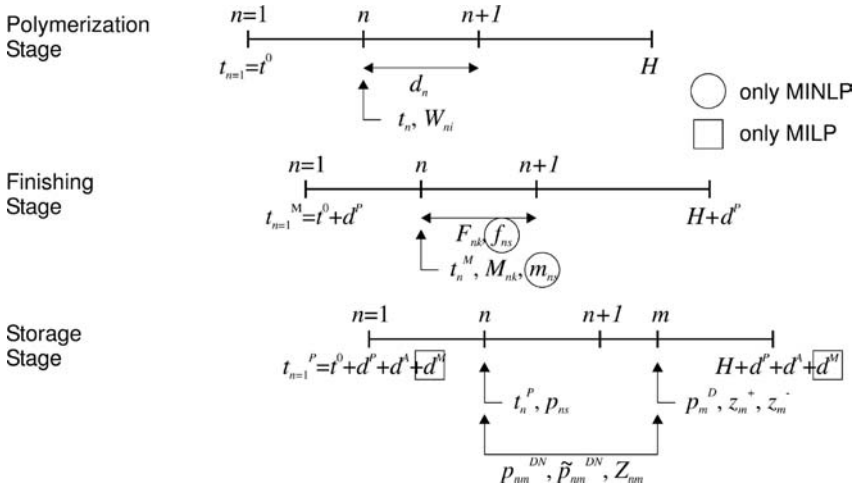

Fig. 7.3 Variables of engineered mixed-integer models.

The events on the polymerization stage time axis are modeled by an aggregated time slot approach: A single time axis is used to model the start times (but not the finish times) of the batch processes that are executed in the four polymerization reactors. The timings are degrees of freedom but the sequence is fixed. The events on the finishing stage time axis and the storage units time axis are modeled similarly: One time axis is used for the two finishing lines and one is used for the ten storage units. The events on the latter two axes are synchronized with the events on the polymerization stage time axis by fixed offsets such that they are no additional degrees of freedom. In contrast, a general precedence-based approach is applied on the storage units time axis to synchronize the variable event times with the fixed due dates of the demands: A binary variable indicates for each pair of event and due date if a certain event happens before or after a certain due date.

For the material balances, again two different concepts are applied to distinct stages of the plant. This mixed approach exploits that the number of batches produced is assumed to be given, but their types (i.e., the recipes) are degrees of freedom. As the capacity of the polymerization stage is not a function of the recipes, the material balance around the polymerization stage can be based on a batch oriented approach. In the finishing stage, batches are mixed and split into the grain size fractions which are stored and sold, such that the finishing stage and the product storage units require material balances based on network flow equations. It is important to notice that the material balance around the finishing stage is accompanied by the nonlinear function characterizing the mixing process.

The presented concept leads to a mixed-integer nonlinear programming (MINLP) model. The binary variables representing assignment decisions and sequencing decisions are complemented by continuous variables representing timings, durations, hold-ups, feed streams, supplied product, under- and overproduction. The

nonlinear mixing process constraints are complemented by nonlinear approximation constraints of the product profiles in addition to linear capacity constraints, assignment constraints, material balances, synchronization constraints and a linear objective function.

7.4.2 Formulation

The formulation of the engineered nonlinear short-term model presented is a variant of an MINLP model described in the dissertation by Schulz [5]. In this subsection, all necessary indices, parameters and variables are introduced, and the constraints and the objective function are derived. In the following section, the nonlinear formulation is linearized yielding a MILP model. In order to keep track of the variables used in the MINLP and in the MILP formulation, they are displayed in Figure 7.3 along with some key parameters.

7.4.2.1 Capacity Constraints of the Reactor Group

In the polymerization stage, the number of events, i.e., of the polymerization starts, N, is given and the events are identified by the index $n = 1 \ldots N$. Start times of polymerizations are represented by continuous variables $t_n \in [0, H]\ \forall n$ with H denoting the given scheduling horizon. As an initial condition, the first polymerization is defined to start at $t_{n=1} = t^0$.

The reactors are not considered individually but they are aggregated to a group of identical processing units with an overall capacity of four. The allocation of the reactors is not modeled explicitly, but they induce constraints on the start times of the polymerizations. Based on the assumption that all four reactors can be used and that they are allocated in turns, the intervals between t_n and t_{n+4} ($n = 1 \ldots N - 4$) must be greater than or equal to the processing time of a polymerization d^P (with $d^P = 17$):

$$t_{n+4} - t_n \geq d^P \quad \forall n \leq N - 4$$

7.4.2.2 Capacity Constraints of the Safety Ventilation System

Similar to the reactors, the allocation of the safety ventilation system is not modeled explicitly. It induces lower bounds on the variable intervals d_n ($n = 1 \ldots N - 1$) between two consecutive polymerization starts n and $n + 1$ given by the duration of the first polymerization phase q, i.e., $d_n \in [q, H]\ \forall n \leq N - 1$ with $q = 4$. The intervals d_n are calculated from equality constraints:

$$t_{n+1} - t_n = d_n \quad \forall n \leq N - 1$$

Note that in contrast to the intervals between t_n and t_{n+4}, the intervals between successive polymerization starts t_n and t_{n+1} are calculated explicitly, since they are essential for the capacity constraints of the mixing vessels (see below).

7.4.2.3 Assignment Constraints

Exactly one recipe $i \in I = \{1 \dots 10\}$ has to be assigned to each polymerization $n = 1 \dots N$. The assignments are indicated by binary variables $W_{ni} \in \{0, 1\}$ for each possible recipe-polymerization-combination ni with $n = 1 \dots N$ and $i = 1 \dots 10$. For a certain combination, $W_{ni} = 1$ indicates that recipe i is assigned to polymerization n, and $W_{ni} = 0$ indicates that it is not assigned to this polymerization. The following equality constraints ensure that exactly one recipe is assigned to each polymerization:

$$\sum_i W_{ni} = 1 \quad \forall n$$

7.4.2.4 Synchronization of the Mixing Vessels

The two mixing vessels $k = A, B$ are employed as semi-continuous storage units. In each vessel, the products $s \in S = \{1 \dots 10\}$ which belong to the corresponding subsets S_k, $S_{k=A} = \{1 \dots 5\}$ and $S_{k=B} = \{6 \dots 10\}$, are stored and mixed. The holdup of a mixing vessel k increases stepwise whenever a polymerization according to the corresponding subset of recipes I_k with $I_{k=A} = \{1 \dots 5\}$ and $I_{k=B} = \{6 \dots 10\}$ is finished. As the batches are transferred into the mixing vessels immediately after a polymerization is finished, there is a constant shift between the timings of the polymerization starts and the timings of the mixing vessel holdup steps $t_n^M \in [0; H]$ given by the processing time of a polymerization d^P:

$$t_n^M = t_n + d^P \quad \forall n$$

The same index n is used for the timings of the polymerization starts and the timings of the mixing vessel holdup steps since each mixing vessel holdup step corresponds exactly to one polymerization start.

In the sequel, it however turns out that t_n^M is not explicitly used. Thus, it does not have to be included in the model.

7.4.2.5 Product Balances Around the Mixing Vessels

Product balances around each of the mixing vessels $k = A, B$ are stated as integral balances over the half-open intervals $]t_{n-1}^M, t_n^M]$ $\forall n \geq 2$ and for the point $t_{n=1}^M$. The holdup of a product s immediately after a step at t_n^M is represented by the nonnegative variable $m_{ns} \in \mathbb{R}^+$ $\forall n, s$; the initial holdup of a product s immediately before a step at $t_{n=1}^M$ is represented by the parameter m_s^0 $\forall s$. The integral feed of product s during interval d_n is represented by the variable $f_{ns} \in \mathbb{R}^+$ $\forall n, s | n \leq N - 1$. The input is given by ρ_{is} (see Figure 7.2), the yield of product s according to the recipe i:

$$m_{ns} = \left\{ \begin{array}{ll} m_s^0 & \text{if } n = 1 \\ m_{n-1,s} & \text{else} \end{array} \right\} + \sum_i \rho_{is} W_{ni} - \left\{ \begin{array}{ll} 0 & \text{if } n = 1 \\ f_{n-1,s} & \text{else} \end{array} \right\} \quad \forall n, s$$

The sum $\sum_{i \in I_k} \rho_{is} W_{ni}$ acts like a filter which "filters out" the yield of the recipe i which is assigned to the polymerization n.

7.4.2.6 Total Feed and Total Holdup

The total integral feeds $F_{nk} \in \mathbb{R}^+$ ($n = 1 \ldots N-1$, $k = A, B$) during intervals d_n and the total holdups $M_{nk} \in [C^{\min}, C^{\max}]$ ($n = 1 \ldots N$, $k = A, B$, with $C^{\min} = 0.1$, $C^{\max} = 3.0$) immediately after steps at t_n^M are calculated from summations over the product specific integral feeds and holdups:

$$F_{nk} = \sum_{s \in S_k} f_{ns} \quad \forall n \leq N-1, k$$

$$M_{nk} = \sum_{s \in S_k} m_{ns} \quad \forall n, k$$

7.4.2.7 Lower Capacity Constraints of the Mixing Vessels

Between two steps, the holdups of the mixings vessels decrease monotonically and continuously. To guarantee feasible holdups it is sufficient to state the lower capacity constraints for the holdups immediately before the steps and the upper capacity constraints for the holdups immediately after the steps. The latter are already given by the upper bounds $C^{\max}$ on the total holdups M_{nk} (see above). The former are based upon integral total balances over the half-open intervals $\left[t_n^M, t_{n+1}^M\right[$ $\forall n \leq N-1$ (see Figure 7.4):

$$M_{nk} - F_{nk} \geq C^{\min} \quad \forall n \leq N-1, k$$

7.4.2.8 Lower and Upper Capacity Constraints of the Finishing Units

The capacities of the continuously operated finishing lines $k = A, B$ are lower and upper bounded by the feed rates, $F^{\min}$ and $F^{\max}$ (with $F^{\min} = 0.10$ and $F^{\max} = 0.25$), respectively. The bounds on the integral feed F_{nk} are given by the case that the feed rate is at its lower or upper bound during the whole interval d_n:

$$F^{\min} d_n \leq F_{nk} \leq F^{\max} d_n \quad \forall n \leq N-1, k$$

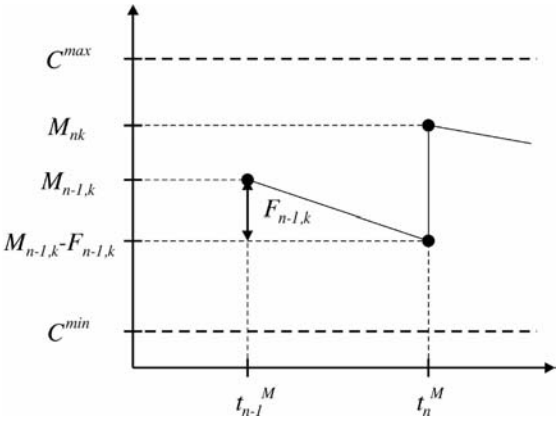

Fig. 7.4 Capacity constraints of the mixing vessels k.

7.4.2.9 Mixing Process

The mixing process is assumed to be ideal, i.e., the product concentration of the feed is equal to the product concentration of the holdup at any time. The concentrations change at the event points t_n^M only, such that the mixing process can be stated based upon the variables as introduced above:

$$\frac{m_{ns}}{M_{nk}} = \frac{f_{ns}}{F_{nk}} \quad \forall n \leq N-1, s \in S_k, k$$

This nonlinear equality constraint makes the continuous part of the optimization problem nonconvex. Nonconvexity with regard to constraints means, that the linear combination of two feasible points may be infeasible. The following two points may serve as an example: $(m_1, M_1, f_1, F_1)^T = (1, 2, 3, 6)^T$ and $(m_2, M_2, f_2, F_2)^T = (1, 3, 2, 6)^T$. Both of them are feasible with respect to the mixing process constraint but their arithmetic mean $0.5\,(m_1, M_1, f_1, F_1)^T + 0.5\,(m_2, M_2, f_2, F_2)^T = (1, 2.5, 2.5, 6)^T$ is infeasible.

7.4.2.10 Synchronization of the Storage Tanks

Similar to the balances around the mixing vessels, the product profiles are calculated based on integrated quantities. The timings of the supporting points of the storage profiles $t_n^P \in [0, H]$ with $n = 1 \ldots N$ correspond to the holdup steps of the mixing vessels t_n^M with a constant shift given by the processing time d^A of the separation (with $d^A = 24$):

$$t_n^P = t_n^M + d^A \quad \forall n$$

Again, the same index n as for the timings of the polymerization starts and the timings of the mixing vessel holdup steps can be used.

As the timings of the mixing vessel holdup steps t_n^M are not explicitly used (see above), they can be eliminated and the synchronization constraints of the mixing vessels can be dropped:

$$t_n^P = t_n + d^P + d^A \quad \forall n$$

7.4.2.11 Supporting Points of the Product Profiles

Each supporting point of the profile of product s at time t_n^P is represented by the variable $p_{ns} \in \mathbb{R}^+$ with $n = 1 \ldots N$ and $s = 1 \ldots 10$; the initial condition is given by $p_s^0\ \forall s$. The difference between two consecutive supporting points p_{ns} and $p_{n+1,s}$ is given by the integral feed $f_{n,s}$ (see Figure 7.5):

$$p_{ns} = \left\{ \begin{array}{ll} p_s^0 & \text{if } n = 1 \\ p_{n-1,s} & \text{else} \end{array} \right\} + \left\{ \begin{array}{ll} 0 & \text{if } n = 1 \\ f_{n-1,s} & \text{else} \end{array} \right\} \quad \forall n, s$$

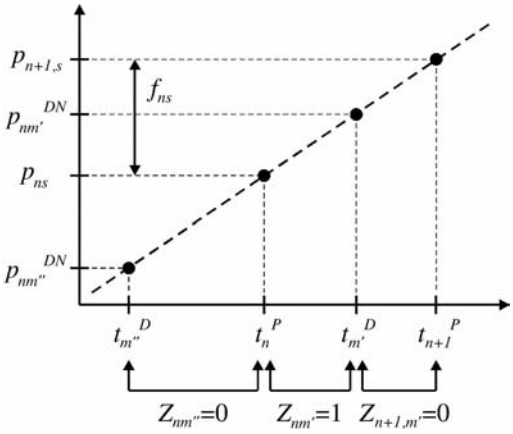

Fig. 7.5 Interpolation and extrapolation of the supporting points.

7.4.2.12 Approximation of the Product Profiles

In order to calculate deviations of the demand profiles from the product profiles at the M due dates t_m^D of the demands $m = 1 \dots M$, the supporting points p_{ns} are linearly interpolated and extrapolated. The amount of product $s \in S_m$ at t_m^D, which is interpolated (see Figure 7.5, Index m') or extrapolated (see Figure 7.5, Index m'') from p_{ns} and $p_{n+1,s}$, is represented by the variable $p_{nm}^{DN} \in \mathbb{R}^+$ with $n = 1 \dots N$ and $m = 1 \dots M$. Note, that each subset S_m contains exactly one element such that the demanded product $s \in S_m$ is uniquely defined by m. For $n = 1 \dots N-1$ it is calculated from the supporting point at t_n^P and the integral feed f_{ns}; for $n = N$ it is equal to the last supporting point as no corresponding integral feed is defined. The approximation constraints are stated for those subsets of products S_m^D with a corresponding demand at t_m^D:

$$p_{nm}^{DN} = p_{ns} + \left\{ \begin{array}{ll} 0 & \text{if } n = N \\ f_{ns} \dfrac{t_m^D - t_n^P}{t_{n+1}^P - t_n^P} & \text{else} \end{array} \right\} \quad \forall n, m, s \in S_m^D$$

Note, that this approximation is a second source of nonconvex nonlinear constraints.

7.4.2.13 Synchronization of the Demands

At a certain due date t_m^D, the demand has to be compared to the corresponding interpolated (not extrapolated) amount of product, i.e., to that p_{nm}^{DN} with $t_m^D \in \left]t_n^P, t_{n+1}^P\right]$. An exception are demands $\overline{m}$ with due dates $t_{\overline{m}}^D$ which are greater than the timing of the last supporting point $t_{n=N}^P$: These cannot be compared to amounts of product but are compared to the product demand at the last supporting point instead. It is important to note that for a given demand m the neighboring supporting points (specified by n and $n+1$), which are used for the interpolation, are not known a

priori. The interpolated (and for $\overline{m}$ extrapolated) amounts are represented by the variables $p_m^D \ \forall m$ and determined by the following synchronization constraints:

$$p_m^D = \sum_n p_{nm}^{DN} \left(Z_{nm} - \left\{ \begin{array}{ll} 0 & \text{if } n = N \\ Z_{n+1,m} & \text{else} \end{array} \right\} \right) \quad \forall m$$

The binary variable $Z_{nm} \in \{0, 1\}$ $(n = 1 \ldots N,\ m = 1 \ldots M)$ indicates if t_n^P precedes t_m^D, or not (i.e., $t_n^P < t_m^D \Leftrightarrow Z_{nm} = 1 \ \forall n, m$, see Figure 7.5 and below). If t_n^P precedes t_m^D and if t_{n+1}^P it does not precede t_m^D (with $n = 1 \ldots N - 1$), i.e., if t_m^D is in the half-open interval $\left]t_n^P, t_{n+1}^P\right]$, then the difference $Z_{nm} - Z_{n+1,m}$ is equal to 1 ($Z_{nm} = 1$ and $Z_{n+1,m} = 0$), otherwise it is 0 ($Z_{nm} = 1$ and $Z_{n+1,m} = 1$ or $Z_{nm} = 0$ and $Z_{n+1,m} = 0$). For $t_{m'}^D$ (i.e., $n = N$), the subtrahend disappears as t_{n+1}^P is not defined.

The products $p_{nm}^{DN} \left(Z_{nm} - Z_{n+1,m} \right)$ can exactly be linearized according to an approach by Williams [6]: For that, the products are substituted by auxiliary variables $\tilde{p}_{nm}^{DN} \in \mathbb{R}^+$ with $n = 1 \ldots N$ and $m = 1 \ldots M$, i.e.,

$$p_m^D = \sum_n \tilde{p}_{nm}^{DN} \quad \forall m$$

which are linearly constrained as follows:

$$\tilde{p}_{nm}^{DN} \leq N \left(Z_{nm} - \left\{ \begin{array}{ll} 0 & \text{if } n = N \\ Z_{n+1,m} & \text{else} \end{array} \right\} \right) \quad \forall n, m$$

$$\tilde{p}_{nm}^{DN} \geq p_{nm}^{DN} - N \left(1 - \left(Z_{nm} - \left\{ \begin{array}{ll} 0 & \text{if } n = N \\ Z_{n+1,m} & \text{else} \end{array} \right\} \right) \right) \quad \forall n, m$$

$$\tilde{p}_{nm}^{DN} \leq p_{nm}^{DN} \quad \forall n, m$$

As the size of a polymerization batch is normalized (the size of a batch is "1") and the initial product amounts are defined to be zero (see below), the number of polymerizations N is a nonbinding upper bound for the product amounts. The first inequality constraint forces $\tilde{p}_{nm}^{DN}$ to zero if the difference $Z_{nm} - Z_{n+1,m}$ or if $Z_{n=N,m}$ is equal to zero; if $Z_{nm} - Z_{n+1,m}$ or if $Z_{n=N,m}$ is equal to one, the inequality constraint is nonbinding. The second inequality constraint forces $\tilde{p}_{nm}^{DN}$ to a value greater than or equal to p_{nm}^{DN} if the difference $Z_{nm} - Z_{n+1,m}$ or if $Z_{n=N,m}$ is equal to one, i.e., if $1 - \left(Z_{nm} - Z_{n+1,m} \right)$ or if $1 - Z_{n=N,m}$ is equal to zero; if $Z_{nm} - Z_{n+1,m}$ or if $Z_{n=N,m}$ is equal to zero, i.e., if $1 - \left(Z_{nm} - Z_{n+1,m} \right)$ or if $1 - Z_{n=N,m}$ is equal to one, the inequality constraint is nonbinding. In combination with the third inequality constraint, $\tilde{p}_{nm}^{DN}$ is forced exactly to p_{nm}^{DN} if $Z_{nm} - Z_{n+1,m}$ or if $Z_{n=N,m}$ is equal to one.

7.4.2.14 **Logic Constraints**

The equivalence $t_n^P < t_m^D \Leftrightarrow Z_{nm} = 1 \ \forall n, m$ (if and only if t_n^P precedes t_m^D, then Z_{nm} is equal to one) is modeled by two inequality constraints. They reflect the

implications $t_n^P < t_m^D \Rightarrow Z_{nm} = 1\ \forall n, m$ (if t_n^P precedes t_m^D, then Z_{nm} is equal to one) and $t_n^P > t_m^D - 0.1 \Rightarrow Z_{nm} = 0\ \forall n, m$ (if t_n^P follows $t_m^D - 0.1$, then Z_{nm} is equal to zero). By the small, artificial subtrahend 0.1, Z_{nm} becomes zero if t_n^P and t_m^D are simultaneous. Without the subtrahend, Z_{nm} would not be uniquely defined for the case $t_n^P = t_m^D$. The implications are stated following a "big M"-approach with the scheduling horizon H as "big M":

$$t_m^D \leq t_n^P + H \cdot Z_{nm} \quad \forall n, m$$

$$t_n^P \leq t_m^D - 0.1 + H(1 - Z_{nm}) \quad \forall n, m$$

If t_n^P precedes t_m^D, then Z_{nm} is forced to 1 by the first inequality constraint; in this case, the second constraint becomes $t_n^P \leq t_m^D - 0.1$. If t_n^P follows $t_m^D - 0.1$, then Z_{nm} is forced to 0 by the second inequality constraint; in this case, the second constraint becomes $t_m^D \leq t_n^P$. Note, that t_n^P with $t_m^D - 0.1 < t_n^P < t_m^D$ are infeasible which affects, however, the model precision only marginally.

7.4.2.15 Objective Function

The objective to be minimized is a weighted sum of deviations of the produced amounts p_m^D from the demanded amounts d_m^D at the due dates $t_m^D\ \forall m$. Overproduction and underproduction, i.e., positive differences $p_m^D - d_m^D$ and $d_m^D - p_m^D$, respectively, are weighted by the nonnegative factors α_m and β_m. If the value of the objective function is represented by $z \in \mathbb{R}$ the objective can be stated as follows:

$$\min z = \sum_m \left(\alpha_m \max\left(0,\ p_m^D - d_m^D\right) + \beta_m \max\left(0,\ d_m^D - p_m^D\right)\right)$$

The nonlinear max-operators are used to distinguish if the differences $p_m^D - d_m^D$ and $d_m^D - p_m^D$ yield positive or negative values. The operators can exactly be linearized using variables $z_m^+ \in \mathbb{R}^+\ \forall m$ and $z_m^- \in \mathbb{R}^+\ \forall m$, representing the values of $\max\left(0,\ p_m^D - d_m^D\right)$ and $\max\left(0,\ d_m^D - p_m^D\right)$, respectively. The objective can then be reformulated to a linear one along with a linear equality constraint:

$$\min z = \sum_m \left(\alpha_m z_m^+ + \beta_m z_m^-\right)$$

$$z_m^+ - z_m^- = p_m^D - d_m^D \quad \forall m$$

For a given positive or negative difference $p_m^D - d_m^D$, the minimization of the weighted sum of z_m^+ and z_m^- makes sure that either z_m^+ or z_m^- is zero in an optimal solution.

7.5 Linearized Short-Term Model

7.5.1 Linearization Approach

Both the mixing process and the approximation of the product profiles establish nonconvex nonlinearities. The inclusion of these nonlinearities in the model leads to a relatively precise determination of the product profiles but do not affect the feasibility of the production schedules. A linear representation of both equations will decrease the precision of the objective but it will also eliminate the nonlinearities yielding a mixed-integer linear programming model which is expected to be less expensive to solve.

In order to linearize the mixed-integer nonlinear programming model, a problem specific approach is applied. The mixing process constraints are dropped such that the product specific quantities (product specific feeds and product specific holdups) are decoupled from the corresponding total quantities (total feeds and total holdups). Constraints on the product specific quantities are summed up yielding constraints on the total quantities, and constraints on the total quantities are maintained to guarantee feasible schedules. On the product specific level, the semi-continuous mixing process is approximated as a batch process with a fixed processing time $d^M = 10$. As a consequence, the product amounts in the storages are piecewise constant with steps at the supporting points, such that the nonlinear approximation constraints of the product profiles can be dropped as well.

7.5.2 Model Adaptation

As indicated above, the variables representing the product specific holdups $m_{ns} \in \mathbb{R}^+ \; \forall n, s$ and the product specific feeds $f_{ns} \in \mathbb{R}^+ \forall n \leq N-1, s$ as well as the equality constraints defining the total feed and the total holdup and the mixing process constraints are dropped. For ease of notation, the constraints which approximate the product profiles are substituted by an identity:

$$p_{nm}^{DN} = p_{ns} \quad \forall n, m, s \in S_m^D$$

The product specific quantities in the product balances around the mixing vessels are summed up and replaced by the total quantities, resulting in the following total balances of the mixing vessels:

$$M_{nk} = \left\{ \begin{array}{ll} M_k^0 & \text{if } n = 1 \\ M_{n-1,k} & \text{else} \end{array} \right\} + \sum_{i, s \in S_k} \rho_{is} W_{ni} - \left\{ \begin{array}{ll} 0 & \text{if } n = 1 \\ F_{n-1,k} & \text{else} \end{array} \right\} \quad \forall n, k$$

The timings of the supporting points of the storage profiles t_n^P are shifted by another processing time, namely that of the approximated mixing process d^M (see

Figure 7.3), leading to the following adapted constraints for the synchronization of the storages:

$$t_n^P = t_n + d^P + d^A + d^M \quad \forall n$$

The height of a step of the product amount at t_n^P is given by the yield ρ_{is} leading to the following adapted constraints for the supporting points of the product profiles:

$$p_{ns} = \left\{ \begin{array}{ll} p_s^0 & \text{if } n = 1 \\ p_{n-1,s} & \text{else} \end{array} \right\} + \sum_i \rho_{is} W_{ni} \quad \forall n, s$$

The remaining variables and linear constraints from the MINLP-model are kept in the model.

7.6 Comparative Numerical Studies

7.6.1 Concept

The nonconvex constraints of the MINLP-model may cause multiple local optima. Note that in mathematical programming a conceptual distinction is made between nonconvexity caused by constraints and by the objective (i.e., nonconvexity in the continuous subspace) and nonconvexity caused by integrality requirements. Classical gradient based solvers converge to that local optimum in the continuous subspace which is "next" to the initial point. In contrast to nonlinear problems, linear problems are always convex and exhibit only one local solution which is also the global one. Schulz [5] addressed the nonconvex MINLP by a heuristic, problem specific depth search algorithm. In contrast, in the following exclusively commercially available "of the shelf" solvers are applied.

In this section, the numerical solutions of the MINLP-model and of the MILP-model as presented in Sections 7.4 and 7.5 are compared with respect to their solution quality (measured by the objective values) and the required solution effort (measured by the computing time). In order to compare the MILP-solution with the MINLP-solution, the optimized values for the start times of polymerizations t_n, the recipe assignments W_{ni}, and the total holdups M_{nk} are inserted into the MINLP-model and the objective is calculated. To guarantee comparability of the results, the models were stated with identical initial conditions, namely $t^0 = 0$, $M_k^0 = 2\ \forall k$, $p_s^0 = 0\ \forall s$, and $m_s^0 = 0.4\ \forall s$ (i.e., the variables defined at the beginning of the corresponding time axes are fixed to the indicated values). For the algorithmic solution procedure, all variables were initialized by 1 (i.e., the search for optimal values starts at values of "1"), and none of the solvers was specifically customized.

Each model was tested for 16 ($= 2^4$) different sets of parameters, which were generated by combining the schemes A and B for the amounts d_m^D, the demanded

Table 7.1 Schemes for the amounts.

Demand m	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Scheme A	0.1	0.3	0.5	0.7	0.9	1.1	1.3	1.5	1.7	1.9
Scheme B	1.9	1.7	1.5	1.3	1.1	0.9	0.7	0.5	0.3	0.1

Table 7.2 Schemes for the demanded products.

Demand m	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Scheme A	1	2	3	4	5	6	7	8	9	10
Scheme B	10	9	8	7	6	5	4	3	2	1

Table 7.3 Schemes for the weighting factors.

Demand m	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Scheme A	1	2	3	4	5	6	7	8	9	10
Scheme B	10	9	8	7	6	5	4	3	2	1

Table 7.4 Schemes for the due dates.

Demand m	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Scheme A	55	60	65	70	75	80	85	90	95	100
Scheme B	75	100	75	100	75	100	75	100	75	100

product S_m, the weighting factors of the underproduction β_m, and the due dates of the demands t_m^D, see Tables 7.1 to 7.4. Each model comprised $M = 10$ demands, the scheduling horizon H was set to 200, and the weighting factors for the overproduction were set to $\alpha_m = 1\ \forall m$.

7.6.2 Algorithms

The MINLP-problems were implemented in GAMS [7, 8] and solved by the outer approximation/equality relaxation/augmented penalty-method [9] as implemented in DICOPT. The algorithm generates a series of NLP and MILP subproblems, which were solved by the generalized reduced gradient method [10] as implemented in CONOPT and the integrality relaxation based branch and cut method as

implemented in CLPEX, respectively. The MILP-problems were implemented in GAMS and solved by CPLEX.

7.6.2.1 Outer Approximation/Equality Relaxation/Augmented Penalty

The algorithm underlying DICOPT starts by solving the NLP in which the integrality conditions are relaxed. If the solution to this problem yields an integer solution the search stops. Otherwise, it continues with an alternating sequence of nonlinear programs (NLP) called subproblems and mixed-integer linear programs (MILP) called master problems. The NLP subproblems are solved for fixed integer variables that are predicted by the MILP master problem at each major iteration. For the case of convex problems, the master problem provides a lower bound (in case of minimization) on the objective function. This lower bound increases monotonically as iterations proceed due to the accumulation of linear approximations.

The term "outer approximation" refers to the fact that the surface described by a convex function lies above the tangent hyper-plane at any interior point of the surface. In the algorithm outer approximations are attained by generating linearizations at each iteration and accumulating them in order to provide successively improved linear approximations of nonlinear convex functions that underestimate the objective function and overestimate the feasible region. The term "equality relaxation" refers to the fact that under certain assumptions concerning the convexity of the nonlinear functions, an equality constraint can be "relaxed" to an inequality constraint. This property is used in the MILP master problem to accumulate linear approximations. "Augmented penalty" refers to the introduction of (nonnegative) slack variables on the right hand sides of the just described inequality constraints and the modification of the objective function when assumptions concerning convexity do not hold [8].

7.6.2.2 Generalized Reduced Gradient

Generalized reduced gradient algorithms search along curves that stay near the feasible set. Essentially they use the constraints to eliminate a subset of the variables, thereby reducing the original problem to a bound-constrained problem in the space of the remaining variables. The eliminated variables are called basic variables, the remainder are called nonbasic variables. The nonbasic variables are furthermore subdivided into fixed variables and superbasic variables. The fixed variables include most of the variables which are at either their upper or lower bounds and that are to be held constant in the current iteration. The superbasic variables are free to move in this iteration. The reduced gradient algorithm implemented in CONOPT searches along the steepest-descent direction in the superbasic variables.

7.6.2.3 Branch and Cut

The integer part of the MILP problem is addressed by a branch and cut algorithm, which is a hybrid of branch and bound and cutting plane algorithms.

A branch and bound algorithm works on a search tree with linear programs (LPs) in the nodes. In the root node, all integrality requirements are dropped and a fully integer-relaxed linear program is solved as the first subproblem. In the simplest case, each subsequent layer of the search tree corresponds to a bisection of the search space according to one integer variable, and the branches on each layer correspond to additional integer bounds on that variable. The leaf nodes finally represent the totality of integer solutions of that problem. Branch and cut employs cutting planes in addition to simple bounds to constrain the search space more tightly.

The LP solutions in the nodes control the sequence in which the nodes are visited and provide conservative lower bounds (in case of minimization problems) with respect to the objective on the subsequent subproblems. If this lower bound is higher than the objective of the best feasible solution found so far, the subsequent nodes can be excluded from the search without excluding the optimal solution. Each feasible solution corresponds to a leaf node and provides a conservative upper bound on the optimal solution. This combination of branching and bounding or cutting steps leads to the implicit enumeration of all integer solutions without having to visit all leaf nodes.

The LP problems were solved by the simplex method. This algorithm solves a linear program by progressing from one extreme point of the feasible polyhedron to an adjacent one.

7.6.3 Results

The MINLP-model instances comprised 200 binary variables, 588 continuous variables and 1038 constraints. The linearization not only eliminates the nonlinearity but also leads to a reduced number of 398 continuous variables and 830 constraints (the number of 200 binary variables is unchanged). The MINLP-problems were solved by the solver architecture DICOPT/CONOPT/CPLEX, and the MILP problems were solved by CPLEX, both on a Windows machine with an Intel Xeon 3 GHz CPU and 4 GB RAM.

Figure 7.6 shows that for 14 out of 16 instances, the solution based upon the simplified MILP model is of better quality, i.e., the objective to be minimized is smaller, than the local solution found for the MINLP problem (ABAB stands for scheme A for the amounts, scheme B for the demanded products, scheme A for the weighting factors, and scheme B for the due dates). Remember that the solution quality of the MILP problem was evaluated by applying the solution vector to the MINLP model. Furthermore, Figure 7.7 shows that the solution effort for the MILP problems is significantly smaller and less volatile than for the MINLP problems. For the MINLP problems, the CPU times range between 3 s and 18,436 s with a mean value of 4277 s; for the MILP problems they range only between 6 s and 1500 s with a much smaller mean value of 286 s.

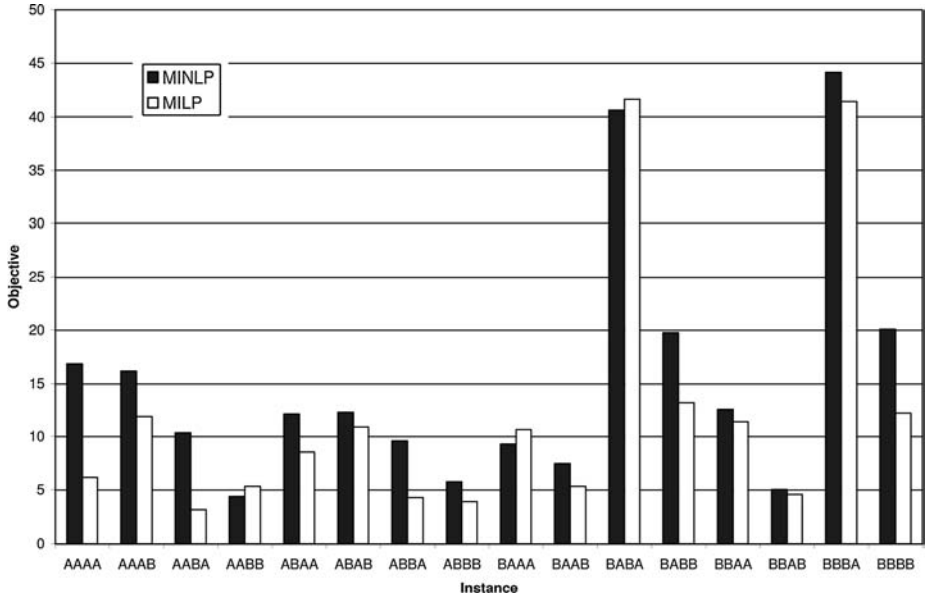

Fig. 7.6 Solution quality.

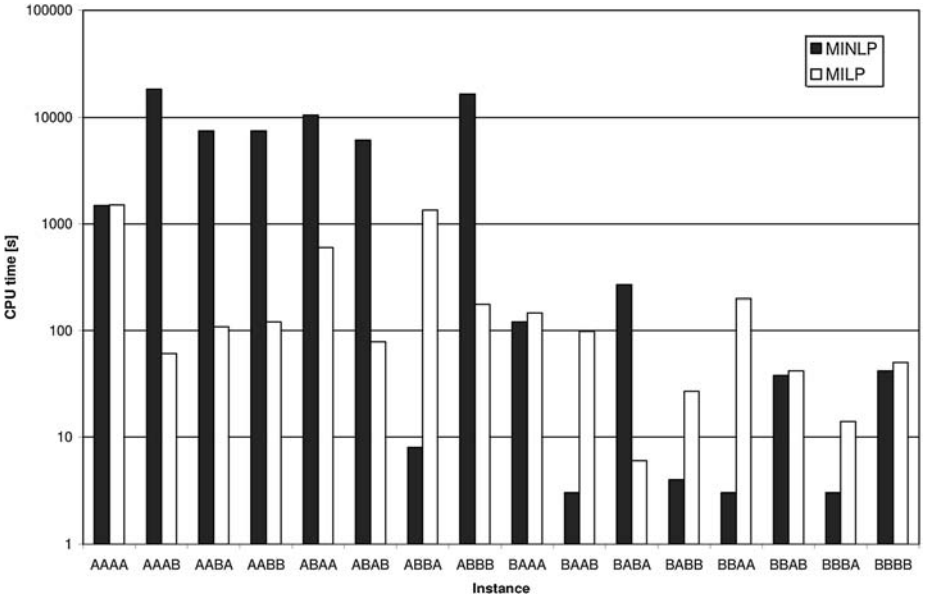

Fig. 7.7 Solution effort.

7.7 Conclusions

The study of a real-world production process from the polymer industries exhibited that the short-term scheduling problem cannot completely be classified according to general schemes for scheduling problems of batch plants. The main reason is its hybrid character given by the coupling of stages which are operated in batch mode with stages which are operated continuously. A more detailed problem analysis revealed that the scheduling problem can be decomposed into a core problem comprising interacting decisions and a subproblem comprising decisions which can be made once the core problem has been solved.

An engineered modeling approach to the short-term core problem led to a mixed-integer nonlinear programming model. The nonconvex nonlinearities are caused by batch mixing processes which are executed in semi-continuously operated mixing vessels and by interpolations and extrapolations of the supporting points of the product profiles. The nonlinearities can be eliminated following a customized linearization approach which led to a mixed-integer linear programming model.

In numerical studies it turned out that the MILP problem can not only be solved much faster than the MINLP problem, but for most of the model instances it provides solutions of significantly higher solution quality. Certainly, the engineered linearization of the nonlinear problem causes a loss in model precision, but on the other hand it enables a globally optimal solution. Since the MILP solutions are feasible for the MINLP problem, it is clear that the inferior quality of the MINLP solutions originates from the fact that only local minima were found.

