DISSERTATION

Titel der Dissertation

„A Combined Approach of Simulation and Optimization in Supply Chain Management”

Verfasserin

Mag. Margaretha Preusser

angestrebter akademischer Grad

Doktorin der Sozial- und Wirtschaftswissenschaften

Wien, im April 2008

Studienkennzahl lt. Studienblatt:	A 084 157
Dissertationsgebiet lt. Studienblatt:	Internationale Betriebswirtschaft
Betreuer:	Univ.-Prof. Dr. Richard Hartl

Danksagung

Mein Dank gilt univ.-Prof. Dr. Richard Hartl und Dr. Christian Almeder. Ihrer erstklassigen Betreuung, laufenden Unterstützung und Spaß an der Forschung verdanke ich alle Erfolge und Erfahrungen, die ich im Rahmen meiner wissenschaftlichen Tätigkeit gemacht habe.

Ich danke meinen Eltern, Geschwistern, Großeltern, Peter, Maus und Luis. Dieses Buch ist für Euch.

Contents

Part I.

Introduction

1. Objective and motivation

In recent years intra-company supply chains have been growing significantly spanning production sites and distribution networks all over the world. Managing supply chains becomes a more and more important but also a highly challenging task. Improved performance and ongoing optimization along the whole value chain are vital. It is impossible to imagine neither management science nor practice without the subject of *Supply Chain Management (SCM)*. According to Stadtler (2005), *Supply Chain Management* is defined as the task of integrating participants involved along a supply chain and of coordinating of materials, information, and financial flows in order to fulfill ultimate customer demands with the aim of improving competitiveness of a supply chain as a whole (cf. Lee and Ng (1998)). Participants may be suppliers, manufacturing sites, warehouses, distribution centers, or any other actor being integrated in a supply network. At the same time global competition has increased, such that there is strong demand for new decision support tools on strategic, tactical and operational levels. Strategic decisions in this context primarily refer to supply chain design problems like, e.g., the warehouse location problem. Tactical decisions on the other hand mainly correspond to enabling of operational processes by, e.g., dimensioning and layout planning. Finally, operational decisions are related to short-term planning of the operational processes themselves. Production scheduling, for example, is a typical problem within the operational decision level. Biswas and Narahari (2004) classify the relevant research on decision support systems into three categories:

1. Optimization models mainly for multi-echelon inventory control. In most cases these models are deterministic and used for strategic or tactical decisions.

2. Analytical performance models, which consider a dynamic and stochastic environment. They are used to investigate design or principal management decisions. Such systems are represented as Markov chains, Petri nets or queuing models.

3. Simulation and information models, which are used to analyze complex dynamic and stochastic situations and to understand issues of supply chain decision making.

For the first and the second categories it is often necessary to make several simplifications from the real-world case in order to develop solvable models. This leads to simple optimization models being too abstract, such that the relation to the real-world scenario is questionable. Neither is our world deterministic nor is it linear, but many planning models are based on these simplifications in order to keep them solvable with reasonable computational effort. Nevertheless, the maximum problem size is usually very limited. Mixed integer programming (MIP) is a commonly used method for analyzing and improving supply chain networks. Category 3 refers to simulation models, e.g. discrete-event models, which are used to mimic the real behaviour of a supply chain providing a playground for various experiments. However, they do not have the ability to provide optimal solutions since they are often too complex to apply an effective optimization technique. According to this, optimization as well as simulation are widely accepted in this context, but definitively both of them have crucial drawbacks. Although there are promising developments of combinations of these two categories (cf. Chapter 2), many of them remain on a strategic level and the stochastic property is considered by a small number of different scenarios.

Most of the time during my PhD studies I focused on developing a new approach for using LP/MIP models in the context of discrete-event simulation. The idea was to become able to combine the advantages of both methods by considering a detailed representation of a dynamic-stochastic environment while applying optimization models in the context of *Supply Chain Management*. As a matter of fact we call this hybrid approach *SimOpt*. The initial proposal for this field of research was delivered by the *Austrian Research Centers Seibersdorf (ARCS)*, with which we had a two years interesting and productive corporation on this topic. Having successfully implemented the combined simulation-optimization approach a second corporation evolved. This time we were asked by a global player of the paper producing industry to analyze potential improvements within their distribution strategy. This real world study gave valuable input concerning the model design requirements in order to generate a method able to handle problems of realistic size. This thesis is partly based on publications resulting from my scientific activities during my PhD studies: Almeder and Preusser (2004), Preusser et al. (2005a), Preusser et al. (2005b), Gronalt, Hartl, and Preusser (2007), Almeder, Preusser, and Hartl (2008).

For our investigations we formulated a general supply chain network model with different facilities (suppliers, manufacturers, distributors) and different transportation modes connecting these facilities. We do not take supply chain design problems into account, but assume a predefined network structure. We furthermore act on the assumption of having a central planner with perfect information such as for intra-company supply chains or supply chains with a dominant member. The supply chain is represented as a discrete-event model and we developed a toolbox for simulating operational decisions such as production, stocking, transportation, and distribution. A simplified version is modeled as an optimization model, which might be a linear program, a mixed integer linear program or just any kind of optimization model being small enough to provide solutions with reasonable computational effort. We are aiming for reduction of total costs by simultaneously optimizing the production/transportation schedule and reducing inventory levels. Thus, the goal is to achieve robust operations plans for supply chain networks by referring to a stochastic environment and additionally combine it with classical optimization approaches.

We do not use the optimization on top of the simulation (parameter optimization), but include simulation and optimization into an iterative process based on consistent mutual data exchange. We start by performing several simulation runs in order to get average values of the parameters (e.g. unit transportation costs) which are then fed into the optimization model. After solving the optimization model the result is transformed into decision rules that consequently are used within the discrete-event model. Then we start again with further simulation experiments, and so on.

Comparing the considered problem classes to the tasks in the well known supply chain matrix (cf. Fleischmann, Meyr, and Wagner (2005)), we are facing combinations of several operational tasks in production planning, distribution planning, and transport planning. Assuming the network design as predefined we clearly remain on the tactical and operational decision level.

The basic contribution to the already well explored field of optimization in supply chain management, is a successful hybridization of two traditional solution methods, i.e. simulation and optimization. By reconciling the advantages of these well established approaches we succeed in finding competitive results for supply chains of different sizes and levels of complexity. Especially when trying to cope stochastic elements within supply chains, our iterative combination of simulation and (mixed-integer) linear programming finds good results in terms of computation times and solution robustness. Since the embedded optimization model is supposed to cover just a minimum of relevant nonlinear elements, we are able to explore real-world settings being far to complex for traditional exact solution approaches.

This thesis is separated into three basic parts. The first one introduces, motivates, and embeds this study into the existing literature. This will be followed by a detailed implementation description and reports on the conducted numerical experiments. Finally, Part III gives deep insight into the already mentioned real world case and some interesting follow-up experiments. We conclude with a summary and an overview of interesting fields for further research in this context.

2. Literature review

2.1. Integrated *Supply Chain Management*

Aspects of the integration of transport and production planning within supply chains have been investigated in several papers. An extensive survey on this topic is given by Erengüc, Simpson, and Vakharia (1999). They claim that, from an operational perspective, there are two relevant features when dealing with a supply chain: the supply chain network, and the nature of the relationship between each stage in the network. Based on these they identify the relevant decisions that need to be considered for an integrated production/distribution planning. Exemplary model formulations for each stage of such a network are presented as well.

Combined hierarchical planning approaches for different decision levels can, for example, be found in Jayaraman and Pirkul (2001) where a combination of strategic and of operational decisions is conducted. On the strategic level they consider, among others, the location of plants and warehouses while on the operational level the distribution strategy from plants to customer outlets through warehouses is addressed. They present a MIP model formulation and an heuristic solution procedure utilizing the Lagrangian relaxation in order to solve this combined planning problem. Arbib and Marinelli (2005) present an integer programming model integrating two hierarchical decision levels (short-term operations and mid-term planning) and functional areas (production and purchase of materials). They aim for minimization of production, holding, and transportation costs in the context of a real cutting process with skiving option.

Further approaches for integrated planning approaches can be found in Meyr (2002) and Schneeweiss (2003). While Meyr (2002) presents a combination of lotsizing and scheduling Schneeweiss (2003) considers two mechanisms for coordinating supply links (cf. Kodnar (2007)). One instrument refers to the determination of an appropriate procurement policy for the producer, and the other mechanism defines optimal penalty costs. The type of coordination is based on the amount of private information that is kept by the participants. Coordination of supply chains has been investigated by several authors. Lee and Wang (1999), for example, consider the case of decentralized multi-echelon supply chains, where site managers have to be provided with incentives in order to align their interests. Consequently, different performance measurement schemes are derived. Problems of decentralized supply chains and the resulting information delays have also been addressed by Chen (1999). Weng (1999) considers the possibility of effectively coordinating a supply chain that is composed of one supplier and several homogeneous buyers through simultaneously applying quantity discounts and franchise fees. Chen, Federgruen, and Zhen (2001) carry on with the presented idea of Weng and show that quantity discounts are not enough to coordinate a supply chain, in case the retailers are not homogeneous. Cachon and Lariviere (2001) focus on shared demand forecasts within a simple supply chain link where the supplier builds his capacities due to the provided demand forecasts.

There are numerous papers dealing with linear or mixed-integer programs for supply chain networks and network flows (cf. Shapiro (1993)). Yaged (1971) discusses in his paper a static network model which includes nonlinearities. He tries to optimize the product flow by solving a linearized version of the network and to improve the flow in the network. Paraschis (1989) discusses several different possibilities to linearize such networks and Fleischmann (1993) presents several applications of network flow models, which are solved through linearization. Pankaj and Fisher (1994) showed that based on an MIP model the coordination of production and distribution can reduce the operating cost substantially. Dogan and Goetschalckx (1999) showed that larger supply chain design problems can be solved using decomposition while others like, e.g., Vidal and Goetschalckx (2001) primarily focus on the transportation aspect. A recent case study about a supply chain in the pulp industry modeled as a MIP is given by Gunnarsson, Rönnqvist, and Carlsson (2007). Problems solved with LPs and MIPs usually include several simplifications in order to keep them solvable. Tsiakis, Shah, and Pantelides (2001), for example, did some work on supply chain network design. They assume fixed manufacturing and customer zones, but the locations of warehouses and distribution centers are to be determined. Recent publications include stochastic elements within the optimization models as well. Santoso et al. (2005) consider a stochastic programming approach for the supply chain network design. They use a simple average approximation and a decomposition method to solve design problems for a supply chain while taking future operational costs into account. For that purpose they developed a linear model with uncertain cost factors and demand. Although they use a fast algorithm, realistic problems with sample sizes of up to 60 scenarios need several hours to be solved. Alonso-Ayuso et al. (2003) consider a similar combined design and operation problem. Their stochastic programming approach is able to solve medium sized problems with about 100 binary decisions within one hour. Leung et al. (2007) present a robust optimization model for a simultaneous production planning for several sites in a supply chain under uncertainty. But still they are restricted to rather small models and consider only four different scenarios.

Concerning the field of supply chain simulation Kleijnen (2005) gives a short overview of simulation tools and techniques used for supply chains. He distinguishes between four different approaches: spreadsheet simulation, system dynamics, discrete-event dynamic systems simulation, and business games. Clearly, discrete-event simulation is the most powerful tool to consider complex stochastic systems, but improving strategies for certain objectives are mainly restricted to a trial-and-error procedure. Numerous software packages for discrete-event simulation are available, both very specialized ones for a specific part of the supply chain and general ones with a high functionality in modeling and visualization of supply chains (cf. Kelton, Sadowski, and Sadowski (2002), Kuhn and Rabe (1998)). One example is the *Supply Net Simulator* presented by Stäblein, Baumgärtel, and Wilke (2007). It allows to simulate the behaviour of individual members in a supply chain network. They use an agent-based approach, where each member optimizes its own operations in the sense of an advanced planning system. But there is no interaction between simulation and optimization.

2.2. Simulation and optimization

Most of today's simulators include possibilities to do a black-box parameter optimization of a simulation model. Glover, Kelly, and Laguna (1999) present the successful development of OptQuest (© OptTek Systems[1]), an optimization toolbox containing different algorithms (mainly metaheuristics) designed to optimize configuration decisions in simulation models. The simulation model is used only for the evaluation of the objective value, no further structural information is considered. However, long computation times for evaluating the simulation-based objective make classical search procedures inefficient. Swisher et al. (2000) describe key issues for doing parameter optimization within a simulation model and Fu (2002) gives a survey on the available software solutions in this context. Both of them state in their papers, that there still is a big gap between optimization methods for simulation-based optimization used in commercial software and methods available in research literature. Truong and Azadivar (2003) developed an environment for solving supply chain design problems, where they combine simulation with genetic algorithms and mixed integer programs. Strategic decisions regarding facility location and partner selection are considered. The work by Lee and Kim (2002), possibly the most related work in the context of this thesis, shows a real combination of simulation and optimization for the case of a production-distribution system. They use simulation to check the capacity assumptions used for a simpler linear model in a more realistic environment with stochastic machine break-downs and for updating these capacity parameters for the optimization. After several iterations they end up with a solution of the optimization model which is also within the constraints of the stochastic simulation model. Their method is quite similar to our approach, but they aim for more realistic capacity estimation for the optimization model. In contrast, we try to find a robust plan for production, stocking, and transportation considering stochastic and nonlinear operations and costs by estimating delays and cost factors based on simulation experiments.

[1]www.opttek.com

Part II.

Implementation and proof of concept

3. Introduction

In order to solve large scale supply chain problems we developed *SimOpt*, a new hybrid solution approach by applying an LP/MIP formulation in the context of discrete-event simulation. In order to profit from advantages of both categories, i.e. simulation and optimization, we couple a detailed reproduction of the given supply chain setting with an optimization model which is basically provided with the structural information of the problem. Our approach provides the possibility to model and solve more realistic problems (incorporating dynamism and uncertainty) in an acceptable way. Thus, we are aiming for a trade-off between finding a good approximation of the optimal decisions within a model being close to reality but even so having reasonable computation times. For this purpose we combine a simulation model representing the best possible reproduction of the real setting, i.e. including all nonlinearities (step functions, binary decisions, etc.) and stochastic elements, with an exact LP/MIP model which represents the simplified, maybe even the linearized, reproduction of reality. In an iterative process where information is handed over from simulation to optimization and reversed we approximate a good solution concerning operational and tactical decisions of the complete network. The total cost of operation is used as the objective function to be minimized. Total cost includes production cost, transaction cost, storage cost, transportation cost, and penalty cost for late deliveries. The simulation model includes nonlinear and stochastic elements, whereas the optimization model just represents a simplified version of the given setting. Figure 3.1 illustrates the interactive cycle of activities between simulation and optimization.

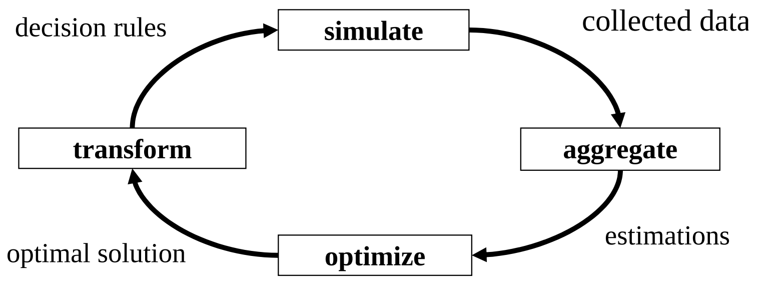

Figure 3.1.: General procedure of the combined simulation-optimization approach

The simulation is the master process of the whole procedure and the general course of an experiment is as follows: based on a set of initial simulation runs different parameters, i.e. costs and delays, are estimated and handed over to the optimization model. Based on these values the optimization model, which is also provided with all necessary information on the network structure and relevant general parameters, computes the optimal solution and translates it into decision rules, e.g., ordering and transportation plans, which are intended to be applied within the forthcoming set of simulation runs. Aiming for an improvement of the overall performance of the network the next set of simulation runs is conducted and the collected data is used for new estimations of special parameters like costs and delays. Again, an optimization run is performed based on the updated parameter estimations. The solution of this run is again translated into decision rules and fed into the simulation which is going to use it for its next set of runs. Clearly, due to the changed decision rules provided by the optimization we might be in a completely different situation which in turn necessitates a recalculation of the parameters and consequently a further optimization run. As long as the objects in the simulation model do not have input from the optimization, i.e. during the initial runs, they act according to autonomous decision rules, e.g., an (s,S)-policy for ordering decisions. This procedure is applied iteratively until the solution of both the simulation and the optimization do not change anymore, i.e. until we reach a stable solution. We are aiming for a robust solution in the sense that we want to find good solutions while considering impacts of stochastic elements. Thus, we want to get the biasing effects of uncertainty under control. More details on this concept will be discussed in Chapter 5.

Clearly, taking nonlinearities into account within the optimization part as well is definitely preferable for finding good solutions but the increase of computation time for solving a MIP instead of an LP model is not negligible, and has to be explicitly considered. Thus, the challenge is to find the best trade-off between the best possible approximation of an optimal solution in a reasonable amount of time. It is apparent that at least essential decisions that may lead to a high increase of costs should be taken into account within the optimization model. This implication is verified based on numeric examples in Chapter 6. As a matter of fact stochastic elements are solely considered within the simulation part. In the following a *SimOpt* experiment is called *SimLP* if a pure LP model is connected to the simulation, and it is called *SimMIP* in case the optimization part refers to nonlinear elements as well, i.e. if a MIP model is connected to the simulation model instead of a pure linear model.

The question of convergence of this looping procedure cannot be answered in general. We are able to construct special examples, where we got trapped in an cycle and hence no convergence is possible (see Section 6). However, all experiments on realistic and randomly generated instances lead to fast convergence, i.e. the gap between the results of the simulation and the optimization is decreasing and after 3-4 iterations it reaches an acceptable low level of less than 1%. Clearly, a stable solution does not guarantee an optimal one. Therefore, we tried to evaluate the solution quality based on a set of small test instances. For them we find good solutions quite quickly which means after less than four iterations. All this is presented and discussed in Chapter 6.

Our framework is based on a general supply chain network model with different facilities (suppliers, manufacturers, customers) and different transportation modes connecting these facilities. Figure 3.2 displays the key elements of the supply chain model within a schematic illustration.

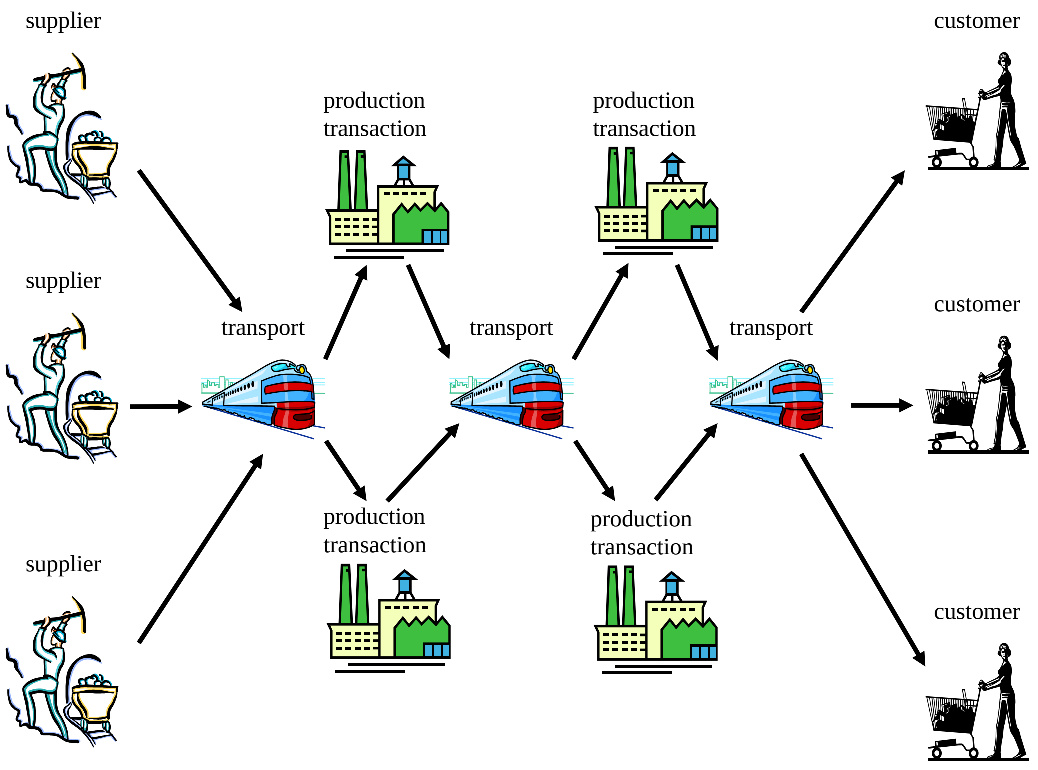

Figure 3.2.: Schematic illustration of a supply chain network

We assume that there is a central planning with perfect information like in intra-company supply chains or networks with a dominant member. Due to its general formulation the model is open for a wide range of adaptions according to the given supply chain setting. The basis for our supply chain model is a predefined network, i.e., the locations of all participants and the connections between them are given. Within the network we differentiate between four types of participants connected by transportation links:

- suppliers providing raw materials
- production/warehouse sites where production, stocking, and transshipment takes place
- customers who demand certain products at a specific time
- transportation objects connecting the other participants

The supply nodes have a given supply and an outbound inventory. They are used as source for raw materials, which are then sent to production/warehouse sites. The latter can neither be located a the very beginning nor at the downstream end of a supply chain and therefore act always as intermediaries between other nodes. Each intermediate node disposes of both an inbound and an outbound inventory. The former is needed for storing incoming products until they are either used within a production process or simply transacted to the outbound inventory. From there they are sent to another intermediate node or to the downstream end of the supply chain, i.e. the customer nodes. The latter have a given demand and an inbound inventory used for buffering premature deliveries which may occur due to stochastic elements, until they are used for the satisfaction of the customer's demand. So these premature deliveries result in a positive inventory at the customer which of course is penalized. Backorders, i.e. a negative inventory status at the customer, are also possible but they are penalized as well.