Symbols

$C^{\min}$, $C^{\max}$	Bounds on total holdups
d_n	Intervals
$d^A = 24$	Processing time of the separation
$d^M = 10$	Processing time for mixing
$d^P = 17$	Processing time of a polymerization
$f_{ns} \in^+ \mathbb{R}^+ \; \forall n, s \vert n \leq N - 1$	Integral feeds
$F_{nk} \in \mathbb{R}^+$	Total integral feeds
$F^{\min}$, $F^{\max}$	Bounds on total integral feeds
H	Scheduling horizon
$i \in I = \{1 \ldots 10\}$	Recipes
$I_{k=A} = \{1 \ldots 5\}$, $I_{k=B} = \{6 \ldots 10\}$	Subsets of recipes
$k = A, B$	Mixing vessels
$m = 1 \ldots M$	Demands
$m_{ns} \in \mathbb{R}^+ \; \forall n, s$	Holdup of mixing vessels
$m_s^0 \; \forall s$	Initial holdups of mixing vessels
M	Number of demands

$M_{nk} \in [C^{\min}, C^{\max}]$	Total holdups
$n = 1 \ldots N$	Events
N	Number of events
$p_{ns} \in \mathbb{R}^+$	Supporting points of the storage profiles
$p_{nm}^{DN} \in \mathbb{R}^+$	Interpolated or extrapolated amounts of products
$\tilde{p}_{nm}^{DN} \in \mathbb{R}^+$	Auxiliary variables
$p_s^0 \; \forall s$	Initial condition for supporting points of the storage profiles
$q = 4$	Duration of the first polymerization phase
$s \in S = \{1 \ldots 10\}$	Products
$S_{k=A} = \{1 \ldots 5\}$, $S_{k=B} = \{6 \ldots 10\}$	Subsets of products
$t_n \in [0, H] \; \forall n$	Timings of polymerization starts
t_m^D	Due dates
$t_n^M \in [0; H]$	Timings of the mixing vessel holdup steps
$t_n^P \in [0, H]$	Timings of the supporting points of the storage profiles
$t^0 = 0$	Initial timing of polymerization starts
$W_{ni} \in \{0, 1\}$	Assignments of recipe to polymerization
$z_m^+ \in \mathbb{R}^+ \, \forall \, m$	Overproductions
$z_m^- \in \mathbb{R}^+ \, \forall \, m$	Underproductions
$Z_{nm} \in \{0, 1\}$	Precedence indicators
α_m	Weighting factors for overproductions
β_m	Weighting factors for underproductions
ρ_{is}	Yields

References

1 Pekny, J. and Reklaitis, G. (1998) *Towards the Convergence of Theory and Practice: A Technology Guide for Scheduling/Planning Methodology*. Proceedings of 3rd Conference on Foundations of Computer-Aided Process Operations (FOCAPO), CACHE, New York, pp. 91–111.

2 Sand, G. and Engell, S. (2003) Modeling and solving real-time scheduling problems by stochastic integer programming. *Comput. Chem. Eng.*, **28**, 1087–1103.

3 Engell, S., Märkert, A., Sand, G. and Schultz, R. (2004) Aggregated scheduling of a multiproduct batch plant by two-stage stochastic integer programming. *Optim. Eng.*, **5**, 335–359.

4 Till, J., Engell, S. and Sand, G. (2005) Rigorous vs stochastic algorithms for two-stage stochastic integer programming applications. *Intl. J. Inf. Technol.*, **11**, 106–115.

5 Schulz, C. (2002) Modeling and optimization of a multiproduct processing plant. Dissertation, Universität Dortmund, Department of Chemical Engineering (in German).

6 Williams, H. (1999) *Model building in Mathematical Programming*, Wiley, New York.

7 Brooke, A, Kendrick, D., Meeraus, A. and Raman, R. (2005) *GAMS – A Users Guide*, GAMS Development Corporation, Washington, DC.

8 GAMS (2005) *The Solver Manuals*, GAMS Development Corporation, Washington, DC.

9 Viswanathan, J. and Grossmann, I.E. (1990) A combined penalty function and outer approximation method for MINLP optimization. *Comput. Chem. Eng.*, **14**, 769–782.

10 Drud, A.S. (1992) CONOPT: a large-scale GRG code. *ORSA J. Comput.*, **6**, 207–216.

11 Glover, F., Laguna, M. and Marti, R. (2003) Scatter search, in *Advances in Evolutionary Computation: Theory and Applications*, (eds A. Ghosh and S. Tsutsui), Springer, New York, pp. 519–537.

8
MILP Optimization Models for Short-term Scheduling of Batch Processes

Carlos A. Méndez, Ignacio E. Grossmann, Iiro Harjunkoski, and Marco Fahl

8.1
Introduction

Scheduling is a critical issue in process operations that is crucial for improving production performance. For batch processes, short-term scheduling deals with the allocation of a set of limited resources to manufacture several products following batch recipes over a time horizon of typically a few days up to two weeks. There have been significant research efforts over the last decade in this area in the development of MILP optimization approaches, and several excellent reviews can be found in Pekny and Reklaitis [1], Pinto and Grossmann [2], Shah [3], Kallrath [4], Floudas and Lin [5], and Méndez et al. [6]. We restrict the discussion of the paper to MILP models because we regard these as being the most general ones compared to special purpose methods like genetic algorithms or similar search methods. Despite significant advances in applying MILP approaches for batch scheduling there are still a number of major challenges and questions that remain unresolved. For instance, it is not clear to what extent general methods aimed at complex network structures such as the one in Figure 8.1, can also be effectively applied to commonly encountered structures such as the multistage process shown in Figure 8.2. There are also many detailed questions related to the specific capabilities of the methods for handling a large number of operational issues (e.g., variable or fixed batch size, storage and transfer policies, changeovers), as well as different objectives (e.g., makespan, earliness, or cost minimization). Finally, there are also questions on the strengths and limitations of the various optimization models that have been reported in the literature and the size of problems that one can realistically solve with these models.

It is the objective of this paper to provide a comprehensive review of the state-of-the art of short-term batch scheduling. Our aim is to provide answers to the questions posed in the above paragraph. The paper is organized as follows. We first present a classification for scheduling problems of batch processes, as well as of the features that characterize the optimization models for scheduling. We then discuss representative MILP optimization approaches for general network and sequential batch plants, focusing on discrete and continuous-time models. Computational

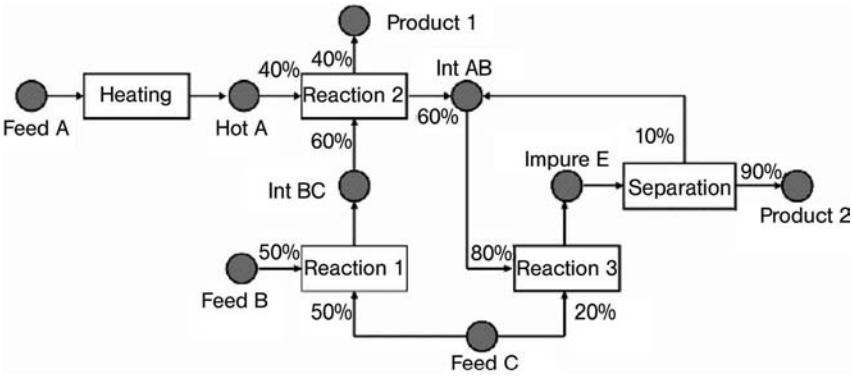

Fig. 8.1 Batch process with complex network structure.

results on a specific case study for a general network are presented in order to compare the performance of several of the methods, particularly discrete- and continuous-time models. We finally conclude the paper with several observations.

8.2 Classification of Batch Scheduling Problems

Different modeling approaches based on mathematical programming techniques, mostly MILP, have been proposed in the literature over the last decade. In order to provide a systematic characterization we present first a general road map for classifying most relevant problem features, which are summarized in Figure 8.3. Here, not only equipment and material issues are considered, but also time representation and demand-related constraints. As can be seen, main features involve 13 major categories, each of which are linked to central problem characteristics. These significantly complicate the task of providing a unified approach that can address all the cases covered in Figure 8.3.

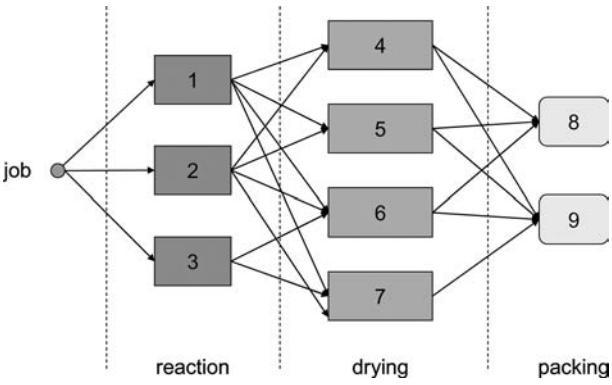

Fig. 8.2 Batch process with sequential structure.

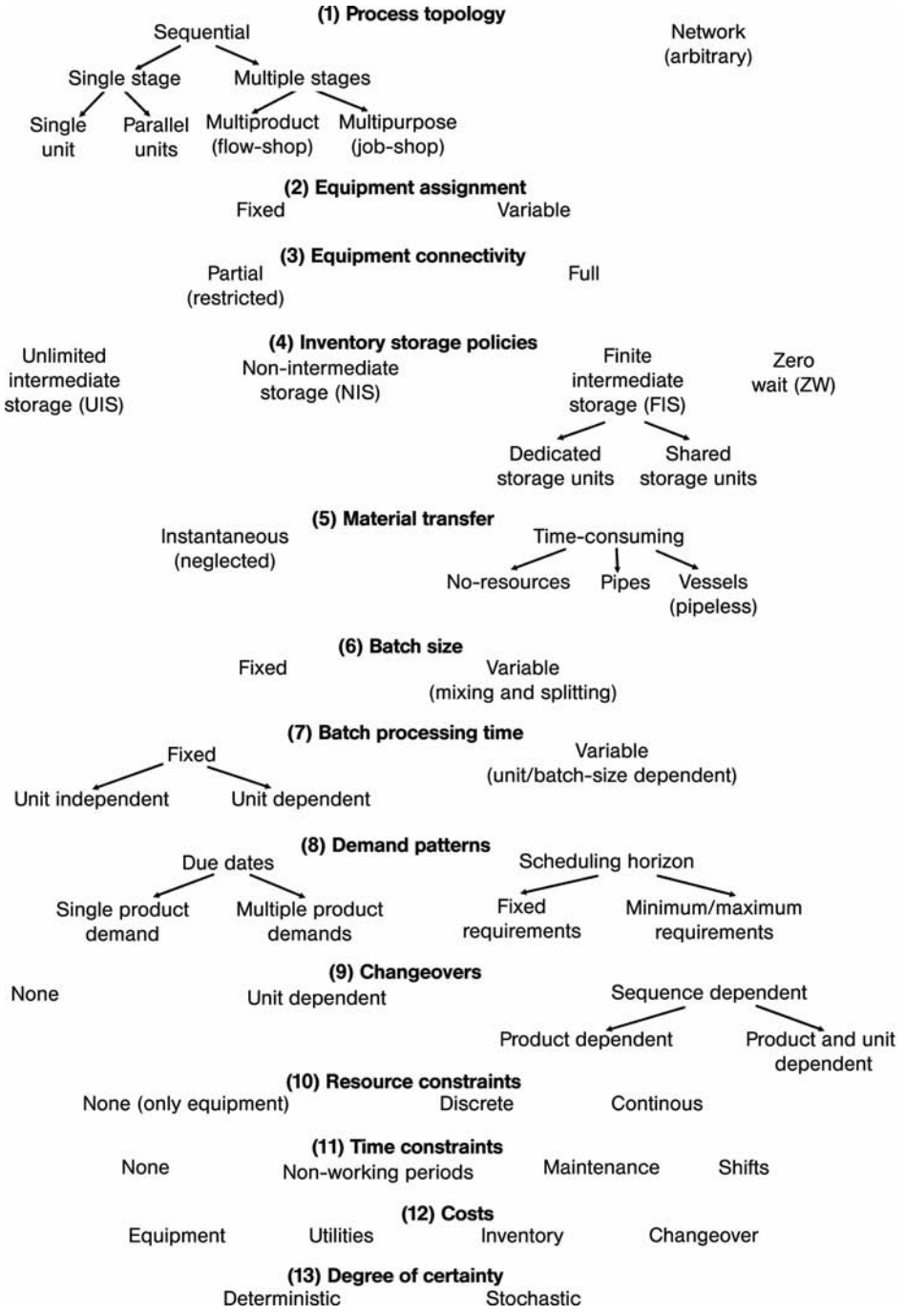

Fig. 8.3 Roadmap for scheduling problems of batch plants.

First, the process layout and its topological implications have a significant influence on problem complexity. In practice many batch processes are sequential, single or multiple stages, where one or several units may be working in parallel in each stage. Each batch needs to be processed following a sequence of stages defined through the product recipes. However, increasingly as the applications become more complex, networks with arbitrary topology must be handled. Complex product recipes involving mixing and splitting operations and material recycles must be handled in these cases. Closely related to topology considerations are requirements/constraints on equipment in terms of its assignment and connectivity, ranging from fixed to flexible arrangements.

Another important aspect of process flow requirements is reflected in inventory policies. These often involve finite and dedicated storage, although frequent cases include shared tanks as well as zero-wait, non-intermediate and unlimited storage policies. Material transfer is often assumed to be instantaneous, but in some cases such as in pipeless plants delay is significant and must be accounted for.

Perhaps a major factor is the handling of batches. For instance, pharmaceutical plants usually handle fixed sizes for which integrity must be maintained (no mixing/splitting), while solvent or polymer plants handle variable sizes that can be split and mixed. Similarly, different requirements on processing times can be found in different industries depending on process characteristics. For example pharmaceutical applications might involve fixed times due to FDA regulations, while solvents or polymers have times that can be adjusted and optimized with process models.

Demand patterns can also vary significantly ranging from cases where due dates must be obeyed to cases where production targets must be met over a time horizon (fixed or minimum). Changeovers are also a very important factor, which is particularly critical in cases of transitions that are sequence dependent on the products, as opposed to simple setups that are only unit dependent.

Resource constraints, aside from equipment (labor, utilities), are also often of great importance and can range from pure discrete to continuous. Practical operating considerations often give rise to time constraints such as non-working periods on the weekend or maintenance periods. Also, while scheduling is often regarded as a feasibility problem, costs such as use of equipment, inventories, changeovers and utilities can have a significant impact in defining an optimal schedule. Finally, there is the issue of the degree to which uncertainty in the data must be accounted for, which is particularly critical for demands as longer time horizons are used.

The classification in Figure 8.3 shows that there is a tremendous diversity of factors that must be accounted for in short-term scheduling making the task of developing unified general methods quite difficult. At the same time, there is the trade-off of having a number of specialized methods that can address specific cases of this classification.

8.3 Classification of Optimization Models for Batch Scheduling

Having presented the general features of typical batch scheduling problems we introduce a roadmap that describes the main features of current optimization

approaches. This section is particularly important because it describes alternative ways of addressing the same problem, which usually have a direct impact on the computational performance, capabilities and limitations of the model. Each option is able to efficiently address a subset of the features described in Figure 8.3.

8.3.1 Time Representation

As illustrated in Figure 8.4, the first and most important issue is the time representation. Depending on whether the events of the schedule can only take place at some predefined time points, or can occur at any moment during the time horizon, optimization approaches can be classified into discrete and continuous-time formulations. Discrete time models are based on the ideas of (1) dividing the scheduling horizon into a finite number of time intervals with predefined duration and (2) allowing the events such as the beginning or ending of tasks to happen only at the boundaries of these time periods. Therefore, scheduling constraints have only to be monitored at specific and known time points, which reduces the problem complexity and makes the model structure simpler and easier to solve, particularly when resource and inventory limitations are taken into account. On the other hand, this type of problem simplification has two major disadvantages. First, the size of the mathematical model as well as its computational efficiency strongly depend on the number of time intervals postulated, which is defined as a function of the problem data and the desired accuracy of the solution. Second, sub-optimal or even infeasible schedules may be generated because of the reduction of the domain of timing decisions. Despite being a simplified version of the original problem, discrete formulations have proved to be very efficient, adaptable and convenient for a wide variety of industrial applications, especially in cases where a reasonable number of intervals is enough to obtain a good approximation.

In order to overcome the previous limitations and generate data-independent models, a wide variety of optimization approaches employ a continuous-time representation. In these formulations, timing decisions are explicitly represented as a set of continuous variables defining the exact times at which the events take place.

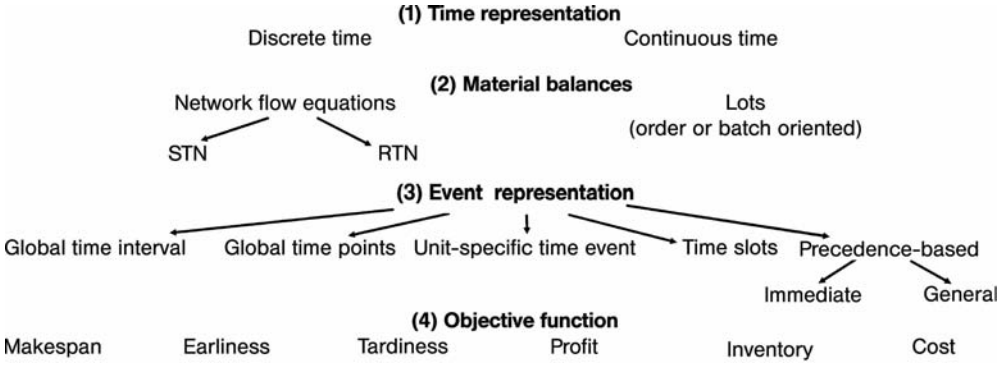

Fig. 8.4 Roadmap for optimization models for short-term scheduling of batch plants.

In the general case, variable time points allow obtaining a significant reduction of the number of variables of the model and at the same time, more flexible solutions in terms of time can be generated. However, the modeling of resource and inventory limitations usually needs the definition of more complicated constraints involving big-M terms which tends to increase the model complexity and the integrality gap and may negatively impact on the capabilities of the method.

8.3.2 Material Balances

The handling of batches gives rise to two types of optimization models. The first category refers to monolithic approaches, which simultaneously deal with the optimal set of batches (number and size), the allocation and sequencing of manufacturing resources and the timing of processing tasks. These methods are able to deal with arbitrary network processes involving complex product recipes. Their generality usually implies large model size formulations and consequently their application are currently restricted to processes involving a small number of processing tasks and rather narrow scheduling horizons. These models employ the state-task network (STN) or the resource-task network (RTN) concept to represent the problem. The STN-based models represent the problem assuming that processing tasks produce and consume states (materials). A special treatment is given to manufacturing resources aside from equipment. In contrast, the RTN-based formulations employ a uniform treatment for all available resources through the idea that processing tasks consume and release resources at their beginning and ending times, respectively.

The second group comprises models that assume that the number of batches of each size is known in advance. These solution algorithms can indeed be regarded as one of the modules of a solution approach for detailed production scheduling, widely used in industry, which decomposes the whole problem into two stages, batching and batch scheduling. The batching problem converts the primary requirements of products into individual batches aiming at optimizing some criterion like the plant workload. Afterwards, the available manufacturing resources are allocated to the batches over time. This approximate two stage approach permits to solve much larger practical problems than monolithic methods. However, they are still restricted to processes comprising sequential product recipes.

8.3.3 Event Representation

In addition to the time representation and material balances, scheduling models are based on different concepts or basic ideas that arrange the events of the schedule over time with the main purpose of guaranteeing that the maximum capacity of the shared resources is never exceeded. As can be seen in Figure 8.5 and Table 8.1, we classified these concepts into five different types of event representations, which have been broadly utilized to develop a variety of mathematical formulations for the batch scheduling problem. Although some event representations are more

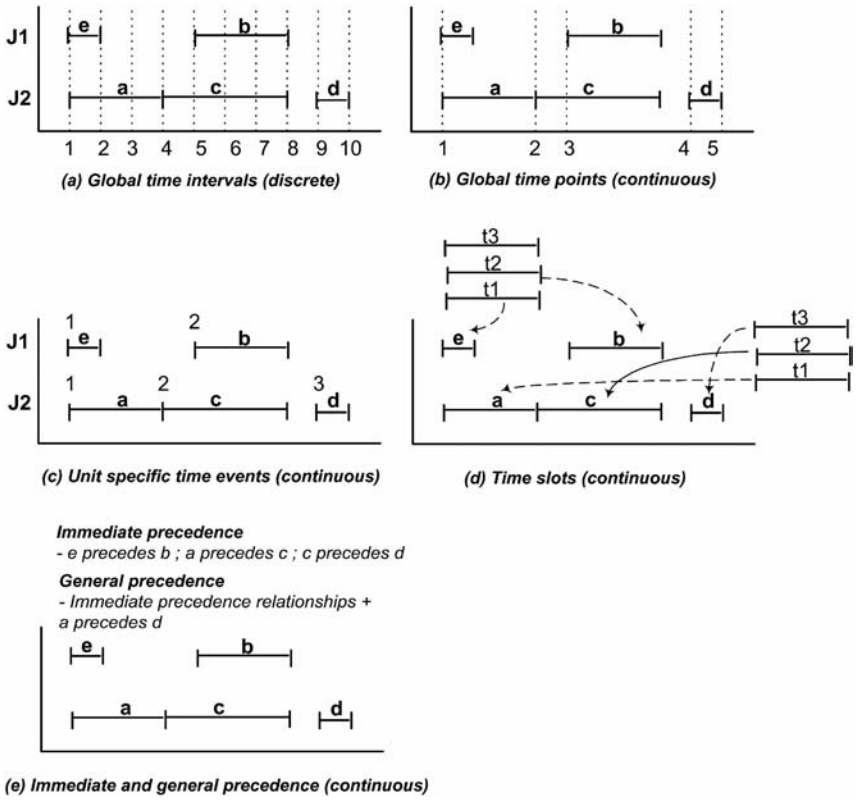

Fig. 8.5 Different concepts for representing time in scheduling problems.

general than others, they are usually oriented towards the solution of either arbitrary network processes requiring network flow equations or sequential batch processes assuming a batch oriented approach.

For discrete time formulations, the definition of global time intervals is the unique option to deal with both general network and sequential processes. In this case, a common time grid valid for all shared resources is predefined and batch tasks are enforced to begin and finish exactly at a point of the grid. Consequently, the original scheduling problem is reduced to a simple allocation problem where the main model decisions define the assignment of the time interval at which every batch task begin, which is modeled through the discrete variable W_{ijt} as shown in Table 8.1. A significant advantage of using a fixed time grid is that time-dependent problem aspects can be modeled in a relatively simple way without compromising the linearity of the model. Some of these aspects comprise hard time constraints, time-dependent utilities cost, multiple product demands and/or raw materials supplies taking place through the scheduling horizon.

In contrast to the discrete-time representation, continuous-time formulations are based on an extensive range of alternative event representations which are focused

Table 8.1 General characteristics of current optimization models.

Time representation	**Discrete**		**Continuous**				
Event representation	Global time intervals	Global time points	Unit-specific time events	Time slots[a)]	Unit-specific immediate precedence[a)]	Immediate precedence[a)]	General precedence[a)]
Main decisions	Lot-sizing, allocation, sequencing, timing	Lot-sizing, allocation, sequencing, timing	Lot-sizing, allocation, sequencing, timing	Allocation, sequencing, timing	Allocation, sequencing, timing	Allocation, sequencing, timing	Allocation, sequencing, timing
Key discrete variables	W_{ijt}, defines if task i starts in unit j at the beginning of time interval t	Ws_{in}/Wf_{in} define if task i starts/ends at time point n $W_{inn'}$, defines if task i starts at time point n and ends at time point n'	$Ws_{in}/W_{in}/Wf_{in}$ define if task i starts/is active/ends at event point n	W_{iljk} define if stage l of batch i is allocated to time slot k of unit j	$X_{ii'j}$ defines if batch i is processed right before of batch i' in unit j XF_{ij} defines if batch i starts the processing sequence of unit j	$X_{ii'}$ defines if batch i is processed right before of batch i' XF_{ij}/W_{ij} defines if batch i starts/is assigned to unit j	$X'_{ii'}$ define if batch i is processed before or after of batch i' W_{ij} defines if batch i is assigned to unit j
Type of process	General network	General network	General network	Sequential	Sequential	Sequential	Sequential
Material balances	Network flow equations (STN or RTN)	Network flow equations (STN or RTN)	Network flow equations (STN)	Batch-oriented	Batch-oriented	Batch-oriented	Batch-oriented

[a)] Batch-oriented formulations assume that the overall problem is decomposed into the lot-sizing and the short-term scheduling issues. The lot-sizing or "batching" problem is solved first in order to determine the number and size of "batches" to be scheduled.

on different types of batch processes. For instance, the available options to deal with general network processes comprise the definition of global time points and unit-specific time events whereas in the case of sequential processes the alternatives involve the use of time slots and different batch precedence-based approaches. The global time point representation corresponds to a generalization of global time intervals where the timing of time intervals is treated as new model variables. In this case, a common and variable time grid is defined for all shared resources. The beginning and the finishing times of the set of batch tasks are linked to specific time points through the key discrete variables reported in Table 8.1. Both for continuous STN as well as RTN-based models, limited capacities of resources just need to be monitored at a small number of variable time points in order to guarantee the feasibility of the solution. Models following this direction are relatively simple to implement even for general scheduling problems. In contrast to global time points, the idea of unit-specific time events defines a different variable time grid for each shared resource, allowing different tasks to start at different moments for the same event point. These models make use of the STN representation. Because of the heterogeneous locations of the event points, the number of events required is usually smaller than in the case of global time points. However, the lack of reference points for checking the limited availability of shared resources makes the formulation much more complicated. Special constraints and additional variables need to be defined for dealing with resource-constrained problems. The usefulness and computational efficiency of the formulations based on global time points or unit-dependent time events strongly depends on the minimum number of time points or events required to generate the optimal solution. Since this number is unknown *a priori*, a practical criterion is to determine it through an iterative procedure, during which the number of variable points or events is increased by 1 until there is no improvement in the objective function. This means that a significant number of instances of the model need to be solved for each scheduling problem which may lead to a high total CPU time. It is worth to mention that this stopping criterion cannot guarantee the optimality of the solution and in some cases may also stop with a poor feasible schedule. In a few words, the iterative procedure aims at generating the best possible schedule with the minimum computational effort, which corresponds to a reasonable practical criterion.

The previous general continuous-time formulations are mostly oriented towards arbitrary network processes. On the other hand, different continuous-time formulations focused their attention on particular features of a wide variety of sequential processes. One of the first contributions following this direction is based on the concept of time slots, which stand for a set of predefined time intervals with unknown durations. The main idea is to postulate an appropriate number of time slots for each processing unit in order to allocate them to the batches to be processed. The definition of the number of time slots required is not a trivial decision and represents an important trade-off between optimality and computational performance. Other alternative approaches for sequential processes were developed based on the concept of batch precedence. Model variables defining the processing sequence of batch tasks are explicitly embedded into these formulations and, consequently,

sequence dependent changeover times can be incorporated in a straightforward manner. The concept of batch precedence can be applied to the immediate or the general batch predecessor, which originates three different types of basic mathematical formulations. The idea of unit-specific immediate precedence determines the processing sequencing of batches in each piece of equipment by defining the sequencing variables $X_{ii'j}$ reported in Table 8.1. In this way, allocation and sequencing decisions are made simultaneously through a unique set of model variables. Because of that, the size of the model in terms of variables is notably increased which represents the major disadvantage of this idea. To overcome the previous limitation, the concept here called immediate batch precedence can be employed. In contrast to the previous model, this idea decouples allocation and sequencing decisions in two different sets of model variables W_{ij} and $X_{ii'}$, as described in Table 8.1. This modification allows one to obtain a significant reduction of the model size and the computational effort. Finally, the generalized precedence notion extends the immediate precedence concept to not only consider the immediate predecessor but also all batches processed before in the same processing sequence. In this way, the basic idea is completely generalized which simplifies the model and decreases the number of sequencing variables. Apart from decoupling allocation and sequencing decision, this concept requires a single sequencing variable for each pair of batch tasks that can be allocated to the same resource. In this way, the formulation results simpler and smaller than those based on the immediate predecessor. In addition, another advantage of this approach is that the utilization of different types of renewable shared resources such as processing units, storage tanks, utilities and manpower can be efficiently handled through a unique set of sequencing variables without compromising the optimality of the solution. A common weakness of precedence-based formulations is that the number of sequencing variables is usually very significant for real-world applications.

8.3.4 Objective Function

Different measures of the quality of the solution can be used for scheduling problems. However, the criterion selected to be optimized usually has a direct effect on the model computational performance. In addition, some objective functions can be very hard to implement for some event representations, requiring additional variables and complex constraints.

8.4 Review of Scheduling Models

Having introduced a general road map for classifying problems and models for batch scheduling we present a brief review on the specific models that have been proposed in the literature (for model details see Méndez et al. [6]) .

8.4.1 Global Time Intervals (Discrete Time)

The event representation based on the definition of global time intervals predefines a unique time grid for all shared resources involved in the scheduling problem, such as processing units and storage tanks. In addition, if a discrete time grid is used in combination with this type of event representation, the boundaries of these intervals are known data and correspond to the only time points at which the set of tasks to be scheduled can take place, i.e., starting and ending times. In this way, the availability of states and resources only has to be monitored at a finite number of predefined time points. Since the duration of the time intervals must be equal to the greatest common factor of the problem data, constant processing times are usually considered, which may not always be the real situation. On the other hand, this assumption in combination with the use of fixed time points allow generating tighter constraints with no big-M terms, which reduce the integrality gap and improve the computational performance.

8.4.1.1 STN-based Discrete Formulation

The most relevant contribution for global discrete time models is the State Task Network representation proposed by Kondili et al. [7] and Shah et al. [8] (see also [9]). The model involves 0–1 variables for allocating tasks to processing units at the beginning of the postulated time intervals. Most important equations comprise mass balances over the states, constraints on batch sizes and resource constraints. The STN model covers all the features that are included at the column on discrete time in Table 8.1.

8.4.1.2 RTN-based Discrete Formulation

A simpler and general discrete time scheduling formulation can also be derived by means of the Resource Task Network concept proposed by Pantelides [10]. The major advantage of the RTN formulation over the STN counterpart arises in some problems involving many identical pieces of equipment. In these cases, the RTN formulation introduces a single binary variable instead of the multiple variables used by the STN model. The RTN-based model also covers all the features at the column on discrete time in Table 8.1. In order to deal with different types of resources in a uniform way, this approach requires only three different classes of constraints in terms of three types of variables defining the task allocation, the batch size, and the resource availability. Briefly, this model reduces the batch scheduling problem to a simple resource balance problem carried out in each predefined time period.

Although resource-task network formulations are able to deal with sequence-dependent changeovers, they need to define explicitly additional tasks associated to each type of cleaning requirement, as well as different states of cleanliness for each processing unit. Since changeover tasks must be performed in a specific unit, the definition of many identical processing units as the same resource can no longer be used. The available processing resources must be defined individually. In this way,

different equipment states allow the model to guarantee that the corresponding cleaning task has been performed before starting a particular processing task. The definition of cleaning tasks significantly increases the model size and the computational requirements and the problem may become intractable even if a modest number of changeovers need to be considered.

We can then conclude that while the discrete time STN and RTN models are quite general and effective in monitoring the level of limited resources at the fixed times, their major weakness in terms of capability is the handling of relatively small processing and changeover times. Regarding the objective function, these models can easily handle profit maximization (cost minimization) for a fixed time horizon. Intermediate due dates can be easily modeled. Other objectives such as makespan minimization are more complex to implement since the time horizon and, in consequence, the number of time intervals, are unknown *a priori* (see [11]).

8.4.2 Global Time Points (Continuous Time)

8.4.2.1 STN-based Continuous Formulation

A wide variety of continuous-time formulations based both on the STN-representation and the definition of global time points have been developed in the last years. Most of the work falling into this category is represented by the approaches proposed by Schilling and Pantelides [12], Zhang and Sargent [13], Mockus and Reklaitis [14, 15], Lee et al. [16], Giannelos and Georgiadis [17], and Maravelias and Grossmann [11].

For instance, the formulation by Maravelias and Grossmann [11] is able to handle most of the aspects found in standard batch processes (see first column for continuous models in Table 8.1). This approach is based on the definition of a common time grid that is variable and valid for all shared resources. This definition involves time points n occurring at unknown time Tn, $\mathrm{n} = 1, 2 \ldots |\mathrm{N}|$, when N is the set of time points. To guarantee the feasibility of the material balances at any time during the time horizon of interest, the model imposes that all tasks starting at a time point n must occur at the same time Tn. However, in order to have more flexibility in terms of timing decisions, the ending time of tasks does not necessarily have to coincide with the occurrence of a time point n, except for those tasks that need to transfer the material with a zero wait policy (ZW). For other storage policies it is assumed that the equipment can be used to store the material until the occurrence of next time point. Given that the model assumes that each task can be performed in just one processing unit, task duplication is required to handle alternative equipment and unit dependent processing times.

8.4.2.2 RTN-based Continuous Formulation

Different continuous-time formulations were also developed based on the RTN concept initially proposed by Pantelides [10]. The work developed by Castro et al. [18] which has been improved by Castro et al. [19] falls into this group. Major assumptions of this approach are (1) processing units are considered individually, i.e., one resource is defined for each available unit, and (2) only one task can be

performed in any given equipment resource at any time (unary resource). These assumptions increase the number of tasks and resources to be defined but, at the same time allow reducing the model complexity. This model also covers all the features given at the column on continuous-time and global time points in Table 8.1.

In the same way as in the previous STN-based continuous-time formulation, a set of global time points N is predefined where the first time point takes place at the beginning $T1 = 0$ whereas the last at the end of the time horizon of interest $Tn = H$. However, the main difference in comparison to the previous model arises in the definition of the allocation variable $W_{inn'}$ which is equal to 1 whenever task i starts at time point n and finishes at or before time point $n' > n$. In this way, the starting and finishing time points for a given task i are defined through only one set of binary variables. It should be noted that this definition on the one hand makes the model simpler and more compact, but on the other hand it significantly increases the number of constraints and variables to be defined.

We can conclude that the continuous-time STN and RTN models based on the definition of global time points are quite general. They are capable of easily accommodating a variety of objective functions such as profit maximization or makespan minimization. However, events taking place during the time horizon such as multiple due dates and raw material receptions are more complex to implement given that the exact position of the time points is unknown.

8.4.3 Unit-specific Time Event

In order to gain more flexibility in timing decisions without increasing the number of time points to be defined, an original concept of event points was introduced by Ierapetritou and Floudas [20], which relaxes the global time point representation by allowing different tasks to start at different moments in different units for the same event point. Subsequently, the original idea was implemented in the work presented by Vin and Ierapetritou [21] and Lin et al. [22] and recently enhanced by Janak et al. [23]. Particularly, the last work corresponds to the most general model falling into this category. A wide variety of new variables and constraints were added to the original formulation in order to deal with relevant aspects of the scheduling problem such as multiple storage policies and resource limitations. Despite its generality, the model still involves a large number of interconnected constraints which makes the formulation difficult to understand and implement. Special constraints are often required for solving some cases as reported in Janak et al. [24].

8.4.4 Time Slots

The previous general continuous-time formulations are mostly oriented towards general network processes. On the other hand, different continuous-time formulations focused their attention on the particular features of a wide variety of sequential

processes. One of the first contributions following this direction is based on the concept of time slots. Relevant work on this area is represented by the formulations developed by Pinto and Grossmann [25, 26], Chen et al. [27], and Lim and Karimi [28]. The central point of this formulation is the definition of the minimum number of time slots to generate the optimal solution. Heuristic approaches are usually used to estimate this critical number.

8.4.5 Precedence-based Approaches

With the main purpose of developing more efficient optimization models for batch sequential processes, especially those involving sequence-dependent changeovers, different approaches were proposed based on the concept of batch precedence.

8.4.5.1 Unit-specific Immediate Precedence

The concept of immediate precedence in each unit defines the processing sequence of batches in each process equipment through the binary variable $X_{ii'j}$ which becomes equal to 1 whenever batch i is processed immediately before batch i' in the processing sequence of unit j. Allocation and sequencing decisions are modeled through a unique set of decision variables. Cerdá et al. [29] presented such a formulation where a single-stage batch plant with multiple equipment working in parallel is assumed.

8.4.5.2 Immediate Precedence

An alternative formulation is based on the concept of immediate batch precedence. In contrast to the previous model, allocation and sequencing decisions are divided into two different sets of binary variables. This idea is described in the work presented by Méndez et al. [30], where a single-stage batch plant with multiple equipment in parallel is assumed. Relevant work following this direction can also be found in Gupta and Karimi [31]. Key variables are defined as follows:

WF_{ij} denotes that batch i is the first processed in unit j; W_{ij} denotes that batch i is allocated to unit j but not in the first place and; $X_{ii'}$ denotes that batch i is processed right before batch i'.

8.4.5.3 General Precedence

The generalized precedence notion extends the immediate precedence concept not only to consider the immediate predecessor, but also all batches processed before in the same processing sequence. In this way, the original basic idea is completely generalized. A single sequencing variable $X_{ii'}$ is defined for each pair of batch tasks that can be allocated to the same shared resource. Thus, whenever a pair of batches (i,i') is allocated to the same manufacturing resource, the sequencing variable $X_{ii'}$ takes the value 0/1 indicating that batch i is processed before/after batch i'. Therefore, the major advantage of this approach is that the utilization of different types of renewable shared resources such as processing units, storage tanks, utilities and manpower can be efficiently handled through the same

set of sequencing variables without compromising the optimality of the solution. All resources are treated in a uniform way for making the required allocation and sequencing decisions. Part of the work falling into this category is represented by the approaches developed by Méndez et al. [32] and Méndez and Cerdá [33–35].

8.5 Computational Comparison Discrete vs Continuous Approaches

In order to test the effectiveness of discrete and continuous-time representations, we performed a computational comparison using MILP models that rely on the definition of global time intervals [8] or global time points [11]. The generality, efficiency and easy implementation of these formulations were the main reasons to choose them within a variety of alternatives. The case study selected is based on the benchmark problem proposed by Westenberger and Kallrath [36]. This case covers most of the features that contribute to the high complexity of batch scheduling (network structure, variable batch size, storage constraints, and different transfer policies). It has, however, the important simplification that neither changeover times nor non-zero transfer times are considered. A process representation that relies on the state task network (STN) concept introduced by Kondili et al. [7] is shown in Figure 8.6. Problem data related to states and processing tasks are also displayed. The STN is a directed graph that consists of three key elements: (1) *state nodes* representing the feeds (state 1), intermediates (states 2 to 14) and final products (states 15 to 19); (2) *task nodes* representing the process operations which transform material from one or more input states into one or more output states and; (3) *arcs* that link states and tasks indicating the flow of materials. State and task nodes are denoted by circles and rectangles, respectively. As shown in Figure 8.6, this batch process involves 17 processing tasks, 19 states and 9 production units. Fractions of input and output goods are marked on the arcs indicating the particular flow of material. In general, these proportions are fixed. However, the output fractions of task 2 are variable, which means that a fraction x of the total output is allotted to state 3 and the remaining amount to state 4, where the fraction x is allowed to vary between 0.2 and 0.7. Moreover, it is assumed that there is sufficient initial stock of raw material (state 1) and unlimited capacity to store the required raw material (state 1) and the final products (states 15 to 19). Different intermediate storage polices are taken into account for different states. For instance, a zero-wait transfer policy (ZW) is assumed for states 6, 10, 11 and 13 whereas a finite dedicated intermediate storage capacity (FIS) is considered for the remaining intermediate states. Based on the roadmap introduced in Section 8.2 (see Figure 8.3), a summary of the main problem features are given in Table 8.2.