The whole supply chain is order-driven, which means that products are manufactured, transshipped, or transported only if a subsequent member of the supply chain requests it. So the origin for all activities is the predefined deterministic demand of the customers. All activities are based on time periods, which might be days or shorter time periods since we are focusing on operational decisions. Each participant, each transportation mode and also each product type can be identified by a unique numeric ID.

Since the aim was to construct an easy-to-handle general framework for diverse network settings we decided to build-up our own supply chain management simulation library on a modular basis by implementing a set of suitable object classes. These classes just provide a basic building block; when used within a simulation project a huge number of parameters (e.g., cost functions) can easily be adjusted within the instances themselves. These user-defined fine-tuning options will be listed for each class separately. In the following this supply chain management library will be denoted by *SCMLib*.

We developed *SCMLib* using the simulation tool AnyLogic 5.5 (© XJ Technologies[1]) which is a multi-paradigm simulation software. It can be applied for discrete-event, as it had been done for our purpose, but also for continuous or hybrid kinds of simulation. The language for user-defined functionalities is Java. The object-oriented model design provides for modular construction of simulation models. We created an object class for each type of participant of the supply chain network plus a general control class necessary for managing the simulation experiments as well as the communication with the optimization model. Simulation experiments based on our library are conducted by creating a new project, i.e. a working unit in AnyLogic, and including predefined objects such that the considered supply chain network can be represented best possible. A project is displayed as root object in the *Project* window of the user interface. The optimization model is written in Xpress-Mosel (© Dash Optimization[2]) and solved by the Xpress-Optimizer (© Dash Optimization[2]).

In the following an extensive description on the implementation and functionality

[1]www.xjtek.com
[2]www.dashoptimization.com

of *SCMLib* as well as on the creation of a fully functional *SimOpt* setting will be given. The presentation of results referring to different sized test instances will conclude this chapter.

4. Object classes and their features

It has already been mentioned above that we implemented a special library destined for simulating supply chain networks. We defined some object classes for our own purpose, but certainly used a number of predefined classes as well. The most relevant among them will be mentioned in the course of this chapter.

In AnyLogic the project internal communication can be organized using Java objects which are exchanged between objects. These interchangeable objects are called *Messages*. Sometimes they are also used for information exchange within one and the same object. One has to define one or more message classes in order to make use of message objects. Typically objects exchange messages like commands or signals but also entity units like virtual products or people are possible. These interchangeable Java objects are prepared to store a predefined set of information and are also quite useful for measuring the duration of certain processes within the network (e.g., transportation times, waiting times, etc.). They are routed via connection lines which can be inserted between any two objects that are intended for data exchange. For this purpose special elements, called *Ports*, are provided which can either be used for receiving ingoing and for sending outgoing messages[1]. *SCMLib* knows three message classes:

- Message class *Lot*
- Message class *Product*
- Message class *Request*

The first one is solely used for objects that are involved in production or transaction. Instances of *Lot* specify a production or transaction lot. Whenever such a message is created by an object it will be automatically fed with the current product ID, the size of the lot, and the lot's starting time (i.e. creation time). Later on this information is used for recording production and transaction times. Instances of this class are not intended for exchange between different objects but for virtual product interchange within one and the same object.

Message class *Product* is, similar to the previously described one, assigned to entity flow. In principle it is used exactly the same way but unlike the first one it is assigned to data exchange between different objects. Additional to the already

[1] www.xjtek.com

mentioned input parameters of an instance of *Lot*, which are again necessary, the creating object automatically fills in information on the lot's source (i.e. its own ID), the destination (i.e. ID of the receiving object), and the ID of the transport mode chosen to execute the delivery.

As already mentioned above the whole supply chain network is order-driven and the entity flow is triggered by requests sent from objects to their predecessors within the network. For the according inter- and also intra-signaling mechanism we use a message class called *Request*. Therefore, a request is created whenever an object invites another one to send products. The same mechanism is used for requests between the outbound and the inbound inventory of intermediate nodes. Here the output inventory generates an instance of *Request* whenever it needs to be supplied with new products from the inbound inventory. In each case the message is automatically provided with the ID of the requested product and the amount needed, source and destination of the potential delivery, and finally the ID of the transport mode that should be chosen.

Another important predefined class in AnyLogic is called *Timer* and it is used to manage timeouts and delays. AnyLogic differs between dynamic and static timers. Both of them have a their own characteristics and fields of applications, but to put it simply the difference between them is that the dynamic ones delete themselves after expiry and static ones survive and can be restarted. The start and later restarts of a static timer are always of an user-defined form. The run-time of a dynamic timer starts whenever it is constructed, i.e. started, by an object. Timeout is passed to the timer's constructor, which makes them very useful for time measurements. Both types have been used in our implementation several times. We incorporated static ones for almost all cyclic activities (e.g., updates of costs, stock levels, or waiting queues, etc.). Since dynamic ones are predestinated for time measurement we primarily used them for measuring transportation, production and transaction delays. For this purpose we implemented different dynamic timer classes each being assigned to a specific object class. During the simulation run instances of those classes are created by particular objects, which in turn will learn about the timer's timeout and therefore the length of its active time[2].

Throughout the network we have to handle occurring waiting queues, e.g., products waiting for transportation, which is done by using `java.util.List` as an interface for creating Java lists. Entities waiting for processing are inserted into specific lists which are then by and by reduced according to first-come-first-serve priorizations.

The external exchange of data, i.e. the communication between simulation and optimization model, is done by using an external database. In the following we differ between three categories of data:

1. *Network structure*: information about the network configuration including the number of actors and the according links between them. The number of products and the number of considered time periods are also to be assigned to this

[2]www.xjtek.com

category. These values are certainly needed by both simulation and optimization model and are imported by both of them. This import happens once in the beginning of each experiment.

2. *Global parameters*: fixed predefined values (e.g., capacity limitations, resource parameters, bill-of-materials, predefined supply at the suppliers and predefined demand at the customers, etc.) that are also imported from the database once at the beginning of each simulation or optimization run. They are used for simulation as well as for optimization and are never changed during the experiments.

3. *Estimations*: values that are collected and then aggregated by the simulation model (e.g., costs, delays, etc.) in order to be exported to the database. The optimization model uses the current entries for forthcoming optimizations. Clearly it may happen that for one or the other parameter no estimation can be made because, for example, the corresponding node or mode has not been in use during the last simulation run. Reporting zero for this parameter would strongly influence the solution of the next optimization run in the sense that all unused connections or nodes would seem to be extra attractive for the cost minimizing optimization model. Since this implication is not desirable and will probably lead to a cycling behaviour we implemented a mechanism for automatically generating a reasonable estimation in this case. In fact, the object concerned creates a fictive lot and uses it to measure the missing data. After that the faked lot disappears and has no further relevance. Estimations might be done by simply calculating the mean of a certain sample of values. However, for critical parameters that have a strong influence on the objective, e.g., delay parameters, we prefer to make a distinct statistical analysis and to determine appropriate values based on the current sample. This issue is addressed in more detail later in this part.

4. *Decision variables*: values calculated by the optimization model (e.g., transportation plans, production plans,..) composing the optimal solution of an optimization run. As soon as the optimal solution is found the values are written to the database to be used by the simulation model. Due to this values each actor of the network arranges the timetable of its activities (e.g., when to start the production of a lot, etc.).

A more detailed description of the connection between simulation and optimization model will be presented in Chapter 5. Each of the following sections is dedicated to an object class appearing in *SCMLib*. Since the corresponding formulations used within the optimization model will also be mentioned there it seems useful to give a few introducing words on this issue at this point: the optimization model is formulated in a quite general way enabling the possibility to make adaptions according to the particular setting. Especially if and where to include nonlinear costs can be decided as the case may be. So the optimization model can be specified as pure LP or as MIP model. The goal should be to describe a simplified deterministic version of the considered network setting. Including nonlinearities, i.e. using a MIP model, should always happen in awareness of the, possibly significant, increase of computation time. Nevertheless it definitely makes sense to consider, primarily substantial, decisions within the optimization model. This issue will be further addressed in Chapter 6.

Within the optimization model the sets of supply, intermediate, and customer nodes are denoted by J_S, J_I, and J_C, respectively. Accordingly the set of all participants in a network is denoted by J. P is the set of products, and T the number of periods. The transportation modes are represented by set V. Figure 4.1 shows an exemplary and schematic illustration of a supply chain network in this context. Selective inclusion of variables serves an improved comprehensibility.

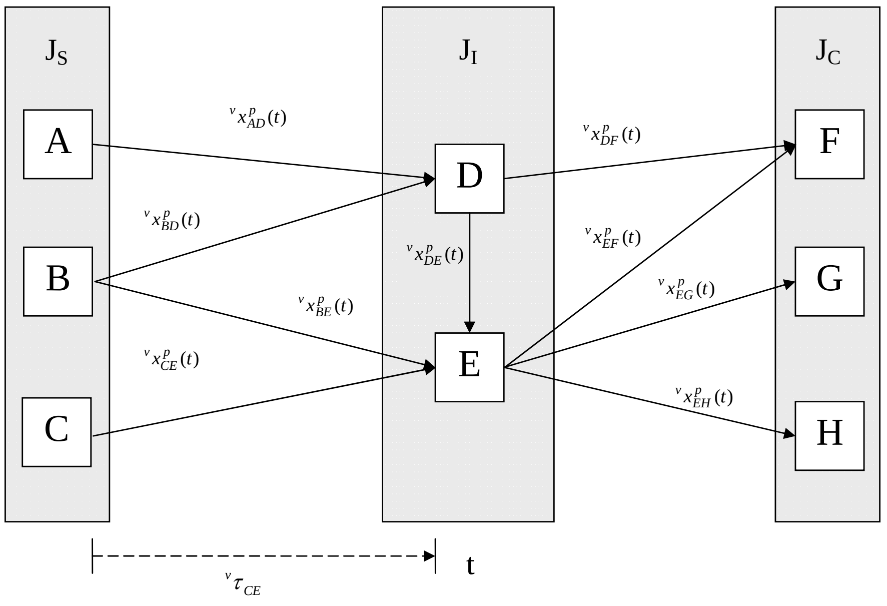

Figure 4.1.: Illustration of a supply chain network in the context of the optimization model

For clarity reasons only one type of decision variables is included in this illustration, which is ${}^v x^p_{ij}(t)$. It represents deliveries of product p from node i to node j done by transportation mode v starting in time period t. The amount of time periods needed for a delivery from node i to node j by transportation mode v is denoted by ${}^v\tau_{ij}$. The general formulation of the optimization model will be addressed piece by piece in the following sections of this chapter. Exemplary model adaptions in order to develop a *SimMIP* experiment, i.e. for considering nonlinearities within the optimization model as well, are suggested in Chapter 6. In there we show how to adapt the model formulation in order to cope best possible with the actual given setting. Please see Appendix A for a concise list of the notation used for the optimization model's formulation.

The following sections focus on object classes building the body of *SCMLib*. As superclass we defined *BaseNodeClass* which in turn is derived from the predefined class *ActiveObject* provided by AnyLogic. *BaseNodeClass* has the subclass *SCM-ControlClass* where we implemented a number of general methods and defined some public attributes. These essential methods and attributes are used by instances of all our self-defined object classes. The major task for instances derived from *SCM-ControlClass* is to control *SimOpt* experiments. They allow automated alternation between simulation and optimization and are furthermore used for administrating the corresponding data exchange.

4.1. *SCMControlClass*

Methods being valid for all existing objects are combined in their own class which is called *SCMControlClass*. Each simulation project should have exactly one instance of *SCMControlClass*, which is usually called *SCMControlObject*. The essential part of this class is that it is responsible for the management of all activities. For this purpose we use an external command file, which is the interface between user and *SCMControlClass*. Here the user basically defines the setting and the schedule of experiments that shall be executed. To some extent the user has to specify commands by a numerical entry. All commands are read in by *SCMControlClass* and are further used as public attributes, i.e. they are valid for all objects. AnyLogic calls them *Parameters*. Commands are passed to the particular objects which in turn have access to the necessary methods defined in *SCMControlClass*. A huge number of commands can be specified in the command file and not all of them are worth mentioning here. Nevertheless some of them should be listed at this point to give a brief insight:

- `useODBCInterface()`: definition of the used database interface by handing over its name.
- `load_lpsolution()`: one can decide if the recent information provided by the optimization model should be used for the forthcoming simulation run or not by handing over parameter 1 or 0, respectively. Typically optimization solutions cannot be used for the first initial runs since no optimization runs have been conducted yet.
- `setVerbose()`: determines how much information should be written into the logfile. This is regulated by inserting a parameter between 0 and 100. The higher the value the more information is provided.
- `setRoundingFactor()`: this command is used for the data aggregation after each simulation run. For some parameters, e.g., transportation times, it would not be reasonable to pass averaged values to the optimization model. For these parameters we use this command in order to make proper estimations. We go into this issue in Section 4.3.

- `setSmoothParameter()`: in most cases it makes sense not only to report the recent estimation of parameters to the optimization model but also to take previous ones into account. Especially for critical estimations like delays it seems more reasonable to use something like exponential smoothing. This can be done by using the given command and handing over any desired smoothing parameter between 0 and 1 (the higher the factor the lower is the impact of old values). Sometimes this increases the speed of convergence and dampens unwanted oscillations. For selected cases we even use exponential smoothing for non-critical parameters, i.e. cost parameters, trying to avoid oscillations.

- `mRun_wSave()`: the number which is handed over specifies the number of simulation runs in a row.

- `useResFile()`: giving the opportunity to create a clearly arranged solution report we use result files with a predefined structure. These can be commanded by the given expression. Handing over 1 activates the generation of result files, 0 deactivates it.

Additionally, *SCMControlClass* contains some essentials methods that are not directly influenced via the command file. They are globally used and automatically executed in any case. The most important among them are:

- *Network initialization*: import of network structure (number of suppliers, intermediates, customers, products, transportation modes, and length of planning horizon), global parameters (capacities, bill-of-materials, etc.), and allocation of IDs at the beginning of each simulation experiment.

- *Cost update*: every object has to check up its cumulated cost values (including total cost) and a few additional parameters, e.g., particular stock levels, production volume, etc., which are used for the final aggregation step and for a complete solution output. This update is restarted after each time unit by a static timer.

- *Write status*: each time period every object has to report its current status which will then be used to provide the relevant information which is then written to the logfile. Again, a static timer has been included for calling the function periodically.

- *Aggregation and estimation*: after a set of simulation runs each object has to aggregate the gathered data in order to provide parameter estimations for the optimization model. Estimations are basically done for two types of parameters: costs and delays. For cost parameters we generally calculate average per unit costs, which means that, for example, the production cost occurring in a certain production object for a certain product type are accumulated for the whole planning horizon and then divided by the number of products manufactured. Delays are estimated on behalf of a more complex calculation taking variances of the observations into account. This happens in instances of class *Production* and will be explained in the corresponding section.

- *Save to database*: each object is called to save the aggregated data and its own total cost into the corresponding tables of the database. The control object itself saves the calculated total cost of the whole network into the database. This value is essential because it is used to evaluate solution quality.

Another component of the controlling class are *Algorithmic Functions*. In Any-Logic one uses this expression when defining mathematical functions which are implemented as Java methods. Accordingly they can be accessed by other objects. In *SCMLib* they are solely used for more complex calculations of parameter estimations. All of them are defined as static functions implying that objects can use them directly by accessing the corresponding class.

4.2. The *Supplier* class

This class is used for supplier nodes which are able to generate products, hold them in its storage, and deliver them if demanded. Objects of this class have an input port to receive requests for products and an output port to send products. If a request is received through the input port the object tries to satisfy the demand as soon as possible. If the amount exceeds the current inventory level, only the available amount is sent. The remaining part of the order is inserted into a list, called *SendingList*, which corresponds to a waiting queue for non-satisfied requests. Every time unit a static timer checks if reductions of the queue are possible. Reductions become feasible as soon as new products are available, i.e. required products are on stock. The appearance of new products is due to given parameters which are imported from the database in the beginning of each simulation experiment. Products that are not sent away are stored in the object's outbound inventory.

Storage of products causes inventory cost which at the same time is the only cost arising in this module. Inventory cost functions may have any user-defined functional form. Each supplier object may have its own cost functions which can furthermore vary between different types of products. Considering fixed holding costs is as well possible as incorporating cost functions depending on stock levels, points in time, or product IDs. The functions have to be defined in specific fields in the objects themselves.

Supplier nodes are sparsely involved in data collection. After the initialization procedure they only have to retrieve the list of occurring supply from the database. They do not use any information provided by the optimization model but they have to report the average of their holding costs. This is done by saving the cumulated inventory cost as well as the cumulated inventory levels for each product type. The periodically restarting cost updating procedure, which has already been presented in Section 4.1, gathers the relevant data based on the user-defined cost functions mentioned above. In the aggregation step at the end of a simulation run the division of the cumulated inventory cost by the cumulated inventory level leads to the average holding costs per time unit for each product type which are then inserted into the database together with the object's total cost.

The representation of the supplier's behaviour in the optimization model can be formulated as follows (J_S denotes the set of supplier nodes within the network, P the set of products, and T the number of time periods):

$$TC_i^S = \sum_{p \in P} \sum_{t=1}^{T} {}^{out}H_i^p({}^{out}l_i^p(t)) \qquad \forall i \in J_S \tag{4.1}$$

$${}^{out}l_i^p(t) = {}^{out}l_i^p(t-1) - {}^{out}f_i^p(t) + S_i^p(t) \qquad \forall i \in J_S, t = 1, \dots, T, p \in P \tag{4.2}$$

$${}^{out}l_i^p(t) \geq 0 \qquad \forall i \in J_S, t = 1, \dots, T, p \in P \tag{4.3}$$

The overall cost of supplier i is denoted by TC_i^S consisting of the sum of holding cost defined by the cost function ${}^{out}H_i^p(\cdot)$ referring to its output inventory. The cost function has a user-defined form and depends on the inventory level ${}^{out}l_i^p(t)$ of each product p in each period t. This is expressed by the right-hand side of Equation (4.1). Equation (4.2) is an inventory balance equation. The inventory level of product p in period t must be equal to the corresponding level in the previous period minus ${}^{out}f_i^p(t)$ which is the amount of product p that has been sent to other nodes in period t plus $S_i^p(t)$ denoting the predefined replenishment of product p in period t. Clearly, inventory levels can never be less than zero which is guaranteed by Constraint (4.3).

Figure 4.2 illustrates the activities of a supplier object within the simulation. Selective inclusion of some of the optimization model's variables serves the purpose to improve transparency of interference between simulation and optimization. The only direct connection between instances of class *Supplier* and the optimization model, i.e. information that effectively is collected by the simulation model and then used by the optimization model, are the averaged inventory costs as it has been explained above.

4.3. The *Production* class

This class is the basis for all intermediary nodes which have the task to produce or to transact products. A lot of essential features are implemented in this class and therefore it can be seen as the core of our library. All instances of *Production* represent both a production site and a transshipment point and they dispose of a capacitated input and capacitated output storage. Incoming items are either used for production of new items according to a bill-of-materials, or simply transferred from the input to the output storage. Instances of *Production* have an input

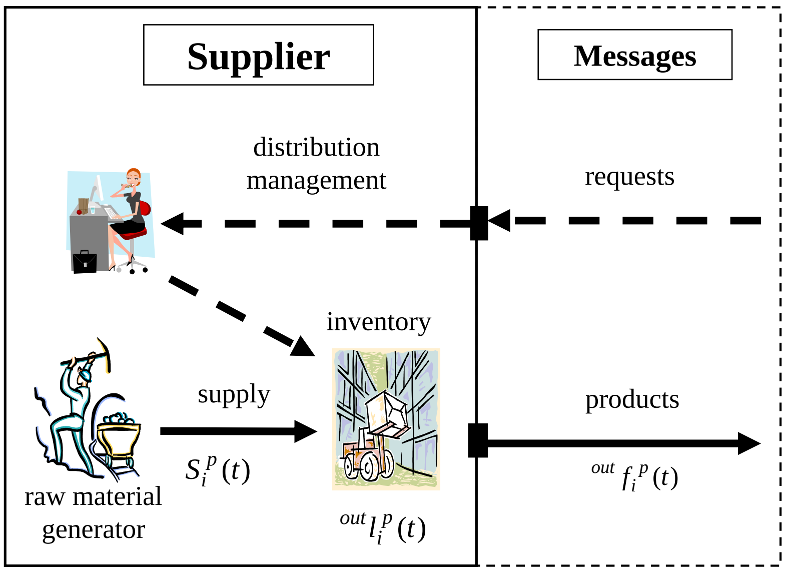

Figure 4.2.: Logical structure and process sequence for an instance of class *Supplier* where some variables used within the optimization model have been included as well

port and an output port for requests, as well as an input and an output port for receiving or for sending products. The input storage is primarily replenished by ordering products from a supplier or another intermediate object. When sending a request the object also determines the transportation mode which should execute the ordered delivery. If no optimization solution is used (which would include the information which transportation mode to use anyway), this is done by using the parameter *StandardTransportMode* which has to be defined in a predefined field within the simulation project. Thus, the user determines the default transportation mode (possibly depending on the ordered product type) which will be transmitted to potential delivery sources within the corresponding sent requests. The same is available for the question where to send a request to, if no optimization solution is used. Accordingly, we included the parameter *StandardProductSource* which has to be provided with the default receiver of the object's requests (again this can be defined as a function which depends on the ordered product type), i.e. the default product source. We incorporated an additional possibility for replenishment which is used in order to avoid empty storages in the first or the last couple of periods. This is done by a special parameter representing the work-in-progress in period 0. When using this parameter one has solely to define the amount of products and the point in time when products should arrive at the outbound inventory of the producer node. Since the arrival of these extra inflows to the storage can be scheduled to any period of the planning horizon it is possible to start simulation runs with full storages throughout the whole network instead of waisting the first couple of periods by waiting until the first regular deliveries have arrived at all nodes. Empty storages at the end of a planning horizon can be avoided by simply using negative values for this parameter which will lead to compensating orders by the affected outbound inventories and therefore to the desired inventory levels. Of course, if this mechanism is not used the solution will most probably be one with low inventory levels in the end of the planning horizon since the optimization model will have no reason to provide solutions with useless inventory levels. The solution will tend to the, in most cases, unrealistic situation of empty storages in the final phase of a planning. Thus, it is quite reasonable to enforce filled storages. A similar mechanism is provided for the inbound inventories of intermediate nodes. This is managed by the *Transport* class and will be mentioned in the respective section.