The computational results for the case studies allow the comparison and study of the efficiency and limitations of specific modeling approaches. However, it is worth mentioning that problem data involves only integer processing times, which represents a fortunate situation for discrete time models since no special provisions

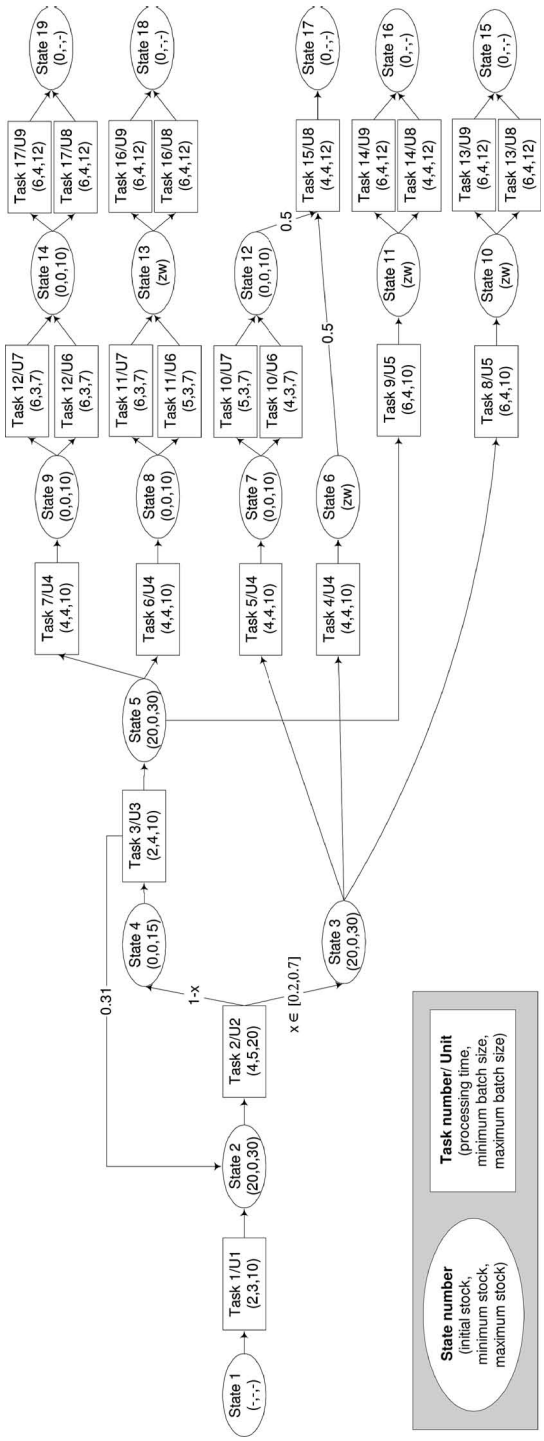

Fig. 8.6 STN-based representation for case study.

Table 8.2 Case study features

Feature	Type
Process topology	Network
Equipment assignment	Variable
Equipment connectivity	Full
Inventory storage policies	FIS (dedicated) ZW and UIS
Material transfer	Instantaneous
Batch size	Variable
Batch processing time	Fixed – unit dependent
Demand patterns	Scheduling horizon
Changeovers	None
Resource constraints	None
Time Constraints	None
Costs	None
Degree of certainty	Deterministic

for rounding are needed. In order to evaluate the influence of the objective function on the computational performance, we solved two different instances: minimizing makespan (case a) and maximizing profit (case b). For the makespan, product demands of 20 tons for states 15, 16 and 17 have to be satisfied. Instances comprising a larger number of demands were not possible to solve in a reasonable time by using the selected optimization approaches, which suggests limitations that may be faced when addressing realworld problems. When the profit was maximized, minimum product demands of 10, 10, 10, 5 and 10 tons for states 15, 16, 17, 18 and 19 were considered. Also, original discrete processing times were slightly modified in order to use more realistic data that enforces a finer discretization. Therefore, processing times of 2, 4, 5 and 6 hours were changed to 1.3, 3.7, 4.2 and 5.6 hours, respectively. Raw material cost, inventory cost, unit operating cost and product values considered to estimate the total profit of the schedule.

Gantt charts for the optimal solutions for the two instances are shown in Figures 8.7 and 8.8. The corresponding state number is shown within each rectangle. Model sizes, computational times and objective values are summarized in Table 8.3. The number of time intervals or points that was required in each case

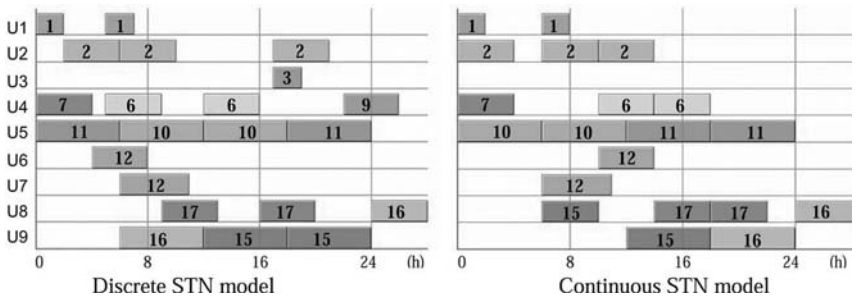

Fig. 8.7 Gantt charts for case a (Makespan minimization).

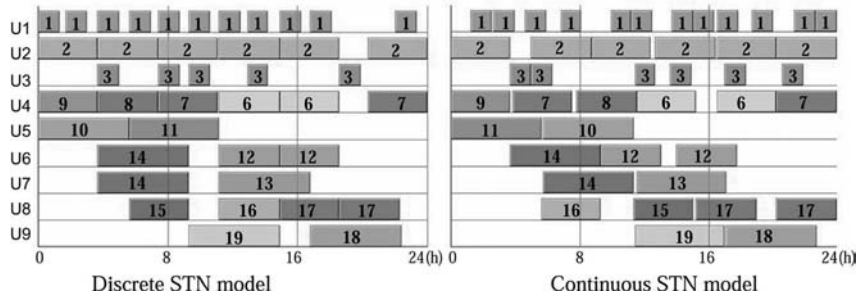

Fig. 8.8 Gantt charts for case b (Profit maximization).

is also reported in brackets. For case a, it can be observed that both formulations are able to reach the same objective value of 28 h. Thirty time intervals of 1 h duration were defined for the discrete time, whereas 8 variable time points were required for the continuous representation. An iterative procedure that increases the time horizon by 1 h was implemented for the discrete time model. The iterative procedure described in Section 8.2 was utilized to define the minimum number of variable time points required in the continuous-time formulation. The computational effort corresponding to the last iteration for each case is only reported in Table 8.3. However, we would like to remark that the total computational cost for both cases is significantly higher and depends on the starting point of the iterative procedure.

For the case of profit maximization, a fixed time horizon of 24 h was defined. This scheduling horizon was represented through 240 fixed time intervals and 14 variable time points in the discrete and continuous-time models, respectively. Longer horizons could not be solved in a reasonable time. In this case, the schedule found through the discrete time representation was slightly better than the continuous one, probably because the number of time points required for generating the discrete solution exceeds the current continuous model capabilities. Continuous models comprising more than 14 time points only generated poor solutions with significant computational effort.

Although the usefulness and performance of continuous and discrete time models strongly depends on the particular problem and solution characteristics, our

Table 8.3 Computational results for discrete and continuous STN-based models

Case study	Event representation (time intervals or points)	Binary vars, cont. vars, constraints	LP relaxation	Objective function	CPU time[a)]	Relative gap
(a)	Global time intervals (30)	720, 3542, 6713	9.9	28	1.34	0.0
	Global time points (8)	384, 2258, 4962	24.2	28	108.39	0.0
(b)	Global time intervals (240)	5760, 28322, 47851	1769.9	1425.8	7202	0.122
	Global time points (14)	672, 3950, 8476	1647	1407.4	258.54	0.042

a) Seconds on Pentium IV PC with CPLEX 8.1 in GAMS 21.

experience in the area and the results obtained from the case study performed allow us to draw the following interesting conclusions for general scheduling problems: (1) despite the fact that discrete time models are usually larger than its continuous counterpart, its simpler model structure tends to significantly reduce the CPU time requirements when a reasonable number of time intervals is postulated (around 250 intervals usually appears as a tractable number); (2) the complex structure of continuous-time models makes them useful only for problems that can be solved with a relatively small number of time points (15 points may be a current upper bound for generic process); (3) discrete time models may generate better solutions than continuous ones whenever the time discretization is a good approximation to the real data; (4) the objective function selected may have a significant impact on the computational cost and the model efficiency. Computational costs ranging from 1 s to 7202 s were obtained for this case study.

8.6 Concluding Remarks and Future Directions

This chapter has shown through a classification of problem types the great diversity involved in short-term batch scheduling problems. A general classification of optimization models was used as framework for describing the major optimization approaches that have emerged over the last decade in this area. A qualitative description of the methods has been given emphasizing the main ideas and highlighting their strengths and limitations. Finally, a specific example problem was presented to illustrate the performance of discrete and continuous-time methods discussed in the review.

While there are clearly still a number of limitations of the MILP optimization models, it is also clear that very significant progress has been made in terms of scope and solution efficiency. For instance, both the STN and RTN models are very general models and cover most of the features presented in the classification of batch processes in Figure 8.3. Also, in terms of efficiency, not only has the quality of MILP models improved through tighter relaxations (e.g., STN model), or more compact formulations (e.g., RTN model and general precedence), but solution methods for MILP have improved dramatically. As an illustration of this point it is interesting to note that the first MILP model for Figure 8.1 by Kondili et al. [7], involving 72 0-1 variables, 179 continuous variables and 250 constraints, required in year 1987 on a VAX computer: 908 s and 1466 nodes and in year 1992 on a SUN-Sparc: 119 s and 419 nodes [8]. In both cases, an ad hoc branch and bound code based on MINOS were used. That same problem requires today 0.45 s and 22 nodes on an IBM laptop using CPLEX 8.1. Thus, what appeared to be computationally very challenging 15 years ago has become computationally trivial. An important lesson here is that computational efficiency is a moving window: MILP problems that today are regarded as computationally unsolvable are no longer several years later.

Despite the advances mentioned above there are still clearly a number of major problems that need to be addressed and that will require significant research. Four major problems that we can cite are the following:

1. Perhaps the biggest gap in terms of effective models is the capability of simultaneously handling changeovers, inventories and resource constraints. Sequential methods can handle well the first, while discrete time models (e.g., STN, RTN), can handle well the last two. While continuous-time models with global time intervals can theoretically handle all of the three issues, they are at this point still much less efficient than discrete time models, and therefore require further research.
2. Despite advances in MILP solution methods, problem size is still a major issue since scheduling problems are known to be NP-hard (i.e., exponential increase of computation time with size in worst case). While effective modeling can help to overcome to some extent the issue of computational efficiency, special solution strategies such as decomposition and aggregation are needed in order to address the ever increasing sizes of real-world problems.
3. Short-term scheduling models rely on simplified models (e.g., fixed processing times, changeovers, etc.). In a number of applications that are not dictated by recipe production (e.g., polymers, specialties), detailed dynamic models for predicting processing and changeover times are required, which when incorporated in scheduling problems give rise to mixed-integer dynamic optimization problems that are at this point still very difficult to solve.
4. While short-term scheduling problems are important by themselves, they rarely arise in isolation, but they have to be considered as part of a production planning problem. Thus, the integration and simultaneous optimization of planning and scheduling so as to achieve consistency and optimality remains a major outstanding problem.

   It is hoped that the above points will stimulate further research in the area as it is clear that significant work is still required in this area.

Acknowledgements

The authors are grateful for financial support from ABB Corporate Research.

References

1 Pekny, J.F. and Reklaitis, G.V. (1998) Towards the convergence of theory and practice: A technology guide for scheduling/planning methodology. *Proceedings of the third international conference on foundations of computer-aided process operations*, pp. 91–111.

2 Pinto, J.M. and Grossmann, I.E. (1998) Assignments and sequencing models of the scheduling of process systems. *Annals of Operations Research*, **81**, 433–466.

3 Shah, N. (1998) Single- and multisite planning and scheduling: Current status and future challenges. *Proceedings of the third international conference on foundations of computer-aided process operations*, pp. 75–90.

4 Kallrath, J. (2002) Planning and scheduling in the process industry. *OR Spectrum*, **24**, 219–250.

5 Floudas, C.A. and Lin, X. (2004). Continuous-time versus discrete-time approaches for scheduling of chemical

processes: a review. *Computers and Chemical Engineering*, **28**, 2109–2129.

6 Méndez, C.A., Cerdá, J., Grossmann, I.E., Harjunkoski, I. and Fahl, M. (2006) State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Computers and Chemical Engineering*, **30**, 6–7, 913–946.

7 Kondili, E, Pantelides, C.C. and Sargent, W.H. (1993) A general algorithm for short-term scheduling of batch operations – I. MILP formulation. *Comput. Chem. Eng.*, **2**, 211–227.

8 Shah, N., Pantelides, C.C. and Sargent, W.H. (1993) A general algorithm for short-term scheduling of batch operations – II. Computational issues. *Comput. Chem. Eng.*, **2**, 229–244.

9 Rodrigues, M.T.M., Latre, L.G. and Rodrigues, L.C.A. (2000) Short-term planning and scheduling in multipurpose batch chemical plants: A multi-level approach. *Comput. Chem. Eng.*, **24**, 2247–2258.

10 Pantelides, C.C. (1994) Unified frameworks for optimal process planning and scheduling, in *Foundations of Computer-Aided Process Operations*, CACHE, New York, pp. 253–274.

11 Maravelias, C.T. and Grossmann, I.E. (2003) Minimization of makespan with discrete-time state-task network formulation.. *Ind. Eng. Chem. Res.*, **42**, 6252–6257.

12 Schilling, G. and Pantelides, C.C. (1996) A simple continuous-time process scheduling formulation and a novel solution algorithm. *Comput. Chem. Eng.* **20**, 1221–1226.

13 Zhang, X. and Sargent, W.H. (1996) The optimal operation of mixed production facilities – A general formulation and some approaches for the solution. *Comput. Chem. Eng.*, **20**, 897–904.

14 Mockus, L. and Reklaitis, G.V. (1999a) Continuous-time representation approach to batch and continuous process scheduling. 1. MINLP formulation. *Ind. Eng. Chem. Res.*, **38**, 197–203.

15 Mockus, L. and Reklaitis, G.V. (1999b) Continuous-time representation approach to batch and continuous process scheduling. 2. Computational issues. *Ind. Eng. Chem. Res.*, **38**, 204–210.

16 Lee, K., Park, H. and Lee, I. (2001) A novel nonuniform discrete time formulation for short-term scheduling of batch and continuous processes. *Ind. Eng. Chem. Res.*, **40**, 4902–4911.

17 Giannelos, N.F. and Georgiadis, M.C. (2002) A simple new continuous-time formulation for short-term scheduling of multipurpose batch processes. *Ind. Eng. Chem. Res.*, **41**, 2178–2184.

18 Castro, P., Barbosa-Póvoa, A.P.F.D. and Matos, H. (2001) An improved RTN continuous-time formulation for the short-term scheduling of multipurpose batch plants. *Ind. Eng. Chem. Res.*, **40**, 2059–2068.

19 Castro, P.M., Barbosa-Póvoa, A.P., Matos, H.A., and Novais, A.Q. (2004) Simple continuous-time formulation for short-term scheduling of batch and continuous processes. *Ind. Eng. Chem. Res.*, **43**, 105–118.

20 Ierapetritou, M.G. and Floudas, C.A. (1998) Effective continuous-time formulation for short-term scheduling. 1. Multipurpose batch processes. *Ind. Eng. Chem. Res.*, **37**, 4341–4359.

21 Vin, J.P. and Ierapetritou, M.G. (2000) A new approach for efficient rescheduling of multiproduct batch plants. *Ind. Eng. Chem. Res.*, **39**, 4228–4238.

22 Lin, X., Floudas, C.A., Modi, S. and Juhasz, N.M. (2002) Continuous-time optimization approach for medium-range production scheduling of a multiproduct batch plant. *Ind. Eng. Chem. Res.*, **41**, 3884–3906.

23 Janak, S.L., Lin, X. and Floudas, C.A. (2004) Enhanced continuous-time unit-specific event-based formulation for short-term scheduling of multipurpose batch processes: Resource constraints and mixed storage policies. *Ind. Eng. Chem. Res.*, **43**, 2516–2533.

24 Janak, S.L., Lin, X. and Floudas, C.A. (2005) Additions and corrections. *Ind. Eng. Chem. Res.*, **44**, 426.

25 Pinto, J.M. and Grossmann, I.E. (1995) A continuous-time mixed integer linear programming model for

short-term scheduling of multistage batch plants. *Ind. Eng. Chem. Res.*, **34**, 3037–3051.

26 Pinto, J.M. and Grossmann, I.E. (1996) An alternate MILP model for short-term scheduling of batch plants with preordering constraints. *Ind. Eng. Chem. Res.*, **35**, 338–342.

27 Chen, C., Liu, C., Feng, X., and Shao, H. (2002) Optimal short-term scheduling of multiproduct single-stage batch plants with parallel lines. *Ind. Eng. Chem. Res.*, **41**, 1249–1260.

28 Lim, M. and Karimi, I.A. (2003) Resource-constrained scheduling of parallel production lines using asynchronous slots. *Ind. Eng. Chem. Res.*, **42**, 6832–6842.

29 Cerdá, J., Henning, G.P., Grossmann, I.E. (1997) A mixed-integer linear programming model for shortterm scheduling of single-stage multiproduct batch plants with parallel lines. *Ind. Eng. Chem. Res.*, **36**, 1695.

30 Méndez, C.A., Henning, G.P. and Cerdá, J. (2000) Optimal scheduling of batch plants satisfying multiple product orders with different due-dates. *Comput. Chem. Eng.*, **24**, 2223–2245.

31 Gupta, S. and Karimi, I.A. (2003) An improved MILP formulation for scheduling multiproduct, multistage batch plants. *Ind. Eng. Chem. Res.*, **42**, 2365–2380.

32 Méndez, C.A., Henning, G.P. and Cerdá, J. (2001) An MILP continuous-time approach to short-term scheduling of resource-constrained multistage flowshop batch facilities. *Comput. Chem. Eng.*, **25**, 701–711.

33 Méndez, C.A. and Cerdá, J. (2003a) An MILP continuous-time framework for short-term scheduling of multipurpose batch processes under different operation strategies. *Optim. Eng.*, **4**, 7.

34 Méndez, C.A. and Cerdá, J. (2003b) Dynamic scheduling in multiproduct batch plants. *Comput. Chem. Eng.*, **27**, 1247–1259.

35 Méndez, C.A. and Cerdá, J. (2004) An MILP framework for batch reactive scheduling with limited discrete resources. *Comput. Chem. Eng.*, **28**, 1059–1068.

36 Westenberger, H. and Kallrath, J. (1995) Formulation of a job shop problem in process industry. Internal report, Bayer AG, Leverkusen, and BASF AG, Ludwigshafen.

9
Uncertainty Conscious Scheduling by Two-Stage Stochastic Optimization

Jochen Till, Guido Sand, and Sebastian Engell

The strong global competition has been increasing the pressure to an efficient operation of chemical batch plants. Flexible batch plants are used to react quickly to changes in customers' demands. The variations in the demands as well as, e.g., the prices of raw materials, or the yields of the production process are not exactly known at the time of scheduling. When these uncertainties are not sufficiently considered in the scheduling, the operations will lead to lower profits or even losses.

The use of uncertainty conscious schedulers – schedulers which consider the uncertain parameters already at the scheduling stage – have the potential to lead to a significant increase in the profit compared to deterministic methods. However, the resulting optimization problems are usually of large scale and it is difficult to solve them within the short period of time available in a real-time environment.

In the first part of this contribution, we show the benefit of using a stochastic model in a moving horizon based real-time scheduler by means of a small example. In the second part we consider the algorithmic issues in solving the resulting large stochastic optimization problems. After a review on stochastic programming, a new hybrid evolutionary algorithmic approach is presented. This algorithm exploits the problem structure of the stochastic optimization model. The new approach is compared to state-of-the-art solvers by means of the real-world scheduling problem discussed in the chapter by Sand ("Engineered Mixed-Integer Programming in Chemical Batch Scheduling"). The new algorithmic approach shows a competitive performance and provides good solutions in short computation times.

9.1
Introduction

In most chemical batch scheduling problems the underlying data is not exactly known at the time the schedule has to be generated. Typical sources of uncertainties are (1) failures of reactors, equipment, and resources, (2) varying processing times, (3) varying product qualities, and (4) varying customer's demands.

If a schedule is computed based on a model that does not consider uncertainties, this schedule can become suboptimal or even infeasible when the situation has changed. For example, a schedule can become suboptimal if a batch is unexpectedly of inferior quality and the revenues are a function of its quality. A schedule can become infeasible if there is an unexpected plant failure that reduces the plant capacity: a batch has to be immediately transferred to another unit, but no unit is available. Then it is impossible to modify the infeasible schedule to a feasible one.

The vast majority of model based chemical batch scheduling approaches ignore the uncertainty by assuming the data to be certainly known. In contrast, *uncertainty conscious* scheduling approaches do not ignore the uncertainties. They can be classified according to two approaches [1, 2]:

- *Reactive scheduling* is an online procedure which modifies nominal schedules in reaction to the occurrence of an unexpected event. Reactive scheduling is traditionally used to handle short-term uncertainties in parameters as, e.g., processing times, or equipment failures. The underlying-models themselves usually do not incorporate information on the uncertainty.

- *Stochastic scheduling* takes the uncertainty into account explicitly by using uncertainty conscious models. Stochastic scheduling is typically applied offline to generate schedules which are robust against parameter variations in the sense that only marginal online adjustments are necessary to maintain the quality and the feasibility of the schedules. However, the modeling of uncertainties typically leads to a significant increase in the size and the complexity of the models such that their solution in reasonable response times becomes a demanding task.

In the first part of this chapter (Section 9.2), we present an uncertainty conscious scheduling approach that combines reactive scheduling and stochastic scheduling by using a moving horizon scheme with an uncertainty conscious model. In this approach, it is assumed that decisions are made sequentially and that the effect of the revealed uncertainties can be partially compensated by later decisions. The sequence of decisions and observations is modeled by a sequence of two-stage stochastic programs.

In the second part starting with Section 9.3, we consider algorithmic issues in solving large two-stage stochastic programs. We start with a review on stochastic programming. Then a new hybrid evolutionary algorithmic approach is presented. This algorithm exploits the specific problem structure of two-stage stochastic programs. Finally, the new approach is compared to other state-of-the-art solvers for the polymer plant scheduling problem presented in Chapter 8. The results show that the new algorithmic approach has a competitive performance and provides good solutions in short computation times.

9.2
Scheduling Under Uncertainty Using a Moving Horizon Approach with Two-Stage Stochastic Optimization

In this section, we explain the key features of a moving horizon approach with two-stage stochastic optimization by considering a simplification of the real-world scheduling example investigated in more detail in Section 9.5. First we apply a moving horizon based deterministic scheduler to this example. Then we investigate the resulting sequence of observations and decisions that represents a multi-stage information and decision structure. We develop the idea of exploiting the knowledge about this structure by using two-stage stochastic models. Finally, we apply a moving horizon based stochastic scheduler to the example and we demonstrate that the stochastic scheduler leads to a much better performance than the deterministic scheduler.

9.2.1
Motivating Example

As a simple example, a medium-term scheduling problem for a single-product batch plant with a single processing unit is considered. The availability of the raw materials and of the product storage capacity are unlimited. All batches have identical processing times and yield one unit of product per batch.

The model of the scheduling problem is based on a discrete representation of time where each period i corresponds to one day. The scheduler assigns the number of batches x_i to be produced in each period. The capacity of the plant is constrained to $x_i \in \{0, 5, 6, \ldots, 12\}$ batches per period; if the required production is less than five units in a period, the plant has to be switched off for this period. A costly set-up procedure has to be performed each time the plant is switched on or off. The occurrence of the set-up procedure in period i is denoted by the binary variable w_i ($0 =$ no, $1 =$ yes). The production costs per batch are denoted by $\beta = 1.0$ and the cost for a set-up is $\gamma = 3.0$. Demands d_i that are satisfied in the same period as requested result in a regular sale M_i with a full revenue of $\alpha = 2.0$ per unit of product. Demands that are satisfied with a tardiness of one period result in a late sale M_i^L with a reduced revenue of $\alpha^L = 1.5$ per unit. Demands which are not satisfied in the same or in the next period result in a deficit B_i^- with a penalty of $\alpha^- = 0.5$ per unit. The surplus production of each period is stored and can be sold later. The amount of batches stored at the end of a period is denoted by M_i^+ and the storage costs are $\alpha^+ = 0.1$ per unit. The objective is to maximize the profit over a horizon of H periods. The cost function P contains terms for sales revenues, penalties, production costs, and storage costs. For technical reasons, the model is reformulated as a minimization problem:

$$\min P = -\sum_{i}^{i+H-1} \left(\alpha M_i + \alpha^L M_i^L - \alpha^+ M_i^+ - \alpha^- B_i^- - \beta x_i - \gamma w_i\right) \qquad (9.1)$$

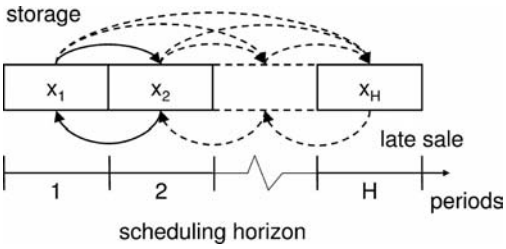

Fig. 9.1 Scheduling problem: couplings of the production decisions x_i over the periods due to storage and due to late sale.

The couplings between the production decisions over the periods due to the storage of batches and due to late sale are shown in Figure 9.1. The storage enables to produce the products for future demands earlier, e.g., when the demand in the next period exceeds the capacity in the next period. The late sale with a tardiness of one period allows for the compensation of a previously incurred deficit, e.g., when current demands cannot be satisfied from current storage and production.

The scheduling problem is subject to uncertainties in the demands. The demands d_i in period i are only known precisely after the period i. Thus, the production decision x_i has to be made under uncertainty without knowing the demand exactly for the current and for later periods. Table 9.1 provides a model of the uncertain demands. The model consists of two possible outcomes of the demands for each period i: d_i^1 and d_i^2. We assume a probability distribution with equal probabilities p_i^2 and p_i^2 for all outcomes.

Table 9.1 Uncertainties: model of the uncertain demands for four periods.

Time period	Probabilities		Demand outcomes		Expected value
i	p_i^1	p_i^2	d_i^1	d_i^2	d_i
1	0.5	0.5	0	12	6
2	0.5	0.5	7	15	11
3	0.5	0.5	4	10	7
4	0.5	0.5	9	11	10

9.2.2 Deterministic Online Scheduler

In order to investigate the performance of a deterministic online scheduler, we apply it to the example problem under demand uncertainty for three periods. The model of the scheduling problem used in the scheduler considers a prediction horizon of $H = 2$ periods. Only the current production decision $x_i(t_i)$ is applied

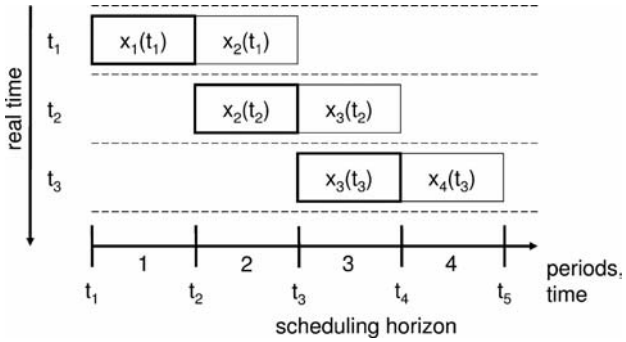

Fig. 9.2 Moving horizon scheme: only the current decision $x_i(t_j)$ is applied to the plant.

to the plant at the beginning of the current period i. The model is deterministic, since it considers the expected (mean) values $\bar{d}_i$ and $\bar{d}_{i+1}$ of the uncertain demands instead of their probability distributions. At the end of the current period i, the realized demand d_i is observed, the horizon is shifted by one period, the model is updated from the observations, and the procedure is repeated. This procedure is called a moving horizon scheme. The scheme is depicted in Figure 9.2 where the symbol t_j denotes the start time of period j. The symbol $x_i(t_j)$ denotes the production decision for the period i as decided at the time t_j.

All possible evolutions of the demands for three periods are depicted by means of a scenario tree in Figure 9.3. The numbers above each node represent the possible outcomes of the demands and thus the possible observations. Each path from the root node to a leaf of the tree represents a single scenario ω. Each scenario contains one of all possible combinations of the demand outcomes. With two different realizations per period, the scenario tree for three periods consists of $\Omega = 2^3 = 8$ scenarios. The demands for period $i = 4$ are not considered in the figure because

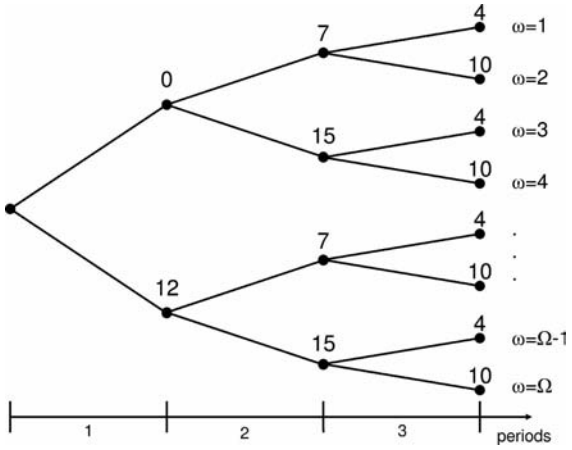

Fig. 9.3 Uncertainties: scenario tree of the demand for three periods.

they are used only in the last period of the prediction horizon of the scheduling model but not in the performance evaluation.

The sequence of decisions obtained from the scheduler for all possible evolutions of the demand for the three periods is shown in Figure 9.4. The plant is started with an empty storage $M_0^+ = 0$, no deficit from previous periods $d_0^L = 0$, and the plant operation state "on". The boxes contain the production decisions for each period while the circles contain the total objective after three periods for each scenario. The average objective value over the eight scenarios after three periods results as $P = -16.05$. The small figures on the right provide the evolution of the storage M_i^+, the deficit B_i^-, the late sale M_i^L, the sale M_i, the production x_i, and the objective p_i per period for each scenario.

The difference between the expected demands $\bar{d}_i$ used in the model and the realization of an actual demand d_i causes a model mismatch. The scheduler corrects this error after the observation of this demand in a reactive manner by the production decisions taken at the beginning of the next period. For example, the decisions taken in period $i = 1$ take the expected value of the demand d_1 into account ($\bar{d}_1 = 6$) while the decisions taken in period $i = 2$ take the true value into account. When $d_1 = 0$, the large storage causes a relatively low production in the next period ($x_2(t_2) = 5$) whereas in case of $d_1 = 12$, a deficit results and the production of the next period is larger ($x_2(t_2) = 12$). The sequence of decisions is a function of the observed demands and thus the sequence varies over the scenarios.

9.2.3 Stochastic Online Scheduler

The performance of the scheduler can be significantly improved by the use of a stochastic model. The stochastic model used here considers not only the probability distribution of the uncertain parameters but also the structure of decisions and observations that result from the moving horizon scheme.

The sequence of decisions obtained from the scheduler (Figure 9.4) has a tree structure. This structure results from the scenario tree of the uncertain demand parameters (Figure 9.3). Due to the moving horizon scheme, the decisions and the observations alternate at each period and the decisions are functions of the observations. Each point in time where a decision is made is called a stage. The result is a multi-stage tree where each stage corresponds to a period.

Mathematical optimization models that explicitly consider such a multi-stage structure belong to the class of multi-stage stochastic programs. A deterministic optimization model with uncertain parameters is extended to a multi-stage model by three measures:

- The uncertain parameters are replaced by the set of their possible outcomes (scenarios).
- The set of variables is extended by the correction decisions that are made after the observations.
- The objective function is replaced by the an expected (average) objective over all scenarios.

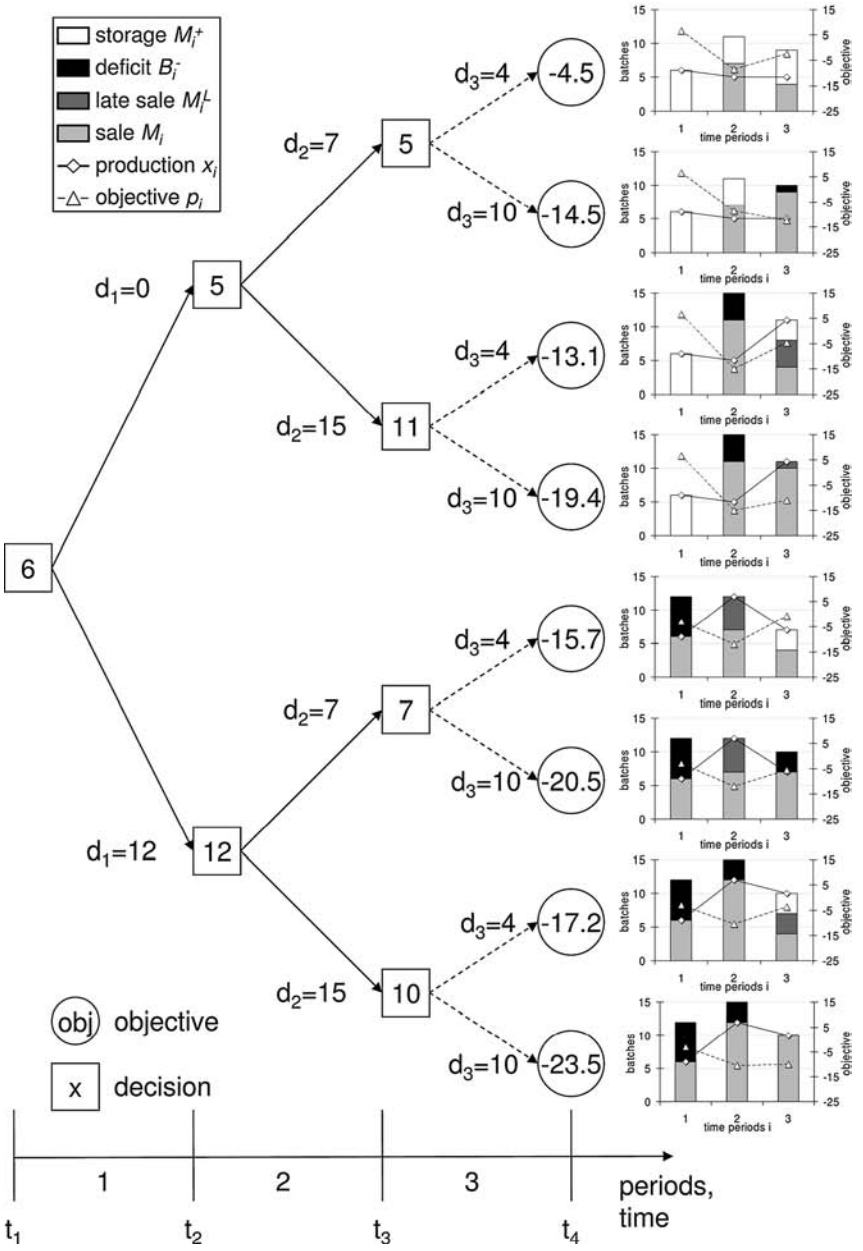

Fig. 9.4 Deterministic scheduler: sequence of decisions and results for all scenarios (average objective after three periods $P = -16.05$).

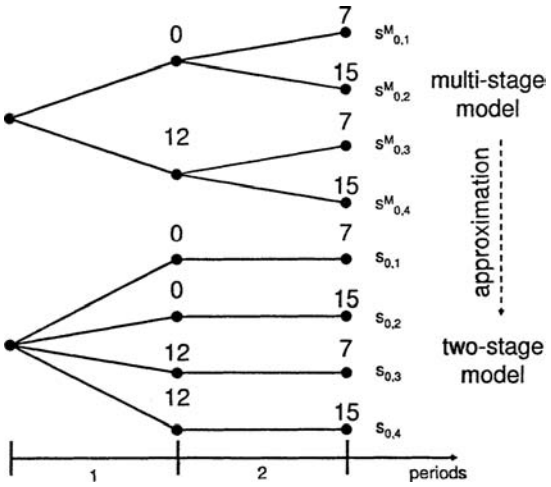

Fig. 9.5 Uncertainties: the multi-stage scenario tree of period 1 and its two-stage approximation (both with a horizon of $H = 2$ periods).

In this fashion, we extend our deterministic model with a prediction horizon of $H = 2$ to a multi-stage model. The multi-stage tree of the possible outcomes of the demand within this horizon (starting from period $i = 1$) with four scenarios is shown in Figure 9.5. Each scenario $s^M_{i,j}$ represents the combination k out of the set of all combinations of the demand outcomes within the horizon. The production decision x_1 has to be taken under uncertainty in all future demands. The decision x_2 can react to each of the two outcomes of d_1, but has to be taken under uncertainty in the demand d_2. The corrective decisions are explicitly modeled by replacing x_2 by two variables: $x_{2,1}$ and $x_{2,2}$.

However, the description of the tree structure of a multi-stage model leads to complicated constraints. To simplify the original multi-stage model, it is approximated by a model with two stages. It consists of only one sequence of decisions-observation-decisions. The two-stage structure leads to considerably simpler optimization problems. It is also adequate from a practical point of view: in the moving horizon scheme, only the first decision x_1 is applied to the plant while all the remaining variables are used to compute the estimated performance only.