Ordering policies of intermediate nodes may either be autonomous (e.g., an (s,S)-policy or any user-defined policy) or, in case the solution from an optimization run is available, determined by these. However, we have to distinguish between dispatching orders sent to any connected node, i.e. intermediate nodes or supplier nodes, and internal production or transaction orders which are sent from an intermediate object's outbound inventory to its own inbound inventory. The former start with a request sent through the output port for requests and result in product deliveries entering the object through the input port for products and finally arriving at the input inventory. Products on stock are available for satisfaction of internal production or transaction orders. These internal orders are placed by the output inventory (again this is either autonomous or based on the solution of the optimization model), by sending a request to the inbound inventory. The final outcome are product arrivals in the outbound inventory coming either from the production or the transaction line. Production and transfer have limited capacities and furthermore production is restricted to the availability of raw materials. If any restriction does not allow producing (or transferring) an ordered lot as a whole, it can be split into several batches which are processed at the earliest possible. Non-processed batches or lots are saved in the waiting queue *WaitList* which is checked every time unit for possible reductions. The delay for production and transfer is a user-defined function. It may contain stochastic elements and depend on other parameters like the size of the currently transported batch, the size of the complete lot, or the percentage of capacity consumption during the current activity. Through the input port for requests the module receives orders from other intermediate or customer objects. These are satisfied as far as possible by sending products through the output port for products. Clearly, incoming orders can only be fulfilled according to availability of products on stock. In case of product shortages deliveries can also be split into batches. Unsatisfied orders or batches of orders are inserted into waiting queue *SendingList* which is checked up for possible shortenings after each time unit. Of course, products that are not scheduled for delivery yet are kept on stock for the time being.

In productions objects costs arise for inventory holding (input and output), for production, and for transfer. The cost functions for all of these components may have any user-defined form. Again, they can be defined for each intermediate object separately and may also differ for each product type. Considering fixed holding costs is as well possible as incorporating cost functions which amongst others depend on stock levels, product IDs, points in time, or capacity consumption.

Instances of *Production* have to handle a huge amount of data. First of all they have to get the information about the general network structure from the database and are subsequently provided with their ID. Additionally, intermediate nodes gather data concerning the bill-of-materials, the work-in-progress, and the capacity restrictions. For instances of *Production* we provide the possibility to consider different capacity consumption factors depending on different products. Each object finds this information, which is used for cost calculations and capacity usage evaluations in production, transaction and storage. Assumed that the solution of an optimization run is available and needed for the forthcoming simulation runs they have to be read in as well since the object has to arrange its activities according to this information. To be considered in this context are the transportation, production, and transaction plans. Intermediate nodes are not responsible for the transportation of products between actors of the network but they need to have the information about the currently calculated optimal transportation plan anyway. This is because transports are always initiated by the demander of a delivery. This implies that intermediate nodes need to know the point in time a delivery to themselves should be started according to the calculations of the optimization model in order to transmit the corresponding request to the object being the source of the delivery in time. The plans referring to production and transaction provided by the optimization model are processed the same way. Accordingly, the intermediate's outbound inventories use the information about optimal production and transaction starting times and amounts for transmitting corresponding requests in time. Due to stochastic production, transaction, or transportation delays it is not always possible to follow the schedules provided by the optimization model. This may lead to out-of-stock or other problematic situations. However, allegations from the optimization model are executed best possible.

Furthermore intermediate nodes have an essential task which is to record of some critical parameters, i.e. the estimation of delay parameters. More precisely, they have to measure and process data on delays for production and transaction activities. This, and also the collection of production and transaction costs, is done by dynamic timers as they have already been introduced in Section 4.1. Whenever a lot, or a part of a lot, is released for production or transaction, an instance of the corresponding dynamic timer class, i.e. *ProdDelay* for production activities or *TransDelay* for transaction activities, is created. On expiry, which is determined by the user-defined function for the corresponding time delay, it checks whether a whole lot has been completed or if just a part of a lot has left the production or transaction line. This check-up is done by automatically generating an instance of message class *Lot* and providing it with the relevant information (i.e. original lot size, current lot size, starting time, etc.). In case just parts of a still unfinished lot have been proceeded the timer just reduces the current lot size by the proceeded amount and finally disappears. If a whole lot has been completed it updates the object's production/transaction cost and some parameters needed for the aggregation step where average per unit costs for production and transaction are finally calculated. Additionally, the timer updates some time-referenced data (e.g., total processing time, variance, etc.) by calling the corresponding algorithmic functions in the *Control* class. Of course, instances of *Production* also calculate estimations for their storage cost. These is done completely apart from dynamic timers. Instead they are calculated by a simple accumulation and linearization as it has already been explained for the holding costs of supplier nodes.

Since the simulation model may contain stochastic and nonlinear elements, it is necessary to perform several simulation runs and combine these results. This is usually done by calculating averaged per unit costs. For parameters having a direct influence on the material flow (especially delays), the use of average values would most probably lead to bad results. In about half of the cases the delay would be longer than assumed and this would consequently lead to further delays in subsequent operations. Therefore, it seems reasonable to use, e.g., a 90%-quantile (assuming a normal distribution with estimations of the mean and the variance calculated based on the last simulation runs) for the estimations of such delay parameters. For this purpose the command `setRoundingFactor()` mentioned in Section 4.1 has to be set by filling in a parameter of 0.9. This will result in an overestimation of the delays for the optimization model, because the value is determined such that 90% of the occurred delays are shorter than this value, but it ensures that a smooth material flow through the network is possible. Any other value between 0 and 1 can be handed over instead and will lead to estimations based on the selected quantile and rounded to the next bigger or smaller integer value. Furthermore it seems useful to combine results from the previous iterations with actual ones, in order to enlarge the sample size and to get better estimations of the mean and the variance. This is done with the command `setSmoothingFactor()` which is used to hand over a smoothing factor between 0 and 1. The higher the factor the lower the impact of old values. Therefore, a factor of 1 will lead to the complete neglection of previous estimations. This complete neglection of old values is primarily done during the very beginning of an experiment where the estimations are done based on rather simple autonomous decisions rules and should preferably not be considered for more than one iteration. Together with the object's total cost all these values are written into the database.

In the optimization model intermediate nodes are represented using the following formulations, where the set of intermediate nodes in the network is denoted by J_I:

$$\begin{aligned} TC_i^I = &\sum_{p\in P}\sum_{t=1}^{T} W_i^p(m_i^p(t)) + \sum_{p\in P}\sum_{t=1}^{T} Z_i^p(u_i^p(t)) \\ &+ \sum_{p\in P}\sum_{t=1}^{T} {}^{in}H_i^p({}^{in}l_i^p(t)) + \sum_{p\in P}\sum_{t=1}^{T} {}^{out}H_i^p({}^{out}l_i^p(t)) \qquad \forall i \in J_I \end{aligned} \tag{4.4}$$

$$m_i^p(t) \leq {}^{prod}Cap_i^p(t) \qquad \forall i \in J_I, t = 1, \ldots, T, p \in P \tag{4.5}$$

$$\sum_{p\in P} a_i^p \cdot m_i^p(t) \leq {}^{prod}C_i(t) \qquad \forall i \in J_I, t = 1,\ldots,T \tag{4.6}$$

$$u_i^p(t) \leq {}^{ta}Cap_i^p(t) \qquad \forall i \in J_I, t = 1,\ldots,T, p \in P \tag{4.7}$$

$$\sum_{p\in P} d_i^p \cdot u_i^p(t) \leq {}^{ta}C_i(t) \qquad \forall i \in J_I, t = 1,\ldots,T \tag{4.8}$$

$$\begin{aligned} {}^{in}l_i^p(t) = {}^{in}l_i^p(t-1) + {}^{in}f_i^p(t) - \sum_{p'\in P} \alpha_i^p(p') \cdot m_i^{p'}(t) \\ - u_i^p(t) + r_i^p(t) \qquad \forall i \in J_I, t = 1,\ldots,T, p \in P \end{aligned} \tag{4.9}$$

$$\begin{aligned} {}^{out}l_i^p(t) = {}^{out}l_i^p(t-1) - {}^{out}f_i^p(t) + \chi_{t\geq\delta_i^p} \cdot m_i^p(t-\delta_i^p) \\ + \chi_{t\geq\sigma_i^p} \cdot u_i^p(t-\sigma_i^p) + s_i^p(t) \qquad \forall i \in J_I, t = 1,\ldots,T, p \in P \end{aligned} \tag{4.10}$$

$${}^{in}l_i^p(t) \leq {}^{invin}Cap_i^p(t) \qquad \forall i \in J_I, t = 1,\ldots,T, p \in P \tag{4.11}$$

$$\sum_{p\in P} q_i^p \cdot {}^{in}l_i^p(t) \leq {}^{in}L_i(t) \qquad \forall i \in J_I, t = 1,\ldots,T \tag{4.12}$$

$${}^{out}l_i^p(t) \leq {}^{invout}Cap_i^p(t) \qquad \forall i \in J_I, t = 1,\ldots,T, p \in P \tag{4.13}$$

$$\sum_{p\in P} q_i^p \cdot {}^{out}l_i^p(t) \leq {}^{out}L_i(t) \qquad \forall i \in J_I, t = 1,\ldots,T \tag{4.14}$$

$$m_i^p(t),\ u_i^p(t),\ {}^{in}l_i^p(t),\ {}^{out}l_i^p(t) \geq 0 \qquad \forall i \in J_I, t = 1,\ldots,T, p \in P \tag{4.15}$$

The overall cost of an intermediate node i is represented by TC_i^I as it is shown in Equation (4.4). It is defined by the production cost function $W_i^p(\cdot)$ depending on production amount $m_i^p(t)$ of product p in time period t, the transaction cost function $Z_i^p(\cdot)$ depending on transaction amount $u_i^p(t)$ of product p in time period t, the cost function for the inbound inventory ${}^{in}H_i^p(\cdot)$ depending on inventory level ${}^{in}l_i^p(t)$ of product p in time period t, and finally the cost function for referring to the outbound inventory which is denoted by ${}^{out}H_i^p(\cdot)$ and depends on inventory level ${}^{out}l_i^p(t)$ of product p in time period t. All these functions are open to any user-defined specification. Equations (4.5) and (4.6) assure that the given production capacities ${}^{prod}Cap_i^p(t)$ and ${}^{prod}C_i(t)$ denoting an individual restriction valid for product type p in time period t as well as a global restriction for the cumulated production amount over all products p in time period t, respectively, is not exceed by the according production amount $m_i^p(t)$ of product p in time period t. For restricting the global production volume the production amounts of products p are multiplied by a corresponding factor a_i^p representing the corresponding resource requirements. The compliance of the time-specific capacity restrictions in transaction activities is guaranteed by Equations (4.7) and (4.8), where $u_i^p(t)$ denotes the amount of transacted products p in period t, and ${}^{ta}Cap_i^p(t)$ and ${}^{ta}C_i(t)$ the available capacities in period t for each product p individually and the global upper bound for them all together, respectively. Again, for the global restriction the amounts, in this case the transaction amounts $u_i^p(t)$ are multiplied by a consumption factor for product p which is here denoted by d_i^p. The distinction between individual capacity constraints for each product and global capacity constraints are necessary to cover general situations where product-specific resources as well as common resources are used for production or transaction. Equation (4.9) is an inventory balance equation for the input inventory level ${}^{in}l_i^p(t)$ of product p in time period t. This has to be equal to the according inventory level in the previous period plus the inflow (${}^{in}f_i^p(t)$) of product p sent by all other nodes and arriving in time period t, minus the required raw materials for producing amount $m_i^{p'}(t)$ of product *p'* in period t . Variable $\alpha_i^p(p')$ represents the units of raw material p that are necessary to produce one unit of product *p'*. The right hand side of the inventory balance equation is completed by subtracting the transaction volume $u_i^p(t)$ of product p in time period t as well as by adding some external inflow ($r_i^p(t)$) which may either be deliveries from periods previous to period t or any kind of external inflow arriving from outside the system. However, the products arrive at intermediate node i in time period t. The simulation model knows these extraordinary inflows as well but there they are considered within class *Transport*. Primarily this parameter is used in order to avoid empty storages in the beginning or in the end of the planning horizon. The latter can be avoided by using negative values for this parameter. Equation (4.10) is the inventory balance equation for the outbound inventory and is very similar to the previous one. This time we balance the inventory level ${}^{out}l_i^p(t)$ of product p in period t with the corresponding inventory level in the previous period reduced by the outflow of product p in period t, which is denoted by ${}^{out}f_i^p(t)$. Furthermore we have to consider the production and transfer delays (δ_i^p, σ_i^p). Since the inventory level of product p can only be increased in period t by lots whose production process has been started δ_i^p periods or whose transaction process has been started σ_i^p periods ago. This is formulated by using the indicator functions $\chi_{t \geq \delta_i^p} \cdot m_i^p(t - \delta_i^p)$ and $\chi_{t \geq \sigma_i^p} \cdot u_i^p(t - \sigma_i^p)$, respectively. The previous one becomes 1 as far as the current period t has a higher value than δ_i^p and otherwise is set to 0. Accordingly, the second indicator function becomes 1 if the current period t is higher than σ_i^p and 0 otherwise. This formulation assures that production or transaction lots do not increase the outbound inventory level before they are finished. Finally, we add a parameter representing the already mentioned work-in-progress to the equation's right hand side. Thus, $s_i^p(t)$ considers production or transaction lots that are already on their way during the run-up, i.e. before period 1. Of course, this can also be used to avoid empty inventories in the beginning or the in the end of the planning horizon. Equation 4.11 and Equation 4.12 are used to restrict the stock of the inbound inventory of product p in time period t which is denoted by ${}^{in}l_i^p(t)$. Again, we differ between the individual capacity ${}^{invin}Cap_i^p(t)$ implying an upper bound for each product type p being on stock in time period t and the global value $(inL_i(t))$ referring to the total volume of stocked entities in time period t. The capacity consumption factor, indicating the space used by each entity of a certain product type p, is denoted by q_i^p. Similar we formulate the capacity restrictions for the outbound inventory level ${}^{out}l_i^p(t)$ in Equation (4.13) and Equation (4.14) for product type p in period t. Accordingly, we have the time-specific and product-specific individual capacity constraint which bounds the stock level of product p in period t to a maximum of ${}^{invout}Cap_i^p(t)$, and the accumulated and solely time-specific one considering the consumption factor q_i^p for each product p. These restrictions can be used to model dedicated-storage as well as random-storage policies. Equations (4.15) ensure the non-negativity of production amounts $(m_i^p(t))$, transaction amounts $(u_i^p(t))$, and inventory levels $({}^{in}l_i^p(t), {}^{out}l_i^p(t))$ for product p and time period t.

Figure 4.3 illustrates an instance of *Production* and its activities within a simulation run. Again, some variables used within the optimization model's formulation have been included as well.

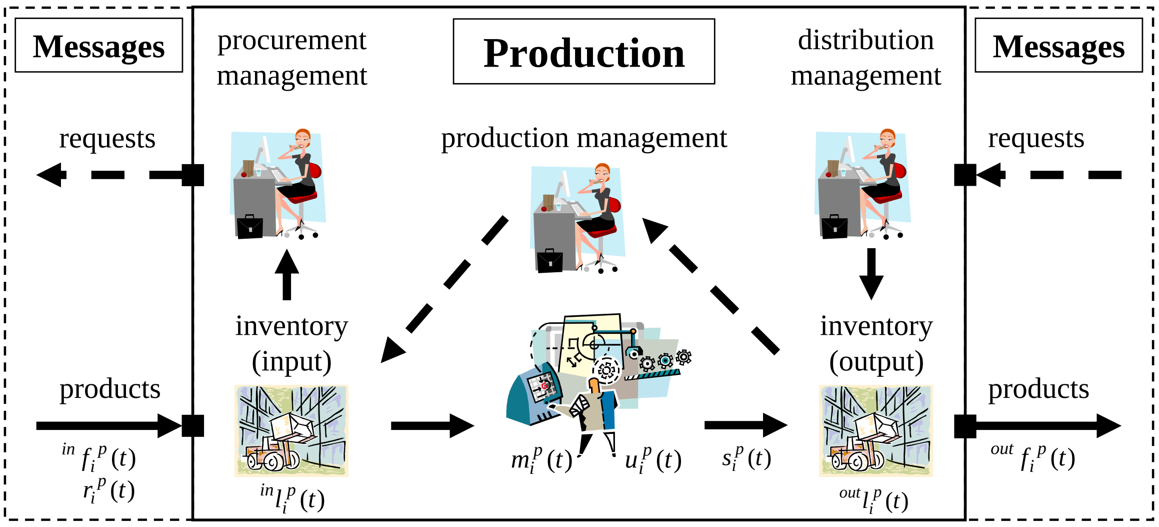

Figure 4.3.: Logical structure and process sequence for an instance of class *Production* where some variables of the optimization model's formulation have been included as well

Instances of *Production* are connected to the optimization model by all cost factors of Equation (4.4), i.e. production, transaction, inventory holding cost, and by production and transaction delays (δ_i^p and σ_i^p). In the simulation model these can have any user-defined functional form possibly containing stochastic and nonlinear elements. On the other hand simulation and optimization are connected by results provided by the optimization model, i.e. production plans (m_i^p) and transaction plans (u_i^p).

4.4. The *Customer* class

This class is used to represent the customers' behaviour. Objects of *Customer* dispose of an input port for products and an output port for sending requests to intermediate nodes. Due to stochastic features within the simulation it is not possible to plan deliveries exactly. Therefore, the customer has an input inventory for keeping products on stock. According to a given demand table the customer orders products which on arrival are either used to promptly satisfy the demand or are stocked in the inventory. The inventory may account for shortages as well as for oversupply situations, in both cases high penalty costs occur. If no results from the optimization model are available the requests are sent according to the given demand. For this purpose the user has to define a *StandardTransportDelay* which is the amount of time periods one expects a delivery to last. Clearly here again it is essential to determine a *StandardProductSource* and a *StandardTransportMode* as it has already been explained in the previous section. These parameters can be defined as functions depending on the ordered product type and have to be inserted into the provided field of the project. The resulting expected delivery delay determines the point in time an object will transmit the corresponding request, which will consequently be the period which is exactly *StandardTransportDelay* periods prior to the period where the demand for the requested product will occur. In case the solution of the optimization model is taken into account within the simulation the requests are transmitted according to the transportation plan provided by the optimization model. This is because deliveries are initiated by the object demanding it, as it has already been explained in the previous section. Thus, instances of *Customer* need to know the points in time transportation lots should be launched by the sending object in order to transmit the corresponding request in time.

Instances of *Customer* solely consider costs arising in their input inventories. Here we distinguish between costs due to a positive inventory level, i.e. oversupply, and those caused by a negative inventory status, i.e. unsatisfied demand. Both can have any user-defined functional form taking fixed costs, product types, stock levels (of a certain product type or cumulated for all product types), or points in time into account. According to the formulation of the optimization model customers are interpreted as just-in-time actors and therefore it is prohibited to cause oversupplies at customer objects within the optimization. Nevertheless we weakened this just-in-time assumption within the simulation model, and thus allow premature deliveries there. The reason is that otherwise we would end up with the unrealistic situation that early deliveries caused by stochastic transportation times do not influence the total cost at all. So we avoid this scenario by incorporating high inventory costs for handling premature deliveries.

So the estimated inventory cost which will be reported by the customer objects to the optimization model are always based on costs which have arised due to undersupply of a customer. Costs occurring due to a positive stock level do not have a direct impact on the forthcoming optimization solution. Although they are not explicitly considered within the optimization model they may have an implicit influence on the result anyway.

Data processing activities in this object class can be explained quickly. The usual initialization of objects is followed by the import of two essential sets of parameters. The first set consists of the given demand values, determining the amounts and periods for stock reducing movements in an object's inventory. Furthermore customer nodes import a part of the optimization's results, i.e. the transportation plans for deliveries between intermediate nodes and themselves. These are needed for initiating deliveries on time, as it has already been explained for the intermediate nodes. Customer objects cumulate arising costs for unsatisfied demand, i.e. for inventory shortages. Based on these the corresponding per unit costs for backorders for each product type are calculated by average determination and are exported to the database together with the object's total cost. Within the optimization model these values are going to be interpreted as penalty costs for backorders per period and per unavailable product.

The optimization model for the customers' behaviour is formulated as follows (the set of intermediate nodes in the network is denoted by J_C):

$$TC_i^C = \sum_{p \in P} \sum_{t=1}^{T} R_i^p({}^{in}b_i^p(t)) \qquad \forall i \in J_C \tag{4.16}$$

$$\begin{aligned} {}^{in}l_i^p(t) - {}^{in}b_i^p(t) &= {}^{in}l_i^p(t-1) - {}^{in}b_i^p(t-1) + {}^{in}f_i^p(t) \\ &- D_i^p(t) + r_i^p(t) \qquad \forall i \in J_C, t = 1, \dots, T, p \in P \end{aligned} \tag{4.17}$$

$${}^{in}l_i^p(t) = 0 \qquad \forall i \in J_C, t = 1, \dots, T, p \in P \tag{4.18}$$

In Equation (4.16) the formulation for costs arising at customer nodes is given. Here we use cost function $R_i^p(\cdot)$ depicting the penalty costs for backorders depending on the amount of backorders for product p in time period t which are denoted by ${}^{in}b_i^p(t)$. These backorders are also the essential component of the customer's inventory balance equation (4.17). Here the inventory level ${}^{in}l_i^p(t)$ of product p in time period t reduced by the open backorders ${}^{in}b_i^p(t)$ for product p in period t must be equal to the corresponding inventory level ${}^{in}l_i^p(t-1)$ of the previous period minus the open backorders of the previous period (${}^{in}b_i^p(t-1)$) plus ${}^{in}f_i^p(t)$, which are the incoming units of product p in time period t. As already mentioned above the stock level is decreasing with occurring demand. Therefore, we have to subtract the demand $D_i^p(t)$ for product p in time period t from the right hand side of the inventory balance equation. Finally we add inflows arriving from outside the system.