In the approximation of the multi-stage model by a two-stage model it is assumed that all future demands can be observed after the first period. The resulting two-stage scenario tree for period 1 of the example problem with four scenarios is shown in Figure 9.5. The set of scenarios $s_{i,k}$ represent the two-stage approximation of a set of scenarios $s^M_{i,k}$. In the two-stage model only the first decision x_1 has to be taken under uncertainty while all remaining decisions can react to the observations. The corrective decisions are explicitly modeled by a corrective variable for each of the four scenarios $[x_{2,1}(t_1), x_{2,2}(t_1), x_{2,3}(t_1), x_{2,4}(t_1)]$ instead of only two variables $x_{2,1}(t_1)$

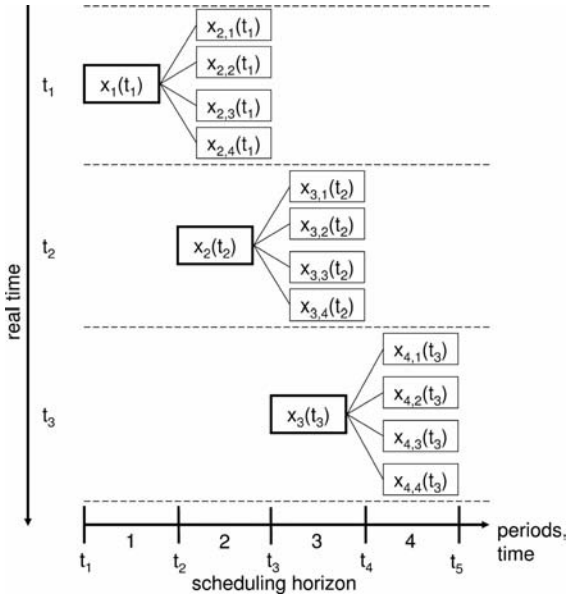

Fig. 9.6 Moving horizon scheme with two-stage stochastic models: the first decisions $x_i(t_i)$ are applied to the plant (compare to Figure 9.2).

and $x_{2,2}$ (t_1). The two-stage approximation has more degrees of freedom and less constraints than the multi-stage model and thus its objective is at least as good as that of the multi-stage model.

The moving horizon scheme using the two-stage model is shown in Figure 9.6. In contrast to the deterministic scheduler which uses the expected value of the demands $\bar{d}_i$ and $\bar{d}_{i+1}$ (see Section 9.2.2), the stochastic scheduler updates the demand in form of the distribution given in Table 9.1: d_i^1, d_i^2, d_{i+1}^1, and d_{i+1}^1.

The sequence of decisions obtained from the stochastic scheduler for all possible evolutions of the demand for the three periods is provided in Figure 9.7. The sequence of decisions obtained by the stochastic scheduler differs from that obtained by the deterministic one, e.g., $x_1(t_1) = 10$ instead of $x_1(t_1) = 6$. The average objective for the stochastic scheduler after three periods is $P = -17.65$.

9.2.4 Comparison and Conclusions

The performance of the deterministic and of the stochastic scheduler is compared in Figure 9.8. The figure shows the objective for all scenarios and the average objective. The stochastic scheduler improves the average objective by approximately 10% and for five out of eight scenarios. On the other hand, the stochastic scheduler produces a larger variation in the objective of the scenarios.

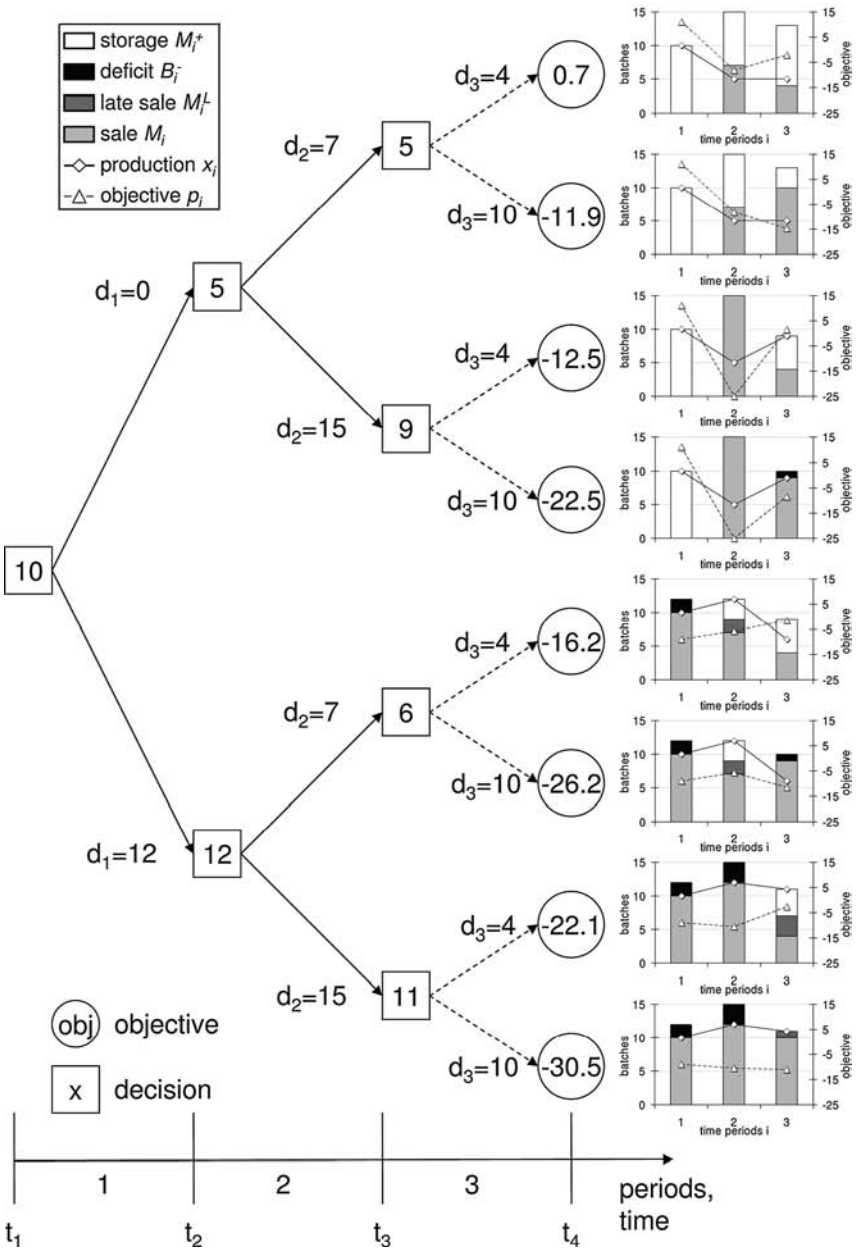

Fig. 9.7 Stochastic scheduler: sequence of decisions and results for all scenarios (average objective after three periods $P = -17.65$), compare to Figure 9.4.

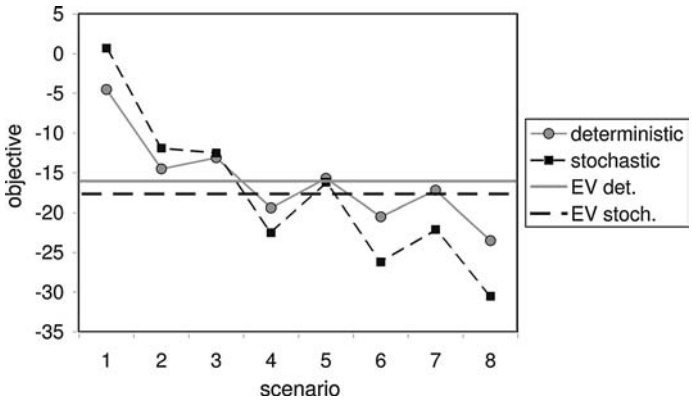

Fig. 9.8 Deterministic vs. stochastic scheduler: comparison of the objective after three periods.

The use of two-stage stochastic models in uncertainty conscious scheduling can provide a significant increase in the average objective compared to the use of deterministic models. The second-stage variables generally contain integer variables since scheduling problems usually consist of integer decisions in each period. For real-world problems, two-stage models become large-scale mixed-integer programs since the number of second-stage variables grows linearly with the number of scenarios. Therefore the remainder of this chapter is focused on the efficient solution of two-stage stochastic programs.

9.3 Two-Stage Stochastic Integer Programming

In the previous section it was shown that the performance of a scheduler can be significantly improved by the use of stochastic models. In this section, we present the mathematical models that represent two-stage stochastic scheduling problems and algorithmic approaches to the optimization of the schedules.

9.3.1 Stochastic Optimization Model

A *stochastic program* is a mathematical program (optimization model) in which some of the problem data is uncertain. More precisely, it is assumed that the uncertain data can be described by a random variable (probability distribution) with sufficient accuracy. Here, it is further assumed that the random variable has a countable number of realizations that is modeled by a discrete set of scenarios $\omega = 1, \ldots, \Omega$.

In a stochastic program *with recourse*, some corrective decisions or recourse actions can be taken after the uncertainty is disclosed. Each point in time where a decisions is made is called a *stage*. The *two-stage* stochastic program is the most

simple recourse program. It considers only one observation and thus the total set of n decisions is divided into two groups:

- Some decisions have to be taken before the uncertainty is disclosed. These are called the *first-stage* decisions and are denoted by the n_1-dimensional vector $\mathbf{x}$. The first-stage decisions cannot anticipate which scenario will realize and thus have to be the same for all scenarios. This is called *non-anticipativity*.
- Some decisions have to be taken after the uncertainty has been disclosed. These are called the *second-stage* decisions and are denoted by a n_2-dimensional vector $\mathbf{y}_\omega$ for each scenario ω. In contrast to the first-stage variables, the second-stage decisions are functions of the realized scenario. The second-stage decisions are a means to compensate the outcome of the first-stage decisions in the face of realized uncertainties.

A two-stage stochastic program is called a two-stage stochastic *mixed-integer* program when integrality requirements are present. The n_1-dimensional vector of first-stage decisions is divided into n_1' integer variables and n_1'' real variables. Each n_2-dimensional vector of second-stage decisions is divided into n_2' integer variables and n_2'' real variables. Integer requirements are present, when $n_1' + n_2' > 0$.

The objective of a two-stage stochastic *linear* program (9.3.1) consists of the first-stage costs and of the expected value of the second-stage costs. The first-stage costs and the scenario-specific second-stage costs are calculated as linear function of the first-stage variables $\mathbf{x}$ and the second-stage variables $\mathbf{y}_\omega$ with vectors of parameters $\mathbf{c} \in \mathbb{R}^{n_1}$ and $\mathbf{q}_\omega \in \mathbb{R}^{n_2}$. The expected value of the second-stage costs is calculated by the summation of the scenario-specific second-stage costs over all scenarios with the corresponding probabilities $\pi_\omega \in \mathbb{R}^1$ as weighting factors.

The constraints of a two-stage stochastic linear program can be classified into constraints on the first-stage variables only (9.3.2) and constraints on the first and on the second-stage variables (9.3.3). The latter represent the interdependency of the stages. All constraints are represented as linear inequalities with the matrices $\mathbf{A} \in \mathbb{R}^{m_1 \times n_1}$, $\mathbf{T}_\omega \in \mathbb{R}^{m_2 \times n_1}$, $\mathbf{W}_\omega \in \mathbb{R}^{m_2 \times n_2}$, and the vectors $\mathbf{b} \in \mathbb{R}^{m_2}$ and $\mathbf{h}_\omega \in \mathbb{R}^{m_2}$.

For a finite number of scenarios with fixed probabilities, a stochastic program can be modeled in the standard form of a mathematical program which is often called the *deterministic equivalent program*. The deterministic equivalent of the two-stage mixed-integer linear program (*2S-MILP*) is modeled by a large *mixed-integer linear program* (MILP) that can be can be written as:

$$(DEP): \min_{\mathbf{x},\mathbf{y}_\omega} f(\mathbf{x}, \mathbf{y}_\omega) = \overbrace{\mathbf{c}^T\mathbf{x}}^{\text{first-stage costs}} + \overbrace{\sum_{\omega=1}^{\Omega} \pi_\omega \mathbf{q}_\omega^T \mathbf{y}_\omega}^{\text{expected second-stage costs}} \tag{9.2}$$

$$s.t. \mathbf{A}\mathbf{x} \leq \mathbf{b} \tag{9.3}$$

$$\begin{aligned} \mathbf{T}_\omega\mathbf{x} + \mathbf{W}_\omega\mathbf{y}_\omega &\leq \mathbf{h}_\omega \\ \mathbf{x} \in X, \mathbf{y}_\omega \in Y, \omega &= 1, \ldots, \Omega \end{aligned} \tag{9.4}$$

where the sets X and Y contain the integer requirements and upper and lower bounds on the decision variables provided by the vectors $\mathbf{x}^{\min}$, $\mathbf{y}_\omega^{\min}$, $\mathbf{x}^{\max}$, and $\mathbf{y}_\omega^{\max}$

$$X = \{\mathbf{x} \in \mathbb{Z}^{n_1'} \times \mathbb{R}^{n_1''} | \mathbf{x}^{min} \leq \mathbf{x} \leq \mathbf{x}^{max}, \mathbf{x}^{min}, \mathbf{x}^{max} \in \mathbb{R}^{n_1'+n_1''}\} \tag{9.5}$$

$$Y = \{\mathbf{y}_\omega \in \mathbb{Z}^{n_2'} \times \mathbb{R}^{n_2''} | \mathbf{y}_\omega^{min} \leq \mathbf{y}_\omega \leq \mathbf{y}_\omega^{max}, \mathbf{y}_\omega^{min}, \mathbf{y}_\omega^{max} \in \mathbb{R}^{n_2'+n_2''}\} \tag{9.6}$$

The model (*DEP*) covers the general case with parametric uncertainties in the objective function ($\mathbf{q}_\omega$), in the left-hand-side multipliers of $\mathbf{x}$ and $\mathbf{y}_\omega$ ($\mathbf{T}_\omega$ and $\mathbf{W}_\omega$, respectively) and in the right-hand-side parameters ($\mathbf{h}_\omega$).

9.3.2 The Value of the Stochastic Solution

The concept of the *value of the stochastic solution* (*VSS*) measures the advantage of using a two-stage stochastic program over using a deterministic one, in other words, it measures the cost of ignoring the uncertainty.

In the deterministic program that corresponds to a stochastic program with discrete scenarios, the uncertain parameters are replaced by their mean values:

$$\overline{\mathbf{q}} = \sum_{\omega=1}^{\Omega} \pi_\omega \mathbf{q}_\omega \tag{9.7}$$

$$\overline{\mathbf{h}} = \sum_{\omega=1}^{\Omega} \pi_\omega \mathbf{h}_\omega \tag{9.8}$$

$$\overline{\mathbf{T}} = \sum_{\omega=1}^{\Omega} \pi_\omega \mathbf{T}_\omega \tag{9.9}$$

$$\overline{\mathbf{W}} = \sum_{\omega=1}^{\Omega} \pi_\omega \mathbf{W}_\omega \tag{9.10}$$

The result is a deterministic program, where the original second-stage decisions are not a function of the realized scenario, i.e., it is assumed the there is a single scenario problem and all decisions $\mathbf{x}^{EV}$ and $\mathbf{y}^{EV}$ have to be made before the observation. The corresponding optimization problem is called the *expected value problem* (*EV* problem) and can be written as follows:

$$EV: \min_{\mathbf{x}^{EV}, \mathbf{y}^{EV}} \quad \mathbf{c}^T\mathbf{x}^{EV} + \overline{\mathbf{q}}^T\mathbf{y}^{EV} \tag{9.11}$$

$$s.t. \quad \mathbf{A}\mathbf{x}^{EV} \leq \mathbf{b} \tag{9.12}$$

$$\overline{\mathbf{T}}\mathbf{x}^{EV} + \overline{\mathbf{W}}\mathbf{y}^{EV} \leq \overline{\mathbf{h}} \tag{9.13}$$

$$\mathbf{x}^{EV} \in X, \mathbf{y}^{EV} \in Y \tag{9.14}$$

Plugging the first-stage solution of the *EV* problem $\mathbf{x}^{EV}$ into the stochastic program (*2S-MILP*) gives the *expected result of using the EV solution* (*EEV* problem). The solution of the *EEV* problem is not necessarily optimal for the original *2S-MILP*. Consequently, the optimal objective value of the *EEV* problem is always greater than (or at least equal to) the optimal objective value of the *2S-MILP*, such that the objective of *EEV* is an upper bound for the optimal solution of the *2S-MILP*:

$$EEV \geq \text{2S-}MILP \tag{9.15}$$

The advantage of using a *2S-MILP* instead of the corresponding deterministic approach is measured by the *value of the stochastic solution* (*VSS*) which is the difference of the respective optimal objective values:

$$VSS = EEV - \text{2}S\text{-}MILP \tag{9.16}$$

9.3.3 General MILP Algorithms

Since the program (*DEP*) represents a mixed-integer linear program (MILP), it can be solved by commercially available state-of-the-art MILP solvers like CPLEX [3] or XPRESS-MP [4]. These solvers are based on implementations of modern branch-and-bound search algorithms with cuts and heuristics.

In general, *branch-and-bound* [5] is an enumerative search space exploration technique that successively constructs a decision tree. In each node, the feasible region is divided into two or more disjoint subsets which are then assigned to child nodes. During the search space exploration for minimization problems, a lower bound of the objective function is computed in each node and compared against the lowest upper bound found so far. If the lower bound is greater than the upper bound, the corresponding branch is said to be fathomed and not explored anymore. The exploration terminates when a certain gap between the upper and the lower bound is reached or when the all possible subsets have been enumerated.

The modern *branch-and-bound algorithms for MILPs* use branch-and-bound with integer relaxation, i.e., the branch-and-bound algorithm performs a search on the integer components while lower bounds are computed from the integer relaxation of the MILP by linear programming methods. The upper bound is taken from the best integer solution found prior to the actual node.

The branch-and-bound algorithms for MILPs belong to the class of *exact* algorithms. In contrast to *metaheuristics*, exact algorithms guarantee to find an optimal solution in finite (though possibly unrealistically large) time and are able to prove optimality by using conservative gaps, the lower bounds [6].

However, the straightforward approach to solve *2S-MILP*s by standard MILP solvers is often computationally prohibitive for real-world problems [7] due to the presence of a large number of integer variables. The reason for the large number of variables is the fact that each scenario adds a copy of the second-stage constraints

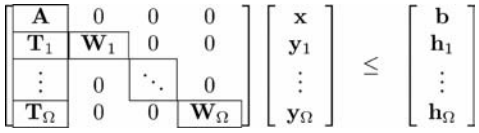

Fig. 9.9 Constraints of the *2S-MILP* (9.3.2)–(9.3.3): staircase structure.

and of the second-stage variables. Thus the total number of integer variables n' grows linearly in the number of scenarios:

$$n' = n'_1 + \Omega\, n'_2 \tag{9.17}$$

However, the constraints of the the *2S-MILP* have a so-called *staircase* structure (see Figure 9.9). Although commercial MILP solvers partially exploit the structure of a MILP, they are not yet able to automatically detect and exploit the structure of the *2S-MILP* [8].

9.3.4
Decomposition of a *2S-MILP*

The staircase matrix structure of the *2S-MILP* (see Figure 9.9) is exploited by *2S-MILP*-specific decomposition based algorithms [9, 10]. The constraint matrix of the *2S-MILP* consists of Ω subproblems $\mathbf{W}_\omega$ that are tied together by the first-stage variables $\mathbf{x}$ and the corresponding matrix column $[\mathbf{AT}_1 \ldots \mathbf{T}_\Omega]^T$. The main steps of *decomposition based algorithms* for *2S-MILPs* are:

1. Remove the links between the subproblems.
2. Solve the resulting set of smaller and easier MILP subproblems instead of the large-scale *2S-MILP*.
3. Generate full solutions for the original *2S-MILP* from the solutions of the subproblems.

However, the last step is not easy because the solutions of the subproblems may be suboptimal or infeasible for the original problem.

Basically, there are two different ways to decompose a *2S-MILP* (see Figure 9.10). The *scenario decomposition* separates the *2S-MILP* by the constraints associated to a scenario, whereas the *stage decomposition* separates the variables into first-stage and second-stage decisions. For both approaches, the resulting subproblems are MILPs which can be solved by standard optimization software.

9.3.5
Scenario Decomposition Based Branch-and-Bound Algorithm

Before a new stage decomposition based hybrid evolutionary algorithm is proposed in Section 9.4, we briefly review the algorithm for general *2S-MILP*s of Carøe and Schultz [11] which is regarded as the state-of-the-art exact algorithm for *2S-MILP*s

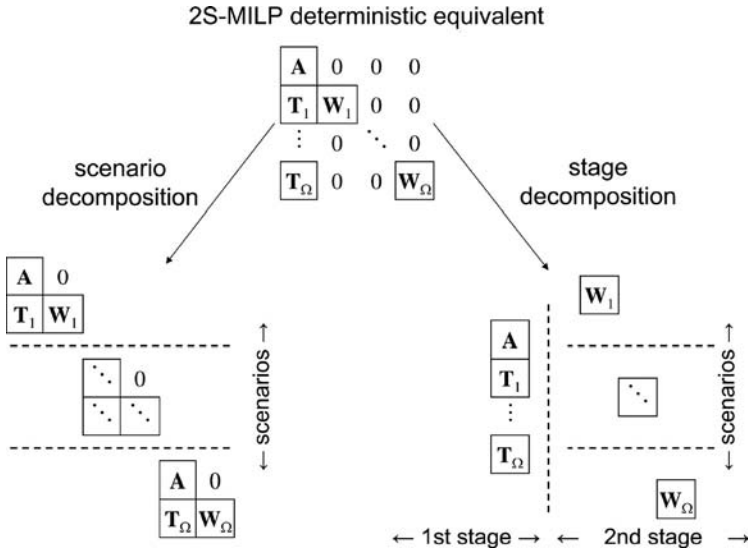

Fig. 9.10 Decomposition: scenario decomposition and stage decomposition of a *2S-MILP*.

[12]. This algorithm has been successfully applied to chemical batch scheduling problems [13–17].

To decompose a *2S-MILP* into single scenario problems, the program (*DEP*) is equivalently rewritten in its *extensive form*. Copies of the first-stage variables $\mathbf{x}_\omega$ are added for each scenario ω. Since the first-stage variables must not vary over the scenarios, explicit non-anticipativity constraints are added and the following problem formulation results:

$$\min_{\mathbf{x}\omega,\mathbf{y}_\Omega} \quad \sum_{\omega=1}^{\Omega} \pi_\omega \left(\mathbf{c}^T\mathbf{x}_\omega + \mathbf{q}_\omega^T\mathbf{y}_\omega\right) \tag{9.18}$$

$$s.t. \quad \mathbf{A}\mathbf{x}_\omega \leq \mathbf{b} \tag{9.19}$$

$$\mathbf{T}_\omega\mathbf{x}_\omega + \mathbf{W}_\omega\mathbf{y}_\omega \leq \mathbf{h}_\omega \tag{9.20}$$

$$\mathbf{x}_1 = \mathbf{x}_2 = \ldots = \mathbf{x}_\Omega \tag{9.21}$$

$$\mathbf{x}_\omega \in X, \mathbf{y}_\omega \in Y, \omega = 1, \ldots, \Omega \tag{9.22}$$

In this form, the scenario subproblems are tied together only by the non-anticipativity constraints (9.22). This naturally leads to a decomposition based on the relaxation of the non-anticipativity constraints.

The main idea of the algorithm of Carøe and Schultz [11] is to decompose a *2S-MILP* into its scenarios by Lagrangian relaxation of the non-anticipativity constraints. In a Lagrangian relaxation, constraints are removed and included in the objective function with a penalty term.

A standard branch-and-bound algorithm is used to explore the first-stage search space while lower bounds are provided by the Lagrangian dual. Candidate solutions

for the upper bounds are generated by rounding heuristics from the possibly different first-stage solutions $\mathbf{x}_\omega$ of the dual subproblems. In other words, the rounding heuristics are used to generate the full solution from the solutions of the decomposed subproblems which correspond to a single scenario MILP each. The result for a candidate solution provides an upper bound of the *2S-MILP*. However, a candidate solution may be infeasible in the primal *2S-MILP*.

9.4 A Stage Decomposition Based Evolutionary Algorithm

This section presents a new stage decomposition based hybrid evolutionary algorithm for *2S-MILP*s that was proposed by Till et al. [7, 18].

9.4.1 Stage Decomposition

The main idea of stage decomposition (see Figure 9.10) is to remove the ties between the scenario subproblems of the *2S-MILP* by fixing the first-stage variables. The *2S-MILP* is written in its *intensive form* [9], where the resulting master problem is a function of the first-stage decisions only:

$$(MASTER): \quad \min_{\mathbf{x}} \qquad f(\mathbf{x}) = \mathbf{c}^T\mathbf{x} + \sum_{\omega=1}^{\Omega} \pi_\omega Q_\omega(\mathbf{x}) \tag{9.23}$$

$$s.t. \qquad \mathbf{A}\mathbf{x} \leq \mathbf{b}, \mathbf{x} \in X \tag{9.24}$$

The evaluation of the implicit *second-stage value function* $Q_\omega(\mathbf{x})$ for a given $\mathbf{x}$ requires the solution of Ω independent MILP subproblems:

$$(SUB): \quad Q_\omega(\mathbf{x}) = \min_{\mathbf{y}_\omega} \qquad \mathbf{q}_\omega^T\mathbf{y}_\omega \tag{9.25}$$

$$s.t. \qquad \mathbf{W}_\omega\mathbf{y}_\omega \leq \mathbf{h}_\omega - \mathbf{T}_\omega\mathbf{x}, \mathbf{y}_\omega \in Y \quad \forall\omega = 1, \ldots, \Omega \tag{9.26}$$

When the second stage decisions are real-valued variables, the value function $Q_\omega(\mathbf{x})$ is piecewise-linear and convex in $\mathbf{x}$. However, when some of the second stage variables are integer-valued, the convexity property is lost. The value function $Q_\omega(\mathbf{x})$ is in general non-convex and non-differentiable in x. The latter property prohibits the use of gradient-based search methods for solving *(MASTER)*.

9.4.2 Main Idea

Metaheuristics as, e.g., evolutionary algorithms are a widely used alternative approach to large-scale or combinatorial optimization problems [19]. In contrast to *exact* algorithms (e.g., branch-and-bound algorithms), metaheuristics have the potential to find good solutions in limited computation times but cannot guarantee

or prove optimality. In fact, metaheuristics may not converge to the optimum at all. A recent promising trend is the use of *hybrid* approaches which combine metaheuristics with exact algorithms for solving large-scale combinatorial optimization problems [6].

The main idea of our stage decomposition based *hybrid evolutionary algorithm* is to tackle the non-linear master problem (*MASTER*) by an evolutionary algorithm and to use a standard MILP solver to evaluate the second-stage value function. In contrast to exact stage decomposition based algorithms like the L-shaped approach [9], the evolutionary algorithm does not rely on convexity properties of the master problem. However, the price paid for this generality is that convergence in finite time cannot be guaranteed.

9.4.3 Evolutionary Algorithms

The term *evolutionary algorithm* (*EA*) refers to a class of population based metaheuristic (probabilistic) optimization algorithms which imitate the Darwinian evolution ("survival of the fittest"). However, the biological terms are used as metaphors rather than in their exact meaning. The *population* of individuals denotes a set of solution candidates or points of the solution space. Each *individual* represents a point in the search space which is coded in the individual's representation (genome). The *fitness* of an individual is usually defined on the basis of the value of the objective function and determines its chances to stay in the population and to be used to generate new solution points.

An *EA* searches for the optimal solution by the iterative application of the *variation-selection-paradigm* to the population. Usually the population is *initialized* by arbitrarily generated individuals. Two probabilistic *variation* operators are used to generate new solution candidates (offspring) from the individuals of a parent population. The *mutation* operator changes the search space parameters of an individual according to a given probability distribution while the *recombination* operator merges information of two or more randomly selected parent individuals. After the fitness of the offspring is evaluated, the *selection* operator, which may contain probabilistic components, chooses the fittest individuals to be the parents of the next generation. The variation and selection operators are iteratively applied until a *termination* criterion is reached. The variation operator produces diversity in order to explore the search space, whereas the selection operator directs the evolutionary search by exploiting the fitness information. Figure 9.11 presents the general schema of an *EA* as a flow chart.

Different classes of *EA*s differ in the representation of the degrees of freedom and in the operators used. The most popular class are *genetic algorithms* [20], which operate on a binary string and almost always apply recombination operators in addition to selection and mutation. The term *evolution strategy* (*ES*) refers to a class of *EA* which adapts the mutation strength to the topology of the search space during the course of the evolution [21]. In an *ES*, the variation operators work on the same representation (called *object parameters*) as the fitness evaluation. The parameters

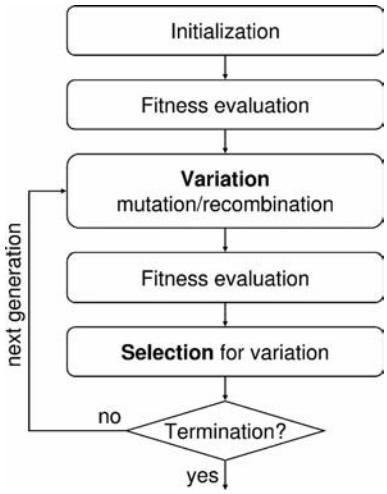

Fig. 9.11 Evolutionary algorithm: general schema as a flow chart.

that determine the behavior of the algorithm as, e.g., the population size or the mutation strength are called *strategy parameters*.

9.4.4 Realization of a Hybrid Evolutionary Algorithm for *2S-MILP*s

General purpose evolutionary algorithms as the genetic algorithm or the evolution strategy were originally designed for unconstrained search spaces. However, the search space X of (*MASTER*) is subject to integrality requirements, as well as to explicit and implicit constraints. The *EA*-literature (e.g., [22]) provides a large variety of *constraint handling techniques*, e.g., penalty functions, special representations and operators, repair algorithms, or separation of objectives and constraints. However, there is no technique that is efficient in all cases. A constraint handling method specific to *2S-MILP*s is proposed in this section. The proposed method handles the integrality requirements and the bounds by an appropriate representation of the degrees of freedom within a mixed-integer evolution strategy. The explicit and the implicit feasibility constraints are considered by a modified objective function.

9.4.4.1 Mixed-Integer Evolution Strategy

The hybrid evolutionary algorithm for *2S-MILP*s is realized by using an evolution strategy (*ES*) to solve the master problem of the intensive *2S-MILP*. Each individual of the *ES* represents a first-stage candidate solution $\mathbf{x}$. The object parameters are encoded by a mixed-integer vector. The fitness of an individual is evaluated by the objective function of the master problem (*MASTER*), $f(\mathbf{x})$.

The *ES* used here is the *mixed-integer ES* for bounded search spaces [23]. It can operate on a general mixed-integer search space. The *ES* uses a population size of μ, with λ offspring per generation. Self-adaptation is realized by extending

Table 9.2 Mixed-integer evolution strategy: user defined strategy parameters.

Symbol	Type	Default	Description
μ	$\mathbb{Z}^+$	10	Number of parent individuals
$\nu = \lambda/\mu$	$\mathbb{R}^+$	7	Offspring to parents ratio
κ	$\mathbb{Z}$	5	Maximum age
r_x	$-, i, d$	d (discrete)	Recombination operator for object parameters
r_s	$-, i, d$	i (intermediate)	Recombination operator for strategy parameters

the object parameters of the individual by a set of mutation strength parameters. The mutation operators first change the mutation strength parameters which then determine the mutation of the object parameters. A new individual with modified parameters modified is then subject to the fitness evaluation and the selection mechanism.

For *recombination* two parents are randomly selected. The *discrete* recombination operator (d) generates an offspring by randomly taking the offspring's object parameters from one of the selected parents with equal probability. The *intermediate* recombination operator (i) takes the arithmetic mean of both parents' parameters. The recombination can be omitted ($-$).

The *mutation* operator has a constant mutation probability and adds random values to the object parameters. The values are randomly generated from (continuous) normal distributions. The variances of the distributions are determined by the adaptive strategy parameters of the mutation strength. The mutation strength parameters are initially set to 10% of the range of the corresponding object parameter. For integer object parameters, the integrality is maintained by the use of (discrete) geometric distributions. The mutated object parameters are kept within their bounds by a transformation function that reflects infeasible parameters back into their domain.

The (μ, κ, λ)-*selection* chooses the best μ individuals from the union of μ parents and λ offspring, except those parent individuals which exceed the maximum age of κ generations. The consideration of a maximum age enables the *ES* to escape from local optima. Table 9.2 summarizes the user defined strategy parameters of the *ES*.

9.4.4.2 Constraint Handling by a Modified Objective

The explicit feasibility constraints of (*MASTER*) are given by the linear first-stage constraints in (9.4.2). In a classical penalty function approach, the explicit feasibility constraints are relaxed while the violation of these constraints is considered by an additional penalty term in the fitness function. However, this method would waste valuable CPU time since the MILP subproblems (*SUB*) have to be solved also for the fitness evaluation of infeasible individuals. A similar method which does not require the solution of the MILP subproblems for infeasible individuals is the use of a *modified objective function* that separates the objective and the feasibility

constraints. The modified objective always prefers feasible solutions over infeasible solutions. The original fitness function $f(\mathbf{x})$ is extended to $F(\mathbf{x})$:

$$F(\mathbf{x}) = \left\{ \begin{array}{ll} f(\mathbf{x}) & \text{if } (MASTER) \text{ and } (SUB) \text{ are feasible} \\ f_{max} + p(\mathbf{x}) & \text{otherwise} \end{array} \right\} \tag{9.27}$$

The parameter $f_{\max}$ denotes a conservative upper bound of $f(\mathbf{x})$. For the *2S-MILP* it is easily calculated by maximizing the integer relaxation of (DEP). A positive penalty term $p(\mathbf{x})$ is used to measure the amount of infeasibility. This steers the search in infeasible regions towards the feasible region. The penalty for the violation of the first-stage constraints is provided by:

$$p(\mathbf{x}) = \sum_{j=1}^{m_1} \max[0, (\mathbf{A}_j \mathbf{x} - b_j)] \tag{9.28}$$

For first-stage infeasible candidates $\mathbf{x}$, the subproblems are not solved.

As first-stage feasible solutions in general do not necessarily have a feasible completion in the second-stage due to the implicit constraints in (SUB), the total set of feasible solutions for $\mathbf{x}$ is a subset of the first-stage feasible solutions. In this case, the program is called a *2S-MILP* without relative complete recourse. For a *2S-MILP* with relative complete recourse, each first stage feasible solution $\mathbf{x}$ has a feasible completion in the second-stage.

In contrast to the first-stage constraints, the amount of infeasibility of the second-stage constraints cannot directly be computed due to the existence of the degrees of freedom $\mathbf{y}_\omega$. Thus,

$$p(\mathbf{x}) = 0 \tag{9.29}$$

is assigned in this case. Solutions which are first-stage feasible and second-stage infeasible are thus preferred to solutions which are first-stage infeasible. The CPU time for the detection of a violation of the second-stage constraints is much higher than for the first-stage constraints, since in the worst case $\Omega - 1$ subproblems have to be evaluated before an infeasible scenario is detected. When a *2S-MILP* without relative complete recourse is infeasible with respect to the first-stage constraints, the evaluation of the second-stage feasibility is not necessary and thus omitted.

Figure 9.12 shows the modified objective function for a one-dimensional continuous object parameter with first- and second-stage infeasibilities.

9.5 Numerical Studies

In this section, the hybrid evolutionary algorithm described above is applied to a real-world scheduling problem under uncertainty. The performance of this algorithm is compared to that of the state-of-the-art MILP solver *CPLEX* and to that of

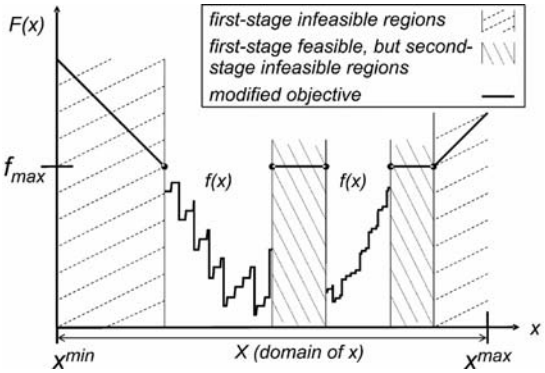

Fig. 9.12 Modified objective function: illustrated for a one-dimensional continuous object parameter with first- and second-stage infeasibilities.

the scenario decomposition based branch-and-bound-algorithm described briefly in Section 9.3.5.

9.5.1 Case Study

The scheduling of the production of polymers in a multi-product batch plant (see Figure 9.13) is investigated here as a real-world case study. The reader is referred to Chapter 8 for a more detailed description.

The plant is used to produce type *A* and type *B* of the polymer *expandable polystyrene* (*EPS*) in $F = 5$ grain size fractions each from a number of raw materials (E). The availability of raw materials and the product storage capacity are assumed to be unlimited. The preparation stage is not limiting the production process

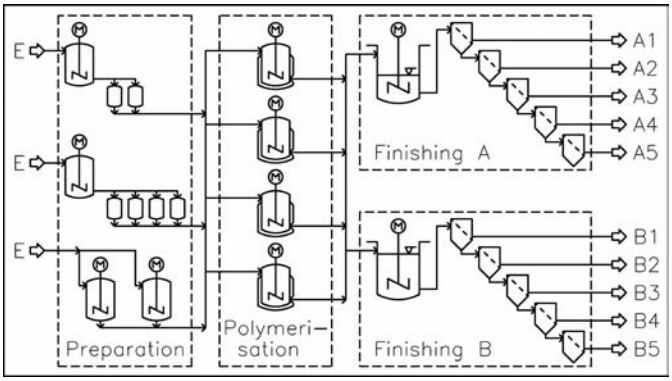

Fig. 9.13 Flow sheet of the EPS-process.

and is thus neglected in the sequel. The polymerization stage operates in batch mode. The production of each batch is controlled by a recipe. For each *EPS*-type p, $R_p = 5$ recipes r_p exist ($r_p \in \{1, \ldots, R_p\}\ \forall p$) which determine the grain size distribution such that each batch yields a main product with $F - 1$ coupled products. The duration of a polymerization is the same for all recipes. After the polymerization of a batch is finished, this batch is directly transferred to the corresponding mixer of the finishing stages A or B. The mixers are semi-continuous storage tanks, the finishing lines operate continuously. If a mixer runs empty, the corresponding finishing line has to be shut down temporarily. After a shutdown, the line has to be stopped for a certain period of time. The degrees of freedom of the scheduling problem are:

- the number of the polymerization batches (discrete);
- the timing of the batches in the polymerization stage (continuous);
- the assignments of recipes to the polymerization batches (discrete);
- the start-up and shut-down times of the finishing lines (continuous);
- the outflows of the mixing vessels (continuous).

The decisions have to be made to maximize the profit (given in 10^3 Euro) which is calculated from sales revenues, production costs, storage costs, and penalties for lateness and for finishing line start-ups and shut-downs. The demand profile is specified by amounts of the products and their due dates. The scheduling problem is complicated by the fact that the coupled production of grain size fractions and the mixing in the finishing lines prohibit a fixed assignment of recipes to products.