Hence, external deliveries or transportation lots having been sent away from an intermediate node prior to the first period of the planning horizon are considered by $r_i^p(t)$, denoting the amount of product p arriving in period t. Within the simulation model such extraordinary deliveries are implemented in class *Transport*. As already mentioned above we assume, within the optimization model, all customers as just-in-time customers. Therefore, Constraint (4.18) ensures that no oversupply (positive stock level) is possible, i.e. inventory level ${}^{in}l_i^p(t)$ has to be equal to 0. This just-in-time assumption may, depending on the given setting, just as well be dropped and holding costs for positive stock, as we consider them within the simulation model, may be included.

Figure 4.4 illustrates an instance of *Customer* used within the simulation. Additionally some of the optimization model's variables have been added.

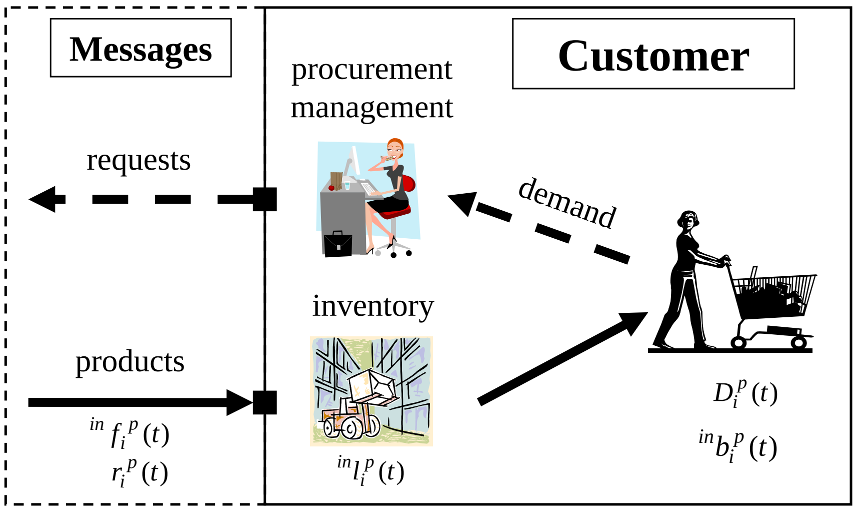

Figure 4.4.: Logical structure and process sequence for an instance of class *Customer*; some variables used within the optimization model have been added as well

The direct connection between the simulation model and its representation as an optimization model is solely the estimated penalty cost factor for delayed deliveries. This is the only parameter that is calculated within the simulation and then handed over to the optimization model.

4.5. The *Transport* class

This object class is used for simulating the material flow through the network. Each transport node is connected to exactly one product source, i.e. supplier objects or intermediate nodes, and exactly one destination node, i.e. intermediate nodes or customers. Objects of *Transport* receive products from their source through their input ports and, according to capacity availability, either hold them in their queue

WaitingList or proceed them to their output ports. If capacities are short it is also possible to split shipments and send them by and by. This again is organized by a static timer checking up the queue periodically. Transports are subject to a time delay, which may be stochastic and may depend on other parameters like the transported product type, the size of the current load, the original order amount of the current shipment (in case an order has had to be split in partial deliveries), the type of the used transport mode, current capacity consumption, or the current time period. The corresponding function determining the delays is again user-defined. It has already been mentioned above that consider deliveries from outside the system and, which actually is the main intention for their application, for avoiding empty inbound storages at the beginning or at the end of the planning horizon. Thus, the according parameters represent deliveries which have already been sent away before the first period of the planning horizon. Their arrival can be scheduled for any time period of the planning horizon. Defining them as less than zero consequently leads to corresponding extra orders from the supplied node. Such it becomes possible to avoid empty storages in the end of the planning horizon.

There are two types of costs arising in transportation nodes: transportation cost and waiting cost. The latter occur in case products have entered the object but can, due to a lack of capacity, not be proceeded in the current time period, i.e. all products appearing in *WaitingList.* Both transportation cost and waiting cost may be defined as functions depending on the currently loaded product type, amount of loaded products, size of original amount (in case an order had to be split in partial deliveries), point in time, transportation mode, and capacity consumption.

Business of a transport object starts with the usual initialization procedure. This is followed by the import of some essential parameters. These are available capacities, per unit usage of capacities, and the already mentioned parameters for extraordinary deliveries. There are no values provided by the optimization model for instances of this class. Data reporting is limited to the per unit cost for transportation and estimations for transportation delays. The former is straightforward: occurring costs, i.e. cost for transport and waiting costs, determined by the user-defined functions, are accumulated and finally divided by the total amount of processed products. Estimating transportation times is more complex. Time measurement is done by the dynamic timer class *TransportDelayTimer.* An instances of it is created automatically whenever a transport is started. Simultaneously the object's cumulated waiting cost is updated. On the timer's expiry, which is due to the user-defined function for transportation delays, an update of the object's total transportation cost and amount as well as time-referenced data (transportation time and variance) is executed. The latter is done just in case a complete transportation order or the final split of a complete transportation order has been finished. Finally the timer disappears. The drawback of using average values for delay estimations has already been mentioned above. Due to these we here again generate estimations based on quantiles, assuming a normal distribution with estimated mean and variance. For this purpose the algorithmic functions implemented in class *SCMControl* are used. Again, the command `setSmoothingFactor()` is used for smoothing estimations according to previous ones. Finally computations for delays and cost parameters and the object's total cost are exported to the database.

Transportation activities are controlled by the following equations of the optimization model (V represents the set of transportation modes):

$$TC_{ij}^F = \sum_{p \in P} \sum_{t=1}^{T} \sum_{v \in V} {}^vC_{ij}^p({}^vx_{ij}^p(t)) \qquad \forall i \in J_S \cup J_I, j \in J_I \cup J_C \tag{4.19}$$

$${}^vx_{ij}^p(t) \leq {}^vCap_{ij}^p(t) \qquad \forall i \in J_S \cup J_I, j \in J_I \cup J_C, t = 1, \dots, T, p \in P, v \in V \tag{4.20}$$

$$\sum_{p \in P} {}^vg^p \cdot {}^vx_{ij}^p(t) \leq {}^vC_{ij}(t) \qquad \forall i \in J_S \cup J_I, j \in J_I \cup J_C, t = 1, \dots, T, v \in V \tag{4.21}$$

$${}^{in}f_j^p(t) = \sum_{\substack{i \in J_S \cup J_I \\ {}^v\tau_{ij} < t}} \sum_{v \in V} {}^vx_{ij}^p(t - {}^v\tau_{ij}) \qquad \forall j \in J_I \cup J_C, p \in P, t = 1, \dots, T \tag{4.22}$$

$${}^{out}f_j^p(t) = \sum_{i \in J_I \cup J_C} \sum_{v \in V} {}^vx_{ji}^p(t) \qquad \forall j \in J_S \cup J_I, p \in P, t = 1, \dots, T \tag{4.23}$$

$${}^vx_{ij}^p(t) \geq 0 \qquad \forall i \in J_S \cup J_I, p \in P, t = 1, \dots, T, v \in V \tag{4.24}$$

The total transportation cost TC_{ij}^F for deliveries from node i to node j is calculated based on the user-defined function ${}^vC_{ij}^p(\cdot)$ which depend on the transportation amounts (${}^vx_{ij}^p(t)$) of product p in period t between nodes i and j with transportation mode v. This is defined in Equation (4.19). Constraint (4.20) limits the transportation amount ${}^vx_{ij}^p(t)$ to a capacity limit valid for product p, transportation mode v, transportation leg ij, and time period t, denoted by ${}^vCap_{ij}^p(t)$. The capacity restriction for all products together is given in Equation (4.21) and ensures that transportation amount ${}^vx_{ij}^p(t)$ times a product-specific and transportation-mode-specific capacity consumption factor ${}^vg^p$ is less or equal to the overall capacity limitation ${}^vC_{ij}(t)$ for transportation mode v on transportation leg ij in time period t. Within the optimization model for intermediate nodes we introduced the auxiliary variables ${}^{in}f_j^p(t)$ and ${}^{out}f_j^p(t)$. These are also used within the transportation-referenced part of the optimization model. Equation (4.22) ensures that the product inflow (${}^{in}f_j^p(t)$) of product p arriving in time period t at node j, which can either be an intermediate or a customer node, is equal to the sum of deliveries sent by all nodes i, i.e. supplier or intermediate nodes, that have a time-specific transportation distance ${}^v\tau_{ij}$ to node j when using transportation mode v of less than the the current time period t.

Thus, the auxiliary product inflow variable ${}^{in}f_j^p(t)$ has to be equal to all deliveries sent away ${}^v\tau_{ij}$ time periods ago (${}^vx_{ij}^p(t - {}^v\tau_{ij})$). A similar equation is incorporated for the product outflow (4.23). Here it is ensured that outgoing deliveries of product p that are sent in any period t by node j, which can either be a supplier or an intermediate node, is equal to the sum of transported lots of product p that are sent from node j to any node i, which in turn can either be an intermediate or a customer node, starting in period t and transported by any transportation mode v. So the product outflow variable ${}^{out}f_j^p(t)$ has to be equal to all deliveries starting at node i in period t (${}^vx_{ij}^p(t)$). Finally, Equation (4.24) ensures that delivery amount ${}^vx_{ij}^p(t)$ of any product type p sent with using any transportation mode v in any time period t from node i to node j is larger or equal to zero. Since deliveries can solely be transmitted starting at supplier or intermediate nodes this equation has not to be valid for customer nodes.

An exemplary instance of *Transport* as it is used within the simulation is displayed in Figure 4.5. Again, the inclusion of some variables from the optimization model intends to give a better insight into the parallels of simulation and optimization.

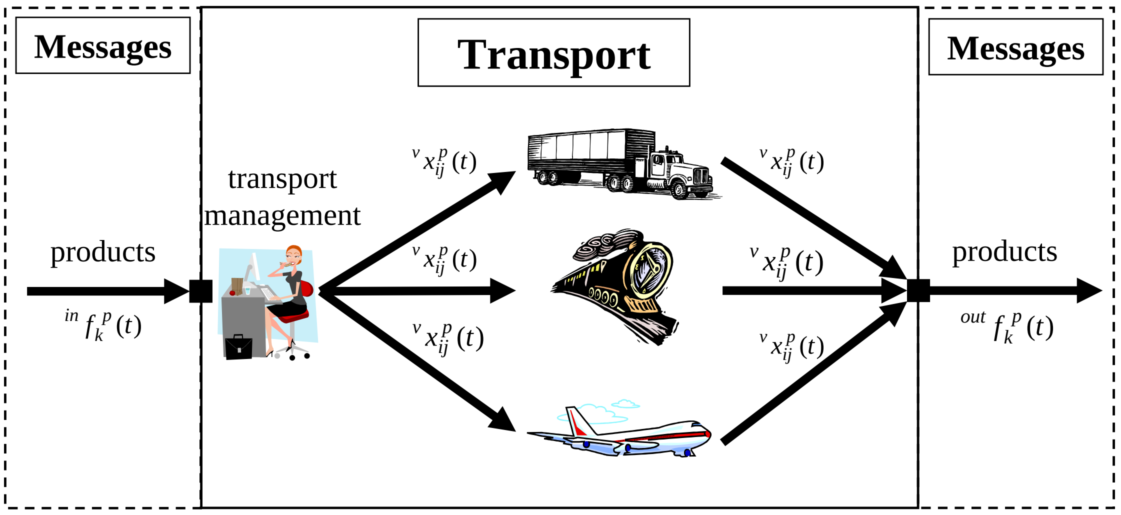

Figure 4.5.: Logical structure and process sequence for an instance of class *Transport* as it is used within the simulation; inclusion of some variables as they are used in the optimization model intends to give a better insight

The connection between the simulation and the optimization model for instances of *Transport* is established by the transportation cost function given in Equation (4.19) and the transportation delay ${}^v\tau_{ij}$ given in Equation 4.22 on the one hand, and the transportation amounts on the other hand. As it has already been explained transportation amounts ${}^vx_{ij}^p(t)$ are used to define ordering schemes for intermediate and customer objects for the simulation model. All other variables, including those appearing in Figure 4.5, are not used for direct information exchange between instances of *Transport* and the optimization model. Indeed, resulting values of variable

${}^{v}x_{ij}^{p}(t)$, representing the transportation amounts, are very well used directly by the simulation model, but explicitly not by transportation objects but by intermediate and customer objects.

4.6. The optimization model

In the previous section the components of the optimization model's general formulation have been introduced. Each of them headed by the definition of the component's total cost. In order to assemble these components we simply define the objective function of the global model, i.e. for the whole supply chain network. The global objective is the minimization of total costs referring to all participants of the network:

$$min \quad \sum_{i \in J_S} TC_i^S + \sum_{i \in J_I} TC_i^I + \sum_{i \in J_C} TC_i^C + \sum_{i \in J_S \cup J_I} \sum_{j \in J_I \cup J_C} TC_{ij}^F \tag{4.25}$$

So the total cost of a network simply is the sum of the suppliers' total cost (TC_i^S), the intermediates' total cost (TC_i^I), the customers' total cost (TC_i^C), and the transportation modes' total cost (TC_i^F). Assuming that Equation (4.25) exclusively consists of pure linear functions it is possible to reformulate it as follows ($J = J_S \cup J_I \cup J_C$):

$$\begin{aligned} min \quad & \sum_{i \in J_S \cup J_I} \sum_{p \in P} \sum_{t=1}^{T} {}^{out}h_i^p \cdot {}^{out}l_i^p(t) + \sum_{i \in J_I} \sum_{p \in P} \sum_{t=1}^{T} {}^{in}h_i^p \cdot {}^{in}l_i^p(t) \\ & + \sum_{i \in J_I} \sum_{p \in P} \sum_{t=1}^{T} w_i^p \cdot m_i^p(t) + \sum_{i \in J_I} \sum_{p \in P} \sum_{t=1}^{T} z_i^p \cdot u_i^p(t) \\ & + \sum_{i \in J_C} \sum_{p \in P} \sum_{t=1}^{T} \rho_i^p \cdot {}^{in}b_i^p(t) + \sum_{ij \in J} \sum_{p \in P} \sum_{t=1}^{T} \sum_{v \in V} {}^{v}c_{ij}^p \cdot {}^{v}x_{ij}^p(t) \end{aligned} \tag{4.26}$$

For the linear formulation of the suppliers' total cost we summarize the outbound inventory levels (${}^{out}l_i^p$) of all supplier nodes i, of all products p, and in all time periods t while multiplying each of them with a cost factor ${}^{out}h_i^p$. The same is done for calculating the inbound and outbound inventory costs at the intermediate nodes. For the latter we consider inventory level ${}^{in}l_i^p$ and cost factor ${}^{in}h_i^p$. Total production cost is determined by production amounts $m_i^p(t)$, which are multiplied with the cost factor w_i^p and cumulated over all production nodes i, products p, and time periods t. Similarly we calculate total transaction cost. Here we use transaction amounts $u_i^p(t)$ times the transaction cost factor z_i^p. And finally the total transportation cost is considered by transportation amounts ${}^{v}x_{ij}^{p}(t)$ times transportation cost factor ${}^{v}c_{ij}^{p}$ and summed up over all possible transportation legs ij, all transportation modes v, all delivered products p, and all time periods t. The objective function is subject to the following side constraints which have already been described in the previous sections (please see Appendix A for a precise list describing the complete notation):

$$ {}^{out}l_i^p(t) = {}^{out}l_i^p(t-1) - {}^{out}f_i^p(t) + S_i^p(t) \qquad \forall i \in J_S, t = 1, \ldots, T, p \in P \tag{4.27}$$

$$ m_i^p(t) \leq {}^{prod}Cap_i^p(t) \qquad \forall i \in J_I, t = 1, \ldots, T, p \in P \tag{4.28}$$

$$ \sum_{p \in P} a_i^p \cdot m_i^p(t) \leq {}^{prod}C_i(t) \qquad \forall i \in J_I, t = 1, \ldots, T \tag{4.29}$$

$$ u_i^p(t) \leq {}^{ta}Cap_i^p(t) \qquad \forall i \in J_I, t = 1, \ldots, T, p \in P \tag{4.30}$$

$$ \sum_{p \in P} d_i^p \cdot u_i^p(t) \leq {}^{ta}C_i(t) \qquad \forall i \in J_I, t = 1, \ldots, T \tag{4.31}$$

$$ \begin{aligned} {}^{in}l_i^p(t) = {}^{in}l_i^p(t-1) + {}^{in}f_i^p(t) - \sum_{p' \in P} \alpha_i^p(p') \cdot m_i^{p'}(t) \\ - u_i^p(t) + r_i^p(t) \qquad \forall i \in J_I, t = 1, \ldots, T, p \in P \end{aligned} \tag{4.32}$$

$$ \begin{aligned} {}^{out}l_i^p(t) = {}^{out}l_i^p(t-1) - {}^{out}f_i^p(t) + \chi_{t \geq \delta_i^p} \cdot m_i^p(t - \delta_i^p) \\ + \chi_{t \geq \sigma_i^p} \cdot u_i^p(t - \sigma_i^p) + s_i^p(t) \qquad \forall i \in J_I, t = 1, \ldots, T, p \in P \end{aligned} \tag{4.33}$$

$$ {}^{in}l_i^p(t) \leq {}^{invin}Cap_i^p(t) \qquad \forall i \in J_I, t = 1, \ldots, T, p \in P \tag{4.34}$$

$$\sum_{p\in P} q_i^p \cdot {}^{in}l_i^p(t) \leq {}^{in}L_i(t) \qquad \forall i \in J_I, t = 1,\dots,T \tag{4.35}$$

$${}^{out}l_i^p(t) \leq {}^{invout}Cap_i^p(t) \qquad \forall i \in J_I, t = 1,\dots,T, p \in P \tag{4.36}$$

$$\sum_{p\in P} q_i^p \cdot {}^{out}l_i^p(t) \leq {}^{out}L_i(t) \qquad \forall i \in J_I, t = 1,\dots,T \tag{4.37}$$

$$\begin{aligned} &{}^{in}l_i^p(t) - {}^{in}b_i^p(t) = {}^{in}l_i^p(t-1) - {}^{in}b_i^p(t-1) + {}^{in}f_i^p(t) \\ &- D_i^p(t) + r_i^p(t) \qquad \forall i \in J_C, t = 1,\dots,T, p \in P \end{aligned} \tag{4.38}$$

$${}^{v}x_{ij}^p(t) \leq {}^{v}Cap_{ij}^p(t) \qquad \forall i \in J_S \cup J_I, j \in J_I \cup J_C, t = 1,\dots,T, p \in P, v \in V \tag{4.39}$$

$$\sum_{p\in P} {}^{v}g^p \cdot {}^{v}x_{ij}^p(t) \leq {}^{v}C_{ij}(t) \qquad \forall i \in J_S \cup J_I, j \in J_I \cup J_C, t = 1,\dots,T, v \in V \tag{4.40}$$

$${}^{in}f_j^p(t) = \sum_{\substack{i\in J_S\cup J_I \\ {}^{v}\tau_{ij}<t}} \sum_{v\in V} {}^{v}x_{ij}^p(t - {}^{v}\tau_{ij}) \qquad \forall j \in J_I \cup J_C, p \in P, t = 1,\dots,T \tag{4.41}$$

$${}^{out}f_j^p(t) = \sum_{i\in J_I\cup J_C} \sum_{v\in V} {}^{v}x_{ji}^p(t) \qquad \forall j \in J_S \cup J_I, p \in P, t = 1,\dots,T \tag{4.42}$$

$$ {}^{v}x_{ij}^{p}(t) \geq 0 \qquad \forall i \in J_S \cup J_I, p \in P, t = 1, \ldots, T, v \in V \tag{4.43} $$

$$ {}^{out}l_i^p(t) \geq 0 \qquad \forall i \in J_S, t = 1, \ldots, T, p \in P \tag{4.44} $$

$$ m_i^p(t),\ u_i^p(t),\ {}^{in}l_i^p(t),\ {}^{out}l_i^p(t) \geq 0 \qquad \forall i \in J_I, t = 1, \ldots, T, p \in P \tag{4.45} $$

$$ {}^{in}l_i^p(t) = 0 \qquad \forall i \in J_C, t = 1, \ldots, T, p \in P \tag{4.46} $$

This pure linear program can be solved easily with any standard LP solver within very short time. The general formulation provides the possibility to adapt the optimization model in consideration of additional features like, e.g., fixed-costs in transportation, binary decisions, step functions, etc. These adaptions lead to a mixed-integer formulation and thus to an increase of computation time. Some examples on this issue are presented and discussed in Chapter 6.

5. Implementing *SimOpt* experiments

5.1. Creating the simulation model

The implementation of a simulation model based on *SCMLib* starts with the creation of a new instance of *ActiveObject* which has to be included in the destined project. This instance is primarily used as a kind of drawing pad where other class objects are going to be inserted and arranged. Hence, required objects, depending on the given network setting, are dragged from the library's tab and dropped on the drawing pad. Here they can be arranged properly and are then interlinked by the insertion of connection lines. In fact, one has to connect any two objects intended for data exchange. Connection lines always go from an input to an output port, or vice versa. Information flow is maintained by a direct connection line between any two nodes that should exchange information, i.e. that send or receive requests. Of course, one and the same object can be connected to more than one object. Enabling material flow is slightly more complex since a particular connection object, i.e. an instance of class *Transport*, has to be inserted referring to any possible transportation leg. This is done by drawing a connection line from the output port for products of a supplying object, i.e. a supplier or an intermediate object, to the product input port of the provided transportation object. Finally, the output port of this transportation object is connected by an additional line to the product input port of the receiving object, i.e. an intermediate node or a customer. Due to the fact that in the given implementation information channels are exclusively used for transmission of requests, i.e. product orders, it has to be ensured that each direct connection line has a corresponding indirect connection, i.e. a connection via a transport object. In case an object is supplied by more than one object one certainly has to provide a transportation object for each supplying object. This is essential because transportation objects initialize themselves by using the unique IDs of their source and their destination, respectively. Thus, having more than one source or destination is not possible for transportation objects.