9.5.1.1 **Aggregated Scheduling Problem**

The scheduling problem is decomposed hierarchically into an aggregated scheduling problem and a detailed scheduling problem with horizons on the order of weeks and days. The aggregated scheduling problem is solved here. Its decisions are the timings of the polymerization batches and of the start-up and shut-down times of the finishing lines. The decisions of the aggregated model within the short-term horizon of the detailed scheduler are provided as the guidelines to the detailed scheduler. The remainder of this work is focused on the aggregated problem.

The aggregated scheduling problem is subject to uncertainties in the following parameters: (1) the capacity of the polymerization stage, i.e., a possibly reduced availability of polymerization reactors due to equipment failures, and (2) the demand profiles.

When these uncertainties are not considered in the computation of a schedule, the uncertainties in the capacity may lead to infeasible schedules, e.g., a schedule requires more capacity than available, whereas the uncertain demands have an effect on the value of the profit, e.g., when a schedule results in more or less product than demanded.

9.5.1.2 MILP Model

A MILP model of the aggregated scheduling problem of the EPS process was proposed by Sand and Engell [16]. The model is formulated as a discrete time multi-period model where each period $i \in \{1, \ldots, I\}$ corresponds to two days. The degrees of freedom of the aggregated problem are the following discrete production decisions:

- The timing and the number of the polymerization batches together with the assignments of the recipes are modeled by an integer variable N_{i,r_p}. This variable denotes how many batches according to recipe r_p are produced in period i.
- The state of the finishing line of product p in period i is modeled by the binary variable $z_{i,p} \in \{0(=\text{off}), 1(=\text{on})\}$.

The feasibility of the production decisions is restricted by the following operation constraints:

- The capacity of the polymerization stage limits the number of batches that can be produced in each period i.
- The feed into the mixing tanks of the finishing lines is defined by the number of polymerization batches produced per period and is required to be either zero or between the minimum and the maximum capacity depending on the state of the finishing lines.
- A finishing line has to stay in the same operation state for at least two successive periods before the operation state can be changed again. This requirement does not comprise the initial operation state.

The objective is to maximize the profit which is calculated by a cost model of sales revenues, production costs, storage costs, and penalties for lateness and for finishing line start-ups and shut-downs. The cost model adds some equality and inequality constraints with associated real valued variables for the sales, deficits, and the storage, but it does not further restrict the feasibility of the production decisions.

9.5.1.3 Stochastic Extension of the MILP Model to a *2S-MILP*

The uncertainties of the aggregated scheduling problem are modeled by discrete scenarios. The demand scenarios are defined by random variations around a nominal profile where the variations represent new or changed orders. The scenarios for the uncertain capacity are generated by assuming the failure of one polymerization reactor with a certain probability for each period.

Some of the production decisions of the aggregated scheduling problem are provided to the detailed scheduler. These decisions have to be taken before any observation of the outcome of the uncertain parameters is available. Thus, they correspond to the first-stage decisions of the two-stage stochastic problem. Consequently, the vector of first-stage decisions x consists of all production decisions of the short-term horizon: N_{i,r_p} and $Z_{i,p}$ for i e $\{1, \ldots, I_1\}$.

All remaining decisions can be made after the observation of the outcome of uncertain parameters, either in the detailed scheduler or by decisions of the aggregated problem which are taken later. Thus, these decisions are considered as second-stage decisions. Consequently, the vector of second-stage decisions $\mathbf{y}_\omega$ consists of all production decisions of the periods $i > I_1$ and all continuous variables of the cost model for all periods.

The resulting two-stage stochastic mixed-integer linear program has integer variables in the first-stage and mixed-integer variables in the second stage. There is a strong interdependency between the stages. The amount of product produced in the first-stage periods is coupled to the later periods by the storages and affects the objective value, since it generates costs for storage of products that may not be sold. The first-stage decisions on the state of the finishing lines may restrict the possible decisions in the second-stage, e.g., when the state of the finishing line must be maintained for the first period of the second-stage. This formulation of the aggregated scheduling model leads to a *2S-MILP* where the uncertainties appear only in the right-hand side $\mathbf{h}_\omega$.

9.5.2 Numerical Experiments

For the numerical experiments, we investigate solving a single *2S-MILP* of the aggregated *EPS* scheduling problem as described above. The long-term horizon comprises $I = 5$ periods while the short-term horizon comprises $I_1 = 3$ periods. The uncertainties in demands and capacity are modeled by $\Omega = 64$ scenarios. Table 9.3 presents the dimensions of the deterministic equivalent MILP (*DEP*) of the scheduling problem. For technical reasons, the objective is reformulated to a minimization problem. All computations were performed on a 2.4 GHz Linux machine, all MILPs were solved using CPLEX 9.1 [3] with a relative optimality gap of 1%.

As a preliminary step, we investigate the result of using a deterministic model, the expected value problem (*EV*). For the first-stage solution obtained from *EV*, the objective of the *2S-MILP* is $EEV = -12.00$.

9.5.2.1 Test of the Hybrid *ES*

Before the performance of the proposed hybrid *ES* is compared to that of other algorithms, we show that the *ES* converges to a good solution and that it is robust with respect to an infeasible initialization. After some experiments, the strategy

Table 9.3 Deterministic equivalent MILP: model dimensions.

Equations	Integer variables	Continuous variables
12 195	3840	20 481

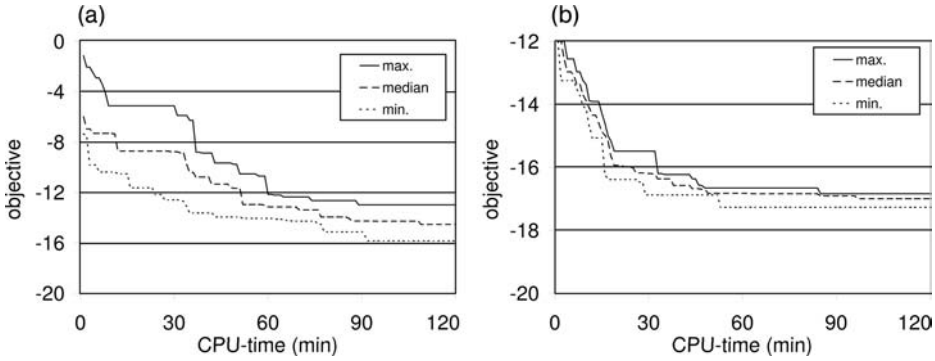

Fig. 9.14 Hybrid *ES:* infeasible vs. feasible initialization, best objective found vs. CPU-time, (a) infeasible initialization, (b) feasible initialization.

parameters of the *ES* (see Table 9.2) were set to $\mu = 10$, $\nu = 7$, $\kappa = \infty$ while recombination was omitted ($r_x = r_s = (-)$) for all results presented. The performance of the *ES* is measured by the best objective found over the CPU-time. Since the *ES* is a randomized algorithm, the performance measure is a random variable. To consider the randomness, we performed a number of $n = 5$ runs for each experiment and show the minimum, the median, and the maximum of the objective found over all runs at a certain CPU-time (see Figure 9.14).

In the first experiment (Figure 9.14a), the *ES* was initialized by a set of randomly generated first-stage infeasible solutions. The first feasible solution was found after 5000 to 6000 fitness evaluations (approximately 70 to 85 generations). The corresponding CPU-time is rather short because on the average 1000 penalty calculations were performed per second. The results show that the proposed penalty function allows to steer the search towards feasible regions. In contrast to the penalty calculation, the CPU-time for the fitness evaluation of a feasible individual is much longer and required on average about one second. The results show, that the *ES* is able to find a feasible solution and to converge towards the optimum independent of a feasible initialization.

In the second experiment (Figure 9.14b), the *ES* was initialized by a feasible initial population that consisted of the *EV*-solution and other randomly generated feasible solutions. Here, the *ES* converges faster than with infeasible initialization. Although the *ES* is robust against infeasible initialization, a feasible initialization is recommended to improve speed of convergence.

9.5.2.2 **Evaluation of the *ES***

The performance of the hybrid *ES* is compared to that of the commercial MILP solver CPLEX and to that of the state-of-the-art scenario decomposition based branch-and-bound algorithm described in Section 9.3.5 (*DDSIP*). The results are shown in Figure 9.15. As the reference for the performance of hybrid *ES*, we take the median from the experiment with feasible initialization (Figure 9.14b).

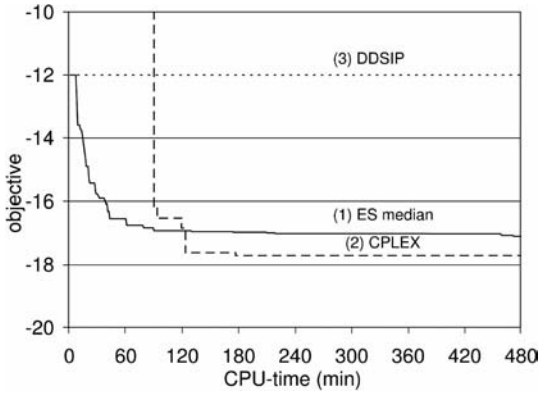

Fig. 9.15 *ES* evaluation: best objective value vs. CPU-time (best lower bound = −20.6).

After eight hours, the median objective was −17.33, the worst out of five runs was −17.25.

CPLEX, a highly advanced commercial MILP solver based on branch-and-bound with cuts and heuristics, and with automatic parameter adaptation is used to address the problem in the form of the large-scale deterministic equivalent program (*DEP*). After two minutes, the first feasible solution $\mathbf{x} = 0$ with an objective of +29.7 was found. The next feasible solution was found after approximately 90 minutes. The best solution found after eight hours was −17.74.

For the branch-and-bound algorithm (*DDSIP*) the configuration recommended by Märkert [24] and Clostermann [25] was used: the branching follows a breadth-first strategy. The Lagrangian dual is solved at the root node only; in the subsequent nodes, the lower bounds are calculated from the Lagrangian relaxation with the same Lagrangian multipliers as obtained in the root node. The rounding heuristic which *takes the average of the subsolutions rounded to the next integer* is selected. *DDSIP* is initialized by the *EV* solution. However, no solution better than the *EEV* was found within eight hours of CPU-time. The solution of the Lagrangian dual required about 40 minutes, then the branch-and-bound algorithm performed a search on 1200 nodes. No other feasible solution was generated. The best lower bound found by *DDSIP* is −20.60.

The comparison of the algorithms in Figure 9.15 shows that the *ES* performed better than the other two algorithms in the first two hours of CPU-time for this example. The proposed *ES* is well-suited for real-world problems, which require good solutions in a short time and is competitive compared to other state-of-the-art algorithms.

9.5.2.3 **Value of the Stochastic Solution**

The best *2S-MILP* solution found for this example was −17.74, thus the value of the stochastic solution (see Section 9.3.2) is

$$VSS = EEV - 2S\text{-}MILP = -12.00 + 17.74 = 5.74 \tag{9.30}$$

or more than 45% of the *EEV*. The use of the stochastic formulation increases the profit significantly compared to deterministic solution. The proposed *ES* is able to exploit the major part of this advantage relatively quickly.

9.6 Conclusions

An uncertainty conscious scheduling approach for real-time scheduling was presented in this chapter. The approach is based on a moving horizon scheme where in each time period a two-stage stochastic program is solved. For the investigated example it was found that the stochastic scheduler improved the objective on average by 10% compared to a deterministic scheduler.

The mathematical model of two-stage stochastic mixed-integer linear optimization problems was discussed as well as state-of-the-art solution algorithms. A new hybrid evolutionary algorithm for solving this class of optimization problems was presented. The new algorithm exploits the specific problem structure by stage decomposition.

By a comparison of the new evolutionary algorithm's performance with state-of-the-art solvers for a real-world scheduling problem it was found that the new algorithm shows a competitive performance. In contrast to the other algorithms the evolutionary algorithm was able to provide relatively good solutions in short computation times.

The hybrid algorithm is in general suitable for any two-stage stochastic mixed-integer linear program with integer requirements in the first-stage and in the second-stage.

Bibliographical Notes

For a recent survey of reactive and stochastic chemical batch scheduling approaches, the reader is referred to Floudas and Lin [2]. For a survey of the different types of probabilistic mathematical models that explicitly take uncertainties into account, see Sahinidis [12]. For detailed information about stochastic programming and its applications, the reader is referred to the books of Birge and Louveaux [9], Ruszczynski and Shapiro [10], or Wallace and Ziemba [26].

Few publications have proposed to use stochastic programming to address scheduling of batch plants under uncertainty. Vin and Ierapetritou [27] as well as Bonfill et al. [28] consider stochastic programs in which all integer decisions are assumed to belong to the first-stage. Sand et al. [13] propose the use of two-stage stochastic mixed-integer programs with integer variables in the second stage within a moving horizon approach. The stochastic programs are solved by applying the scenario-decomposition based branch-and-bound algorithm of Carøe and Schultz [11]. The same approach was further investigated in Engell and Sand [14], Engell et al. [15], and Sand and Engell [16]. Balasubramanian and Grossmann [29]

motivate the use of multi-stage stochastic models with integer recourse, but approximate these models by a series of two-stage models with continuous recourse.

Acknowledgements

The financial support by the Deutsche Forschungsgemeinschaft (DFG) for the Collaborative Research Center *Design and Management of Complex Technical Processes and Systems by Means of Computational Intelligence Methods* (Sonderforschungsbereich 531) at the Universitat Dortmund, Project CIO, is gratefully acknowledged.

References

1 Shah, N. (1998) *Single- and Multi-Site Planning and Scheduling: Current Status and Future Challenges*. Proceedings of the 3rd Conference Foundations of Computer-Aided Process Operations, CACHE, Michigan, pp. 75–90.

2 Floudas, C.A. and Lin, X. (2004) Continuous-time versus discrete-time approaches for scheduling of chemical processes: a review. *Computers and Chemical Engineering*, **28**, 2109–2129.

3 CPLEX (2002) *Using the CPLEX Callable Library*, ILOG Inc., Mountain View, CA.

4 XPRESS-MP (2002) *Users manual. Dash Optimization*, Englewood Cliffs.

5 Nemhauser, G. and Wolsey, A. (1999) *Integer and Combinatorial Optimization*, Wiley, New York.

6 Puchinger, J. and Raidl, G. (2005) *Combining Metaheuristics and Exact Algorithms in Combinatorial Optimization: A Survey and Classification*. in Proceedings of the First International Work-Conference on the Interplay Between Natural and Artificial Computation, vol. 3562 of *LNCS*, Springer, New York, pp. 41–53.

7 Till, J., Sand, G., Engell, S., *et al*. (2005b) A hybrid algorithm for solving two-stage stochastic integer problems by combining evolutionary algorithms and mathematical programming methods, in *Computer-Aided Chemical Engineering* (ed. L. Puig-Janer), vol. **20** *A of European Symposium on Computer Aided Process Engineering – 15*, Elsevier, Amsterdam, pp. 187–192.

8 Parija, G.R., Ahmed, S. and King, A.J. (2004) On bridging the gap between stochastic integer programming and mip solver technologies. *INFORMS Journal on Computing*, **16** (1), 73–83.

9 Birge, J. and Louveaux, F. (1997) *Introduction to Stochastic Programmig*, Springer.

10 Ruszczynski, A. and Shapiro, A. (ed.) (2003), *Stochastic Programming*, vol. 10 of *Handbooks in Operations Research and Management*, Elsevier, Amsterdam.

11 Carøe, C. and Schultz, R. (1999) Dual decomposition in stochastic integer programming. *Operations Research Letters*, **24**, 37–45.

12 Sahinidis, N.V. (2004) Optimization under uncertainty: state-of-the-art and opportunities. *Computers and Chemical Engineering*, **28**, 971–983.

13 Sand, G., Engell, S., Märkert, A., *et al*. 2000, Approximation of an ideal online scheduler for a multiproduct batch plant. *Computers and Chemical Engineering*, **24**, 361–397.

14 Engell, S. and Sand, G. (2003) A two-stage stochastic integer programming approach to realtime scheduling, in *Fourth International Conference on Foundations of Computer-Aided Process Operations* (eds I. Grossmann and C. McDonald), CACHE Corp, Austin, TX, pp. 347–350.

15 Engell, S., Märkert, A., Sand, G. and Schultz, R. (2004) Aggregated scheduling of a multiproduct batch plant by two-stage stochastic integer

programming. *Optimization and Engineering*, **5**, 335–359.

16 Sand, G. and Engell, S. (2004) Modelling and solving real-time scheduling problems by stochastic integer programming. *Computers and Chemical Engineering*, **28**, 1087–1103.

17 Till, J., Engell, S. and Sand, G. (2005a) Rigorous vs. stochastic algorithms for two-stage stochastic integer programming applications. *International Journal of Information Technology*, **11** (5), 106–115.

18 Till, J., Engell, S. and Sand, G. (2006) Hybrid evolutionary algorithms for solving two-stage stochastic integer programs in chemical batch scheduling. *Computers and Chemical Engineering*, **31** (5–6), 630–647.

19 Bäck, T., Fogel D.B. and Michalewicz, Z. (eds.) (1997) *Handbook of Evolutionary Computation*, Oxford University Press, New York.

20 Goldberg, D.E. (1998) *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, MA.

21 Beyer, H. and Schwefel, H. (2002) Evolution strategies. *Natural Computing* (1), 3–52.

22 Coello Coello, C.A. (2002) Theoretical and numerical constraint handling techniques used with evolutionary algorithms: A survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering*, **191** (11–12), 1245–1287.

23 Emmerich, M., Schütz, M., Groß, B., and Grötzner, M. (2000) Mixed-integer evolution strategy for chemical plant optimization with simulators, in *Evolutionary Design and Manufacture (ACDM)*, (ed. I. Parmee), Springer, New York, pp. 55–67.

24 Märkert, A. (2004) *User's guide to ddsip – AC Package for the Dual Decomposition of Stochastic Programs with Mixed-Integer Recourse*. Institut für Mathematik, Universität Duisburg, Duisburg (http://www.uni-duisburg.de/FBll/disma/ddsip-eng.shtml, March 23, 2005).

25 Clostermann, E. (2005) Empirical analysis of scenario decomposition in chemical batch scheduling. Diplomarbeit, Universität Duisburg-Essen.

26 Wallace, S.W. and Ziemba, W.T. (eds.) (2005) *Applications of Stochastic Programming*. MPS-SIAM series in optimization. SIAM, Philadelphia, PA.

27 Vin, J. and Ierapetritou, M. (2001) Robust short-term scheduling of multiproduct batch plants under demand uncertainty. *Industrial and Engineering Chemistry Research*, **40**, 4543–4554.

28 Bonfill, A., Bagajewicz, M., Espuna, A. and Puigjaner, L. (2004) Risk management in the scheduling of batch plants under uncertain market demand. *Industrial and Engineering Chemistry Research*, **43**, 741–750.

29 Balasubramanian, J. and Grossmann, I. (2004) Approximation to multi-stage stochastic optimization in multi-period batch plant scheduling under demand uncertainty. *Industrial and Engineering Chemistry Research*, **43**, 3695–3713.

10
Scheduling Based on Reachability Analysis of Timed Automata

Sebastian Panek, Olaf Stursberg, and Sebastian Engell

10.1
Introduction

10.1.1
Flexible Batch Plants

Efficient and flexible multiproduct batch plants have gained increasing attention as they are suitable to meet the market demands for strongly diversified high-quality products in small quantities. Limited markets for such products make traditional large-scale continuous single-product plants less profitable. This trend can be particularly observed in the fine chemicals, food additives, and pharmaceutical ingredients industries. Flexible multiproduct and multipurpose batch plants enable the production of a large variety of similar products in a batch-wise (discrete) fashion. Small quantities can be produced such that the market demand is satisfied exactly and just-in-time. This flexibility requires a flexible design of the production facilities and advanced production planning techniques. Flexible multipurpose production units can be used for different operations (e.g., mixing, heating, reaction, storing). In addition, flexible connections between the facilities (switched pipes, mobile vessels) allow for rapid and automated material transfer where and when needed. On the other hand, appropriate scheduling and planning requires the planner making a large number of decisions about *what, where, how* and *when* is to be produced. These decisions may be subject to various constraints resulting from process topology, equipment assignments, equipment connectivity, inventory policies, material transfers, batch sizes and processing times, demand patterns, changeover procedures, resource and time constraints, cost functions and degrees of certainty [1].

10.1.2
Example Process

As an introductory example, we consider a two-stage chemical process: in the first stage, the raw material S_0 is separated into two intermediate products S_1 and

S_2. This separation step, denoted by T_1, is performed in the chemical separation unit R_1 (e.g., a batch distillation column) within d_1 minutes. During this time, a batch of b_1 units of S_0 is taken from the storage tank and pumped into R_1. At the end of the separation, $0.4 \cdot b_1$ units of S_1 are produced and pumped into the corresponding storage tank. According to the mass balance principle, $0.6 \cdot b_1$ units of S_2 are produced and stored in the tank for S_2. The duration d_1 of T_1 includes the transportation of the materials to the storage tanks. When T_1 finishes, pumping actions are performed such that S_1 and S_2 can be used in the second stage. In the second stage, both intermediate products, S_1 and S_2, are processed separately to obtain the final products S_3 and S_4, respectively. This stage involves heating and chemical reactions in reactor units. The production of S_3 is represented by T_2 and takes d_2 minutes to (a) pump b_2 units of S_1 from the storage tank into the reaction unit R_2, (b) to heat the material and to perform the reaction, and (c) to pump the final product S_3 into the corresponding storage tank. The storage policy for S_1 is limited: S_1 becomes unstable after γ minutes and thus must not stay longer than γ minutes in the storage tank. This timing constraint must be satisfied in each production schedule because an out-of-date batch of S_1 is useless and very expensive to dispose.

The production of S_4 involves another task T_3 in the second stage. T_3 is similar to T_2. The product S_2 is pumped into the second reaction unit R_3, and after d_3 minutes of heating and reaction, the final product S_4 is pumped into the storage tank. Since S_2 is stable, no extra timing constraints are imposed here. We assume in the sequel that dedicated storage tanks are available for each of the products and are given the same names as the products: $S_0, \ldots, S_4$. The maximum storage capacities are denoted by $B_0, \ldots, B_4$. Note also that all process data, the availability of the equipment and the production orders are known at the beginning of the scheduling time horizon. Additional information (machine breaks or new production orders) obtained in real-time during the production is not taken into account. For real-time scheduling techniques in which such information is used, see [2, 3] and the contribution by Till et al. in this volume (see Chapter 9).

10.1.3 Requirements

The scheduling of the process requires the scheduler to decide (a) how many times each task needs to be started in order to satisfy the market demand and (b) to assign starting and finishing dates to the tasks. This is nontrivial because a valid schedule must meet the following requirements:

1. A production step can start only if the required quantities of raw and intermediate materials are available in the storage tanks, e.g., T_2 can only be started when at least b_2 units of S_1 are present.
2. A production step can finish only if the subsequent storage tanks have sufficient free capacities. For instance, T_2 can finish only if the free capacity within S_3 is at

least $B_3 - b_2$ units (we assume implicitly that the reaction units cannot be used as storage units).

3. No two production steps can run in parallel on the same resource. This is obvious as the resources are always occupied in an exclusive fashion. When a batch of S_1 is being processed in R_2 by T_2, no second batch can be started until this batch has finished. This constraint becomes particularly important when several steps are assigned to the same unit.
4. The timing constraints must be satisfied. Each production step has an individual duration and finishes exactly after its processing time has expired. Preemption is not allowed during processing. Another type of timing constraints arises when unstable products as S_1 are produced. The maximum duration limit γ cannot be exceeded and thus T_2 must be started in time. Minimum waiting times can occur in cases when, e.g., a heated intermediate product needs to cool down for the next processing step.
5. The market demand must be satisfied. This means that the demanded quantities of the final products S_3 and S_4 must be present in the storage tanks when they have to be delivered to the customer. Such due dates or deadlines define timing constraints for the end of the production. Missing a due date often causes a penalty (or the customer pays less to the plant operator). Missing a deadline is not allowed here: the customer must be satisfied before the deadline expires.

10.1.4 Resource Task Networks

The scheduling and planning of batch plants may be very complex and motivates the use of intelligent, computer-based decision support systems. These, however, require that the process and the plant are described in a formal way that can be understood by computer programs. Several modeling and solution approaches as state and resource task networks (STN/RTN) emerged in academia during the 1990s to meet this demand [4–6]. In the context of mathematical programming, such networks serve as starting points for the formulation of integrated mathematical models which can be solved by mixed-integer programming (MIP) to obtain the schedules. This section focuses on the description of scheduling problems by RTN since they can be transformed straightforwardly into TA as shown in the following parts of the paper.

In the RTN approach, the process and the plant are represented by networks of different types of nodes which are connected by arcs. The nodes represent states (materials, products), tasks (production steps, operations), or resources (machines, pieces of equipment).

The arcs connect the different nodes with each other and represent either the utilization of equipment or material flows. Material flows occur when materials are produced or consumed by tasks. All arcs are labeled by fractional numbers which describe the percentage of material transported to the successor node. The overall amounts of transported materials are products of the labels of the arcs and the batch sizes.

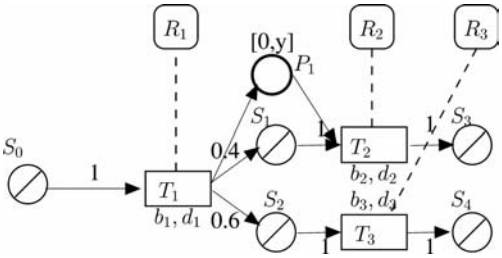

Fig. 10.1 The RTN description of the example process.

The RTN of the example process is depicted in Figure 10.1.

The raw materials, intermediate and final products $S_0, \ldots, S_5$ are shown as circles with diagonal lines. Tasks T_1, T_2, T_3 are depicted as rectangles and connected to the corresponding resources R_1, R_2, R_3 by dashed lines. The diverging material flow at the end of the separation step T_1 is represented by the fractional numbers 0.4 and 0.6 while the flows of all other arcs are 1. For each task i the parameters d_i, b_i are given and each storage S_j can be labeled by its maximal capacity B_j. Some attention must be paid to the timing constraint $[0, \gamma]$ of the unstable product S_1. The minimum storage time for S_1 is 0 and the maximum storage time is γ. The standard RTN framework offers only limited possibilities to describe such timing constraints. Zero-waiting (ZW) and no-intermediate-storage (NIS) policies have been formulated within this framework in the literature [6].

In order to model more complex timing constraints, we introduce here another type of node: *places*. The notion of places is motivated by *timed and hybrid Petri nets* [7]. Places can be connected to tasks in the same way as states, but they do not represent physical products. They become active when a predecessor task has finished. An imaginary product (called token) is then put on to the place and stays there until another task consumes it. When a token is produced, a clock that is attached to the place is started and when the minimum waiting time, e.g., 0, has expired, the token is enabled and can be consumed by the subsequent tasks. This must happen before the maximum waiting time, e.g., γ, has expired. When the token is consumed, the place becomes inactive again. Note that the capacity of places is always 1, i.e., no two tokens can be present in the same place at the same time. When a token has no predecessor tasks, it becomes active at time 0. The place P_1 in Figure 10.1 is depicted by a thick circle and limits the waiting time to a value between T_1 and T_2. A token is placed in P_1 when T_1 finishes and consumed when T_2 starts. Due to the label $[0, \gamma]$ of the place, the waiting in P_1 must not be longer than γ minutes. In general, places are useful to express rigid timing and sequencing constraints between operations, to define release and due dates of production orders, to describe binary resources, and to distinguish between parallel and alternative production paths.

Following the RTN philosophy, one may argue that the equipment could be represented by places as well. The decision to represent resources by separate elements in the extension proposed here was motivated by the clear structure, which enables the reader to distinguish between timing constraints from the recipes on one hand, and exclusive resources on the other hand.

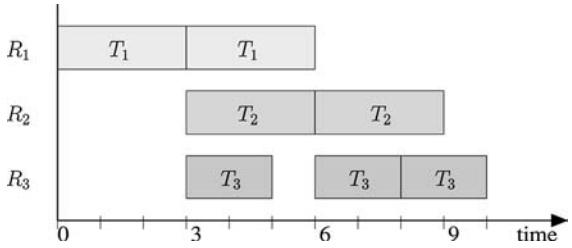

Fig. 10.2 The Gantt chart of the optimal schedule for the example process.

10.1.5 Example Schedule

We define the following parameters for the example process: $B_0 = B_1 = B_2 = B_3 = B_4 = 20$; $d_1 = d_2 = 3$; $d_3 = 2$; $b_1 = 10$; $b_2 = b_3 = 4$. The initial amount of S_1 is 20 units and the goal of the production is to produce 8 units of S_3 and 12 units of S_4. The optimal schedule with respect to the makespan (the overall production time) is shown in Figure 10.2. Each occurrence of a task in the schedule is called an operation and the number of operations must be determined by the scheduler. The schedule in Figure 10.2 is obviously valid because it satisfies the requirements from Section 10.1.3. It is also optimal, because none of the operations can be shifted to the left to reduce the makespan which is determined by the last operation of T_3. In the optimal schedule, 2 operations of both, T_1 and T_2, as well as 3 operations of T_3 are necessary to meet the market demand. The optimal makespan is 10 and all raw and intermediate materials have been processed without overproduction of final products.

It is easy to see that the schedule can be described as a sequence of discrete events and waiting periods. Discrete events represent either starting or finishing of operations on resources. Waiting periods occur when operations are running or when a product is waiting until the conditions to start the next task are satisfied. Let the starting events be denoted by $s(T)$, the finishing events be denoted by $f(T)$, and waiting periods be represented by $w(t)$ (waiting for t time units). The schedule in Figure 10.2 can be described by the following sequence of discrete events and waiting periods: $s(T_1)$, $w(3)$, $f(T_1)$, $s(T_1)$, $s(T_2)$, $s(T_3)$, $w(2)$, $f(T_3)$, $w(1)$, $f(T_1)$, $f(T_2)$, $s(T_2)$, $s(T_3)$, $w(2)$, $f(T_3)$, $s(T_3)$, $w(1)$, $f(T_2)$, $w(1)$, $f(T_3)$.

10.2 Scheduling with Timed Automata

10.2.1 Motivation

In the previous section, a simple chemical production process has been introduced and formally described by the means of RTN extended by places. The solution of the above problem is a straightforward task and can be performed without

special expertise. For more complex problems, however, efficient scheduling is not possible without a computational model and a scheduling algorithm. The extended RTN offers an intuitive graphical representation of the production process. Unfortunately, it does not offer a clear representation of states and time which would straightforwardly translate into analysis and scheduling techniques.

A very popular scheduling framework is based on mixed-integer programming. Herein, the scheduling problem is modeled in terms of variables and algebraic inequalities and solved by mathematical optimization techniques. In opposition to this well-established framework, a different approach is advocated in the paper by Alur and Dill [8] on timed automata (TA).

TA are used to model and analyze dynamic systems with discrete and timed behavior. One of their strengths is the easy modeling in a decomposed fashion: as a set of often small and individually acting automata. Time in TA is modeled in a very natural way by a set of clocks that simply measure the time between events. This is a major difference to MIP techniques, where time and dynamic components are described in a rather artificial way by providing variables and inequalities for every point of time within a discretized time horizon. In addition to the advantages in modeling, TA serve as a computational model which can be analyzed by techniques for reachability analysis. These techniques are widely used in the context of verification, in which the objective is to detect possible undesired (bad or forbidden) behaviors [9–11]. The success of these techniques was pushed by the availability and increasing performance of tools for TA, e.g., *Uppaal* [9, 10, 12, 13].

A recent and very promising application of the reachability analysis of TA is scheduling. This development, however, required the introduction of the notion of cost. In contrast to the verification of whether a behavior fulfills a specification or not, costs introduce a quantitative measure to evaluate the individual behaviors. Successful applications of *priced TA* [15, 16] by using the standard and a special version of Uppaal are documented in [17–19]. The first application to deterministic job-shop scheduling was published by Abdeddaim [20] and Abdeddaim and Maler [21]. In order to further motivate the use of TA, the next section shows how the schedule in Figure 10.2 and the plant on which the operations are scheduled naturally translate into a set of timed automata.

10.2.2 The Schedule as Automaton

As shown in Section 10.1.5, the scheduling can be understood as the generation of a sequence of discrete events and waiting periods while satisfying all conditions described in Section 10.1.3. The number of possible schedules is obviously infinite because the waiting periods represent continuous degrees of freedom: each waiting period can be any number from a bounded interval, e.g., $[0, \gamma]$. The sequence of events and waiting periods in Section 10.1.5 easily translates in a special automaton called *scheduler*. This automaton is a sequence of *locations* (circles) and *transitions* (arcs between circles), as shown in Figure 10.3.

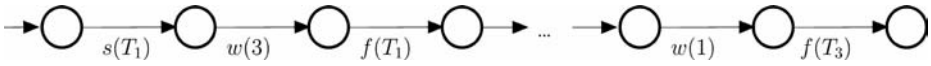

Fig. 10.3 The example schedule described as a control automaton.

The leftmost location is the initial location because of the short arc pointing to it. It is active when the automaton starts running. A running automaton has always exactly one active location which changes when an outgoing transition is taken. In Figure 10.3, the active location changes from left to right following the transitions (arcs) which connect the subsequent locations. Each transition sends an output *signal*, when it is taken. The receiver of the signals is the process/plant to be scheduled. Hence, the process/plant must be modeled in such a way that it accepts and appropriately reacts to the signals from the scheduler.

10.2.3 The Process as Timed Automata

The scheduler in the previous section has been designed to schedule the process by sending signals to it. An appropriate model of the process must take into account all relevant pieces of equipment and their properties: storage tanks, products, the connectivity of the equipment and timing constraints. It must also react to the external signals from the scheduler automaton, i.e., wait, start and terminate tasks when the scheduler expects it to do so. In order to design the process model using TA, the following observations are helpful:

- Each resource is either *idle* or *busy* when it is processing a task assigned to it.
- A resource becomes busy, when it receives the signal $s(T)$ to start the task T.
- A resource becomes idle, when it is busy and receives the signal $f(T)$ to finish the task T.
- When a task is started, it must consume all required materials from predecessor states and tokens from predecessor places.
- When a task is finished, it must transport the products to successor states and tokens to successor places.
- Each resource needs a clock to measure the durations of tasks assigned to it.

Following the above observations, the process model can be formulated by TA. For formal definitions of the syntax and semantics of TA, see [15]. TA are used to model the individual resources by *resource automata* and to describe timing constraints by *place automata*. The former are used to start and to finish tasks which are uniquely assigned to resources, the latter establish timing constraints of places.

Passive components (e.g., materials in storage tanks) can be described by *shared variables* denoted by $v_1, \ldots, v_{n_v}$. The production and consumption of materials can be represented by *actions*, which either increase or decrease the value of the shared variables. The term *shared* refers to the fact that these variables can be manipulated by any automaton in the model.

Special clock variables called *clocks* c_i, $i = 1, \ldots, N_c$ are used to measure the time between events. The values of the clocks increase permanently with the rate $\dot{c}_i = 1$. Clocks are used to measure the durations of tasks for the resource automata, and to measure the waiting between two tasks for place automata.

Each resource automaton has a cyclic structure with two locations, *idle* and *busy*, and two transitions connecting these two locations. When the signal to start a task is received, the corresponding resource automaton changes its location from *idle* to *busy* by taking the first transition. When a task is finished, the resource automaton returns to the *idle* location by taking the second transition.

Transitions must always be enabled before they can be taken. Enabling means that all *enabling conditions* associated with the transition must evaluate to true. While enabling conditions for transitions are called *guards*, a different type of enforcing conditions, called *invariants*, can be formulated for locations. A TA can stay in a location as long as the invariant attached to it evaluates to true. The location must be left before the invariants start evaluating to false. The following types of conditions are possible in most TA implementations:

1. enabling signals from external components, e.g., the signals $s(T)$ and $f(T)$ from the scheduler;
2. invariants which are conjunctions of *clock constraints* $c_i \leq k$, $k \in \mathbb{R}^{\geq 0}$ which limit the waiting in locations and thus force the automata to leave the locations when the limit has been reached;
3. guards which are conjunctions of clock constraints $c_i \leq k$ or $c_i \geq k$, $k \in \mathbb{R}^{\geq 0}$ and enable transitions;
4. guards that are given as conjunctions of arbitrary arithmetic formulae involving shared variables.

Transitions force the automaton to change its location and to perform associated actions. All actions are executed instantaneously when the transition is being taken. The order of the execution depends on the implementation. The following types of actions are defined in most TA implementations:

1. Clock resets: When a transition is taken, a clock reset action $c_i := 0$ attached to it simply assigns the value 0 to the clock c_i. Note that the clock is not stopped, i.e., the clock rate $\dot{c} = 1$ remains unchanged.
2. Actions: Each action $v_j := f(v_1, v_2, v_3, \ldots)$ evaluates an arbitrary function f of shared variables and assigns the result to the variable v_j.

10.2.4 Modeling of the Process

The considerations in the previous section about shared variables, automata, locations, transitions, enabling conditions and actions can be straightforwardly applied to the example process. It can be modeled by TA as shown in Figure 10.4. For the

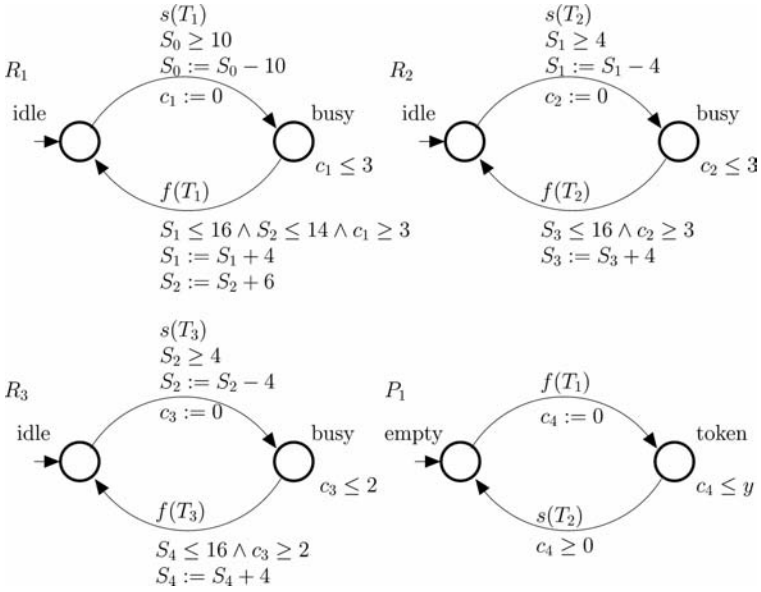

Fig. 10.4 The timed automata models of the resources R_1, R_2, R_3 and the place automaton P_1.

sake of simplicity, the shared variables are given the same names as the materials: $S_0, \ldots, S_4$. All three resources are represented by individual automata R_1, R_2, R_3 and each resource i is equipped with its own clock c_i to measure the durations of the tasks. The *busy* locations have invariants $c_i \leq d_i$ which limit the waiting according to the task durations d_i. Correspondingly, the guards $c_i \geq d_i$ on transitions from *busy* to *idle* enable the transitions when the duration of the task has expired. Each transition that consumes some material must check whether there is enough material to consume. This is performed by guards which evaluate and compare the shared variables, e.g., $S_0 \geq 10$. When a consuming transition is being taken, the material stock becomes updated by appropriate actions, e.g., $S_0 := S_0 - 10$. Correspondingly, each transition that produces some materials must check whether the free capacity of the storage is sufficient. The material stock is updated by increasing the corresponding shared variable, e.g., $S_1 := S_1 + 4$.