Finally, an instance of *SCMControlClass* has to be included and adjusted according to the explanations in Section 4.1. For calibration of simulation configurations (e.g., speed scale, stopping criterion, seed of random number generator, etc.) the software automatically generates an item *Simulation* which is open for user-defined adaptions. Since setting the random number generator is crucial whenever comparable simulation runs are to be conducted it is worth mentioning here that the seed is not necessarily defined in the project itself but it can also be passed through the command line as well. This is quite convenient whenever multiple simulation runs have to be batched with consistently changing seeds. Furthermore the user-defined cost and delay functions for each object have to be specified as it has been described above. Figure 5.1 illustrates the combination of objects derived from *SCMLib's* classes in order to simulate a simple supply chain with one supplier, one intermediate player, and one customer. Material flow and information flow are represented by arcs with solid and dashed lines, respectively.

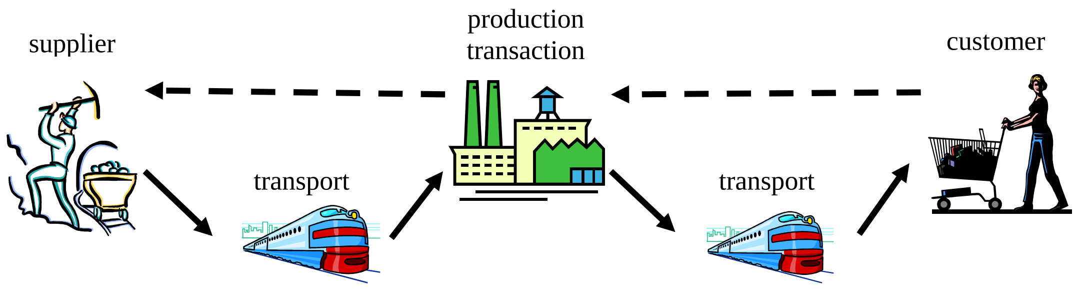

Figure 5.1.: Illustration of an object configuration for a simple supply chain consisting of 1 supplier, 1 intermediate node, and 1 customer; dashed lines indicate information flow and solid lines material flow

In case one has to simulate large supply chain network problems it may happen that due to the huge number of necessary objects (for each transportation link an appropriate instance of *Transport* has to be included), complexity and memory usage may lead to hardly solvable problems. Unfortunately it is apparently not possible to keep AnyLogic from automatically reserving quite a huge amount of memory for each single object of the experiment, e.g., for visualization (even if no visualization is intended). This certainly directly leads to a lack of memory or unacceptable processing times. Even the pure implementation of a large experiment without any computations may turn out to be a very time-consuming task. Especially using our model formulation, which furthermore includes a large number of memory-consuming variables with multiple indices, not even the possibility to increase the Java heap size provides sufficient improvement. Therefore, we implemented another class which is used to divide large simulation models into some smaller parts. It implicitly decreases memory usage considerably. This class is called *SCMContainerClass*. An instance of this class is something like a container that is used in order to accommodate other objects. A container has an input and an output port for requests as well as an input and an output port for products. A group of objects is packed into a container and communicates or interchanges with nodes outside the container through the input and output ports. The content of the container is displaced to a separate chart. This eases problems occurring due to lack of memory.

Under certain conditions, e.g., a huge number of homogeneous transportation objects, using containers provides the possibility to reduce the number of items in a network because in it is no longer necessary to insert a particular transportation object between any couple of objects that are intended for product interchange. Still each object packed into a container needs its own transportation object for sending or receiving products, but this single transportation object can be used for product exchange with an indefinite number of objects positioned outside the container. In fact, such a transportation item operates as representative for a indefinite number of replications of itself. All of them are provided automatically with their unique ID but beside that they have to have homogeneous characteristics regarding cost and delay functions. This way the number of transportation items, i.e. objects and connection lines, can be reduced considerably and the simulation model becomes much more overviewable. Applications of this class will be presented in Part III.

5.2. *SimOpt's* logical configuration

In this section we basically are going to have insight into the conceptional design of *SimOpt*. The basic idea of simulation and optimization exchanging information has already been introduced in Chapter 3. Since the two types of models have been described in detail in the previous sections we are now going to add some information on the connecting element between them: the database which is used for storing all necessary information.

As the connecting element we decided to use an *MS Access* database using the Open Database Connectivity (ODBC) interface. In the beginning of a *SimOpt* experiment it contains a number of predefined parameters (e.g., network structure, capacities, etc.) which are retrieved by both simulation and optimization model. During the course of an experiment the optimization model inserts selected parts of its results into the database. These are then imported by the simulation model and will be taken into account for forthcoming simulation runs. In turn the simulation model exports its parameter estimations to the database that will consequently be the basis for the next optimization run, and so on.

Having described the basic components of *SimOpt* just the overall control mechanism needs to be outlined. *SimOpt* is controlled by the simulation model. In fact, the simulation model is designed as the master process initiating data communication and calling the LP/MIP solver. See Figure 5.2 for a schematic illustration of the logical connection between simulation and optimization focusing on data exchange activities.

To initiate the optimization process in our system and in order to get basic information about the mean and the variances of the parameters (e.g., estimated transportation cost per unit, estimated production delays, etc.) a few simulation runs are performed. Missing decision rules needed in order to operate the supply chain network, which, in later iterations, are generated using the results of the optimization model, are substituted by autonomous decision rules (like, e.g., the already mentioned (s,S)-policy for replenishment). The results of each simulation run (e.g., estimated transportation delays, per unit production costs, etc.) are stored

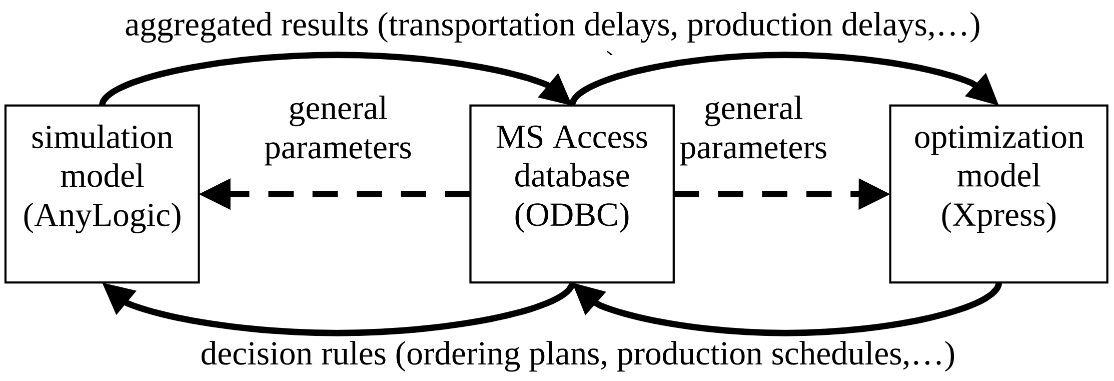
aggregated results (transportation delays, production delays,…)
simulation model (AnyLogic)
general parameters
MS Access database (ODBC)
general parameters
optimization model (Xpress)
decision rules (ordering plans, production schedules,…)

Figure 5.2.: This scheme shows the data exchange between the simulation and the optimization model via the intermediate database

and after the last run the new mean costs and newly estimated critical parameters, e.g., delays, based on mean and variance are exported to the database. These first simulation runs are only necessary to generate initial parameter values for the optimization model, but their results will be ignored for the exponential smoothing in further simulation iterations. Ignoring them is a reasonable procedure in order to avoid unwanted biasing effects caused by the probably bad results caused by the application of simple autonomous decision rules. Nevertheless they are exported to the database since they are going to be used for the first optimization run. Consequently, the next task for the simulation model is to start the optimization model. The latter loads the predefined data and the simulation results from the database, computes the optimal solution based on these and stores its results in the database. Then the next five simulation runs are executed using now the currently computed results from the optimization model, i.e. new ordering and delivery schemes etc., leading to an update of the parameter estimations and of the objective value. This procedure is repeated iteratively until a stable solution is reached, i.e. until the objective and the estimated parameters do not change anymore. It has already been mentioned that *SimOpt* is the collective term for *SimLP*, in case the simulation is coupled with a pure linear optimization model, and *SimMIP*, in case the optimization is based on a mixed-integer formulation. The pseudo code for a *SimOpt* experiment can be summarized as follows:

```
SimOpt:
Load necessary simulation parameters from the database
Perform a few simulation runs using autonomous decision rules
Aggregate results and store them in the database
while stopping criteria are not met
      Load aggregated parameters into LP/MIP solver
      Solve the optimization model
      Write results to the database
      Load new information into simulation model
      Perform simulation runs using the new information
      Aggregate results and store them in the database
end-while
```

Concerning the results coming from the optimization model we wish to add that there of course are several possible ways to use them within the simulation model. The quite simple method that we apply here is to interpret the transportation-, production-, and transfer-specific results as new ordering plans. The precise procedure has been explained in 4. More complex processing of the optimization's results would be, e.g., to use information provided by a sensitivity analysis (dual variables, reduced costs) for determining some critical parameters. Observation of these parameters during the simulation runs and, in case the parameters reach a certain threshold, a proper adaption of the provided ordering plans seems to be even more reasonable. However, for the currently available test results we used the firstly described simple interpretation approach for the solution provided by the optimization model. The analysis of more complex decision rules goes beyond the scope of this thesis and might by a subject for further research (cf. Chapter 13).

6. Empirical Tests

In this chapter we are going to investigate the following research questions:

- Does this method converge in practice for realistic test cases?
- If we can observe convergence, is the result optimal or at least a good approximation?
- Is this method advantageous compared with traditional planning methods?

In order to answer these questions we conducteded a number of numerical tests using a set of self-prepared small instances. This chapter starts with some information concerning solution convergence between simulation and optimization. It will go on with the analysis of test runs facing both deterministic and stochastic problems. The corresponding *SimOpt* results will be compared to those found by a classical exact solution approach in order to give a statement on solution quality of *SimOpt*. The chapter concludes with reports on experiments based on larger tests instances. For these we increased the problem size such that an exact solution cannot be found in reasonable time. By means of these larger instances we tried to find the best setting concerning the quantile for estimating delay parameters.

6.1. Solution convergence

Although it is not possible to prove general convergence we observed fast convergence for all our test instances where we faced realistic cost structures. For all these the objective values converged to a stable value after only a few iterations. Figure 6.1 shows a typical course of objective values of both simulation and optimization. This exemplary illustration is the outcome of test instance D1-L that will be described in detail in Section 6.3. Here it simply serves as graphical demonstration of an usual solution development within a *SimOpt* experiment.

In general we start with the simulation model using autonomous rules for replenishing the inventories. Since we start with all inventories empty, it takes a long time, until orders can be fulfilled. This causes high penalty cost, very long lead times, and consequently overestimations of transportation and production delays. So the first solutions of both simulation and LP model lead to very high total costs mainly consisting of penalty costs for late (or even no) deliveries. Now the simulation model ends up with an improved solution since it uses the delivery plans provided by the optimization model. Due to the fact that the solution of the optimization model

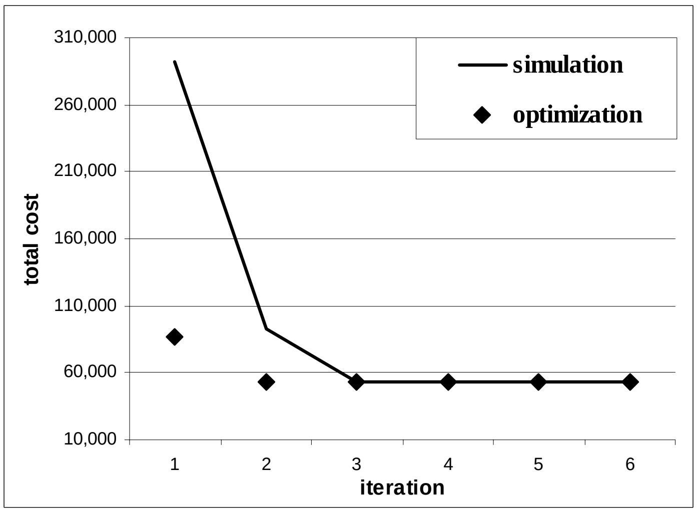

Figure 6.1.: Course of objective values of the optimization model and the simulation model for each iteration of a deterministic *SimLP* experiment considering fixed costs for production, transfer, and transport

causes a somehow synchronized material flow the measured delays are much smaller now and consequently costs are decreasing. After three iterations the simulation and the linear model have converged to the same solution which stays unchanged for the remaining iterations.

As already mentioned above convergence cannot be guaranteed. For exemplification we constructed a fictive setting including nonlinear costs in transportation where *SimLP* gets trapped in a cycle and is not able to find a stable solution. The structure of the example is similar to that used for the previous tests. Again, we took a very small supply chain consisting of one supplier, one intermediate, and one customer. This test instance considers only one product which is to be transacted at the intermediate node. Transaction lasts one period. But for this experiment we provide two different transportation modes, a slow and a fast one, connecting the participants. Customer demand of ten units occurs in the 5^{th} of total six periods. If the slow transportation mode is used, the products will arrive in the 6^{th} period and penalty cost will occur. If the fast mode is used the products arrive on time, i.e. in the 5^{th} period, at the customer. Due to capacity limitations only 5 products can be sent via the fast mode and eight units via the slow mode. The cost for the fast transportation is a fixed value for the whole lot assumed that there are at most four units loaded. This fixed value is quadrupled in case an additional 5^{th} unit is transported. After a few iterations *SimLP* comes to the solution that both modes, the slow and the fast, should deliver 5 products. Obviously the estimated value of customer's per unit penalty cost at that time is higher than the per unit transportation cost of the fast mode. In the next iteration this plan is executed by the simulation model resulting in a rise of the estimated per unit transportation cost for the fast mode since the critical load of 5 products has been reached which leads to a tremendous increase of cost. Provided with this new estimation the optimization model ends up concluding that the fast mode must not load 5 products but should deliver the minimum amount of two products instead. The remaining 8 products are transported with the slow mode accepting that they arrive one period too late which is the better choice anyway. The customer's per unit penalty cost is now exceeded by the fast mode's per unit transportation cost. In the following iteration we face the same situation as before: transporting two units with the fast mode leads to decreased per unit transportation cost for the fast mode which in turn are then exceeded by the customer's per unit penalty cost. Again, it does not pay off to accept the late arrival of 8 products but to send 5 of them with the fast mode and such we go back to the beginning and are not able to leave this cycle. The solutions of the linear model as well as for the simulation jump between two values but the optimal solution of transporting four units with the fast mode will not be reached. This kind of cycling cannot be avoided by using an exponential smoothing technique; only the cycle length will increase. Please see Appendix B.1 for a complete list of data records used. Figure 6.2 illustrates the course of objective values for both simulation and optimization for this special experiment.

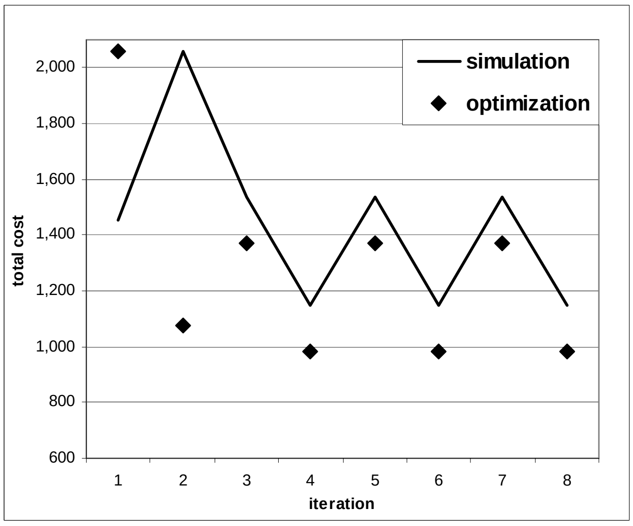

Figure 6.2.: Example for generating a cycle; the solution jumps between two possible realizations, because the nonlinear cost structure in the simulation model leads to heavily fluctuating estimations for the linear model

Apart from this special constructed situation we observed fast convergence in all test cases. Usually, the gap between the objective value determined by the simulation and that found by the optimization decreases continuously until it reaches an acceptable level of less than 1%.

6.2. Design of test instances

In order to verify the quality of solutions found by *SimOpt* we use a self-created set of 12 examples. For these test instances we refer to a simple supply chain consisting of three participants (a supplier, a producer, and a customer) and a time horizon of 30 periods. For delivery of products transportation objects are used connecting the supplier and the producer as well as the producer and the customer. Two types of products are demanded by the customer: *Product 1* which is provided by the supplier and sent via the producer to the customer and *Product 2* which is manufactured by the producer using *Product 1* as a raw material. The cost structure in the network is designed as follows:

- Transportation costs, which are identical on both transportation legs, consist of fixed cost per delivery, which are subject to a step function. A delivery costs 100, 200 or 300 monetary units, depending on the consumed transportation capacity.

- Costs for production and transfer are separated into variable cost and fixed cost. The variable production cost is set to 30 monetary units per item and per period. The fixed part constitutes 50 monetary units per lot. Transferring products costs 15 units per item and per period plus a fixed part of 10 units per lot.

- Unsatisfied demand at the customer is penalized by 100 monetary units per missing item and period.

Concerning the demand at the customer we distinguish between instances with high demand and others with low demand. The difference lies in the frequency of orders sent off by the customer. In high demand cases the occurring orders in each period are around the maximum possible quantities deliverable referring to the capacities of the supplier and the producer. In low demand models the ordered amounts cover approximately 70% of the possible deliveries in each period. Instances D1-L to D5-L (see Table 6.3) are low demand cases. They comply with one and the same scheme for customer demand, but each of them referes to an unique realization for low demand models. Accordingly, instances D6-H to D10-H consider 5 different realizations of high demand models. Exemplary realizations of both schemes are given in Appendix B.2. The last two instances, D1a-L and D6a-H, are modifications of instances D1-L and D6-H, respectively. The former ones consider exactly the same ordering amounts as D1-L and D6-H but the fixed costs for production and transfer are increased to 1,000 respectively 500 monetary units per lot. Transportation delays for deterministic tests are set to a value of 3 and production or transaction needs one time period. See Appendix B.2 for precise data tables.

6.3. Deterministic experiments

For examples of the size presented above it is definitely possible to formulate an exact MIP model and determine the optimal solution within reasonable computation time. This we did in order to generate benchmarks used for the evaluation of the solutions found by *SimOpt*. The resulting formulation consists of 1,342 constraints, 1,080 continuous, and 300 binary decision variables. When applying *SimMIP* the solution of our approach is identical to that provided by the exact MIP model. This is clear because the MIP model embedded in *SimMIP* is exactly the same as the stand-alone MIP model and we do not consider stochastic elements in here (which would most probably lead to solution variabilities). Thus, a comparison of the exact solution to that found by *SimMIP* does not lead to any interesting statement at all. Consequently, we compared the solution found by the stand-alone MIP to the *SimLP* solution. The nonlinear parts are only referred to within the simulation model itself; the connected pure linear model does not include any of them. See Table 6.3 for the resulting total costs established by *SimLP* as well as those found by the stand-alone MIP model.

Table 6.1.: Comparison of total costs between *SimLP* and the exact MIP model for deterministic test cases classified by the occurrence of customer demand (H - high demand, L - low demand). MIP solutions marked with (*) are best solutions found after 60 minutes calculation time; all instances were solved on an Intel P4-M 2GHz, 768MB RAM using Windows XP

instance	SimLP	exact MIP	difference
D1-L	53,640	52,947	1.31%
D2-L	55,032	53,860	2.18%
D3-L	52,626	52,394	0.44%
D4-L	54,442	53,600	1.57%
D5-L	55,198	54,057	2.11%
D6-H	59,885	58,830	1.79%
D7-H	61,257	60,129	1.88%
D8-H	59,028	58,347	1.17%
D9-H	60,403	59,501	1.52%
D10-H	61,436	60,365	1.77%
D1a-L	63,720	61,587	3.46%
D6a-H	76,165	73,761*	3.26%
Average	77,165	73,760	1.87%

The gap between the *SimLP* solutions and the optimal solutions found by exact MIP model varies between 0.44% and 3.46% and averages in 1.87%. For instance D6a-H the computation time for the MIP model exceeded 60 minutes and was therefore terminated before finishing. We report the best solution found so far. As we would expect the two test instances with high fixed cost lead to an increased gap of over 3%. For low demand settings the variation of the gap seems higher than for test instances with high demand. But on average there seems to be no significant difference in solution quality between low and high demand cases.

Based on these results we may conclude that the error caused by neglecting fixed costs is low as long as the fixed costs are low compared with other costs. If fixed costs increase (relatively to the other costs) the nonlinear elements should be considered as far as possible in the optimization model used for the *SimOpt* experiment, i.e. *SimMIP* should be conducted instead of *SimLP*. All in all we got promising results, since, even while neglecting nonlinearities entirely, the solutions found with *SimLP* are not too far away from the exact ones. Per default we conducted 8 *SimLP* iterations for each instance which was far too much since for all of them convergence was reached with the 3^{rd} iteration. Concluding we want to highlight that we are able to find these promising results in a significant shorter time compared to traditional methods. While all *SimLP* experiments were finished after some seconds the stand-alone MIP model needed up to 60 minutes computation time.

6.4. Experiments including stochastic elements

In order to measure the quality of our solutions in a stochastic environment we conducted a set of test runs assuming stochasticity for the transportation times. Other delays, i.e. those for production or transaction are still deterministic and have the fixed value 1. We again compare solutions found by our *SimLP* approach with those found by a stand-alone deterministic MIP model. Since this MIP model does not cover stochastic features we need to provide estimated values of the transportation times for the deterministic MIP model. Within the simulation part of *SimLP* we consider uniformly distributed transportation delays between 1 and 9 for transportations from the supplier to the producer, and between 1 and 5 for transportations from the producer to the customer. For estimating the delay parameters we decided to perform runs using 90%-, 70%-, and 50%-quantiles. At first glance this choice may appear unreasonable, since service levels usually are desired to range around at least 90%. However, for these small test instances there would probably be no difference between the results of a 99%- and a 90%- quantile since both of them would always lead to almost the same risk averse solutions. Therefore, we decided to evaluate a high quantile of 90% referring to a risk averse behaviour, an average quantile of 50%, representing a risky attitude, and an intermediate value of 70%. As a matter of fact the transportation delays provided for the stand-alone deterministic MIP model are set aligned with the used quantile. For deliveries between supplier and intermediate we use the information about the distribution function and set the delays to 8 (90%-quantile), 7 (70%-quantile), and 5 (50%-quantile) time units (the estimated delay within *SimLP* will be based on the quantile and finally rounded to next bigger or smaller integer value). For the second transportation leg, i.e. between intermediate and customer, we chose delays of 5 (90%-quantile), 4 (70%-quantile), and 3 (50%-

quantile) time units. This way we tried to make the comparison between *SimLP* and the stand-alone MIP model as fair as possible. Apart from the transporation delays the instances have exactly the same setting as those presented in the previous section. Again, the first five instances, S1-L to S5-L (see Table 6.4), consider the low demand case. All of them comply with the same scheme of customer demand but each represents an unique realization of this scheme. Accordingly, instances S6-H to S10-H refer to 5 different realizations of high demand models. Exemplary realizations of both schemes are given in Appendix B. Instances S1a-L and S6a-H are replications of instances S1a-L and S6a-H with distinctly increased fixed costs in production and transaction (cf. Section 6.3). See also Appendix B.2 for complete data tables.

The maximum runtime for the stand-alone MIP model was set to 30 minutes while *SimLP* converged after a few seconds anyway. For the latter we processed 8 iterations, each consisting of five simulation runs and one LP computation which is usually absolutely sufficient since convergence is typically reached much earlier. Finally the solution of the stand-alone MIP model found after 30 minutes and the solution found by *SimLP* have been evaluated by performing 20 independent simulation runs based on both solutions. The averaged total costs over these 20 runs were taken as final result. As a matter of fact the seed used by the random number generator is chosen to be identical for the comparative runs. This seed certainly differed from the seed used during the *SimLP* experiment since otherwise our approach would have a clear advantage.

When conducting instances including stochastic elements a comparison between the stand-alone MIP model and *SimMIP* experiments lead to interesting findings. Therefore, we used exactly the same data setting in order to evaluate the performance of *SimMIP* experiments. We simply replaced the embedded linear model by a proper MIP model considering all nonlinear features. While we solely presented a general formulation of *SimOpt's* optimization part so far, at this point an exemplary MIP formulation as it could be used within *SimOpt* should be presented. For this purpose we simply formulate the given test setting as MIP model which then, coupled to the simulation model, represents a proper *SimMIP* environment. According to the given cost structure presented in Section 6.2 there are two nonlinear aspects which are now to be considered within the mathematical formulation given in Chapter 4. Namely, fixed-charge costs in production and transaction as well as a step function for transportation cost have to be taken into account within the model formulation. The additional constraints and the adapted objective function can be formulated as follows. Please see Appendix A for a precise list of the notation used for the formulations of the optimization model.

First we are going to formulate the extensions corresponding to the fixed costs in production and transaction. For this purpose the respective general cost functions $W_i^p(m_i^p(t))$ and $Z_i^p(u_i^p(t))$ are further specified as follows:

$$W_i^p(m_i^p(t)) = \begin{cases} n_i^p & \text{if } m_i^p(t) > 0 \\ 0 & \text{otherwise} \end{cases} \qquad \forall i \in J_I, p \in P, t \in T \tag{6.1}$$

$$Z_i^p(u_i^p(t)) = \begin{cases} o_i^p & \text{if } u_i^p(t) > 0 \\ 0 & \text{otherwise} \end{cases} \qquad \forall i \in J_I, p \in P, t \in T \tag{6.2}$$

So the fixed-charge part of production and transaction cost at intermediate i and for product p is represented by n_i^p and o_i^p, respectively. In order to use these within the objective function we introduce the binary decision variables $\phi_i^p(t)$ and $\psi_i^p(t)$ indicating by a value of 1 or 0 if there is a positive production amount $m_i^p(t)$ or transaction amount $u_i^p(t)$ of product p at intermediate i in period t or not:

$$\phi_i^p(t) = \begin{cases} 1 & \text{if } m_i^p(t) > 0 \\ 0 & \text{otherwise} \end{cases} \qquad \forall i \in J_I, p \in P, t \in T \tag{6.3}$$

$$\psi_i^p(t) = \begin{cases} 1 & \text{if } u_i^p(t) > 0 \\ 0 & \text{otherwise} \end{cases} \qquad \forall i \in J_I, p \in P, t \in T \tag{6.4}$$

Further we add the following constraints:

$$m_i^p(t) \leq M \cdot \phi_i^p(t) \quad \forall i \in J_I, p \in P, t \in T \tag{6.5}$$

$$u_i^p(t) \leq N \cdot \psi_i^p(t) \quad \forall i \in J_I, p \in P, t \in T \tag{6.6}$$

$$\phi_i^p(t), \psi_i^p(t) \in \{0, 1\} \quad \forall i \in J_I, p \in P, t \in T \tag{6.7}$$

M represents a large value and therefore Equation (6.4) ensures that in case binary variable $\phi_i^p(t)$ becomes 1 the production amount $m_i^p(t)$ in period t at intermediate i of product p can take any value smaller or equal to M. Accordingly, Equation (6.4) does the same as the previous one but it refers to transaction activities at intermediate i. Equation (6.4) defines that $\phi_i^p(t)$ and $\psi_i^p(t)$ are binary variables. The resulting MIP model includes $2 \times |J_I| \times |P| \times |T|$ binary decision variables so far.

The second nonlinear aspect which is intended to include into this MIP formulation is the step function concerning the transportation costs. As it is mentioned in Section 6.2 these costs are load-dependent and have three different fixed values.

Whenever a predefined capacity amount is exceeded the cost jumps to the next higher value. First of all we divide the corresponding transportation cost function ${}^vC^p_{ij}({}^vx^p_{ij}(t))$ into three subfunctions:

$$ {}^vC^p_{ij}({}^vx^p_{ij}(t)) = \begin{cases} {}^vy^p_{ij} & \text{if } {}^vx^p_{ij}(t) \in ({}^vE^p_{ij}, {}^vY^p_{ij}] \\ {}^ve^p_{ij} & \text{if } {}^vx^p_{ij}(t) \in ({}^vK^p_{ij}, {}^vE^p_{ij}] \\ {}^vk^p_{ij} & \text{if } {}^vx^p_{ij}(t) \in (0, {}^vK^p_{ij}] \\ 0 & otherwise \end{cases} \tag{6.8} $$

$$\forall i \in J_S \cup J_I, j \in J_I \cup J_C, p \in P, t \in T, v \in V$$

According to this definition transportation load ${}^vx^p_{ij}(t)$ of product p delivered from node i to node j with transportation mode v lying within a given interval $({}^vE^p_{ij}, {}^vY^p_{ij}]$ lead to a fixed transportation cost of ${}^vy^p_{ij}$. Clearly, the definition of interval $({}^vE^p_{ij}, {}^vY^p_{ij}]$ has to be done in awareness of the capacity consumption factor ${}^vg^p_i$ for product p transported with mode v. Assumed transportation load ${}^vx^p_{ij}(t)$ is within the given borders $({}^vK^p_{ij}, {}^vE^p_{ij}]$ this causes fixed transportation costs of ${}^ve^p_{ij}$. And if the transported amount is on the lowest step of capacity consumption, i.e. it lies in interval $(0, {}^vK^p_{ij}]$, the arising fixed cost is denoted by parameter ${}^vk^p_{ij}$. For our test instance as it has been defined above parameters ${}^vk^p_{ij}, {}^ve^p_{ij}$, and ${}^vy^p_{ij}$ are set to 100, 200, and 300 monetary units, respectively. Again, we have to include binary decision variables which have value 1 whenever transportation amount ${}^vx^p_{ij}(t)$ is within their unique interval:

$$ {}^v\gamma^p_{ij}(t) = \begin{cases} 1 & \text{if } {}^vx^p_{ij}(t) \in ({}^vE^p_{ij}, {}^vY^p_{ij}] \\ 0 & \text{otherwise} \end{cases} \tag{6.9} $$

$$\forall i \in J_S \cup J_I, j \in J_I \cup J_C, p \in P, t \in T, v \in V$$

$$ {}^v\epsilon^p_{ij}(t) = \begin{cases} 1 & \text{if } {}^vx^p_{ij}(t) \in ({}^vK^p_{ij}, {}^vE^p_{ij}] \\ 0 & \text{otherwise} \end{cases} \tag{6.10} $$

$$\forall i \in J_S \cup J_I, j \in J_I \cup J_C, p \in P, t \in T, v \in V$$

$$ {}^v\kappa^p_{ij}(t) = \begin{cases} 1 & \text{if } {}^vx^p_{ij}(t) \in (0, {}^vK^p_{ij}] \\ 0 & \text{otherwise} \end{cases} \tag{6.11} $$

$$\forall i \in J_S \cup J_I, j \in J_I \cup J_C, p \in P, t \in T, v \in V$$

Therefore, binary decision variable ${}^v\gamma^p_{ij}(t)$ is set to 1 if the amount ${}^vx^p_{ij}(t)$ of product p in period t transported from node i to node j with mode v is within the highest interval of capacity consumption, and is 0 otherwise. Accordingly, ${}^v\epsilon^p_{ij}(t)$ or ${}^v\kappa^p_{ij}(t)$ are set to 1 if the transported amount is within the medium or lowest stage of capacity usage, respectively. Finally a set of constraints is added to the given formulation where parameter A denotes a given large value:

$$ {}^{v}x_{ij}^{p}(t) \leq A \cdot \ {}^{v}\gamma_{i}^{p}(t) + {}^{v}E_{ij}^{p} \quad \forall i \in J_S \cup J_I, j \in J_I \cup J_C, p \in P, t \in T, v \in V \tag{6.12} $$

$$ {}^{v}x_{ij}^{p}(t) \leq A \cdot \ {}^{v}\epsilon_{i}^{p}(t) + {}^{v}K_{ij}^{p} \quad \forall i \in J_S \cup J_I, j \in J_I \cup J_C, p \in P, t \in T, v \in V \tag{6.13} $$

$$ {}^{v}x_{ij}^{p}(t) \leq A \cdot \ {}^{v}\kappa_{i}^{p}(t) \quad \forall i \in J_S \cup J_I, j \in J_I \cup J_C, p \in P, t \in T, v \in V \tag{6.14} $$

$$ {}^{v}\gamma_{i}^{p}(t), {}^{v}\epsilon_{i}^{p}(t), {}^{v}\kappa_{i}^{p}(t) \in \{0,1\} \quad \forall i \in J_S \cup J_I, j \in J_I \cup J_C, p \in P, t \in T \tag{6.15} $$

Equations (6.12), (6.13), and (6.14) ensure that transportation amount ${}^{v}x_{ij}^{p}(t)$ is exclusively set to a value larger than 0 in case one of the given binary variables ${}^{v}\gamma_{ij}^{p}(t)$, ${}^{v}\epsilon_{ij}^{p}(t)$, and ${}^{v}\kappa_{ij}^{p}(t)$ are 1. Furthermore they ensure the allocation of transported amounts of product p in period t between node i and node j with mode v to the proper interval. Equation (6.15) defines ${}^{v}\gamma_{ij}^{p}(t)$, ${}^{v}\epsilon_{ij}^{p}(t)$, and ${}^{v}\kappa_{ij}^{p}(t)$ as binary decision variables.

The resulting model now consists of $(4 \times |J_I| \times |P| \times |T| + 3 \times |J_S| \times |J_I| \times |P| \times |T| \times |V| + 3 \times |J_I| \times |J_C| \times |P| \times |T| \times |V|)$ binary decision variables. The objective function (4.26) is now defined as follows:

$$ \begin{aligned} min \quad & \sum_{i \in J_S \cup J_I} \sum_{p \in P} \sum_{t=1}^{T} {}^{out}h_i^p \cdot {}^{out}l_i^p(t) + \sum_{i \in J_I} \sum_{p \in P} \sum_{t=1}^{T} {}^{in}h_i^p \cdot {}^{in}l_i^p(t) \\ & + \sum_{i \in J_I} \sum_{p \in P} \sum_{t=1}^{T} w_i^p \cdot m_i^p(t) + \sum_{i \in J_I} \sum_{p \in P} \sum_{t=1}^{T} n_i^p \cdot \phi_i^p(t) \\ & + \sum_{i \in J_I} \sum_{p \in P} \sum_{t=1}^{T} z_i^p \cdot u_i^p(t) + \sum_{i \in J_I} \sum_{p \in P} \sum_{t=1}^{T} o_i^p \cdot \psi_i^p(t) \\ & + \sum_{i \in J_C} \sum_{p \in P} \sum_{t=1}^{T} \rho_i^p \cdot {}^{in}b_i^p(t) + \sum_{ij \in J} \sum_{p \in P} \sum_{t=1}^{T} \sum_{v \in V} {}^{v}k_{ij}^p \cdot {}^{v}\kappa_{ij}^p(t) \\ & + \sum_{ij \in J} \sum_{p \in P} \sum_{t=1}^{T} \sum_{v \in V} ({}^{v}e_{ij}^p - {}^{v}k_{ij}^p) \cdot {}^{v}\epsilon_{ij}^p(t) \\ & + \sum_{ij \in J} \sum_{p \in P} \sum_{t=1}^{T} \sum_{v \in V} ({}^{v}y_{ij}^p - {}^{v}e_{ij}^p - {}^{v}k_{ij}^p) \cdot {}^{v}\gamma_{ij}^p(t) \end{aligned} \tag{6.16} $$

This objective function is subject to Equations (4.2)-(4.3), (4.5)-(4.15), (4.17)-(4.18), (4.20)-(4.24), and (6.1)-(6.15). Applying this exemplary adaption of the model formulation all nonlinear elements of the previously presented declaration of instances can be referred to within the optimization part, i.e. we are able to conduct a *SimMIP* experiment. By now the difference between the *SimMIP* and the stand-alone MIP is only, that in the within *SimMIP* delays are estimated based on simulation experiments and for the stand-alone MIP model delays are predefined knowing the corresponding distribution functions. This constellation is finally used for performing the same experiments as for the pure *SimLP* tests. The results for all three methods (i.e. stand-alone MIP, *SimLP*, and *SimMIP*) are displayed in Table 6.4 where negative percentage values imply that the *SimMIP* achieved a better result than the stand-alone MIP model.

Table 6.2.: Difference of the mean total costs after 20 simulation runs based on the solution found by *SimMIP*, *SimLP*, and the deterministic MIP model classified by the occurrence of customer demand (H - high demand, L - low demand); total costs are reported in columns *cost*, quantiles leading to the best results are reported in columns *quant*, and the difference with respect to the solution of the deterministic MIP is denoted in column *diff*

	MIP		SimLP			SimMIP		
instance	cost	quant	cost	quant	diff	cost	quant	diff
S1-L	66,400	90%	62,601	90%	-5.72%	61,637	90%	-7.17%
S2-L	61,338	90%	60,635	90%	-1.15%	60,282	90%	-1.72%
S3-L	63,323	90%	63,566	70%	0.38%	63,618	70%	0.47%
S4-L	63,122	90%	64,067	90%	1.50%	64,060	90%	1.49%
S5-L	60,954	90%	62,399	90%	2.37%	62,229	90%	2.09%
S6-H	72,485	90%	72,342	90%	-0.20%	70,871	90%	-2.23%
S7-H	70,928	90%	70,751	90%	-0.25%	71,040	90%	0.16%
S8-H	73,257	90%	77,537	70%	5.84%	74,999	70%	2.38%
S9-H	73,501	90%	74,637	70%	1.55%	72,845	90%	-0.89%
S10-H	71,606	90%	70,230	90%	-1.92%	73,934	90%	3.25%
S1a-L	71,511	90%	71,686	90%	0.25%	70,350	90%	-1.62%
S6a-H	88,442	70%	90,582	90%	2.42%	88,442	70%	0.00%
Avg.-L	64,441		64,159		-0.39%	63,696		-1.08%
Avg.-H	75,037		76,013		1.24%	75,355		0.44%
Average	69,739		70,086		0.42%	69,526		-0.32%

The results of *SimLP* are on average slightly worse compared with the deterministic stand-alone MIP solutions. For low demand cases we observe that *SimLP* performs slightly better than the MIP solution but considering the variability of results these differences are not significant. However, it should be noted that the computation time for the stand-alone MIP varied between several minutes and several hours, whereas the *SimLP* algorithm needed on average 30 seconds for each test instance. Hence, we succeeded in finding high quality solutions within a short computation time compared to traditional methods.

We see that applying *SimMIP* the solution quality can be improved for 9 out of 12 test instances and on average the result is slightly better than the deterministic stand-alone MIP approach. In low demand cases *SimMIP* performs on average 1.08% better than the stand-alone MIP. Again, we want to mention that including integer decisions into the optimization model should be done in awareness of possibly tremendously increased computation time. Running a *SimMIP* experiment means to solve several MIP models which might be a quite time-consuming task.

Even if for some instances the 70%-quantile yields the best results, the 90%-quantile, i.e. a low risk behaviour, seems to be the best choice. Using the 50%-quantile, i.e. using the expected value, always caused much higher costs. This outcome is not surprising since using the 50%-quantile comes up with a quite risky behaviour assuming the delays to be according to their expected values and, therefore, to high total penalty costs. Additionally we analyzed the variability of the 20 final simulation runs for each method. A list of minimum total cost, maximum total cost, and coefficient of variance for each instance and each method is provided in Appendix B.3.6. Summarizing we state that there is no significant difference of the variation for all methods. The averaged coefficient of variation for the *SimOpt* methods are 6.91% (*SimLP*) and 7.63% (*SimMIP*). The solutions found by the stand-alone MIP have a coefficient of variance of 6.83%.

Since *SimMIP* and the stand-alone MIP model had almost the same level of information one may expect that both of them conclude with the same the solution. Nevertheless *SimMIP* performs worse in 3 out of 12 instances. This can be explained by little cost differences occurring due to the fact that we have some non-integer variables within the MIP models as well. Therefore, it may happen that the stand-alone MIP model and the embedded MIP model provide slightly different solutions and lead to the situation that the isolated exact approach performs a little bit better than the combination of simulation and optimization. Maybe it seems also questionable that in 3 cases *SimLP* performed better than *SimMIP* in three cases. Since *SimMIP* includeed all information concerning the nonlinear cost in production, transaction and transportation it definitely was able to benefit from making aggregation decisions, e.g, by arranging transportation lots. Due to the lack of this information *SimLP* had a clear disadvantage especially if high fixed costs were considered. But consequently, penalty costs caused by stochastic transportation delays, had a much more harmful effect on total costs of *SimMIP* than of *SimLP*. While for the latter we ended up with rather small lots arriving late the aggregation decisions of *SimMIP* may have lead to quite huge amounts of delayed products. Clearly, in all cases *SimMIP* ended up with lower operational costs compared to those of *SimLP*. However, in few cases the gap of penalty costs was that large that it cannot be compensated by lower operational costs. This way it happens that in few cases *SimLP* in total performed better than *SimMIP*. Disadvantages of *SimLP* compared to *SimMIP* concerning the aggregation of lots could be softened by incorporating period-specific costs. Based on these *SimLP* would recognize per unit cost variations within one and the same run and would thus try to aggregate lots in cheaper periods instead of partitioning them over the planning horizon. This would help *SimLP* to ameliorate its results for models including fixed costs. However, an implementation of period-specific costs has to be planned precisely since it involves some critical questions which are not trivial to solve (e.g., data gathering becomes much more challenging since a huge number of additional cost informations are necessary).

For larger test instances it would not be possible to solve the MIP or to apply the *SimMIP* approach within reasonable computation time. In a preliminary study Mitrovic (2006) we focused on that issue and tried to find the approximate limits of solving MIP formulations of supply chain problems by the means of three state-of-the-art LP/MIP solvers. These tests were conducted on a PC (Intel P4 2.4GHz, 1GB RAM) using Windows 2000. We used a set of different sized supply chain network problems considering fixed costs in transportation, production, and transfer. The best performing solver succeeded in solving problems with 20 supply chain actors, 8 products, 5 periods and 2 transportation modes, considering 3,360 binary variables before and 250 binary variables after presolving. The next in size, which included 10 products instead of 8, could not be solved within a time limit of 60 minutes. Thus, one has to identify a good trade-off between including nonlinear elements into the optimization part, i.e. referring to a more realistic model, and keeping down computation time. Definitely, important decisions involving high fixed costs should be considered within the embedded optimization model of *SimOpt*. However, we mainly focus on finding good solutions by simultaneously coping with stochastic elements. For this purpose applying *SimOpt* in awareness of an upper limit for the problem complexity is a reasonable procedure. Using heuristic problem solving methods in order to support solution finding of complex MIP models is certainly a subject for future research as it is highlighted in the conclusion of this thesis.

6.5. Experiments on larger instances

In addition to the small instances used in the previous subsections we generated a set of 12 instances representing larger supply chain networks. Using these test instances we analyze the influence of the quantile used for estimating the delay parameters. For this purpose we evaluated two different scenarios. The first one covers the case that uncertainty is concentrated in the beginning of the supply network, and the second focuses on the case that stochastic elements occur at the downstream end of the network. The size and structure of these test cases are shown in Figure 6.3.