The fourth automaton in the model corresponds to the place P_1 which limits the storage time of S_1. The structure of this automaton is similar to the structure of the resource automata. The difference is that it does not manipulate the material stocks. It accepts the signal $f(T_1)$, resets its own clock c_4, waits for at most y time units and forces the transition back by $s(T_2)$ when the time has expired. The reader might argue that all automata ignore the waiting signals $w(t)$. These signals are not sent to a particular automaton but are modeled by guards and invariants, and force all automata to wait for some time t.

An important point is the *synchronized execution* of the automata R_1, P_1 and R_2. They obviously share the same signals $f(T_1)$ and $s(T_2)$. In order to ensure correct execution, it is important that both transitions enabled by $f(T_1)$ are taken at the same time. For the same reason, both transitions enabled by $s(T_2)$ must be taken synchronously. Both signals act in this context as *synchronization signals*. Synchronized transitions share the same synchronization signals and thus must be taken at the same time. Evolutions in which one of them is taken alone are not possible. Note that many implementations of TA support *binary synchronizations* only.

10.3 Reachability Analysis

10.3.1 Motivation

The above model consists of two main parts: a scheduler and the plant to schedule. Since the scheduler defines exactly when to start and finish the tasks, the behavior of the entire system is deterministic. Running both parts, the scheduler and the plant, corresponds to a simulation in which one single behavior of the composed system is obtained. Obviously, this situation requires the presence of a scheduler which "knows" the (optimal) schedule. If such a scheduler is absent, then the resource automata in the plant receive no signals. Scheduling of a plant can be understood as the task of finding a scheduler automaton which provides the optimal schedule with respect to an optimization criterion. In the sequel it is assumed that a scheduler does not exist and the automata model is designed as shown in Figure 10.4.

Consider the situation when sufficient quantities of raw and intermediate materials are present and the resource automata in Figure 10.4 are waiting in the *idle* locations. Without a scheduler which exactly determines the next production step, either R_1 or R_2 or R_3 can start processing a batch. The possible actions are $s(T_1)$, $s(T_2)$, $s(T_3)$ or $w(\epsilon)$ with $\epsilon \in \mathbb{R}^{\geq 0}$. Hence, the scheduler has to choose from a set of possible decisions. This set is infinite because the waiting time ϵ is a real number.

Obviously, the model without the scheduler and the signals is *nondeterministic*. Nondeterminism means that the model can evolve in different directions depending on *degrees of freedom*. The degrees of freedom result from the set of concurrent timed automata which represent concurrent and partly independent machines. Fixing a degree of freedom is equivalent to making a decision at some point of time and picking one of the (infinite) set of decision alternatives. It is *a priori* not clear which decisions will lead to the optimal schedule. Different decisions lead to different evolutions, and different evolutions often (but not always) lead to different schedules in which, e.g., the number and the order of operations vary. The infinite number of possible schedules corresponds to the infinite number of possible evolutions of the automata.

Each valid evolution starts in a predefined initial state in which sufficient quantities of the raw materials are given. Furthermore, it must meet the requirements from Section 10.1.3. A schedule will be accepted only if the market demand has been satisfied. Hence, at the end of the evolution, the demanded quantities of final products must be present in the storages. Each evolution which does not satisfy the above requirements leads to an invalid schedule and thus must be rejected. In order to synthesize a valid and optimal schedule, the scheduler has to fix all degrees of freedom by choosing appropriate signals for the resource automata. A valid and optimal schedule corresponds to evolutions which minimize a given optimality criterion.

10.3.2 Parallel Composition

The TA model introduced in Figure 10.4 is nondeterministic and describes an infinite number of different possible evolutions which correspond to different schedules. Scheduling based on this model is equivalent to picking the evolution that minimizes the optimization criterion. Since the consideration of an infinite number of alternatives is difficult, a technique called *symbolic reachability analysis* is applied and will be briefly explained in the next section. In order to perform the symbolic reachability analysis, the individual automata A_i can be composed into one automaton A by *parallel composition*. A is composed such that it combines all locations, transitions, clocks and variables of the individual automata. Any evolution of A corresponds to interleaved evolutions of the single automata. The composed automaton is often very large and complex, and thus a manual composition is difficult. Fortunately, this procedure is well-supported by most tools for TA: it is performed *on-the-fly* during the reachability analysis. For the sake of better understanding, the initial part of the parallel composition of the automata in Figure 10.4 is exemplarily presented in Figure 10.5. The symbol of the parallel composition is $||$ and the automaton A is composed by $A := R_1||R_2||R_3||P_1$. The initial location of A is a combination of initial locations of the individual automata: $\mathbf{l}_0 =$ (idle, idle, idle, empty) and the other locations depicted in Figure 10.5 are those in which either T_1, T_2 or T_3 is running. The composed locations are as follows: $\mathbf{l}' =$ (busy, idle, idle, empty), $\mathbf{l}'' =$ (idle, busy, idle, empty), $\mathbf{l}''' =$ (idle, idle, busy, empty). An evolution of A is a sequence of states and transitions, e.g., $\mathbf{q}_0 \xrightarrow{t} \mathbf{q}_1 \xrightarrow{\delta} \mathbf{q}_2 \xrightarrow{t'} \mathbf{q}_3 \xrightarrow{\delta'} \cdots \xrightarrow{\delta_n} \mathbf{q}_n$, and corresponds to interleaving evolutions of the single automata.

The transitions are either time transitions $t, t', \ldots, t^n$ which represent waiting, or discrete transitions $\delta, \delta', \ldots, \delta^n$ which correspond to events $s(T)$ and $f(T)$. Each valid schedule is an evolution of A which starts at an initial state $\mathbf{q}_0 = (\mathbf{l}_0, \mathbf{u}_0, \mathbf{v}_0)$ with $\mathbf{u}_0 = (0, 0, 0, 0)$, $\mathbf{v}_0 = (20, 0, 0, 0, 0)$, and ends in a state in which the market demand is satisfied, e.g., $\mathbf{q}_n^* = ((idle, idle, idle, empty), (*, *, *, *), (0, 0, 0, 8, 12))$. Note that the number of possible evolutions and the *state space* of A (a subset of the reachable states $\mathbf{q}$), are infinite.

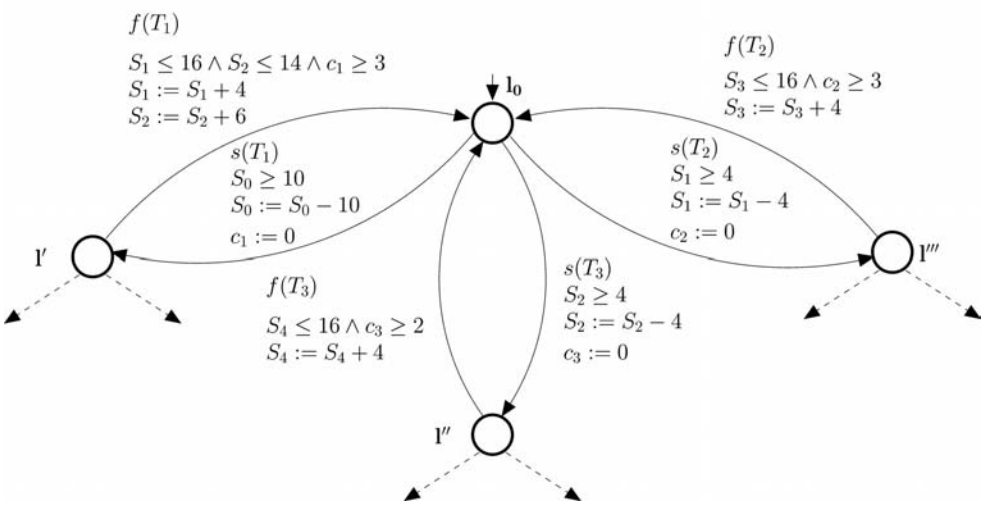

Fig. 10.5 The initial part of the parallel composition A of the automata R_1, R_2, R_3 and the place automaton P_1.

10.3.3 Reachability Algorithms

Symbolic reachability algorithms for TA overcome the problem of the infinite state space by defining (a) *zones* as finite representations of infinitely many clock valuations, or (b) by reducing the possible waiting intervals to single values. Zones are useful for verification where all possible evolutions must be taken into account. They represent regions within the clock space by polyhedra for which finite representations are used. The second possibility is particularly efficient and useful for scheduling where only the optimal evolution is of interest. In many scheduling problems it is possible to collapse zones to single vectors of optimal clock valuations. Both techniques reduce the infinite *state space* to a finite *symbolic search space* which can be represented as a finite, directed and priced graph. The *nodes* of this graph are called *symbolic states* and have the form $(\mathbf{l}, Z, \boldsymbol{\nu})$ with a location vector $\boldsymbol{l}$, a zone Z and values of variables $\boldsymbol{\nu}$.

In the context of reachability analysis, this graph is called *symbolic reachability graph* of the automaton A and can be searched using *shortest path search techniques* as widely used in computer science. Hence, the task of finding the cost-optimal schedule is to find the shortest (or cheapest) path in a (priced) symbolic reachability graph.

The following reachability algorithm is a *best-first forward reachability* algorithm [15, 21]. It is designed for the evaluation of costs of symbolic states and thus particularly useful for scheduling. The main difference to reachability algorithms for verification is that it ignores evolutions which exhibit costs higher than the costs of known solutions.

```
cost* = ∞; Put q0 into W;
While W ≠ ∅
        Pick the cheapest q from W
        If cost (q) < cost*
                If final (q)
                        cost* = cost (q)
                Else
                        W : = W ∪ Succ (q)
                End
        End
End;
```

The search starts from the initial state $\mathbf{q}_0$. The set W is a data structure which stores symbolic states which are not yet explored. The function *final* decides whether the given symbolic state is a target state in which the production is completed. The symbolic reachability graph is step-wise constructed by evaluating the successor relation *Succ*(**q**), which computes the successor symbolic states of **q**. The best solution is returned in *cost**. Existing tools implement numerous extensions of the standard algorithm to improve the efficiency:

- Keeping track of explored nodes to avoid visiting the same node twice (*passed list lookups*) [12, 21].
- Various *guiding functions* as criteria to pick the most promising states from W: depth-first search, breadth-first search, bounded-width search [21], best-lower-bound search, random search, combined search criteria [22], etc.
- Computation of lower bounds of costs to the final state. This technique is common in branch-and-bound algorithms, lower bounds can be obtained by heuristics [21] or by linear programming [22–25].
- Nonlaziness techniques [21] and the sleep set method [22, 26].

10.3.4 The Reachability Graph

The reachability analysis of A performed by the above algorithm explores the symbolic reachability graph step-by-step. The result is the path from the initial state to the target symbolic state which has the lowest makespan. The total graph of A for the example is composed of 145 nodes, each of which corresponds to a symbolic state. The initial part of the reachability graph and the optimal trace are presented in Figure 10.6. The optimal path consists of 15 nodes and represents the optimal schedule depicted in Figure 10.2. These nodes are connected by 14 discrete transitions, each of which is preceded by a time transition. The nodes of the optimal path without clock valuations are shown in Table 10.1. The second column shows the absolute time, the third column contains the waiting times of the time transitions. The remaining columns comprise the locations of the individual automata and the values of the shared variables.

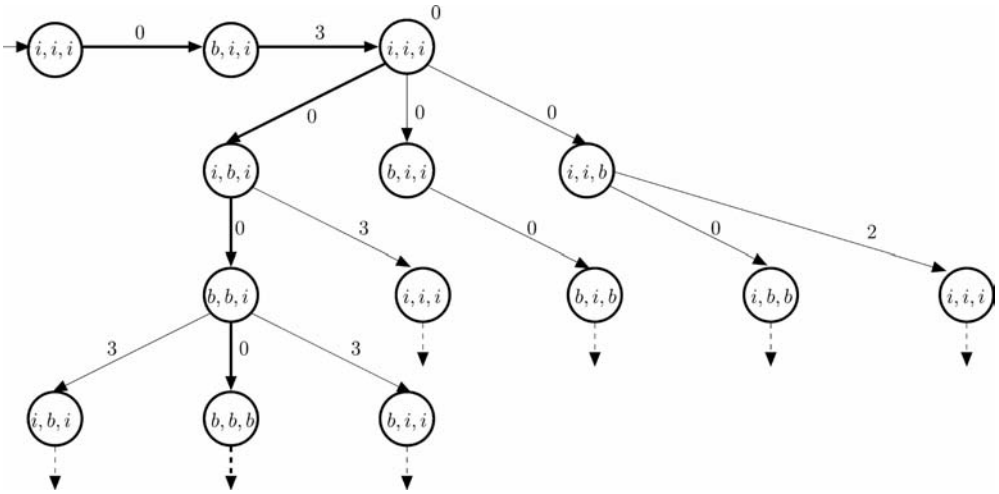

Fig. 10.6 The initial part of the reachability graph of *A*. The node names give information about the locations of the automata R_1, R_2, R_3: *i* is the idle and *b* is the busy location. The arcs are annotated with the time transitions preceding the discrete transitions. The bold arcs represent the optimal trace.

Table 10.1 The optimal path of the reachability graph. The length is 15 nodes and the makespan is 10. The clock valuations of the individual clocks c_i are not shown, only the absolute time is presented in the second column.

			Locations				Variables				
No.	**time**	**wait.**	R_1	R_2	R_3	P_1	S_0	S_1	S_2	S_3	S_4
1	0	0	*idle*	*idle*	*idle*	*empty*	20	0	0	0	0
2	0	0	*busy*	*idle*	*idle*	*empty*	10	0	0	0	0
3	3	3	*idle*	*idle*	*idle*	*token*	10	4	6	0	0
4	3	0	*idle*	*busy*	*idle*	*empty*	10	0	6	0	0
5	3	0	*busy*	*busy*	*idle*	*empty*	0	0	6	0	0
6	3	0	*busy*	*busy*	*busy*	*empty*	0	0	2	0	0
7	5	2	*busy*	*busy*	*idle*	*empty*	0	0	2	0	4
8	6	1	*idle*	*busy*	*idle*	*token*	0	4	8	0	4
9	6	0	*idle*	*idle*	*idle*	*token*	0	4	8	4	4
10	6	0	*idle*	*busy*	*idle*	*empty*	0	0	8	4	4
11	6	0	*idle*	*busy*	*busy*	*empty*	0	0	4	4	4
12	8	2	*idle*	*busy*	*idle*	*empty*	0	0	4	4	8
13	8	0	*idle*	*busy*	*busy*	*empty*	0	0	0	4	8
14	9	1	*idle*	*idle*	*busy*	*empty*	0	0	0	8	8
15	10	1	*idle*	*idle*	*idle*	*empty*	0	0	0	8	12

Note also that the waiting periods in the third column directly correspond to the signals $w(t)$ in Section 10.1.5 and the makespan of the schedule is specified in the second column of the last row. The effort of the reachability analysis is mainly determined by the size of the reachability graph. In most cases, however, the above algorithm will not explore the entire graph, but only a part of it. The reason is the comparison of the costs of nodes with the value $cost^*$ of the best solution found by the algorithm. When a solution is known, it enables the algorithm to cut those nodes which have higher costs. As long as no solution is known, $cost^* = \infty$ applies and no cuts are possible.

The effort to find the best solution can be reduced by defining an appropriate *guiding function*. A guiding function (third line of the algorithm) attempts to select those nodes from W, which are assessed to be best or cheapest. If a perfect guiding function was given (similar to a scheduler automaton which knows the optimal solution), it would always make the optimal decisions and step-by-step select the nodes in Table 10.1 from W.

The search could be stopped after 15 nodes because the last node corresponds to the optimal solution. Backtracking in the reachability graph would not be necessary. Since no such perfect function exists in the general case, other criteria must be used. In this context, a combination of depth-first and best-first search was found to be particularly successful [22]. The nodes from the waiting list W are selected such that (a) nodes with the maximum depth are selected first, (b) if there exist nodes with equal depths, then those nodes with minimal cost are selected. With this setting, the best solution is found after exploring 25 nodes and the overall search effort is reduced from 145 to 84 nodes. The search from node 26 to node 84 is only necessary to make sure that no better solutions exist. The distance from 25 to the minimum of 15 nodes leads to the conclusion that only a very moderate amount of backtracking was necessary to find the best symbolic state and thus the guiding function performed very well for this small example.

10.4 Benchmark Example

10.4.1 Description

In this section, the TA-based modeling and solution approach is applied to a popular benchmark scheduling problem from Kallrath [1]. It is a multistage and multiproduct chemical batch plant.

Figure 10.7 shows the extended RTN formulated for the benchmark problem. The production process includes diverging and converging material flows, flexible proportions of output goods (task T_2), cyclic material flows (recycling of output from task T_3 into state S_1), intermediate products which cannot be stored (state nodes S_5, S_9, S_{10}, S_{12}), and blending of products in task T_{15}. All processing tasks are operated batch-wise with lower and upper bounds on batch sizes. Batch sizes are

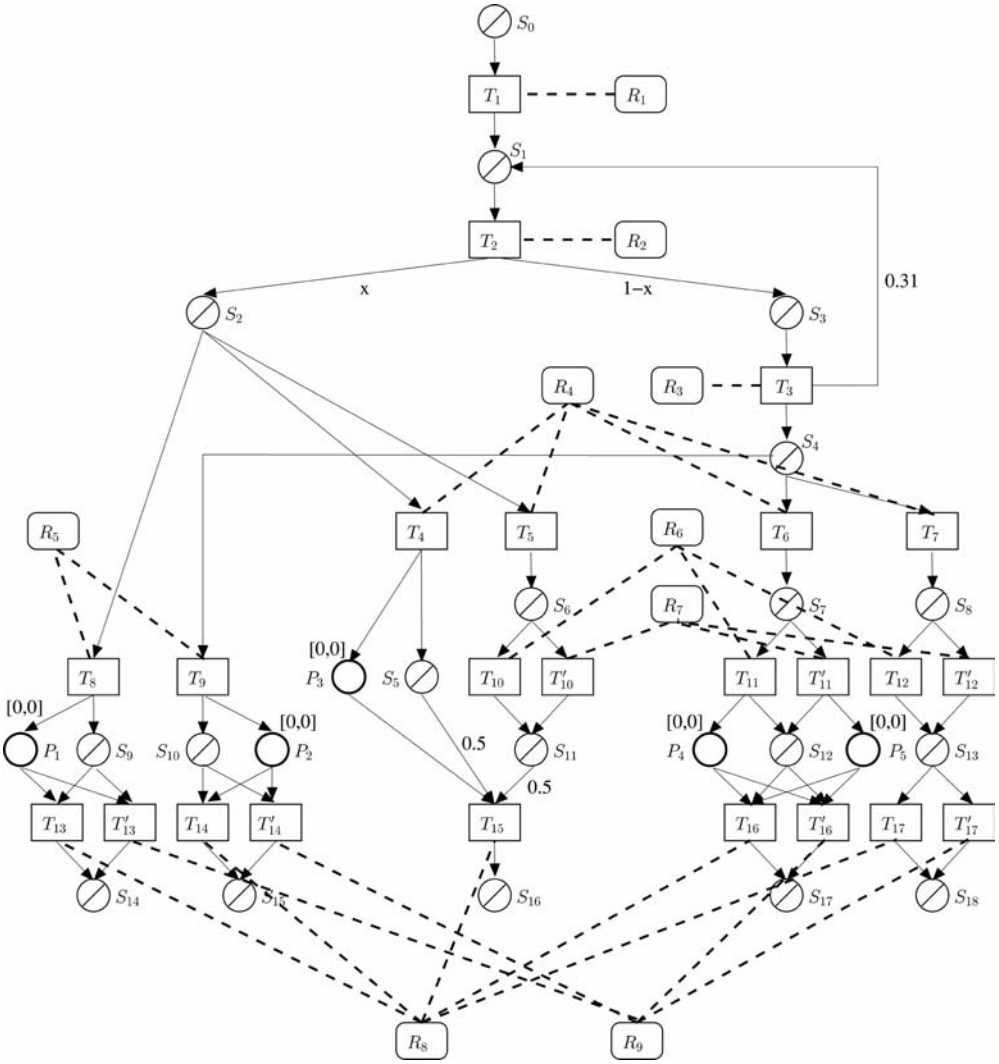

Fig. 10.7 The extended RTN for the benchmark problem.

mainly restricted by the capacities of the units. All units are utilized in an exclusive mode to process one batch by one task. Each task allocates the corresponding unit for a fixed time period which does not depend on the actual batch size. The initial stock values for the raw materials S_1, S_2, S_4 are defined as 20, 20, 20, the initial stock for S_0 is not prescribed and can be freely chosen. All other material stocks are zero at the beginning. The goal of the production is to produce 30, 30, 40, 20, 40 tons of the products $S_{14}, \ldots, S_{18}$. The objective function is to minimize the makespan. An illustrative RTN description of the original problem is given by Bloemer and Guenther [27].

A manual batch sizing procedure was used to determine constant batch sizes for all operations. Batch sizes for tasks running on the resources R_1, $R_3 \ldots$, R_5, R_8, R_9 are fixed to 10. The batch size of T_2, which was experimentally identified as a bottleneck, was chosen as 20. With respect to R_6 and R_7, the batch sizes for the corresponding tasks $T_{10}, \ldots, T'_{12}$ are chosen as 5 because of the resource capacities.

The task durations are as follows: 4 time units for the tasks T_1, T_3, 8 for T_2, T_4, T_5, T_6, T_7, T_{10}, T_{13}, T_{14}, T_{15}, 10 for T'_{10} and 12 for T_8, T_9, T_{11}, T'_{11}, T_{12}, T'_{12}, T'_{13}, T'_{14}, T_{16}, T'_{16}, T_{17}, T'_{17}.

The fact that the unstable products S_6, S_9, S_{10}, S_{12} cannot be stored (NIS) is modeled as follows: separate storages with infinite capacities are defined for these products. Waiting is forbidden by adding timing constraints [0, 0] as places $P_1, \ldots, P_5$ between the corresponding tasks. Hence, after finishing the production the following tasks have to be started immediately. Note that the tasks T_{11} and T'_{11} must always run in parallel to produce as much of S_{12} as is necessary to start either T_{16} or T'_{16} immediately afterwards. The variable material flows are fixed to constant values $x = 0.5$ and $1 - x = 0.5$ for both outgoing arcs of the task T_2.

This choice is motivated by the satisfiability of the production goal and the limited capacity of S_2 and S_3. Note that some values of x, e.g., $x = 0.2$ or $x = 0.3$, can lead to an infeasible problem because the limited capacity of the storage S_3 can prevent any successive T_2 from being finished and any T_3 from being started.

10.4.2 Computational Results

The tool *TAopt* [22–24] is used to solve the benchmark problem. TAopt combines cost-optimal reachability analysis as described in Section 10.3.3 with heuristics and advanced state space reduction techniques [22]. It is used (a) to read the problem description formulated as an extended RTN as presented in Figure 10.7, (b) to build the corresponding TA model automatically from the RTN description, and (c) to perform the cost-optimal reachability analysis augmented by additional search space reduction techniques. The output is a set of schedules and corresponding Gantt charts. The TA modeling performed by TAopt follows the scheme described in Section 10.2.4: one resource automaton is created for each of the 9 resources and 5 additional place automata are constructed to establish the zero-waiting constraints between some tasks. Each of the states is modeled by one shared variable such that 19 variables are used in total. Each automaton is given its own clock and one additional global clock is defined to measure the absolute time and the makespan.

The computational equipment was a 3.06 GHz Xeon machine with 2 GB of main memory and Linux operating system. All tests were limited to five million of visited nodes and this limit was reached in all test runs. The configuration of the waiting list W of TAopt was *depth-first search* combined with *cost minimization*. In order to reduce the search effort, the search space was reduced by reduction techniques described in Panek et al. [22] with small modifications. Results of test runs with different initial quantities for S_0 are shown in Table 10.2. The results show the computation times, the numbers of nodes visited to find the solution,

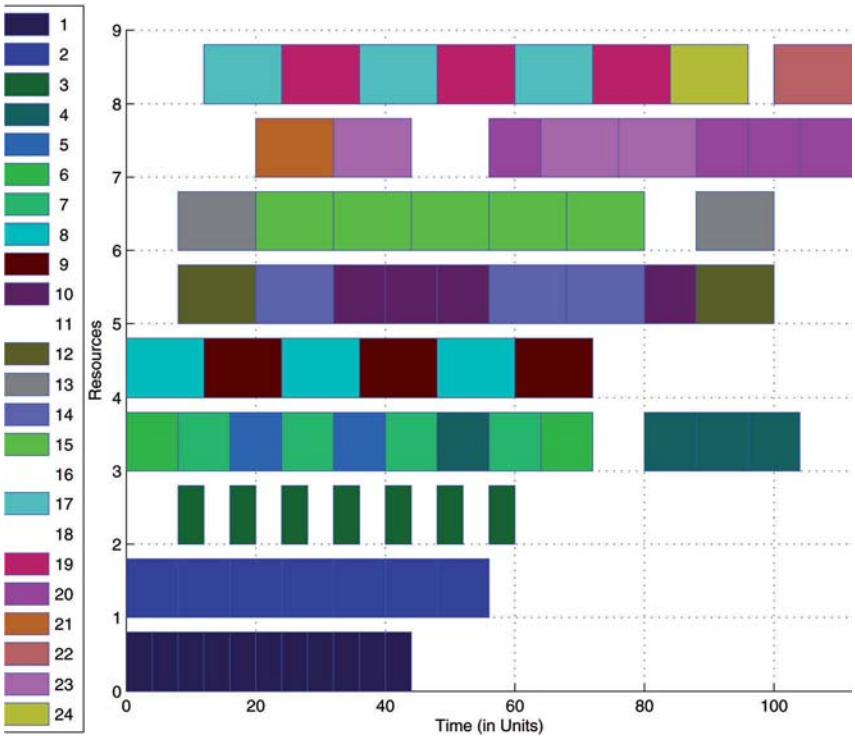

Fig. 10.8 A Gantt chart of the schedule with the best makespan of 112 time units.

the makespans (mspan) and the overproductions (op). Since the solution can be improved during the exploration of the reachability graph, the first and the best solutions are shown in each line. The Gantt chart of the best makespan solution (first line) is depicted in Figure 10.8. The indices of the resources are shown on the left side and the task numbers can be concluded from the legend. It is obvious that some tasks are not used in this schedule. It should be emphasized that the same scheduling problem could *not* be solved by MILP techniques discussed in the contribution of Mendez et al. in this volume (Chapter 9). The reason might be that the MILP model was designed to determine the optimal batch sizes, but the TA model presented here uses predefined batch sizes and thus has fewer degrees of freedom.

10.4.3 Observations and Conclusions

The results presented in Table 10.2 lead to some interesting observations. Generally, all problem instances could be scheduled within reasonable computation times and thus the method is suitable for on-line scheduling. The next observation is that the overproduction generally increases when higher amounts of raw materials are given

Table 10.2 Test results. The values refer to the points of time at which the first-best solution was found, the total effort to terminate was nearly the same ($5 \cdot 10^6$ nodes). Times are in seconds, initial quantities of S_0 are in tons, objective values (mspan) are in time units and the overproduction (op) is measured in tons.

	First				Best			
$S_0(0)$	time	nodes	mspan	op	time	nodes	mspan	op
110	13.83	234788	116	0	23.83	364146	112	0
120	34.28	298488	128	10	34.29	298750	120	10
140	22.43	390427	142	40	32.45	491816	128	20
160	5.472	146387	142	40	23.50	385252	132	30
180	32.22	753853	158	60	71.65	1318555	136	30

at the beginning. This is not surprising because the overproduction is not penalized directly but only indirectly when the makespan increases. The results lead to the conclusion that it is not advantageous to provide more raw materials than necessary, because the scheduler would try to start superfluous tasks. Both, the makespan and the overproduction would probably increase and the solutions found by the algorithm would be worse. When comparing the first and the best (i.e., last) solution the following can be concluded: in all test runs the first solution is considerably improved by the successive solutions in moderate time. The improvement applies to both, the makespan and the overproduction in most cases. The Gantt chart in Figure 10.8 reveals that the best schedule is dense and probably close to the global optimum.

10.5 Summary

A scheduling framework for flexible multiproduct batch plants based on timed automata was explained by discussing a toy-size scheduling problem. A simple TA model composed of four automata was developed and used to compute schedules by reachability analysis techniques. The applicability of the approach to real-world scheduling problems was demonstrated by modeling and solving a popular benchmark scheduling problem. The encouraging results emphasize the strength of the TA framework: a graphical, intuitive and decomposed modeling combined with an impressive solution performance resulting from recent advances in reachability algorithms and the availability of powerful computational tools.

Acknowledgment

This work was financially supported by the European Union within the project AMETIST (IST-2001-35304).

References

1 Kallrath, J. (2002) Planning and scheduling in the process industry. *OR Spectrum*, **24**, 219–250.

2 Engell, S., Maerkert, A., Sand, G., and Schultz, R. (2004) Aggregated scheduling of a multiproduct batch plant by two-stage stochastic integer programming. *Optimiz. Eng.*, **5**, 335–359.

3 Engell, S. and Sand, G. (2003) *A Two-Stage Stochastic Integer Programming Approach to Real-Time Scheduling*. Proceedings of the 4th International Conference on Foundations of Computer-Aided Process Operations (eds. I.E. Grossmann and C.M. McDonald), pp. 347–350.

4 Kondili, E., Pantelides, C.C. and Sargent, R.W.H. (1993) A general algorithm for short-term scheduling of batch operations–MILP formulation. *Comp. Chem. Eng.*, **17**, 211–227.

5 Shah, N., Pantelides, C.C. and Sargent, R.W.H. (1993) A general algorithm for short-term scheduling of batch operations. ii. Computational issues, i. MILP formulation. *Comp. Chem. Eng.*, **17**(2), 229–244.

6 Ierapetritou, M. and Floudas, C. (1998) Effective continuous-time formulation for short-term scheduling–multipurpose batch processes. *Ind. Eng. Chem. Res.*, **37**, 4341–4359.

7 Barthomieu, B. and Diaz, M. (1991) Modeling and verification of time dependent systems using time Petri nets. *IEEE Trans. Software Eng.*, **17**(3), 259–273.

8 Alur, R. and Dill, D.L. (1994) A theory of timed automata. *Theor. Comp. Science*, **126**(2), 183–235.

9 Larsen, K.G., Petterson, P. and Yi, W. (1997) Uppaal in a nutshell. *Int. J. Software Tools Technol. Transfer*, **1**(1), 134–152.

10 Yovine, S. (1998) Model checking timed automata, in *Embedded Systems*, vol. **1494** of *LNCS*, Springer, Berlin, pp. 114–152.

11 Behrmann, G., Larsen, K.G., Pearson, J., *et al.* (1999) *Efficient Timed Reachability Analysis Using Clock Difference Diagrams*. Proceedings of CAV'99, Springer, Berlin, pp. 341–353.

12 Behrmann, G., Bengtsson, J., David, A., *et al.* (2002) *Uppaal Implementation Secrets*. Proceedings of 7th International Symposium on Formal Techniques in Real-Time and Fault Tolerant Systems, Springer, Berlin, pp. 3–22.

13 Yovine, S. (1997) Kronos: a verification tool for real-time systems. *Int. J. Software Tools Technol. Transfer*, **1**(1/2), 123–133.

14 Mounier, L., Graf, S. and Bozga, M. (2002) *If-2.0: a Validation Environment for Componentbased Real-Time Systems*. Proceedings of CAV'02, *vol. 2404 of* LNCS, Springer, Berlin, pp. 343–348.

15 Behrmann, G., Fehnker, A., Hune, T.S., *et al.* (2001) *Efficient Guiding Towards Cost-Optimality in Uppaal*. Proceedings of TACAS'01, Springer, London, pp. 174–188.

16 Larsen, K., Behrmann, G., Brinksma, E., *et al.* (2001) *As Cheap as Possible: Efficient Cost-Optimal Reachability for Priced Timed Automata*. Proceedings of CAV'01, vol. 2102 of *LNCS*, Springer, Berlin, pp. 493–505.

17 Fehnker, A. (2002) Citius, Vilius, Melius: Guiding and Cost-Optimality in Model Checking of Timed and Hybrid Systems. Dissertation, KU Nijmegen.

18 Behrmann, G., Larsen, K.G. and Rasmussen, J.I. (2005b) Optimal scheduling using priced timed automata. *ACM Sigmetric*, **32**(4), 34–40.

19 Behrmann, G., Brinksma, E., Hendriks, M. and Mader, A. (2005a) *Scheduling Lacquer Production by Reachability Analysis– a Case Study*. Proceedings of the 16-th IFAC World Congress. International Federation of Automatic Control, Laxenburg, Austria.

20 Abdeddaim, Y. (2002) Scheduling with Timed Automata. Dissertation, VERIMAG, Institut National Polytechnique de Grenoble.

21 Abdeddaim, Y. and Maler, O. (2001) Job-shop scheduling using timed automata. *Computer-Aided Verification (CAV)* (eds G. Berry and H. Comon), vol. **2102** of *LNCS*, Springer, Berlin, pp. 478–492.

22 Panek, S., Stursberg, O. and Engell, S. (2006) Efficient synthesis of production schedules by optimization of timed automata. *Contr. Eng. Pract.*, in press.

23 Panek, S., Stursberg, O. and Engell, S. (2004b) *Optimization of Timed Automata Models Using Mixed-Integer Programming*. Proceedings of Formal Modeling And Analysis of Timed Systems, vol. 2791 of *LNCS*, Springer, Berlin, pp. 73–87.

24 Panek, S., Stursberg, O. and Engell, S. (2004a) *Job-Shop Scheduling by Combining Reachability Analysis with Linear Programming*. Proceedings of the IFAC Workshop on Discrete Event Systems, Springer, Berlin, pp. 199–204.

25 Rasmussen, J.I., Larsen, K.G. and Subramani, K. (2004) *Resource-Optimal Scheduling Using Priced Timed Automata*. Proceedings of TACAS'04, Springer, London.

26 Godefroid, P. (1991) *Using Partial Orders to Improve Automatic Verification Methods*. Proceedings of the 2nd Workshop on Computer-Aided Verification, vol. **531** of *LNCS*, Springer, Berlin, pp. 176–185.

27 Bloemer, F. and Guenther, H.O. (2000) Lp-based heuristics for scheduling chemical batch processes. *Int. J. Product. Res.*, **35**, 1029–1051.

Part V
Interaction with ERP System

11
Integrated Short and Midterm Scheduling of Chemical Production Processes – A Case Study

Mathias Göbelt, Thomas Kasper†, and Christopher Sürie

11.1
Introduction

Optimization of logistic processes is an important task in the chemical industry. The aim of this chapter is to demonstrate how a case study can be solved using state-of-the-art advanced planning software from SAP. Therefore, the concept of advanced planning systems is introduced first. Then characteristics that distinguish production processes in the chemical industry from those in discrete manufacturing are highlighted because these require particular attention in model building. Based on these foundations, the case study is presented. Its solution based on the SAP Supply Chain Management software is then explained, focusing first on how to model the business requirements within the software and second on how to solve these models.

11.2
Advanced Planning in Chemical Industries

11.2.1
Planning Framework

Different planning philosophies prevail in industry: simultaneous planning approaches and successive planning approaches. While the first one is clearly the best choice in smaller well-defined planning situations, in supply chains often thousands of individual decisions need to be made and coordinated. Due to the high degree of complexity successive planning approaches are therefore often chosen in practice.

The concept of successive planning is to decompose the overall decision problem into smaller subproblems and to tackle each of these with a suitable solution methodology. This decomposition often follows the principals of hierarchical planning, as most practical problems can be structured hierarchically. In the area of supply chain management, the so-called supply chain planning matrix is an

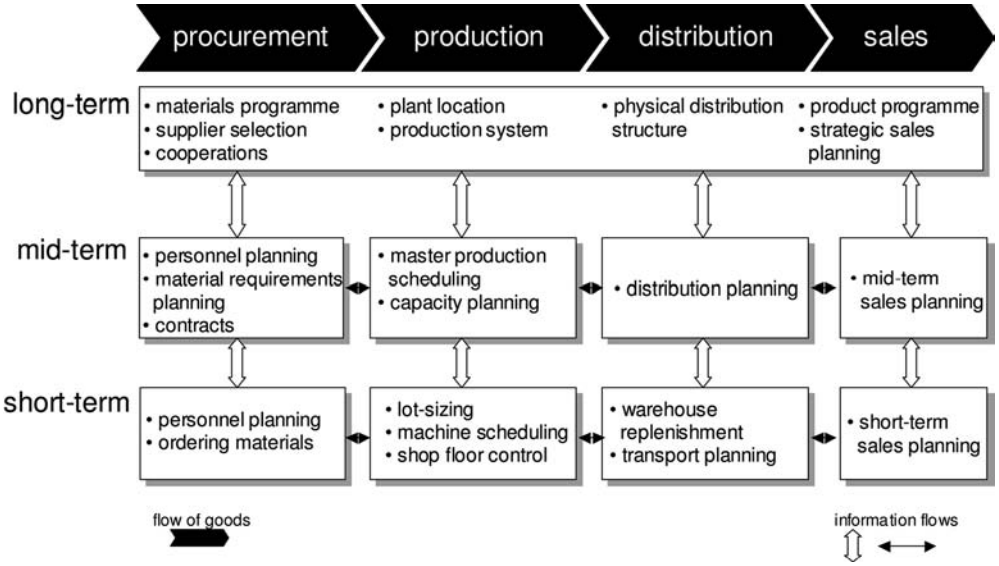

Fig. 11.1 Supply chain planning matrix (planning tasks, p. 87 in [1]).

established concept to hierarchically structure the planning tasks that arise in supply chains (e.g., see [1]).

Figure 11.1 shows typical planning tasks that arise in supply chains. These planning tasks are arranged in two dimensions. The first dimension is the *supply chain process*. In this dimension, planning tasks are arranged focusing on the most important processes following the flow of goods in supply chains. These are procurement, production, distribution and sales. The second dimension is the *planning horizon*. In this dimension the planning tasks are distinguished by their temporal impact on the supply chain. These may be strategic decisions with a long-term impact or operational decisions, which have only an immediate impact in the near future (short-term).

However, the assignment of planning tasks to certain positions in the supply chain planning matrix in Figure 11.1 is not fixed, but depends on the individual supply chain. For example, the decision upon lot sizes is often seen as a short-term planning task in the production process. On the other hand, as will be shown in the case study below, in chemical industries lot sizes must often be decided well in advance, because of long throughput times or because certain active ingredients in pharmaceutics require long lead times. In this case, lot sizing is a decision that has to be made on a midterm planning horizon (1–2 years).

To support decision making in supply chains, advanced planning systems (APS) have been developed and successfully applied in industry by several software vendors. Most of these systems rely on some kind of decomposition of planning tasks into software modules as shown in Figures 11.1 and 11.2.

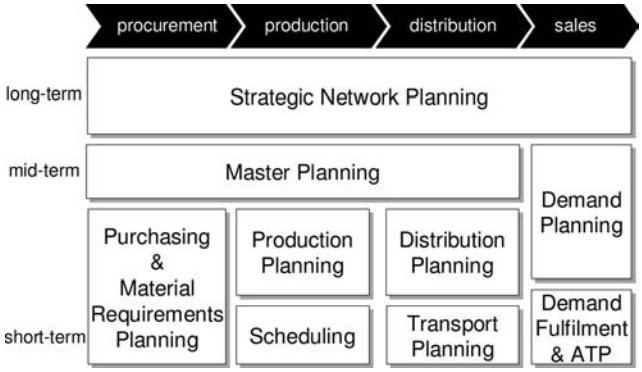

Fig. 11.2 Supply chain planning matrix (APS modules, p. 109 in [2]).

The strategic network planning module covers the long-term decisions across all supply chain processes. It supports the user to determine the structure of the supply network (plant location, distribution system) as well as the product program. Although its results are important for the long-term profitability of supply chains, it is often not a core functionality of APS. This is because APS are primarily built to support daily business, whereas strategic decisions are only reviewed periodically and most often not within the regular organization, but rather on a project basis.

The demand planning module is used for short-term and midterm sales planning. It covers basic statistical forecasting methods, but is also capable of taking additional aspects into account. For example, these may be promotions in short-term sales planning or the consideration of product lifecycles in midterm sales planning.

Short-term sales planning is sometimes supported by a demand fulfillment and available-to-promise (ATP) module. Its purpose is to match inventories and production orders with demands, if customers require reliable quotes with only short notice.

The master planning module coordinates procurement, production and distribution on a midterm level. Its major decision support is about sourcing: which product is produced at which location and when. Thus, in this module the master production schedule is fixed. However, it is important to anticipate the key characteristics of the lower (short-term) planning levels within this module, because otherwise inconsistent plans (for procurement, production and distribution) will result on the lower planning level.

In the area of distribution and transport planning, distribution related planning tasks are addressed, the latter on a more detailed level (e.g., scheduling of transports, vehicle loading and routing). Production planning and scheduling on the other hand are the two modules that support production related issues in the short-term planning horizon. Finally, purchasing and material requirements planning support the (short-term) procurement of materials and components.

11.2.2 Characteristics of Chemical Industries

Different types of industries require different characteristics to be taken into account, because in model-based planning the real decision situation must be represented adequately, as the solution will otherwise not provide any benefit. Along the lines of Meyr and Stadtler [3], the characteristics of different supply chains can be classified into functional attributes (procurement type, production type, distribution type, and sales type) and structural attributes (topography of a supply chain, integration, and coordination).

In the remainder, the focus will be on the special characteristics of chemical industry supply chains in contrast to manufacturing (discrete) supply chains.

First, in the chemical industry products or intermediates are often perishable. As a consequence stocking policies have to be obeyed, when production is planned (e.g., see [4]). Therefore, products and intermediates are classified into different categories according to their perishability: zero-wait, unlimited-wait and finite-wait (e.g., see [5]). Zero-wait products require steady processing as they cannot be stored. In their case, it is important that resources are cleaned and set up properly, when the product or the intermediate is released from a preceding processing step. As resources are often highly specialized and require a large utilization to recover the capital investment, good production plans are crucial. On the other hand, unlimited-wait products allow for buffers in the supply chain and are used for the decoupling of production processes. However, the relatively high value of products and intermediates prohibit the build up of too much stock as well as restrictions on the storage capacities (see below). Finally, finite-wait products allow storage for a limited amount of time.

Second, tied to perishability is the question of storage. Storage capacity is often scarce in chemical production facilities, as special equipment (e.g., tanks) is needed, which might additionally be dedicated to a certain set of products. Tanks can only store one single product at each point in time. Furthermore, if a tank needs to be used for another product, cleaning operations are required.

Third, processing times may require special modeling in chemical industry. While in discrete manufacturing processing times for a certain lot are usually dependent on the lot size, i.e., the number of units to be produced, this is often not true in the chemical industry. Here, processing times are often constant, irrespective of whether a reactor is filled to 70% or 90% of its capacity. This is often referred to as batch production [5]. On the other hand, the quality of the material produced may depend on resource utilization. Certain reactions may not even be feasible, if a minimum bound of the procured material is not exceeded. This implies additional restrictions regarding the resource utilization level on the planning situation.

Finally, an important characteristic identified in chemical production processes are time-consuming and costly setup operations. Thus, the representation of time is a critical issue in the modeling of chemical production processes. Three fundamental deficiencies of the representation result if continuous processes are modeled by standard bucket-oriented lot-sizing models. These are the carry-over of setup states

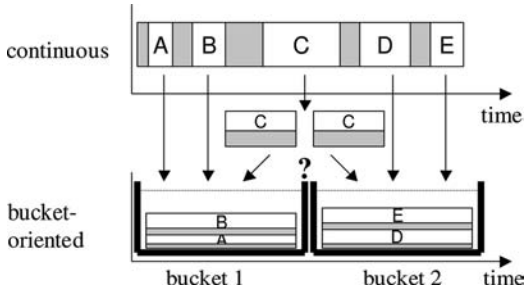

Fig. 11.3 Representation defect of a bucket-oriented model formulation (see p. 33 in [6]).

between adjacent periods, the lot size of production lots spanning over several periods and – if setup times are relatively long compared to bucket sizes – setup operations (setup times) spanning over several periods [6].

The first issue, a setup state carry-over between adjacent periods, is depicted in Figure 11.3. On a continuous time scale a sequence of five products (A to E) can be modeled straightforwardly. However, if the same is to be done in a bucket-oriented setting some difficulties arise. Production within a bucket is coupled with a corresponding setup operation which causes a lot of problems.

One option to deal with the situation is to neglect setup times and setup carry-over. This results in a situation, in which the available capacity is overestimated. On the other hand, if setup times are modeled, but setup carry-overs are not in the scope of the model formulation, the available capacity would be underestimated. Both results are not satisfactory; in the first situation plans will result which are too optimistic and not feasible, whereas in the second situation in which capacity estimation is too conservative, resources will be underutilized. Consequently, only a model formulation which takes setup times and setup carry-overs into account captures the characteristics of the real world adequately and provides a realistic capacity allocation.

Porkka et al. [7] conducted an experimental study to evaluate the effectiveness of setup carry-over. Based on their results it can be concluded that a considerable amount of capacity is released for production, if setup carry-over is included in the model formulation. The released capacity increases if setup times are relatively long. This effect is even enhanced, if the capacity is tightly restricted. Moreover, the released capacity is not only used for additional production, but roughly two-thirds of the theoretically freed capacity is reallocated to new setup operations. This leads to the presumption that plans created by a model formulation that allows for setup carry-over are fundamentally different from those that do not. This is another strong argument for incorporating the preservation of setup states into lot-sizing model formulations, whenever this is required by the underlying production process [6].

After having motivated the importance of coupling the production of adjacent periods via setup carry-over, now the lot size itself will be focused on. In the chemical industry, production quantities are often constrained such that a lower and/or upper bound is imposed on a continuous production run or that production has to be

in multiples of a predefined batch size [8]. This problem is usually referred to as campaign planning, with a campaign defined as the amount of a specific product type of one continuous production run, which can and generally will span over several planning periods.

Minimal bounds on the production quantity are most often process dependent. Typically, a minimal campaign length is required if for example a critical mass is necessary to initiate a chemical reaction. The same is valid for maximal bounds on the production quantity. The rationale here is that a cleaning operation may be required every time a certain amount has been produced. Finally, batch size restrictions often arise in the chemical industry, if for example the batch size is determined by a reactor load or, as discussed above, the processing time for a certain production step is independent of the amount of material processed. In these scenarios, when working with model formulations using a discrete time scale, it is important that the model formulation takes into account that lot sizes may comprise of production in several adjacent periods.

11.3 Case Study

11.3.1 Problem Description

To illustrate the capabilities of the SAP SCM solution on planning problems in the chemical industry, in this section we describe a small case study. The problem that is subject to our investigation is a *multilevel campaign planning* problem. The goal is to get a feasible plan ready for execution for the first two months and a more aggregate plan for a period of two years. The plan itself should have the property to yield a high demand satisfaction while at the same time it should keep inventories low. Moreover, certain resources have to be planned in campaign mode.

The underlying supply chain consists of several production sites that are closely connected. This is a typical situation in the chemical industry as many companies own several sites representing the different production facilities each focusing on special products or product groups. The different sites may be closely coupled, e.g., by pipelines, ensuring that production across several sites are possible. Figure 11.4 shows a simplified overview of the supply chain. Black nodes correspond to production sites. White nodes represent distribution centers or suppliers. Arcs represent possibilities of transportation between two sites.

In our case study, several products can be produced at different sites, thus there is a nontrivial sourcing problem to solve. The production process is organized roughly on three main levels. The first level produces active ingredients while the second production level corresponds to the finished goods level. Finally, the third level is responsible for packaging. The production process is divergent, i.e., the number of different products is increasing from level to level. On the active ingredients level, there are roughly one hundred products, on the finished good level the number

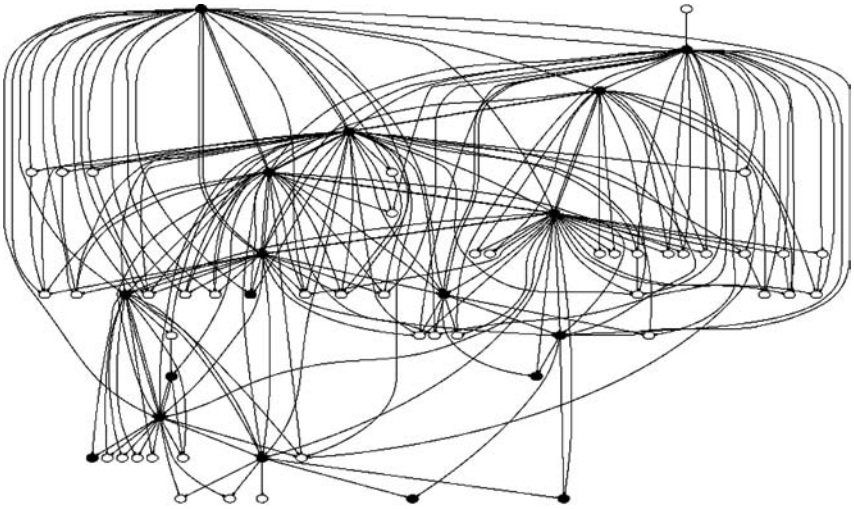

Fig. 11.4 Supply chain overview.

of products has increased to several hundred, while on the packaging level there are up to several thousand products. This high number occurs as the consequence of various packaging possibilities of the finished goods in, e.g., differently sized bottles and language specific documentations/descriptions.

The active ingredient production level is responsible for the main value creation. Therefore, it is important that the production resources on this production level are planned with a high quality in order to ensure good resource utilization. Changing from one product to another on the active ingredient level causes setup efforts of up to two weeks. Therefore, the production is organized in campaigns. That means that once a production resource is setup for a product, this product will be produced for a longer period in order to ensure efficient resource utilization. As a consequence, there is a trade off between production for future demands and inventory costs are incurred for storing these products until they are consumed (resp. delivered). Adjusting this trade off is crucial for achieving high quality production plans and low inventories. In addition to high setup efforts while changing from one product to another, there are also smaller setup efforts caused by the necessary cleaning

Table 11.1 Size of the planning problem.

Number of sites	~80
Number of products	~10000
Number of recipes	~10000
Number of production resources	~400
Number of production resources with campaign planning	~10
Number of products per campaign resource	~3–10
Number of buckets	24

operations after a certain number of batches have been produced. The machines on the finished goods and packaging level run smaller lots than those on the active ingredients level. There, each product is usually produced for one to two weeks. The size of the planning problem is summarized in Table 11.1.

11.3.2 APS Requirements

In our case study, the chemical company has performed their planning task prior to using APS by a plain MRP run. While MRP is an easy to use planning procedure, it has several drawbacks that can be overcome with the introduction of an APS system like SAP APO, which is part of the SAP SCM solution. One of the most important drawbacks is that MRP is an infinite capacity planning procedure, i.e., all resource or personnel capacities are ignored. In practice, this results in an MRP run requiring a postprocessing in order to yield production plans that can be executed. This postprocessing can be complex. For example, changing planned production on one level either in the schedule or with respect to its production quantity may require adaptations of the plan on the previous and subsequent levels. In order to overcome these problems, the new planning approach should create plans from a global view on the complete supply chain, thus reflecting a synchronous product flow over all levels of the value chain while simultaneously respecting capacities.

In addition, the new planning process should reflect the campaign-type production on the active ingredient level. Campaign production can be viewed as a special kind of lot-sizing problem. For controlling the trade off between campaign length/quantity and increased inventory costs should be used. This makes sense since the campaign production level is responsible for the main value creation in the supply chain and furthermore it helps to decide which products are premanufactured in which sequence based on their amount of capital binding. A proper planning of the campaigns will furthermore ensure high resource utilization.

11.4 Modeling the Case Study Scenario in SAP SCM

11.4.1 Modules of SAP SCM

The case study scenario is modeled using SAP's SAP SCM software package, the supply chain solution within the SAP Business Suite. The SAP SCM solution map in Figure 11.5 shows the complete scope of functionalities offered in the 2005 release.

The capabilities offered within SAP SCM extend far beyond the scope of this chapter. The key functionalities we will describe in the following are highlighted in Figure 11.6, which is based on the generic supply chain planning matrix (Figure 11.2) introduced in Section 11.2.1. They are part of the SAP Advanced Planner and

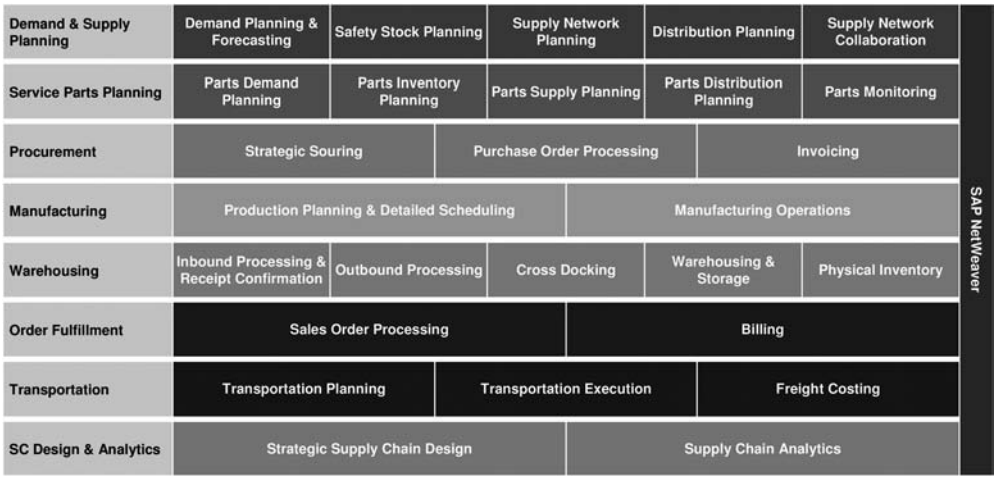

Fig. 11.5 SAP Supply Chain Management solution map, release 2005 (Copyright by SAP AG).

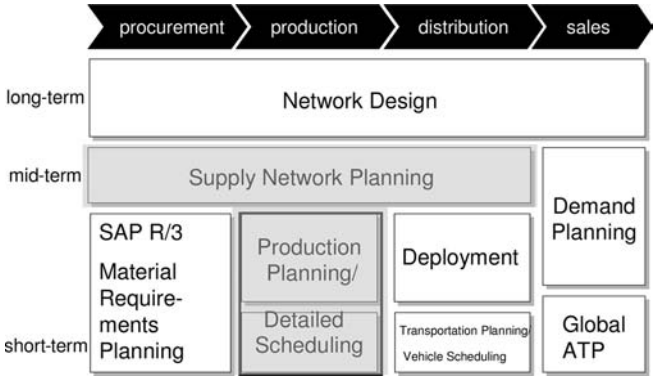

Fig. 11.6 Supply chain planning matrix using SAP SCM terminology.

Optimizer (SAP APO), which is the advanced planning component within the SAP SCM solution. For more information on SAP APO refer to [9, 10].

11.4.2 Reduction of Planning Complexity

The case study problem described in the preceding section is characterized by a high degree of planning complexity.

Solving a planning problem of this complexity in its entirety within one planning step is neither feasible from an algorithmic perspective nor sensible from a planning process point of view. A hierarchical decomposition of the complete planning problem into a master planning and a production planning and scheduling

part – as outlined in the section on planning problems in the chemical industry – addresses planning complexity as well as planning process design issues.

In the following, we describe how the planning problem is partitioned into a master planning and a production planning part and which business requirements are addressed on which planning level. We also show how both planning levels are integrated into a consistent, rolling planning process.

11.4.3 Multilocation Optimization in Supply Network Planning (SNP)

In SAP APO, the master planning process is implemented in the Supply Network Planning (SNP) module. SNP offers a multitude of functionalities, not all of which can be described in the limited scope of this chapter. More details on the SNP module can be found, among others, in [10].

From the different planning methods available within SNP, SNP optimization is selected because it offers the best fit to the customer requirements outlined above. The main reasons for this decision are the multisourcing characteristics of the supply network as well as the fact that the objective functions used by the SNP optimizer, profit maximization or cost minimization, correspond to the planning philosophy favored by the customer. In addition to SNP optimization with its cost-based approach, SNP offers several heuristic-based planning methods which follow a rule-based logic.

In this scenario, the planning horizon is two years. The two year period is divided into 24 monthly periods. SNP offers a high degree of flexibility in the definition of planning horizon and planning period profile. One option is to use a telescopic profile with shorter (e.g., daily) periods (also referred to as buckets) at the beginning of the planning horizon and longer periods (for instance weeks and then months) towards the end of the horizon. For the scenario at hand, this option was not chosen as the average duration of production campaigns is rather long and a shorter period length does not significantly improve the usability of the planning result. Furthermore, an added benefit of the monthly period profile versus a more precise period profile is a reduction of the size of the resulting optimization problem.

The first two months of the planning horizon are fixed for SNP planning. In this period, all planning takes place in Production Planning/Detailed Scheduling (PP/DS) module. Month three is shared by SNP and PP/DS. More details on the rationale of overlapping planning horizons for master planning and production planning are given in the section on the integration of SNP and PP/DS.

The SNP model contains all relevant locations, i.e., production plants and distribution centers, in the supply network. The cross-locational sourcing aspect of the planning scenario is handled within the master planning process. SNP determines which of the plants produces which quantities of which products in which time periods. On a rough level, SNP also determines which production alternative is used at a specific plant, for instance with regard to ingredients and general process characteristics.

To reduce the complexity of the master planning model, not all products are considered in the SNP optimization run. The selection is made by flagging specific products as not relevant for SNP planning. SNP planning takes into account:

- all products produced in a location,
- all products for which a stock transfer between locations is possible,
- externally procured active ingredients,
- goods for resale,
- selected forming auxiliaries.

Not SNP relevant are:

- most raw materials,
- most forming auxiliaries,
- packaging materials.

A similar logic is used for resources. Only bottleneck resources are selected for SNP planning.

The concentration on key products and bottleneck resources also results in a significant simplification of the recipes[1)] used in SNP, which are derived from the more detailed recipes used in PP/DS and the attached enterprise resource planning (ERP) system. Furthermore, compared to the recipes used on the PP/DS level, not all setup and cleaning operations are considered in SNP recipes. Small setup operations are normally neglected while key setup activities, which are relevant for campaign planning on bottleneck resources due to their long duration or high costs, are considered. To account for the resource capacity consumed by small setup and cleaning operations, a loss factor is applied to calculate the resource capacity for SNP planning.

One of the main aspects of the SNP planning process is the cost-based plan determination. The following cost types are used to build a cost model which represents the business scenario of value-based planning:

- penalties for not meeting customer demand/forecast for a cost minimization scenario, or alternatively, location-product specific sale prices for a profit maximization scenario (location-product specific);[2)]
- penalties for late satisfaction of customer demand/forecast (location-product specific);
- penalties for not meeting safety stock/safety days' supply requirements (location-product specific, linear or piece-wise linear);
- storage cost (location-product specific);
- penalty for exceeding maximum stock level/maximum coverage (location-product specific, linear or piece-wise linear);
- external procurement cost (linear or piece-wise linear, location-product specific);
- handling in/out cost (location-product specific);

1) In APO, a recipe is commonly referred to as PPM (production process models) or PDS (production data structure).

2) Location-product specific means that the granularity of the data is dependent on the product and the location.

- transportation cost (transportation lane, product and means of transport specific, linear or piece-wise linear);
- variable production cost (production process specific, linear or piece-wise linear);
- fixed production cost/setup cost (production process specific);
- resource utilization cost (resource specific);
- costs for additional resource utilization (e.g., use of additional shifts, resource specific);
- cost for falling below minimum resource utilization.

The definition of the cost model is of crucial importance for controlling the behavior of the SNP optimizer. One of the central questions is whether to maximize service level, which usually means using high penalties for non and late delivery, or to maximize profits, which requires the use of realistic sale prices. In the case study scenario, the nondelivery cost levels reflect real sale prices sufficiently close to enable a profit maximization logic.

Another important feature of the case study scenario and the resulting cost model is inventory control. High seasonality effects and long campaign durations necessitate considerable build-up of stocks. To avoid an unbalanced build-up of stock, soft constraints for safety stock and maximum stock levels are used. To achieve an even better inventory leveling across products and locations, piece-wise linear cost functions for falling below safety stock as well as for exceeding maximum stock levels are employed.

The other main factor in the selection of SNP optimization of the planning method is that all relevant constraints can be considered in planning, including:

- capacities for production, transportation, handling and storage resources,
- maximum location-product specific storage quantities,
- minimum, maximum and fixed production lot sizes,
- minimum, maximum and fixed transportation lot sizes,
- minimum production campaign lot sizes.

An optimization model which considers all these constraints – especially those which can only be modeled using binary or general integer variables – can be highly complex.

In the section on model solving we discuss how this complexity is addressed within the SNP optimization engine.

11.4.4 Production Planning and Detailed Scheduling (PP/DS)

The short-term planning process is dealt with in the Production Planning and Detailed Scheduling module within SAP APO. PP/DS considers a horizon of two to three months. In contrast to SNP planning, there is no cross-location planning in PP/DS.

PP/DS focuses on determining an optimal production sequence on key resources. In PP/DS, a more detailed modeling than on the SNP planning level is chosen. This does not mean that all products and resources within a real-world production process need to be considered for PP/DS planning as nonplanning relevant products and resources can be excluded from the integration process between the ERP system and the planning system.

The requirements for short-term planning, especially with regard to campaign-handling as well as the need to consider sequence-dependent setup and finite resource capacities on most resources lead to the selection of the PP/DS optimizer as the most suitable planning method. In addition to the optimizer, PP/DS offers numerous heuristics for automating production planning and detailed scheduling tasks.

The main focus of the optimization process in PP/DS is the determination of production campaigns. On the basis of the results determined in SNP optimization, a detailed schedule which considers additional resources and products is created. This schedule is fully executable and there is no need for manual planner intervention, even though manual replanning and adjustments are fully supported within the PP/DS module. An executable plan can only be ensured by considering additional complex constraints in PP/DS optimization. These additional constraints include:

- time-continuous planning,
- sequence-dependent setup and cleaning operations,
- large cleaning/setup operations between production campaigns,
- small cleaning/setup within production campaigns.

As the value-based planning part is handled within SNP, the PP/DS optimizer uses a different objective function than the SNP optimizer. The following goals can be weighted in the objective function, which is subject to minimization:

- sum of delays against due dates,
- maximum delay against due dates,
- setup time,
- setup cost,
- makespan,
- mode cost (i.e., costs associated with the selection of alternative resources),
- storage cost – only for campaign optimization.

The main objective of the PP/DS optimizer run in the scenario at hand is to minimize setup times and costs on campaign and noncampaign resources without incurring too much delay against the order due dates. For some resource groups, mode costs are also used to ensure that priority is given to the *best* (i.e., fastest, cheapest, etc.) resources.

PP/DS optimization also supports handling of single or multiproduct campaigns. The PP/DS optimizer can either respect existing campaigns – with or without being able to extend a campaign by additional orders – or completely replan campaigns.

More details on campaign optimization are given in the section on model solving in PP/DS.

11.4.5 Integration of Supply Network Planning and Production Planning and Detailed Scheduling

The integration between the two planning levels outlined above is of crucial importance for the consistency of the overall planning process. The main aspects of the integration (planning horizons, master data integration and transactional data integration) are outlined in the following.

11.4.5.1 Planning Horizons

In SAP APO, there are two different horizons which can be used to control the division of the complete planning horizon between the two planning levels SNP and PP/DS. On the SNP side, the SNP production horizon defines which time period (starting from the current date) is not planned in SNP. On the PP/DS side, the PP/DS horizon defines up to which point in time (starting from the current date) planning in PP/DS takes place. An overlap between the SNP planning horizon, which extends from the SNP production horizon to the end of the SNP planning horizon (two years in the case at hand), and the PP/DS planning horizon, which extends from the current date (assuming there is no fixing horizon in which only manual adjustments to the plan are allowed) to the PP/DS horizon, is allowed.

All horizons can be defined specifically for each location and each product. This allows to define different planning horizons for SNP and PP/DS for different types of products. In the scenario at hand, this possibility is used to differentiate between products on different levels in the supply chain. This allows to reflect lead time differences on the individual supply chain levels. In the case study scenario, the longest SNP production and PP/DS horizons are defined on active ingredient level, medium ones on finished goods level and the shortest horizons on the packaging levels. The horizons differ across product lines, but a typical choice is a SNP production horizon of two months and a PP/DS horizon of three months on the packaging level, resulting in one month of overlap. On the active ingredient level, both horizons can be longer by up to one month.

11.4.5.2 Master Data Integration

As discussed in the SNP section, certain products and certain resources can be flagged as not relevant for SNP planning. For SNP relevant products, no special integration settings are required.

For SNP relevant resources, however, integration issues arise due to different time profiles used by SNP, which is based on a periodic time profile with, in the scenario at hand, monthly time buckets, and PP/DS, which utilizes a continuous time representation of the planning horizon. This means that the time-continuous resource capacity profiles used in PP/DS, which are usually synchronized with the

capacity profile of the integrated ERP system, need to be translated to bucketized capacity profiles. This can be done either by calculating the bucket capacity on-the-fly by aggregating the time-continuous capacities or by defining a bucket capacity beforehand. Because of tighter integration, the first option is chosen. To account for inaccuracies caused by not considering certain (small) setups or cleaning activities, a resource-specific loss factor is applied when deriving the bucket capacity from the time-continuous capacity.

The other master data elements for which an integration process is required are recipes. Recipes used within the PP/DS module are tightly integrated with the recipes in the ERP system. For complexity reduction in master planning, as outlined in the SNP section above, simplified recipes are used in SNP. These are derived from the PP/DS recipes by a flexible, automated conversion process.

11.4.5.3 Transactional Data Integration

A direct result of the different levels of detail of recipes is that the planned production orders created by planning in SNP and PP/DS are incongruent. Based on the planning horizon concepts, SNP planned orders can be manually converted to PP/DS planned orders as soon as they move into the overlapping part of the SNP and PP/DS planning horizons. An automated conversion process takes place for SNP planned orders which move into the part of the planning horizon which exclusively belongs to PP/DS.

In order to correctly consider PP/DS planned orders in the overlapping period for SNP planning and vice versa, the following order integration logic is utilized.

In consideration of PP/DS planned orders for SNP optimization based planning:

- PP/DS planned orders are fixed for SNP planning.
- Available resource capacity for SNP planning is reduced by the capacity consumption of PP/DS planned orders.
- Material flow of PP/DS planned orders is considered for SNP planning.
- Setup state on production resources induced by PP/DS planned orders is considered for SNP optimization.

In consideration of SNP planned orders for PP/DS optimization based planning:

- SNP planned orders are not changed by PP/DS planning.
- Resource capacity consumption by SNP planned orders is not considered for PP/DS planning.
- Material flow of SNP planned orders is considered for PP/DS planning.

This logic guarantees that SNP planning on the mid- to long-term aggregated planning level is consistent with all constraints imposed by short-term, detailed planning in PP/DS. On the other hand, by only considering constraints related to material flow induced by SNP orders in PP/DS planning, no unnecessary restrictions are imposed on the more detailed planning level.

11.5 Solving the Case Study Scenario in SAP SCM

11.5.1 Solving the SNP Problem

11.5.1.1 Expressing Optimization Models in SAP SCM

As described in Section 11.4.3, the SNP optimizer is used in order to perform cost-based planning. The cost model is derived from real costs in order to anticipate the value aspect as much as possible.

The SNP optimizer is based on (mixed-integer) linear programming (MILP) techniques. For a general introduction into MILP we refer to [11]. An SAP APO user has no access to the mathematical MILP model. Instead, the modeling is done in notions of master data of example products, recipes, resources and transportation lanes. Each master data object corresponds to a set of constraints in the mathematical model used in the optimizer. For example, the definition of a location-product in combination with the bucket definition is translated into inventory balance constraints for describing the development of the stock level over time. Additional location-product properties have further influence on the mathematical model, e.g., whether there is a maximum stock-level for a product or whether it has a finite shelf-life. For further information on the master data expressiveness of SAP SCM we refer to [9].

The production is modeled via recipes representing a combination of bill-of-materials and required resource consumption. Lot-sizing constraints for production can be expressed in different ways. One class of lot-sizing constraints allows to directly manipulate the production quantities by defining a minimum lot-size or by requiring production in integer multiples of a certain batch size. A further way to control lot-sizing is to include setup costs and setup resource consumption for production. In this way, the lot-sizes are determined based on the optimal balance between costs resulting from the setup effort and inventory. While lot-for-lot production can be modeled without additional constraints, expressing the latter lot-size constraints requires that some variables in the mathematical model become integers. As a consequence, mixed-integer linear programming models are required for solving these lot-sizing problems.

11.5.1.2 Cross-Period Lot-Sizing with the SNP Optimizer

For controlling lot-sizing based on setup efforts and inventory costs, the SNP optimizer offers two standard models known as (*multilevel*) *capacitated lot-sizing* (ML) CLSP and (*multilevel*) *proportional lot-sizing* (ML) PLSP. For more details and related models we refer to [6]. Capacitated lot-sizing is expressed in SAP SCM via a production resource and recipes having fixed and variable resource consumption and optionally by maintaining a (piece-wise linear) cost function for expressing setup costs. In this way, the fixed resource consumption and the fixed cost are taken into account whenever the recipe is selected for production in a bucket. In particular, this happens also in cases in which a product is produced without interruption over

several buckets. Although the setup efforts in case of consecutive production for a product do not precisely reflect the reality, the error in the model can be ignored on the SNP level if resource consumption by setups is small compared to the bucket capacity. However, as described in Section 11.3.1, in our case the fixed resource consumption is big compared to the bucket capacity as it can consume up to one-half of it (e.g., two weeks setup in a one month bucket).

In order to properly model problems of this kind, the SNP optimizer allows also for setup carry-over across buckets. In this case, the setup state is conserved across the bucket-boundary such that in case of consecutive production of a product over one or more buckets, only a setup at the beginning and no additional setups are required. In the literature, there are several models proposed which fulfil these requirements. As already mentioned, the SNP optimizer supports the PLSP model which can be characterized by the property that at most one product changeover per bucket is allowed. Although this property seems to be quite restricting it is a good compromise between expressiveness and model complexity. In the case that there is more than one product change per bucket the CLSP model might be sufficient since in this case the fixed resource consumption by the involved recipes is usually smaller. To express proportional lot-sizing in SAP SCM the corresponding production resources have to be marked for cross-period lot-sizing.

Solving CLSP or PLSP optimization problems is known to be NP-hard [12, 13]. Nevertheless, in many cases it is possible to solve problem instances of practical interest with good solution quality in reasonable time. Having a good mathematical model is essential for good runtime performance. Roughly, a MILP model is said to be good if its linear relaxation is a tight approximation of the convex hull of feasible MILP solutions. There are several MILP models known for expressing PLSP problems [6]. A formulation of the single level PLSP that turned out to be efficient in computational experiments is given by Eqs. (11.1) to (11.8):

$$\text{Min} \sum_{j\in\mathcal{J}}\sum_{t\in\mathcal{T}} h_{jt}\cdot I_{jt} + \sum_{j\in\mathcal{J}}\sum_{t\in\mathcal{T}} sc_j\cdot Y_{jt} \tag{11.1}$$

$$I_{jt-1} + X_{jt} = d_{jt} + I_{jt} \qquad \forall j\in\mathcal{J}, t\in\mathcal{T} \tag{11.2}$$

$$\sum_{j\in\mathcal{J}} a_j\cdot X_{jt} + \sum_{j\in\mathcal{J}} st_j\cdot Y_{jt} \le c_t \qquad \forall\, t\in\mathcal{T} \tag{11.3}$$

$$X_{jt} \le \frac{c_t}{a_j}\cdot(Z_{jt} + Z_{jt-1}) \qquad \forall\, j\in\mathcal{J}, t\in\mathcal{T} \tag{11.4}$$

$$\sum_{j\in\mathcal{J}} Z_{jt} \le 1 \qquad \forall\, t\in\mathcal{T} \tag{11.5}$$

$$Y_{jt} \ge Z_{jt} - Z_{jt-1} \qquad \forall\, j\in\mathcal{J}, t\in\mathcal{T} \tag{11.6}$$

$$X_{jt}\ge 0,\ \ Y_{jt}\ge 0,\ \ I_{jt}\ge 0,\ \ I_{j0}=0 \qquad \forall\, j\in\mathcal{J}, t\in\mathcal{T} \tag{11.7}$$

$$Z_{jt}\in\{0;1\},\quad Z_{j0}=0 \qquad \forall\, j\in\mathcal{J}, t\in\mathcal{T} \tag{11.8}$$

The index set J represents the set of products. The set T represents the set of all buckets. The variable X_{jt} corresponds to the recipe describing the production

quantity of product j in bucket t. The variable I_{jt} models the inventory of product j at the end of period t. The variable Y_{jt} takes only the value one if there is a setup in period t for recipe j and is zero otherwise. The variable Z_{jt} expresses whether the resource is setup for recipe j at the end of bucket t (and consequently at the beginning of the bucket $t+1$). The constant c_t expresses the bucket capacity of the resource in bucket t. The value st_j represents the fixed resource consumption and a_j represents the variable resource consumption of recipe j. The demand for the product produced by recipe j is modeled by the constant d_{jt}. The objective is to minimize the sum of storage costs and setup costs (11.1). Constraints (11.2) are regular inventory balance constraints and (11.3) are capacity constraints, stating that production and setup operations never exceed available capacity. Constraints (11.4) link production variables X_{jt} with setup state variables Z_{jt}. Production of j in t is allowed, if the product j is set up either at the beginning of t ($Z_{jt-1} = 1$) or at the end of t ($Z_{jt} = 1$). Inequalities (11.5) take care that there is at most one setup state at the end of each period and (11.6) force the setup operation variable Y_{jt} to a value of 1, if the setup state is changed within a period. Constraints (11.6) together with (11.5) guarantee that at most one setup operation is performed in each period and therefore the sequence of products is determined by this model formulation. Finally, (11.7) and (11.8) impose nonnegativity and binary conditions, respectively.

In case of cross-period lot-sizing on a resource, we call the cumulative quantity of all consecutive bucket production quantities for a product the campaign quantity. Analogously to the bucket-specific lot-sizing constraints, the SNP optimizer is able to handle constraints for a minimal campaign quantity or impose that the campaign quantity is an integer multiple of a certain batch size. As it is not known at modeling time when a campaign starts respectively when it ends, there is no linear model known which expresses this quantity directly as a sum. In [6] several models are investigated. In order to express the campaign quantity, the constraints (11.2) to (11.8) have to be extended by further variables. These variables K_{jt} will be called campaign variables to distinguish them from production variables X_{jt}. The campaign variables K_{jt} will be defined to represent the cumulated lot size (or campaign quantity) of product j in period t of the last (or current) campaign. As long as j is produced, current production of j is added to variable K_{jt}, whereas if production has ceased, variables K_{jt} will remain constant until they are reset to zero at the beginning of the next campaign of product j. The following model extension (11.9) to (11.12) defines the new set of variables K_{jt}:

$$K_{jt} \leq K_{jt-1} + X_{jt} \qquad \forall j \in \mathcal{J}, t \in \mathcal{T} \tag{11.9}$$

$$K_{jt} \geq K_{jt-1} + X_{jt} - maxlot_j \cdot Y_{jt+1} \qquad \forall j \in \mathcal{J}, t \in \mathcal{T}\backslash\{T\} \tag{11.10}$$

$$K_{jt} \leq maxlot_j \cdot (1 - Y_{jt+1}) \qquad \forall j \in \mathcal{J}, t \in \mathcal{T}\backslash\{T\} \tag{11.11}$$

$$K_{j0} \leq minlot_j \cdot (1 - Y_{j1}) \qquad \forall j \in \mathcal{J} \tag{11.12}$$

Unless a new campaign of product j starts in the next period ($Y_{jt+1} = 1$) current production (X_{jt}) is added to the campaign variables K_{jt} [(9) and (10)]. These two constraints provide upper and lower bounds on the campaign variables K_{jt}. In this

case, (11.11) and (11.12) take no effect. On the other hand, if there is a start of a new campaign, (11.10) is lifted (due to the last term on the right-hand side) and (11.11) which dominates (11.9) in this case forces the campaign variables to zero.

At the beginning of the planning horizon variables K_{jt} need to be initialized. This can be done by constraints (11.12).