This fictitious supply chain network consists of 10 actors: 3 suppliers, 4 intermediate nodes, and 3 customers. The intermediate nodes are separated into two layers and all of them are authorized to produce and also to transfer products. There

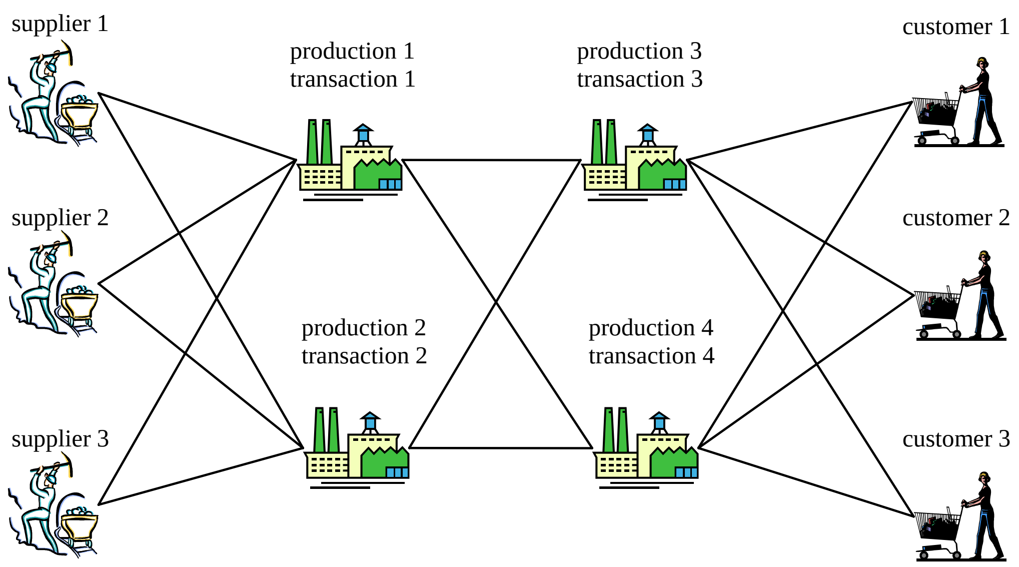

Figure 6.3.: Structure of the increased test instances where for simplicity reasons transport modules have been omitted

is one transportation mode available. The customers request 4 different products. Products 1 and 2 are on the one hand final products, which have to be delivered to the customers and on the other hand raw materials used to produce Products 3 and 4 which are as well demanded by the customers. First we refer to the case with stochasticity concentrated at the beginning of the supply chain. Hence, for the connection between the suppliers and the first layer of production sites we assumed stochastic transportation times, which are uniformly distributed between 1 and 5. The transportation times between the two layers of production nodes are uniformly distributed between 1 and 3. For the remaining links we assume deterministic transportation times of 1. The cost functions for production and transaction consist of a fixed and a variable part. Again, we classify the instances according to occurring customer demand (H - high demand, L - low demand). Instances L1-L to L5-L (see Table 6.5) each refer to an unique realization of the low demand scheme. Accordingly, L6-H to D10-H consider 5 different realizations of high demand models. Exemplary realizations of both schemes are given in Appendix B. The last two instances, L1a-L and L6a-H, are modifications of instances L1-L and L6-H considering extra high fixed costs in production and transaction. Please see Appendix B.3 for assembled lists of data input. If all binary decisions were considered within a proper MIP Model, this would have lead to more than 3,800 binary variables, which is beyond the size of problems we succeeded to solve within several hours during the preliminary study mentioned above Mitrovic (2006). In comparison, our *SimLP* algorithm took about 12 minutes per test instance to converge to a solution. As already mentioned we now focused on achieving information concerning the most preferable quantile for estimating the delay parameters. For this purpose we evaluated three different quantiles, i.e. 90%, 70%, and 50%, as they had also been used in the previous test experiments. See Table 6.5 for the results.

Table 6.3.: Total costs of *SimLP* for test cases with 10 supply chain actors and stochastic transportation delays at the beginning of the supply chain (H - high demand, L - low demand); the results for the 90%-quantile are taken as basic values for the remaining quantiles the percentage difference to the corresponding basic value is given. *S* indicates that stochasticity occurs near the supplier

	quantile		
instance	90%	70%	50%
L1-L-S	274,239	-0.19%	11.20%
L2-L-S	274,241	9.98%	6.00%
L3-L-S	274,995	5.76%	10.85%
L4-L-S	275,366	5.83%	11.51%
L5-L-S	273,214	1.33%	9.80%
L6-H-S	270,286	2.72%	10.10%
L7-H-S	270,491	2.23%	4.51%
L8-H-S	270,438	-0.59%	10.92%
L9-H-S	267,766	2.71%	8.65%
L10-H-S	270,155	-0.57%	10.12%
L1a-L-S	333,772	2.77%	4.04%
L6a-H-S	346,953	1.42%	4.52%
Total Avg.	283,493	2.78%	8.52%

In this case it seemed that the 90%-quantile is the most robust choice, although the 70%-quantile delivers on average only slightly worse results and in some cases even better ones. The 50%-quantile lead to the worst results for all instances. Due to the fact that there are less stochastics near the customer, it is possible to reduce the safety factors for the delays to some extent without increasing the costs too much, because lost time at the beginning of the supply chain can be made up at the downstream end. Most probably this is the reason why the results found with the 70%-quantile do not come off too badly compared to the risk minimizing strategy of choosing the 90%-quantile. However, estimating delays the risky way, i.e. choosing the 50%-quantile, does not pay off since underestimations arising in the beginning of the network cannot be coped anymore and lots of deliveries arrive too late at the customer.

We also conducted experiments where stochastic transportation delays are to be found at the downstream end of the supply chain network. In fact, the connection between intermediate nodes and customers are uniformly distributed between 1 and 5. Transportation times between the two layers of intermediate nodes are again uniformly distributed between 1 and 3 and remaining transportation times are set to 1. The corresponding results are summarized in Table 6.4.

Table 6.4.: Total costs found by *SimLP* for test cases with 10 actors and stochasticity concentrated near the downstream end of the supply network (H - high demand, L - low demand); the results for the 90%-quantile are taken as basic values; for the remaining quantiles the percentage difference to the corresponding basic value is given. *C* indicates that there stochasticity occurs near the customer

	quantile		
instance	90%	70%	50%
L1-L-C	263,957	11.70%	43.96%
L2-L-C	263,252	16.81%	45.02%
L3-L-C	263,948	13.79%	48.81%
L4-L-C	263,114	12.65%	50.60%
L5-L-C	263,707	13.98%	51.41%
L6-H-C	258,440	18.11%	52.09%
L7-H-C	257,963	14.60%	59.29%
L8-H-C	260,606	12.67%	41.68%
L9-H-C	258,762	24.74%	52.44%
L10-H-C	258,936	17.52%	43.16%
L1a-L-C	335,837	7.15%	32.56%
L6a-H-C	335,973	9.61%	32.95%
Total Avg.	273,708	14.45%	46.16%

For these instances the best choice clearly is to use the highest safety factor, i.e. the 90%-quantile, because there are stochastic delays in the downstream end of the supply chain network and there is no chance to catch them up later.

For test instances L1a-H-C and L6a-H-C where extra high fixed costs in production and transaction are included, we tentatively tried to apply *SimMIP* but solely included the extra high fixed costs referring to production and transaction. The relatively low fixed costs in transportation were still neglected within the optimization part. The results are presented in Table 6.5.

For all quantiles the result can be improved slightly since the optimization model could act in awareness of the high fixed costs and benefits from the resulting crucial aggregations of production and transaction lots. However, the calculation times were more than 5 times longer compared to those of *SimLP*. This again highlights that it is absolutely essential that the inclusion of binary decisions within the optimization part of *SimOpt* has to be decided carefully.

If stochastic transportation times are assumed throughout the whole supply chain network, i.e. all transportation delays are defined to be uniformly distributed be-

Table 6.5.: Total costs found by *SimMIP* for 2 selected instances with 10 actors and extra high fixed costs in production and transaction; all percentage values show the difference to the corresponding results found by *SimLP* as they are presented in 6.4

	quantile		
instance	90%	70%	50%
L1a-L-C	-4.48%	-10.86%	-1.19%
L6a-H-C	-3.47%	-2.96%	-0.37%

tween 1 and 5, the results for all 12 instances are similar to those in Table 6.4, i.e. the 90%-quantile is always the best choice.

Having conducted a number of theoretical experiments using *SimOpt* we decided to apply this method in a real-world environment. Joint work with a paper manufacturer provided a perfect possibility to evaluate *SimOpt's* performance on a realistic sized distribution network. For the first phase of this case study we simply solved an LP model since dynamic and nonlinear aspects were omitted so far. For the sake of completeness, a detailed description of the first project phase initiates the following part. This is followed by an implementation description and reports on numerical experiments referring to the second, more challenging, phase were we considered dynamic and nonlinear aspects as well. This increase in complexity made it necessary to apply a more sophisticated solution method which we did by using *SimOpt*.

Part III.
Real world application

7. Introduction

The modeling design of *SimOpt* is motivated by a case study dealing with the medium-term optimization of a paper manufacturer's distribution network. Within this corporative project our main task was to determine production quantities and preferable transportation links in a predefined network and for a planning horizon of one year (cf. Gronalt, Hartl, and Preusser (2007)).

The given network consists of four European manufacturing sites (A, B, C, and D) which deliver 10 different products (I-X) to customers that are partly located within Europe, but also distributed in oversea regions all over the world. The sites differ in their configuration and none of them is able to fabricate all 10 kinds of products, but all of them have a fixed product-mix. Furthermore, they differ in production quality which means that two of them are able to deliver high quality while the others deliver low quality products. Distribution is done either directly from a site to a customer or indirectly using hubs or ports as intermediate stations. Indirect deliveries within Europe are executed using hubs, which are to be chosen out of a set of 18 predefined locations. Deliveries to oversea destinations are forced to pass exactly one port anyway. There is a set of 9 available ports throughout Europe. Figure 7.1 illustrates the given network structure.

Since this case study was based on highly confidential corporate information we consistently present distorted values of the real data. The ratio of intra-European compared to oversea deliveries based on the total delivery quantity as well as of direct to indirect deliveries (via hubs or via ports) are reported in Table 7.1.

Table 7.1.: Percentage of deliveries within Europe in relation to oversea deliveries; the relation of direct to indirect deliveries is reported as well

site	sum (tons)	Europe	oversea	direct	via hub	via port
A	210,040	40%	60%	31%	9%	60%
B	204,130	42%	58%	13%	29%	58%
C	124,950	54%	46%	54%	0%	46%
D	89,510	88%	12%	84%	4%	12%

Due to various reasons like mergers and the strict attempt to keep traditional supply relationships alive the company, over the years, got deadlocked in a distribution strategy which is suboptimal at first glance. Thus, a project covering a widespread analysis of the existent network and a survey of improvement strategies became

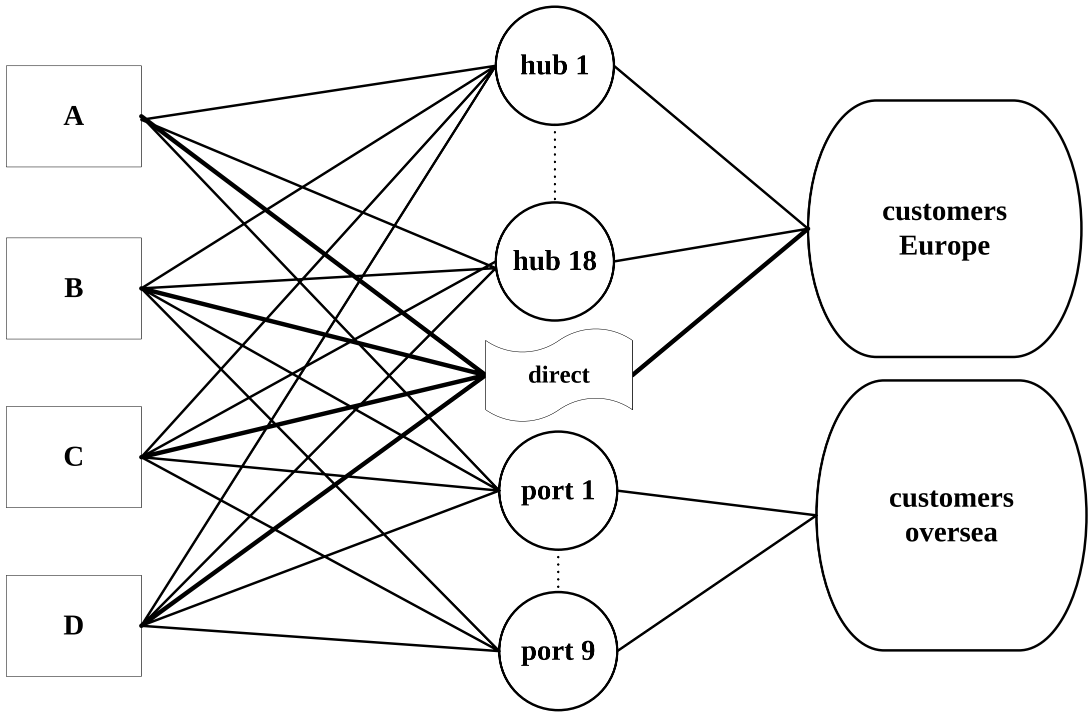

Figure 7.1.: Given structure of the distribution network

essential. In the initial phase of this project we determined the *Quick Wins*, i.e. savings based on the elimination of clear inefficiencies within the distribution strategy. Clearly, this isolated examination of the problem did not lead to a convenient improvement and thus the development of a more sophisticated optimization method was initiated. We decided to calculate the optimal distribution strategy by formulating and solving an adequate LP model. Due to intra-company and geographical circumstances the LP model was subject to a number of side constraints which lead to a considerable limitation of improvement possibilities. However, we wanted to present the whole optimization range anyway and therefore specified different scenarios by consecutively loosening side constraints. Thus, we were finally able to present possible reductions in transportation cost for different levels of network modifications. Figure 7.2 illustrates the project's course with a brief description for each milestone.

The following chapter is going to focus on the isolated examination of each production site's distribution strategy (*Quick Wins*) and will be followed by the introduction of the static LP model which we used since we aimed at getting rough estimations of potential improvements quickly. Subsequently, the challenging task of gathering all necessary data and the main findings of our investigations are discussed. Apart from this consulting project we conducted an interesting follow-up analysis which will be addressed in Chapter 12. It focuses on an extended model where we included dynamic and nonlinear aspects and faced problems like stochastic

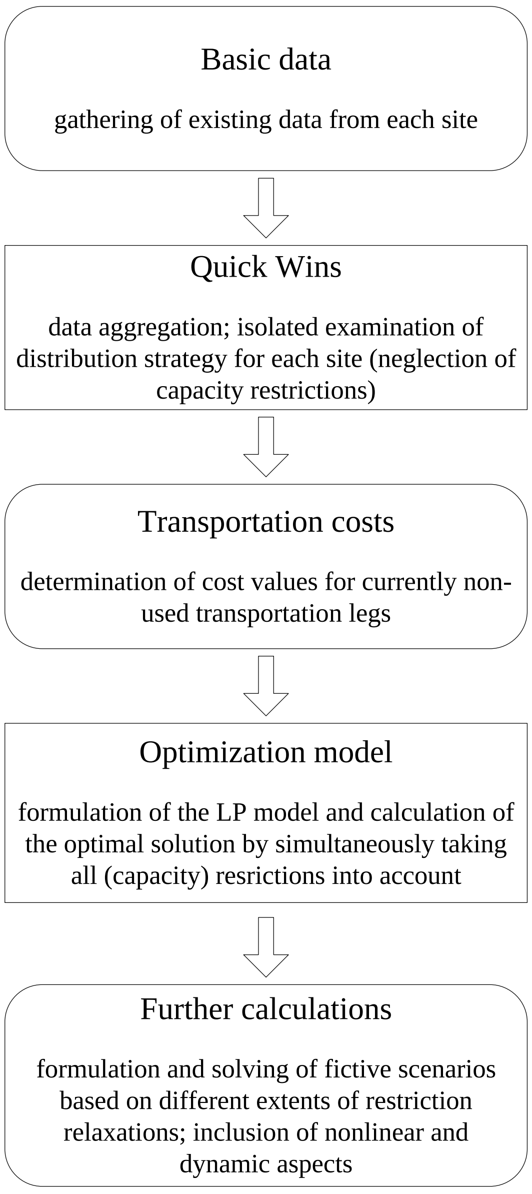
Basic data
gathering of existing data from each site
Quick Wins
data aggregation; isolated examination of distribution strategy for each site (neglection of capacity restrictions)
Transportation costs
determination of cost values for currently non-used transportation legs
Optimization model
formulation of the LP model and calculation of the optimal solution by simultaneously taking all (capacity) resrictions into account
Further calculations
formulation and solving of fictive scenarios based on different extents of restriction relaxations; inclusion of nonlinear and dynamic aspects

Figure 7.2.: Milestones of the joint project with the paper manufacturing company

lead times and fixed production costs under changing demand behaviour. Using the real world data we highlight the clear advantage of central planning compared to decentral planning in this context and that the combined simulation-optimization method (*SimOpt*) is able to handle real-world-sized networks sufficiently.

8. Quick Wins

In order to identify the clear inefficiencies of the system we requested lists of all deliveries that have been processed at one of the manufacturing sites in the base year. Using them we picked out deliveries where

- a customer got supplied with one and the same product type by more than one production site,
- a customer requested different kinds of products at more than one site, even if single sourcing would have been possible as well,
- a site sent deliveries to one and the same customer via different distribution channels.

Since there are no capacity restrictions in transportation to be taken into account all these deliveries are, at first glance, evidence for the inefficiency of the system. So for all these cases we calculated the added transportation cost having arised due to the insufficiently adjusted planning on the corporate level. In Table 8.1 we present some examples of inefficiencies leading to *Quick Wins*.

Table 8.1.: Examples for calculations of added costs due to clear inefficiencies within the distribution network

site	customer	via	product	cost/ton	tonst	*Quick Wins*
B	AUS1	Hamburg	I	160	5,288	211,520
C	AUS1	Hamburg	I	200	5,288	
A	AT1	-	II	6	1,015	22,330
C	AT1	-	II	28	1,015	
A	D1	Salzburg	III	41	148	2,812
C	D1	Gohfled	III	60	148	

We were able to identify a huge number of such apparent inefficiencies in transportation planning. Discussing this topic with the logistics executives we found out that a number of these distribution decisions were made deliberately knowing that, due to various reasons, they are inevitable. Finally we ended up with only 17 deliveries where a suboptimal distribution decision was made without any reproducible reason leading to total added costs in transportation of € 587,000 per year.

We extended the analysis of *Quick Wins* from the single customer related level to an aggregated level considering customer regions instead. Clustering was made based on the first two ZIP-Code digits for European customers and based on countries for oversea ones leading to an amount of 25 European regions and 44 oversea distribution destinations. The added value was small - we solely found 6 more cases of evident inefficiencies leading to additional *Quick Wins* of €157,300 per year. Accordingly, an adjustment of distribution planning on manufacturing sites' level would, in total, have saved transportation cost of up to €744,300 per year. Clustering of customers to ZIP-Code-based and country-based regions was retained for all following analyses since an optimization on the single customer level was decided to be too detailed and unnecessary.

This initial analysis we primarily used in order to roughly estimate the optimization potential in distribution planning to the corporate authorities. Related for the calculation of *Quick Wins* we had, in contrast to the following global optimization, the advantage that all essential data was already existent and quite easily accessible. Determining an optimal solution on the global level was much more challenging since we had to gather cost values for an huge number of transportation legs that have not been used for product distribution so far. However, the result of this isolated examination of *Quick Wins* could, due to the implicit ignorance of global capacity restrictions, in most cases not hold within the global context. In the optimal solution only a few of them are intended to be realized.

9. Model formulation

Having finished the quite myopic determination of *Quick Wins* we went on with implementing a systematic solution method aiming for the optimal solution in a global context. In fact, we formulated an LP model considering the transportation amounts on each leg of the network and the transaction amounts at hubs and ports as decision variables. Our model is based on a standard network flow formulation as it is, for example, presented in Fleischmann, Meyr, and Wagner (2005). This formulation is a special case derived from the general formulation presented in Chapter 4. Some features, which we are referring to within the general model formulation, are not relevant for this real world case. Thus, we adapted the general formulation leading us to the following model formulation:

$J = J^S \cup J^L \cup J^H \cup J^E \cup J^A$	set of network participants
$j \in J^S$	manufacturing sites
$j \in J^L$	hubs
$j \in J^H$	ports
$j \in J^E$	European customer regions
$j \in J^A$	oversea customer regions
$p \in P$	products
u_i	transaction amount at hub or port i
x_{ij}^p	amount of product p transported on leg ij
c_{ij}	transportation cost factor per ton on leg ij
D_j^p	demand for product p in customer region j
h_i	transaction cost factor per ton at hub or port i
Max_i	maximum total transaction amount at hub or port i
Min_i	minimum total transaction amount at hub or port i
$prod_i^p$	maximum production amount of product p at site i

$$min \quad \sum_{i,j \in J} \sum_{p \in P} c_{ij} x_{ij}^p + \sum_{i \in J^L \cup J^H} h_i u_i \tag{9.1}$$

$$\sum_{j \in J^L \cup J^H \cup J^E} x_{ij}^p \leq prod_i^p \qquad \forall i \in J^S, p \in P \tag{9.2}$$

$$\sum_{k\in J^S}\sum_{p\in P} x_{ki}^p = u_i \qquad \forall i \in J^L \tag{9.3}$$

$$\sum_{k\in J^S}\sum_{p\in P} x_{ki}^p = u_i \qquad \forall i \in J^H \tag{9.4}$$

$$\sum_{j\in J^E}\sum_{p\in P} x_{ij}^p = u_i \qquad \forall i \in J^L \tag{9.5}$$

$$\sum_{j\in J^A}\sum_{p\in P} x_{ij}^p = u_i \qquad \forall i \in J^H \tag{9.6}$$

$$\sum_{j\in J^S\cup J^L} x_{ij}^p = D_i^p \qquad \forall i \in J^E, p \in P \tag{9.7}$$

$$\sum_{j\in J^H} x_{ij}^p = D_i^p \qquad \forall i \in J^A, p \in P \tag{9.8}$$

$$u_i \leq Max_i \qquad \forall i \in J^L \cup J^H \tag{9.9}$$

$$u_i \geq Min_i \qquad \forall i \in J^L \cup J^H \tag{9.10}$$

$$x_{ij}^p \geq 0 \qquad \forall ij \in J, p \in P \tag{9.11}$$

$$u_i \geq 0 \qquad \forall i \in J \tag{9.12}$$

The objective function (9.1) minimizes total transportation and transaction cost while Equation (9.2) limits the amount delivered by a production site to the respective maximum production amount. Equations (9.3) and (9.4) balance the amounts arriving at hub or port i to the amount transacted at location i. Since hubs are used for deliveries to European customers alone, we introduced Equation (9.5) ensuring that the transaction amount at hub i is equal to the amount transported from hub i to European customer regions. This implicitly prohibits deliveries between hubs and ports, which indeed is one of the company's hard constraints. Indirect deliveries are not allowed to pass more than one hub or port. For oversea deliveries we have the additional specification that deliveries have to be transacted at a port from where they are finally sent to the destined oversea region. Thus, we formulated Equation (9.6) balancing the amount sent by port i to oversea regions with the transaction amount of port i. Equations (9.7) and (9.8) restrict the amounts arriving at customer regions to be exactly covering the respective region's demand. Hubs and ports are subject to given maximum and minimum transaction amounts. These are taken into account by Equations (9.9) and (9.10). Finally, Equations (9.11) and (9.12) ensure nonnegativity for the decision variables.