With variables K_{jt} properly defined, additional constraints can be identified to cope with the different restrictions posed to the decision problem by the production environment. As outlined in Section 11.3.1, minimal lot sizes, maximal lot sizes and lot sizes which are multiples of an integer batch size are relevant restrictions to be considered here.

$$K_{jt-1} + X_{jt} \geq minlot_j \cdot \sum_{\substack{k \in \mathcal{J} \\ k \neq j}} Y_{kt} \quad \forall\, j \in \mathcal{J}, t \in \mathcal{T} \tag{11.13}$$

$$K_{jt-1} + X_{jt} \leq maxlot_j \quad \forall\, j \in \mathcal{J}, t \in \mathcal{T}\backslash\{1\} \tag{11.14}$$

$$K_{jt-1} + X_{jt} = bs_j \cdot R_{jt} + S_{jt} \quad \forall\, j \in \mathcal{J}, t \in \mathcal{T}\backslash\{1\} \tag{11.15}$$

$$S_{jt} \leq bs_j \cdot \left(1 - \sum_{\substack{k \in \mathcal{J} \\ k \neq j}} Y_{kt}\right) \quad \forall\, j \in \mathcal{J}, t \in \mathcal{T}\backslash\{1\} \tag{11.16}$$

$$S_{jt} \geq 0 \quad \forall\, j \in \mathcal{J} \quad t \in \mathcal{T}\backslash\{T\} \tag{11.17}$$

$$R_{jt} \geq 0 \text{ and integer} \quad \forall\, j \in \mathcal{J}, t \in \mathcal{T}\backslash\{T\} \tag{11.18}$$

To obey minimal lot sizes, (11.13) are necessary. They state that as soon as any new campaign of product k starts, the minimal lot size of product j must have been produced. These constraints also hold true if product j is not produced right before k because the reinitialization of campaign variables is just prior to its own next production start (11.11).

Maximal lot sizes are obeyed due to constraints (11.14). Any campaign may not exceed its maximal production quantity $maxlot_j$.

Dealing with batch size restrictions is slightly more elaborate. Two new variables need to be defined, one of them being integer. In each period t the current campaign quantity ($K_{jt-1} + X_{jt}$) is split into two variables. One of them (R_{jt}) counts the number of full batches already produced in the current campaign and the second one (S_{jt}) takes the rest. This is done by constraints (11.15) and (11.16). The latter one takes care that no more than a full batch is contained in slack variables S_{jt} nor any rest remains if production of another campaign starts. Finally, (11.17) and (11.18) state the domain of the variables R_{jt} and S_{jt}.

11.5.1.3 Consequences from Integration with PP/DS

In Section 11.4.5, we have already seen that the SNP optimizer is restricted to plan in the SNP horizon. If campaign production problems are solved in the context of hierarchical planning it is important that the SNP optimizer respects setup operations of PP/DS orders in the overlap of the SNP and PP/DS horizon. This

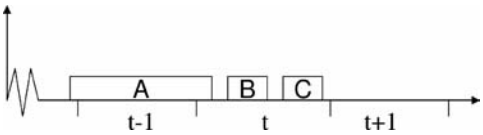

Fig. 11.7 PLSP inconsistent setup situation.

will allow on the one hand that campaigns in this time interval can be increased without planning a new setup in SNP in case that there are new demands and available capacity. On the other hand, the SNP optimizer has a correct setup state if new campaigns are planned in the SNP horizon but after the PP/DS horizon as the last setup state of the PP/DS horizon is taken over. Again no setup has to be planned by the SNP optimizer

Unfortunately, the continuous time representation of PP/DS can cause PP/DS campaigns not to satisfy the restrictions of the PLSP model that there is at most one setup operation per bucket. Another reason might be that due to manual modifications of the PP/DS plan the PLSP constraints are violated. Therefore, the PLSP model in the SNP optimizer has been extended in order to work with setup situations that are inconsistent with the PLSP. Given an initial setup state situation, the SNP optimizer tries to construct a model that keeps the cross bucket setup state conservation as good as possible. For example, in buckets in which there are already more than two different products no additional products may be planned. It is only allowed to extend the campaigns.

In the example in Figure 11.7, either product A or B can be extended in bucket t. Product C can be extended in bucket t and its setup state is conserved to bucket $t+1$. Other approaches for handling the problem, e.g., by splitting bucket t into subbuckets by introducing artificial bucket boundaries can be critical in multilevel cases since they might change the plan compared to the given bucket profile with respect to the makespan. The example in Figure 11.7 shows only the simplest case in which an initial setup situation does not satisfy the PLSP model. The SNP optimizer has several further heuristics for dealing with all kinds of inconsistencies in a reasonable way.

In our case study, the problem instances for SNP optimization result from master data of approximately 10 000 location-products, 10 000 recipes, 400 production resources and 10 resources relevant for campaign production with 3–10 different products per campaign resource. This translates into a MIP model with about 700 000 continuous variables, 1000 binary variables and 300 000 linear constraints.

Due to the campaign structure, the existing decomposition techniques in the SNP optimizer like time decomposition and product decomposition are not applicable. For problems with this structure it is possible to use the resource decomposition in case a good sequence of planning of the campaign resources can be derived. However, in our case, problem instances could be solved without decomposition on a Pentium IV with 2 GHz in one hour to a solution quality of which the objective value deviates at most one percent from the optimal objective function value.

11.5.2 Solving the PP/DS Problem

As described in Section 11.4.4, the PP/DS optimizer is used in this case study to generate a feasible production plan for the immediate future. An aggregate plan, not taking into account each single step in the production sequence has been created by the SNP optimizer. As time goes by, SNP planned orders migrate into the PP/DS horizon and become subject to planning within this module. This plan has to cover a higher level of detail than the SNP plan. Therefore, recipes incorporating a higher level of detail are used within PP/DS.

The main objective of PP/DS optimization is the building of campaigns taking into account inventories and setup costs. Additional objectives in this scenario are the minimization of delays with respect to the due dates given by the SNP optimization run as well as the selection of suitable resources. The PP/DS optimizer is based on a genetic algorithm. As setup carry-overs have been modeled already in the upper planning level, PP/DS receives input that will yield a consistent plan.

The concept of campaign planning within the PP/DS module is based on resource decomposition. The first set of resources for which a sequence is determined is the set of resources on which campaigns have to be built. These are the bottleneck resources of the production system. After a good sequence has been generated for these resources, in subsequent planning steps the sequence of activities on resources preceding the bottleneck stage and the sequence of activities on resources following the bottleneck stage are planned.

Time decomposition, which is another option in PP/DS optimization to decompose the overall planning problem, has not been used in this scenario.

Important features of the PP/DS optimizer in this scenario are its ability to handle campaigns. Campaigns consist of products using the same setup group. Changeovers from one campaign to another require a major setup operation, while there are minor setup operations allowed between products belonging to the same campaign. For each location and for each setup group, a campaign profile is defined. The campaign profile includes planning relevant information like the maximum number of orders in a campaign or whether setup or clean-out orders should be generated at the beginning or at the end of a campaign. Furthermore, administrative information, e.g., who is the responsible production planner is stored here.

As planning takes place in a rolling horizon environment, the PP/DS optimizer needs information whether campaigns that already exist in the planning horizon should be retained or whether they are allowed to be extended or to be dissolved. Retaining campaigns means that the PP/DS optimizer may change the sequence of campaigns, but it is not allowed to remove orders from campaigns, to add new orders to campaigns, or to change the resource the campaign is processed on. In contrast, extending campaigns allows to add orders to a campaign, whereas dissolving of campaigns gives the optimizer all degrees of freedom. Campaigns that have already been started will generally receive a different status than the others within the planning horizon. For example, the first campaign should be retained or (at most) be extended, while the other campaigns will be dissolved.

In standard PP/DS optimization, the PP/DS optimizer does not create any orders, but it is used to schedule the set of existing orders. However, in campaign planning, the creation of campaigns usually has the consequence that additional orders result (setup orders at the beginning of campaigns or cleaning orders at the end of campaigns). Therefore, when using campaign optimization it is possible to generate these orders automatically at the end of an optimization run.

11.6 Conclusion

In this chapter, the SAP SCM solution was applied to a typical multilevel campaign planning problem in the chemical industry. The focus lies on modeling and solving the problem using an optimization based hierarchical planning approach, which is implemented within SAP APO, the advanced planning component of SAP SCM. The planning problem, which is characterized by a high degree of complexity, is split into a master planning part and a production planning and scheduling part. On the master planning level, the optimization engine of the Supply Network Planning (SNP) module is used to solve the multilevel proportional lot-sizing problem with mixed-integer linear programming techniques. The production planning and detailed scheduling (PP/DS) part is solved using the genetic-algorithm based PP/DS optimizer engine, which contains special campaign optimization capabilities. Consistency of the plans generated on the different planning levels is ensured by a sophisticated integration concept.

References

1 Fleischmann, B., Meyr, H. and Wagner, M. (2005) Advanced planning, in *Supply Chain Management and Advanced Planning. Concepts, Models, Software and Case Studies*, 3rd edn (eds H. Stadtler and C. Kilger), Springer, Berlin, pp. 81–106.

2 Meyr, H., Wagner, M. and Rohde, J. (2005) Structure of advanced planning systems, in *Supply Chain Management and Advanced Planning. Concepts, Models, Software and Case Studies*, 3rd edn, (eds H. Stadtler and C. Kilger), Springer, Berlin, pp. 109–115.

3 Meyr, H. and Stadtler, H. (2005) Types of supply chains, in *Supply Chain Management and Advanced Planning. Concepts, Models, Software and Case Studies*, 3rd edn, (eds H. Stadtler and C. Kilger) Springer, Berlin, pp. 65–80.

4 Crama, Y., Pochet, Y. and Wera, Y. (2001) *A Discussion of Production Planning Approaches in the Process Industry*, CORE Discussion Paper No. 42, Louvain-la-Neuve, Belgium.

5 Blömer, F. (1999) *Produktionsplanung und steuerung in der chemischen Industrie*. Ressourceneinsatzplanung von Batchprozessen auf Mehrzweckanlagen. Deutscher UniversitätsVerlag, Wiesbaden.

6 Suerie, C. (2005) *Time Continuity in Discrete Time Models. New Approaches for Production Planning in Process Industries*, Springer, Berlin.

7 Porkka, P., Vepsäläinen, A.P.J. and Kuula, M. (2003) Multiperiod production planning carrying over setup time. *Int. J. Product. Res.*, **41**, 1133–1148.

8 Kallrath, J. (2002) Planning and scheduling in the process industry. *OR Spectrum*, **24**, 219–250.

9 Bartsch, H. and Bickenbach, P. (2001) *Supply Chain Management mit SAP APO*, Galileo, Bonn.

10 Dickersbach, J.T. (2005) *Supply Chain Management with APO*, 2nd edn, Springer, Berlin.

11 Nemhauser, G.L. and Wolsey, L.A. (1988) *Integer Programming and Combinatorial Optimization*, Wiley, New York.

12 Florian, M., Lenstra J. and Kan, A.R. (1980) Deterministic production planning: algorithms and complexity. *Manage. Sci.*, **26**, 669–679.

13 Kimms A. and Drexl, A. (1998) Some insights into proportional lot sizing and scheduling. *J. Oper. Res. Soc.*, **49**, 1196–1205.

12
Integration of Scheduling with ERP Systems

Winfried Jaenicke and Robert Seeger

Typographic Conventions and Terminology

This article uses the terminology of the market leader in business software (SAP AG) and denotes those terms in *italic* (e.g., see [1] for descriptions).

The terminology defined by NAMUR (compare [2]) has not been successful, for those readers that are familiar with this terminology we note that a *master recipe* is the same as a NAMUR *base recipe* and a *process order* is the same as a NAMUR *control recipe*.

The authors use quotation marks to denote citations of typical terms and sentences that are commonly used by business people but do not always have the exact meaning in the world of science, e.g., "optimization" does not always mean the same for business people as for mathematicians.

12.1
Introduction

Production planning is one of the activities that needs to be carried out in preparation of the industrial production of chemical products.

The task of production planning is to determine if and how a quantity of products can be produced at a required date or in a required time range. The aim of production planning is to ensure availability of the required product quantities *in time* and *cost-efficiently*.

Requirements like:

- minimization of stock levels for raw materials, intermediates and finished products,
- maximization of capacity utilization,
- reduction of setup costs

are derived from this aim.

Fig. 12.1. Hierarchical approach: decomposition into strategic planning, scheduling and execution.

Production planning has to determine on the detailed level which production step (*operation*) has to be carried out at which time and on which resource. For this purpose a resource allocation problem has to be solved (this is denoted as a *scheduling problem* in mathematical theory). It is natural to desire this allocation to be optimal in a certain sense (minimal number of setups and/or violations of requirements dates, etc.).

Production planning is not an isolated process, rather it is embedded into a control loop that includes materials management, maintenance, demand planning, strategic planning, logistics, sales forecasting, etc. Most of these tasks are supported by ERP (enterprise resource planning) systems by means of data management and planning functions. These systems like SAP ERP (R/3), Microsoft Navision, JDE Enterprise One (formerly JD Edwards World Software resp. One World) are based on relational databases. They are well-suited for data storage and batch processing of large amounts of data records. They are less suited for complicated scheduling operations. They are transaction-based and work on client/server architectures.

Despite the fact that production planning is embedded into a control loop, basic attitudes differ on how to solve production planning tasks.

The *hierarchical approach* was developed in the early days of computer-aided planning and can be visualized by a pyramid (Fig. 12.1). It was the right solution to the challenges of the restricted capability of computer systems at that time that required a decomposition of problems to reduce their size.

The hierarchical approach assumes that the ERP system provides rough cut planning functions that organize material availability and determine rough capacity requirements. The results are then given to the "subordinate" scheduling system for execution. In the end this means that material requirements planning is separated from resource allocation scheduling. It is further assumed that scheduling has to be done for a shorter time interval. Thus, only, small-scale scheduling problems have to be solved.

Currently two new approaches are competing.

In the *independence approach* the planning functions are stripped from the old-fashioned ERP system and a modern complete APS system is added as a separate server system with an independent persistent data model and integrated by an interface (Fig. 12.2). Some users of the ERP system also use this separate APS system. This approach is supported, e.g., by the software products SAP APO resp.

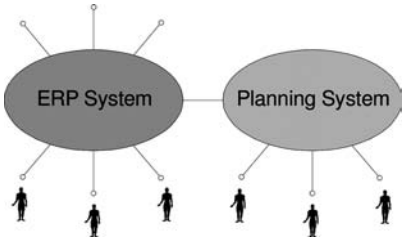

Fig. 12.2 Independence approach: decomposition into separate planning systems.

SAP SCM, I2 Production Scheduler, Wassermann waySCS, AspenONE resp. Aspen MIMI and Axxom OR*ion*-PI.

In the *embedding approach* the ERP system is enhanced with subordinate planning systems that are integrated into the user interface of the ERP system and create a local temporary data storage (LiveCache). All data is still held persistently only in the ERP system (Fig. 12.3). This allows for the LiveCache to use a projection of the ERP data model that is more suitable for APS purposes. This approach is used, e.g., by the software product OR Soft SCHEDULE++.

Both new approaches have some advantages and disadvantages.

The independence approach allows for a good encapsulation of the APS issue but requires major changes in business processes. It requires the creation and maintenance of an independent data model with its own data structure and the definition of new business processes. Introduction of the APS system must be done in a "big bang." Integration is not fully guaranteed, its quality depends on the throughput and the error tolerance of the integration interface.

The embedding approach may require an improvement of modeling in the ERP system (i.e., to maintain additional detailed information for APS purposes) but it can utilize all established business processes, data models and infrastructures. Introduction of the subordinate planning system can be done step-by-step with minimum impact on established business processes. Integration is fully guaranteed

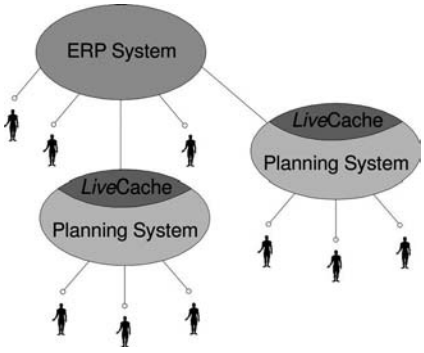

Fig. 12.3 Subordinate planning systems.

if only certified access methods of the ERP system are used to write information back into the ERP system.

The authors follow the embedding approach in this chapter.

12.2 Production Scenarios

12.2.1 Multipurpose Batch Chemical Plants

Multipurpose batch plants are considered to be the most complicated type of chemical plant with regard to production planning. On the other hand, good planning of such plants can lead to large benefits by better nesting of the batches.

As a typical pattern of a multipurpose batch plant, the authors consider a plant with multiple floors that produces various products (e.g., fine chemicals or dyes) in batches and/or semicontinuously. Raw materials are entered into premixing devices and placed together in a reactor where the chemical reaction takes place. The resulting product is separated from the mother liquor by a filter press and then packaged in various types of packaging. The mother liquor is stored and eventually recycled. The same product may even take different paths through the plant by use of alternate devices or production lines.

This multistage production process can be characterized by the description of the material flows and the resource utilization for planning purposes (Fig. 12.4).

The left-hand side of Fig. 12.4 shows the plant structure by way of *resources* and their possible connections (*resource network*) The right-hand side of Fig. 12.4 shows

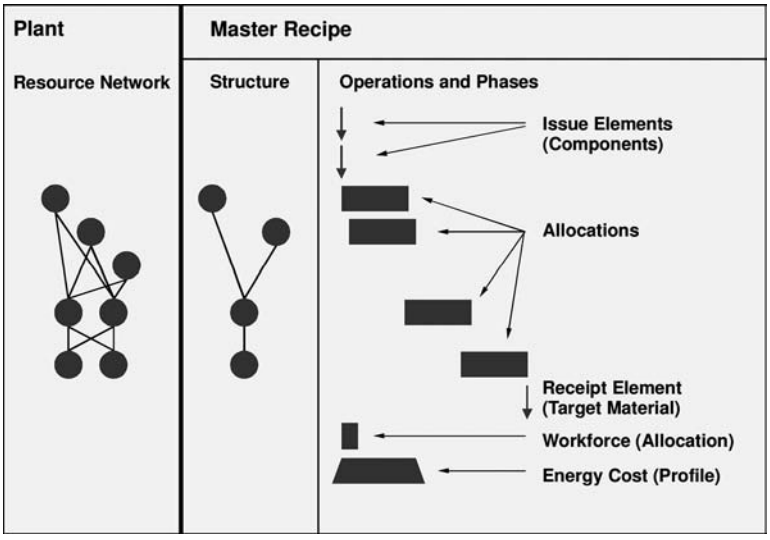

Fig. 12.4 Multipurpose batch chemical plant: plant structure and master recipes.

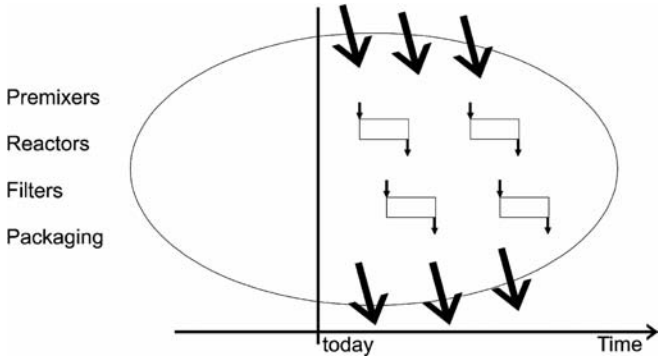

Fig. 12.5 Multipurpose batch chemical plant: resource allocation and material flows over time horizon.

the main part of a *master recipe*. One can see which *resources* are needed at which time interval for the production of a batch (allocations), the vertical arrows represent the *materials* that are either consumed (*issue elements*) or produced (*receipt elements*) during production.

A *process order* is a request asking production to produce a specific quantity of a material on a specific date. Creation of a process order uses a *master recipe* that is valid for the specific material and quantity as a template to determine the specific timing and component quantities (material flow).

Figure 12.5 depicts an abstraction of the production schedule for the sample plant, expressed by process order steps allocated to resources over the time scale. The small arrows represent the receipt and issue elements that are directly related to the process order. The large arrows represent material flows that cross the balance area (i.e., they have either a source or a target outside of the organizational unit considered, e.g., purchase orders and sales orders).

12.2.2 Semicontinuous Production

Semicontinuous production processes can be described in a similar way as batch production processes by adding a duration information to a material flow. This makes it possible to handle the overlapping of independent process orders over multiple stages in campaign planning and semicontinuous production planning.

In some industry branches semicontinuous production occurs together with both multistage joint production (often with multiple side products per stage that have to be processed further just like the main product) and the opportunity to vary the throughputs of the production devices.

The use of this extended planning model will only be problematic if extra reference points, e.g., initial tank storage levels, have to be considered. This may lead to "overdetermination" of the model (i.e., conflicting level values for a given point in time) and it may be necessary to solve a "data reconciliation problem."

Due to this overdetermination problem there exists a strong prejudice in the chemical industry that production planning for batch production is completely different from production planning for continuous production. This leads to the ineffective separation between "process engineering" and "business" data processing in continuous and semicontinuous production.

12.3 The Planning Problem and a Solution Approach

12.3.1 Planning of a Multipurpose Batch Plant

For the planning of a multipurpose plant one has to map the structure of a *master recipe* to the detailed device structure of the plant with regard to a given time in the future. This process can be done automatically, e.g., with the SAP ERP system where it is called "convert." The result is a *process order* as a concretization of a *master recipe*. A *process order* tells the production operator at which time and on which device a given batch production step has to be executed. This simple *conversion* can result in a situation where a selected device is already allocated to a different process order at the given time, thus the production plan may not be feasible.

In case of allocation conflicts one has to choose a different time, a different structural variant (*production version*) or even both. The process of creating feasible production plans is denoted as scheduling. To support the solution of scheduling problems, a scheduling system is used.

Scheduling is complicated by the need to create multiple process orders that compete for the free devices. It gets even more complicated if more degrees of freedom, e.g., planning buffers (variable timing distances between operations) or variable throughputs, are introduced into the *master recipe*, and if more constraints (availability of workforce, limited waste disposal capacity, limited intermediate storage, etc.) have to be considered in the planning model. In this case "scheduling" becomes "advanced planning and scheduling."

Advanced planning and scheduling represents a combinatorial optimization problem of the highest degree of difficulty. For this kind of problem it is impossible to apply exact solution algorithms even for medium-sized problems. This is why suboptimal feasible solutions must be found, in many cases by the humans who are responsible for planning. In doing so, one combines the human capability of pattern recognition to find gaps with the strength of the computer to execute complex calculations, to check constraints and show the consequences. For the planner this looks like a puzzle that has to be solved by the creation of nested structures. To support this puzzle-solving aspect, people use pen and paper, manual planning boards, electronic planning boards or scheduling systems. Advanced planning boards utilize backtracking algorithms to generate structure variants that may fit into a gap that has been spotted by the planner and then check the satisfaction of constraints automatically (Fig. 12.6).

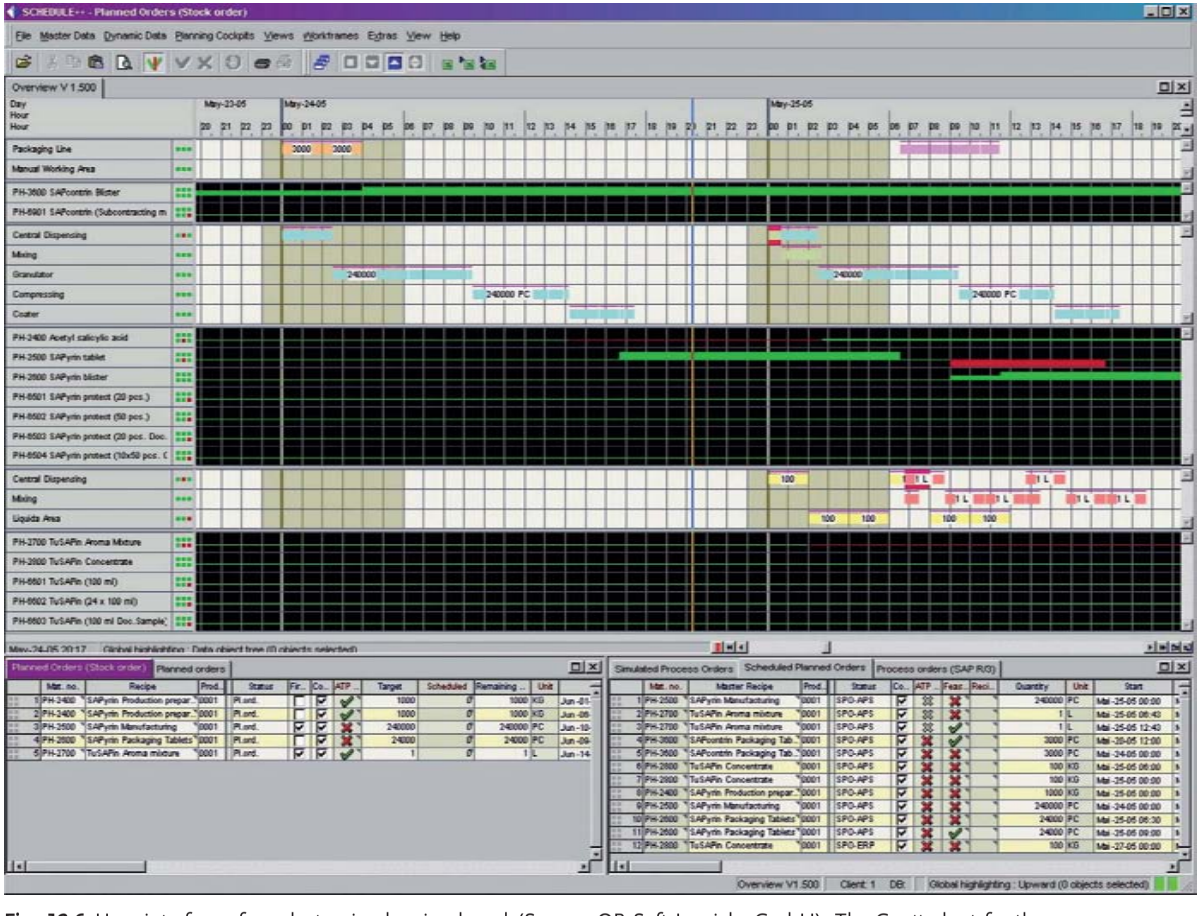

Fig. 12.6 User interface of an electronic planning board (Source: OR Soft Jaenicke GmbH). The Gantt chart for the resources (white background) shows some gaps in the allocation where another process order may fit in, the histograms (black background) show the resulting stock levels for the materials involved.

The scheduling task can be simplified by a reduction of the model (e.g., using single-step process orders that only consider the bottleneck resources). As usual, one has to compromise between effort and precision.

12.3.2 All Do Their Duty

A planning model describes the future with regard to material flows and resource allocation, as it is visualized in Fig. 12.4 and described in more detail in Section 12.4. There is no formal change if one shifts "today" to the right on the time axis. This way the same planning model can be used to describe the past as well as the future. All employees which participate in the production process and related business processes (planning, production, purchasing, sales) have to update the information about material flows that cross the balance area (i.e., they have either a source or a target outside of the organizational unit considered, e.g., purchase orders and sales orders) and process orders inside the balance area (rescheduling, confirmation reporting). If all do their duty, i.e., update the information they are responsible for, then it is easy to recognize which consequences to the total situation are caused by the decisions of a single participant. This makes it necessary to integrate planning into the overall business processes. This implies that there should be no separate isolated planning system.

12.4 Data Model

An APS planning model contains master data and dynamic data.

12.4.1 Master Data

The master data consists of

- materials (products),
- resources (devices, vessels, tanks),
- master recipes (routings, bills of materials).

All data objects contain an identification key and structured information. Materials and resources can be defined by relatively simple property tables. Master recipes require more complex structures to describe which resources have to be used at which time interval by which operation and with which operation parameters, and which materials are needed or produced at which point in time and in which quantity.

12.4.2 Dynamic Data

Dynamic data contains:

- Material flows that cross the balance area, i.e., they have either a source or a target outside of the organizational unit considered (*sales orders*, characterized by customer, quantity, material, requirements date; *purchase orders*, characterized by vendor, quantity, material, availability date).
- Material transformations in connection with resource allocations (*process orders*, concretizations of master recipes with specific quantities, timing and allocation information).
- Initial situation with regard to material stocks (*plant stock*, material stock at a given time, calculated from the last physical inventory, confirmed goods issues and goods receipts).

12.4.3 Remark on "Definite" and "Fuzzy" Data

Material flow and resource allocation can be defined by time, duration, type and quantity in the planning model. They describe a definite (by best current knowledge) change of the planning model in the future. In addition to this there are a number of "fuzzy" information data that have to be included in the planning model but are only weak assumptions about the future planning situation. These include, e.g., *planned orders* and *planned independent requirements*.

Planned orders are place holders for process orders that have yet to be checked for planning feasibility by detailed scheduling. In a hierarchical planning model they are interpreted as a hint to the details planner that they should create and schedule a process order. They are often the result of an automated MRP (material requirements planning) run that is based on *planned independent requirements* and does not consider resource capacities.

A planned independent requirement is a planned requirement quantity for a product for a certain period of time. It is not necessarily created on the basis of any customer requirement.

Planned independent requirements give the impression of a defined cross-scope material flow (i.e., a flow with a possible target outside of the organizational unit considered) at a given time with a given quantity but in most cases they are only based on a rough monthly sales demand forecast figure for the finished product. This means that the requirement quantity and date of a planned independent requirement are less reliable and stable than the requirement quantity and date of a confirmed customer requirement.

12.4.4 Derived Dynamic Data

For some devices (resources) one wishes to know their future allocation at any point in time over a given time interval (planning horizon). For this purpose one can use a resource allocation list that lists all operations from process orders that use the resource. The resource allocation list is visualized by bar graphs (Gantt chart, original version defined around from 1900 by Henry Laurence Gantt).

Additionally, one wishes to know the development of material stock levels at any point in time over a given time interval. For this purpose one notes all receipts and issues, both from process orders inside the balance area and incoming to or outgoing from the balance area. This *stock/requirements list* can be evaluated by various algorithms. For example one only considers *plant stock* and "definite" receipts and issues to calculate the definite future stocks by best current knowledge. On the other hand one may include the planned independent requirements in the consideration to calculate the expected future stocks.

The stock/requirements list may also be evaluated for different storage locations. There are also material types for which no stock/requirements list is necessary (e.g., consumables and cleaning materials).

12.4.5 Constraints

Constraints are given as resource utilization restrictions (e.g., shift regimes, workforce availability, energy, catalyzer). In addition relations between different dynamic data objects are defined (e.g., setup matrix, order network). Constraints can be considered as hard (must be satisfied) or soft (violations are visualized) in a given time interval with regard to calculation of derived dynamic data.

12.5 Planning Software

12.5.1 ERP System

The big players in the chemical industry have developed very powerful company-specific ERP systems (enterprise resource planning; earlier this type of system was denoted as production planning and control) since the 1960s. The growing costs to maintain and adapt these systems have motivated the move to integrated standard business software systems in the last decade. The majority of the big chemical companies use SAP R/3 resp. SAP ERP from SAP AG; other well-known solution vendors include Marcam (now part of Infor) and JD Edwards (now part of PeopleSoft resp. Oracle).

These software products are used company-wide and integrate all operational, business and financial information in huge databases.

To integrate an organizational unit that uses multipurpose batch plants into the company, IT infrastructure with its ERP system, a hierarchical planning approach is most often used. Starting from a material requirements planning (MRP) run, capacity requirements are determined and roughly checked, although the check of the capacity requirements is not directly combined with the material requirements.

In general the overall (company-wide) MRP run will create *planned orders* for the multipurpose batch plant which have to be further processed by the plant details planner.

12.5.2 Resource Allocation Planning

If one pursues this hierarchical approach to separate rough planning and detailed planning (scheduling) and to separate material and capacity requirements, then the details planner of the multipurpose batch plant has to solve a resource allocation problem for the production devices. The higher-level rough planner using the ERP system is responsible for the material flow and gives instructions about the required production to the details planner by way of planned orders. The planner has roughly checked the capacity requirements of these instructions. Based on this set of planned orders it is now the task of the details planner to schedule devices in a way that makes it really possible to produce all necessary intermediates and finished products in the required time and quantity.

This task is generally denoted as a scheduling problem in mathematical publications (e.g., see [3]).

A lot of contradictions in planning processes have their root in the separation of material requirements planning and resource scheduling. This leads to the situation that detailed planning of multipurpose batch plants is still the domain of experienced production schedulers and shift managers who have gained superior knowledge over the years that makes them indispensable.

A number of software solutions still support this classical approach:

Approach 1 (Classical Approach):
Planned orders and other information are sent from the ERP system to a scheduling system via an interface. The scheduling system has its own database and a number of different automatic and interactive scheduling functions. Scheduling results are sent back to the superior system via an interface at the end of the scheduling process. This loose integration respects the hierarchical concept and the autonomy of the planners.

Confirmations from production execution are often not fully retransmitted to the superior system in all detail, thus deviations from the schedule may not be considered in the high-level rough planning.

12.5.3 Advanced Planning and Scheduling (APS)

The task of the details scheduler is to create and modify process orders. There are different approaches to support this task.

Approach 2 (Independence Approach):
The APS system is supported by an independent modern data model that is often based on generic input/output nodes and their dynamic combination in complex networks and thus better suited for algorithmic processing and optimization than the transaction-oriented business data model of the ERP system.

An integration interface transforms master data and dynamic data by mapping objects to the respective target model.

Due to the fact that one might not want to use all ERP business data objects in the APS system, an integration model is used to define for which plants, resources, materials, customers, etc., the integration interface should be active. The APS system also allows to maintain additional master data or to modify and enhance master data that came from the ERP system. Sometimes there are also additional types of master data (e.g. resource setup matrix) or information fields to master data (e.g., scheduling horizon) that really have no analogy in the ERP system and thus have to be maintained directly in the APS system.

The consequence is that data has to be maintained and consolidated in two systems. Because also some planning and production execution functions will stay in the ERP system, there is quite often the need for the planner to work with both systems and to gather information from both systems (e.g., customer order details, confirmations from production execution, which is handled in the ERP system) to make planning decisions.

A representative for this approach is the product APO resp. SAP SCM of SAP AG that is most often used in conjunction with the ERP system SAP R/3 resp. SAP ERP.

Approach 3 (Embedding Approach, Add-on Approach):
Quite often not all modeling capabilities of an ERP system are used to their full extent. For instance it is possible to model continuous material flow, alternative devices, campaigns, resource nets, operation relationships, etc., in the ERP system, thus there is no need for an enhanced model in the scheduling system. The ERP system however lacks the proper algorithms to use the enhanced data (e.g., for detailed scheduling instead of rough capacity leveling).

Thus the alternative approach is to create enhancements (add-ons) to an ERP system that do not have their own persistent database and provide additional interactive functions, visualization and algorithmic processing of enhanced model data. The add-ons create a temporary local data storage (LiveCache) to effectively process the enhanced data. However, all data is stored persistently exclusively in the ERP system. This allows for the local LiveCache to use a mapping of the ERP data model that is structured in a way that is more suitable for APS purposes. Additionally the

add-on should be integrated as seamlessly as possible into the user interface of the ERP system.

This approach has the advantage that it is not necessary to start a huge implementation project that leads to a "big bang" Go-live to improve the detailed scheduling process, but rather it is possible to improve an existing situation evolutionary in a rapid, step-by-step and minimum-risk procedure. Because the mapping in the APS system is not persistent, it can be modified and adapted to the growing needs of the users step-by-step, starting from a standard configuration.

A representative for this approach is the product SCHEDULE++ of OR Soft Jaenicke GmbH.

12.6 Remarks on Planning Philosophy

The topic of scheduling is covered by thousands of scientific publications.

The opinion of the authors of this chapter (who have dealt with the matter as mathematicians for some decades) can be summarized in two simple statements:

- Scheduling problems are combinatorial optimization problems.
- One can find an arbitrary number of both theoretical and practical examples that are either unsolvable or that have a solution that cannot be proven to be optimal.

However, the fast development of hardware performance in the last decade and the current university training seems to lead many IT experts and decision makers in the chemical industry to the wishful thinking that any problem can be solved by "optimization" and sheer calculation power.

One can apply as serious solution approaches:

- branch/bound or backtracking,
- heuristics.

By transformation into mixed integer linear optimization problems one can:

- transform the problem solution to the solution of a different difficult problem,
- try out approximation approaches.

By introduction of penalty functions one can try to combine the various fuzzy constraints into a single evaluation criterion.

By partitioning one can try to split the problem into a number of smaller problems that may be easier to solve and then recombine the local optimal solutions into a global solution.

All these approaches follow the hope to be able to solve a complicated problem by a "one-button solution."

This may be possible in some special cases (stationary bottleneck situation, "friendly" requirements situation, etc.). Quite often one happily accepts a solution that is declared to be "optimal" by the program.

The following situations are more unpleasant:

- One can watch desperate users trying to manipulate the parameters of optimization programs until the program will give them the apparently obvious solution.
- The program gives an infeasible solution (e.g., when penalty functions are used).
- The program shows that no solution exists (this is a common case).
- The solution given by the program (that is most likely correct) contradicts the expectations of the user gathered by practical experience (quite often due to the fact that the user has additional fuzzy constraints "at the back of their minds" that are not known to the program).

In most cases known to the authors planning software is used in situations where in fact one has to deal with unsolvable planning problems (not enough capacity, requirement peaks, lack of workforce etc.).

This is why from the point of view of the authors APS systems are mostly used to support human planners.

Planning is done by humans who use computer support tools like simulation, optimization and production planning programs. They gather information from these systems, make trial decisions and monitor the consequences.

Production planning/optimization always requires a compromise to be found by the humans in charge of decision-making. In the process of finding this compromise, one needs scheduling programs that are integrated into the ERP systems.

From the point of view of the authors it is advisable not to overemphasize the ideas of optimization and the desire for "one-button solutions." A good compromise can be found if the overall production situation can be made transparent just-in-time to the motivated employees and if reasonable business processes have been established.

12.7 Remarks on Technical Issues

There are many possibilities for the implementation of interfaces. They include file transfer, the use of transactions and the modern technology to define an integration layer and adapters (e.g., SAP NetWeaver).

The implementation teams of the authors have in particular gathered experience with interfaces to the systems SAP R/3 and SAP APO. They have noted that the time to establish a stable interface is essential for the success of an APS project. The establishment of a first stable interface between ERP and APS system should be completed in not more than one week.

References

1 Striebeck, U.B. (2000) *SAP-Handbuch*, IDW, Düsseldorf. For SAP terminology see also: http://help.sap.com/content/additional/glossary/index.htm.

2 NAMUR-Recommendation (2003) *NE 33 Requirements to be met by Systems for Recipe-Based Operations*, available at: http://www.namur.de.

3 Brucker, P. (1998, 2001) *Scheduling Algorithms*, Springer, Berlin.

Index