We see that this model is basically focused on the flow of products between the actors of the supply chain. Since the production amounts at the manufacturing sites are fixed we do not consider them within our formulation. Furthermore, we do not explicitly consider inventory levels at the production sites. Supplier and customer nodes do not cause any additional costs. This is due to the fact that the company was solely interested in their internal distribution strategy, leaving procurement or penalty costs aside. Since we do not consider any nonlinear aspects so far, this simplified model can be easily solved by using a standard LP solver. Not until we refer to dynamic aspects and nonlinear elements we use the combined simulation-optimization method, *SimOpt*, as a solution method.

Over time we identified a number of additional restriction that we did not consider within this first model formulation but took into account within the following numeric experiments. Worth mentioning is, that for a couple of customers it is not possible to supply them on direct distribution channels since they do not dispose of sufficient inventory capacities for themselves. Deliveries to regions including one of these special customers have to pass an hub anyway. Furthermore, one of the manufacturing sites has scarce inventory capacities as well which lead to the side constraint that the amount of products sent away directly from this site to customers is limited to a total sum of 60,000 tons per year. Remaining products are proceeded, just after they leave the production process, to an hub from where they are then transported to the intended customer region.

10. Basic data

Gathering data for a profound global optimization was a much more challenging and time-consuming task than initially expected. Best effort was invested in order to develop a valid, realistic, and widespread database while various inconsistencies within the disposed data caused discouraging throwbacks. We designed a sheet requesting detailed information on each delivery carried out in the base year. It was proceeded to the logistics authorities of each manufacturing site. See Table 10.1 for an exemplary excerpt of the data sheet.

Table 10.1.: Excerpt of data sheet requesting essential information about all deliveries in the base year

customer	ID	city	direct	via	tons	total cost
RAP	CI23	Yamoussoukro		Hamburg	1,092	129,400
DNT	UA12	Odessa	X		55	11,880
TTB	AT80	Graz		Salzburg	84	11,088

The production capacities were directly derived from the production amounts of the base year. Predefined limits for minor deviation of these production amounts defined the maximum and minimum output for each manufacturing site. It has already been mentioned above that we had to consider differences in production quality depending on the manufacturing site. Within the implementation we distinguish between high quality and low quality products, distributed by Western European and Eastern European sites, respectively. Customers being formerly supplied with products coming from Western Europe may further on not be supplied by Eastern European manufacturing sites. Furthermore, we had to collect data concerning transaction costs at hubs and ports. In addition to the ex-post calculations for the base year we also conducted experiments concerning the subsequent year. Though we did not perform an additional data survey but took cost and capacity factors from the base year and solely adapted the demand values which were for both the base year and the subsequent year provided by the company's distribution department.

Finally it remained to determine the transportation cost factors for all arcs of the distribution network. Since we knew the occurred total transportation costs of all deliveries proceeded in the base year we simply had to adapt the costs of indirect deliveries in the sense that we separated them into costs on arc 1, i.e. from a site to an hub or port, and costs on Arc 2, i.e. from hub or port to the customer region. Thus, we ended up with the total transportation costs of each already used arc in the network and simply divided these value by the respective delivery amount in order to have per ton cost factors for each arc. The separation of total costs into the part arising on Arc 1 and the part arising on Arc 2 was easy, since the company provided detailed information on transportation costs between their sites and hubs/ports assumed the connecting arc has already been used at least once. The challenging task now was to determine cost factors for arcs that had been unused so far. Even after building clusters of customers based on their geographical position we had 4,000 arcs open for determination of costs. This huge amount of missing data eliminated the initial idea to request an appropriate prize list from the involved freight forwarder. However, using the standard prizes would not have been a good idea anyway since for the existing legs the company already obtained significant rebates. Within the optimization rebate-based arcs would always have seemed to be preferable compared to those evaluated based on standard prizes. This certainly would have distorted all subsequent calculations. As a matter of fact we decided to derive the missing cost factors based on the available data values by calculating average per ton and per kilometer costs and applying them to the, till then, unused legs. For this purpose we developed a distance matrix using MS MapPoint. We determined the necessary per ton and kilometer cost factors by dividing the known cost factors by the corresponding distance. After calculating average values over the obtained values we simply multiplied them with the corresponding entries in the distance matrix. Maybe it's worth mentioning that we sorted out very small deliveries, i.e. deliveries with a total amount of less than 25 tons, in advance assuming that they were disproportionally expensive cases of emergency and would have a distorting impact on the calculation of average cost factors. Thus, we ensured that obtained rebates on existing legs were implicitly considered for the evaluation of new ones. Due to simplicity reasons we did not consider till then unused arcs between ports and oversea destinations. For this part of the network solely already frequented legs were taken into account. After some mutual reviews with the logistics authorities we finally succeeded in creating a database agreeable for all corporation participants. Exemplary calculations of cost factors per ton and kilometer and of transportation costs per ton are presented in Table 10.2 and 10.3, respectively.

Of course we did not revise already existing cost values by applying these calculations, but but used their original cost factors. See Table 10.4 for an excerpt of the resulting transportation cost matrix.

Some time after finishing this project we came across an interesting report which probably could have saved us a lot of time and effort. Werr and Scheuerer (2007) show how they adapted existing cost schedules on long distance traffic for the optimization of a distribution network where they face, similar as we did, the problem of cost determination for previously unused transportation legs. Using the adapted schedules they generate an high-quality cost matrix under consideration of degressive behaviour of cost in distance and load. Fruthermore, the authors address base price levels, meaning a fixed minimum transportation cost, disregarding the delivery

Table 10.2.: Exemplary generation of cost factors per ton and kilometer based on transportation costs on existing legs

from/to	hub SB1	region AT34	region DE15	average
A	$=\frac{costperton}{distance}$	-	$=\frac{costperton}{distance}$	a
B	-	-	$=\frac{costperton}{distance}$	b
C	-	-	-	c
hub SB1	-	$=\frac{costperton}{distance}$	-	d

Table 10.3.: Exemplary generation of transportation costs per ton based on calculations presented in Table 10.2

from/to	hub SB1	region AT34	region DE15
A	-	$= distance * a$	-
B	$= distance * b$	$= distance * b$	-
C	$= distance * c$	$= distance * c$	$= distance * c$
hub SB1	$= distance * d$	-	$= distance * d$

Table 10.4.: Excerpt of the final transportation cost matrix (E - Europe, O - oversea)

from/to		customers (E)		hubs		ports		customers (O)	
		F62	HU26	h1	h2	p1	p2	Iran	Taiwan
sites	A	60	29	32	38	29	74	-	-
	B	92	155	95	89	29	62	-	-
	C	52	32	48	39	35	27	-	-
	D	49	40	35	35	26	8	-	-
hubs	h1	244	226	-	-	-	-	-	-
	h2	90	165	-	-	-	-	-	-
ports	p1	-	-	-	-	-	-	89	18
	p2	-	-	-	-	-	-	80	-

amount. This fixed transportation cost does not increase with the delivery amount until a given threshold is reached. Having exceeded this threshold the base price level is supplemented by a linear cost function depending on the transportation amount.

In retrospect, we checked the transportation cost values provided by the paper manufacturing company and figured out that the assumption of cost degression in distance and load perfectly fits in our case. Thus, the approach proposed by Werr and Scheuerer would most probably have lead to more realistic cost estimations for our data gathering problem. By applying a simple kind of data exploration, as we did, one takes the risk of loosing important characteristics of the given data set. Nevertheless, since we cross-checked our cost estimations again and again with different authorities, we concluded to have a valid data set and did not apply the approach presented by Werr and Scheuerer (2007) anymore.

11. Results

Beside the determination of the optimal solution for the basic scenario as it has been described above we specified three additional problem settings by successively loosening selected restrictions. These are the fixed product mix at the manufacturing sites, the predefined allocation of particular customer regions to a supplier, and finally the fixed, or almost fixed, proportional assignment of production capacities. Though quite unrealistic, we modeled and solved the situation of complete elimination of side constraints as well. All scenarios were implemented with Xpress-MP and solved on a 1 Ghz Pentium 3 processor within a couple of seconds. Again we want to mention that for the following results the stand-alone LP model described in Section 9 was absolutely reasonable and efficient. Please see Section 12 for the more challenging computations were we used *SimOpt* to cope with the high level of complexity.

11.1. Basic scenario

The huge amount of restrictions that had to be considered for the basic case reduced the range of potential improvements considerably. In fact, the optimal solution for this setting just lead to a reduction in total transportation cost of €117,200 per year, with is a marginal improvement, indeed. According to the optimal solution, the site with the highest level of necessary modifications in its distribution strategy is site D. Thus, for clarity reasons the following discussion of results focuses on implications for site D. The current and the optimized distribution network of site D are illustrated in Figures 11.1 and 11.2, respectively.

As we can in see in Table 11.1 the intra-European deliveries of site D should be limited to five countries while the effort in oversea activities is recommended to be increased. Especially USA, Africa, Asia and Australia should move into site D's distribution focus. 73% of site D's total production volume of 71,200 tons per year should be used for the supply of oversea countries, leaving 27%, i.e. 19,000 tons, for intra-European distribution. Furthermore we found out that deliveries coming from site D should not pass any European hub, but be sent directly to the European customer regions instead. All oversea deliveries should be handled by two selected ports. The optimal solution basically implies that site D replaces site C concerning the supply of oversea countries committing the latter with an increased amount of distributions within Europe. It has already been mentioned that the *Quick Wins*, which we determined in the initial phase, could not hold in the global context. Table 11.1 contains the optimized percentage rates of intra-European and oversea

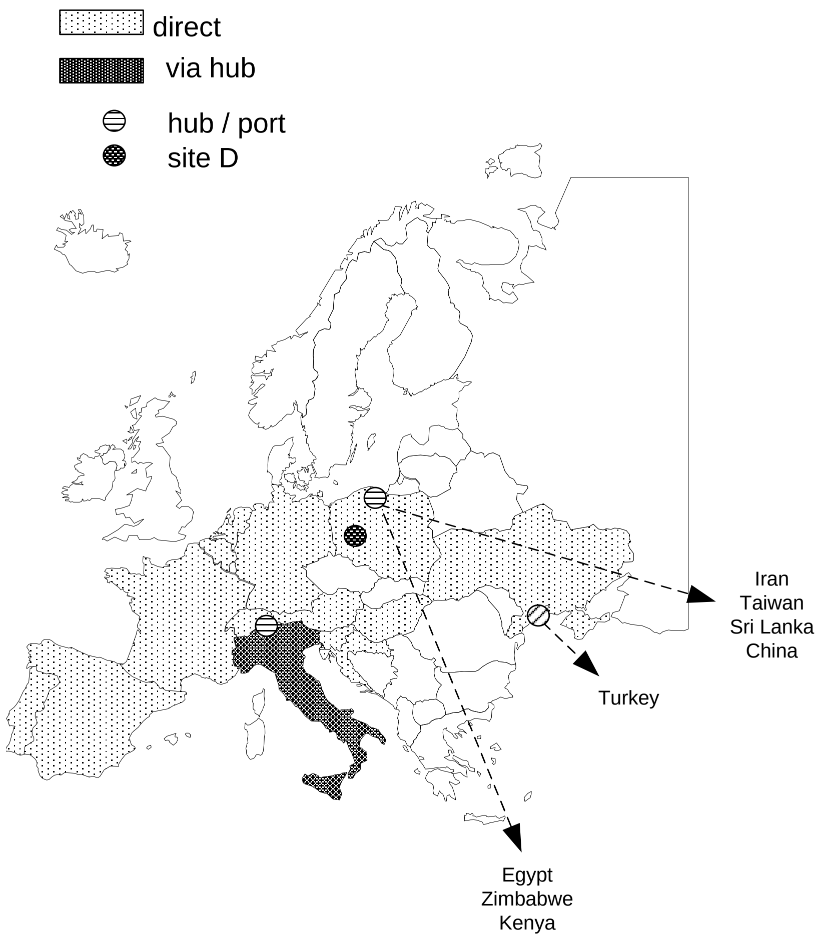

Figure 11.1.: Current distribution network of site D

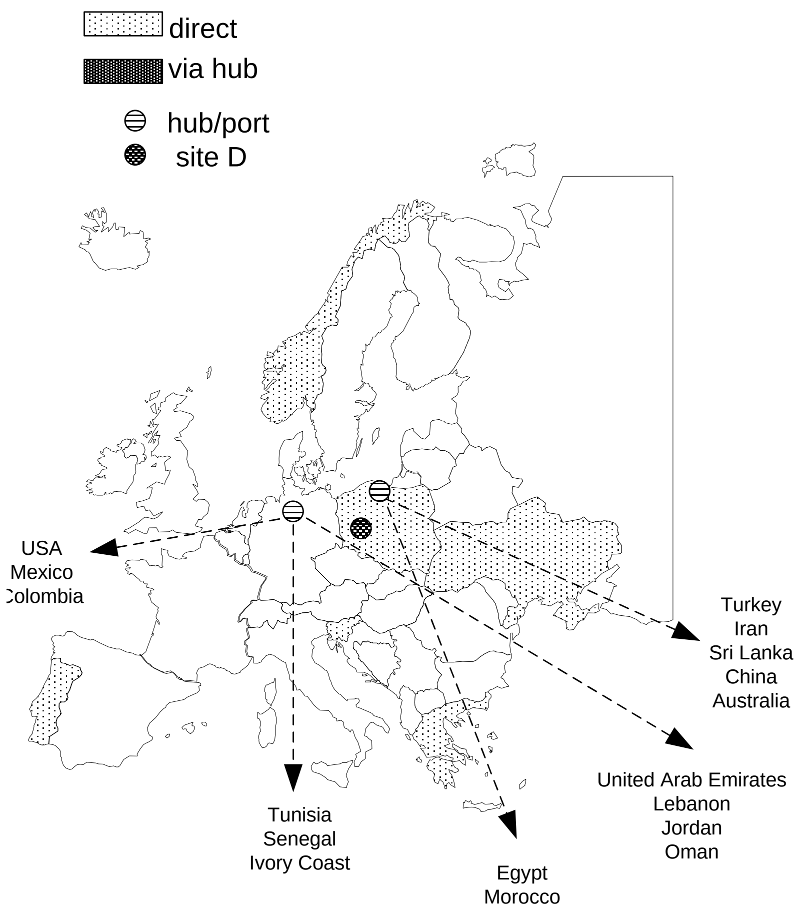

Figure 11.2.: Optimized distribution network of site D

deliveries, and of direct and indirect deliveries, for each manufacturing site.

Table 11.1.: Optimal distribution in percentage of deliveries within Europe and oversea deliveries; the relation of direct to indirect deliveries is reported as well

site	sum (tons)	Europe	oversea	direct	via hub	via port
A	209,890	48%	52%	33%	15%	52%
B	200,760	37%	63%	19%	18%	63%
C	125,070	82%	18%	82%	0%	18%
D	92,940	23%	73%	27%	0%	73%

Applying the optimization model to the demand rates of the subsequent year lead to potential reductions of transportation cost of € 561,000 per year. The quite low optimization potential confirmed the company authorities in their opinion to leave the distribution network unchanged for the moment. For us this result was not surprising since, given the huge amount of side constraints, the range of possible changes within the distribution strategy was fairly tight. Especially the strict assignments of production amounts for each site diminished a number of improving strategies considerably. Thus, we decided to conduct some further calculations and thus to evaluate scenarios based on different relaxations of side constraints. Beside that we implemented a sensitivity analysis in order to use the optimization model's dual information as well. In fact, we tried to figure out which of the manufacturing sites should be favorized in case of possible capacity extensions. The sensitivity analysis showed high shadow prices for sites A and D which lets us conclude that the increase of production capacities for sites A and D is more preferable than for sites B and C. This was confirmed by the results of the subsequent calculations where we evaluated scenarios with loosened capacity restrictions.

11.2. Flexibility scenarios

The first scenario focused on the predefined product mix for each site. This restrictive assumptions we relaxed by adding product IX, a very high demanded one, to the portfolio of manufacturing site A. The examination of this setting was requested by the company since the extension of site A's product portfolio was at that time an open issue for them. Our calculations lead to the conclusion that realizing this setting would be a good idea; in the optimal case site A would no longer have product V in its portfolio but adopted the complete amount of product IX instead. Manufacturing of product V is completely overtaken by site D. This scenario lead to reductions of distribution cost of € 1,295,000 in the base year and of € 1,522,000 in the subsequent year. These high reductions are due to the fact that product IX is highly demanded in Central Europe, which is the location of site A as well, and almost not demanded in Northern Europe where site D, the single supplier of product IX in the current situation, is settled.

A further extension of this setting is specified as the second flexibility scenario. In addition to the increased product portfolio of site A we eliminated all side constraints related to individual requests of single customers (e.g. the strict prevention of direct deliveries to some customers). The already mentioned limitation of direct deliveries from one of the sites to a total amount of 60,000 tons per year was omitted within this scenario as well. This time we ended up with reduced transportation costs of €313,000 in the base year and of €713,000 in the subsequent year.

Within the third scenario we implemented the consideration of variable production costs. In order to enable shirtings of production volumes between sites we doubled the capacities for all of them. Due to missing detail information on production costs we assumed that variable costs do not differ in product type. The optimization model found a solution leading to a reduction of total transportation cost of €300,000 in the base year and €500,000 in the subsequent year. Furthermore, we were able to identify a reflection of the information provided by the sensitivity analysis: sites A and D ended up with significantly increased production volumes while sites B and C provided a strongly decreased output. However, the quite small savings in terms of distribution cost prove the reasonable assignment of production capacities in the current situation.

For the final scenario we loosened all side constraints related to the product portfolios at all manufacturing sites by assuming that all sites are provided to supply all kinds of products. The differences concerning product quality between the sites was neglected as well, all being €2,092,000 under those of scenario 3. Hence, compared to the basic case we reached a transportation cost reduction of €4,000,000 for the base year. For the subsequent year Scenario 4 lead to an additional cost reduction of €1,765,000 in comparison to scenario 3, and of €4,500,000 compared to the initial case. See Table 11.2 for an overview of the cost reductions found for each scenario. The respective bar diagram is illustrated in Figure 11.3.

Table 11.2.: Potential reductions of total transportation cost for each scenario compared to the basic scenario

scenario	description	reduction (base year)	reduction (subsequent year)
1	extended product mix a site A	€1,295,000	€1,522,000
2	relaxed customer specific constraints	€1,608,000	€2,235,000
3	increased production capacities	€1,908,000	€2,735,000
4	relaxed remaining constraints	€4,000,000	€4,500,000

An implementation of one of those scenarios in reality is, due to company internal reasons, difficult at the moment. The analysis was worth the effort anyway, since it distinctively reflects optimization potential for different strategic directions.

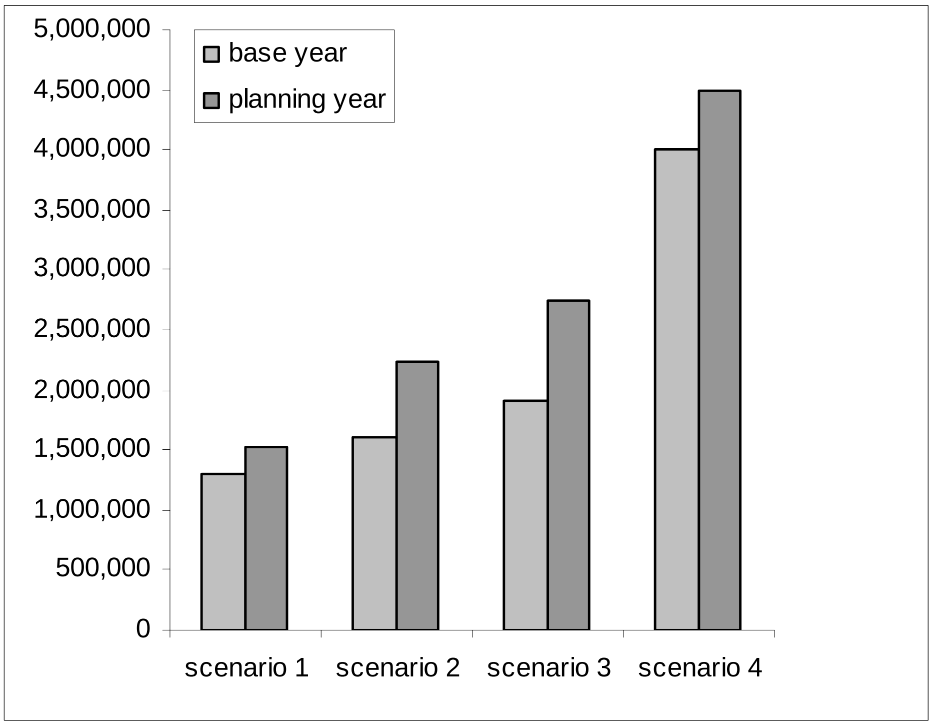

Figure 11.3.: Bar diagram illustrating potential cost reductions (in Euros) for all scenarios

12. Dynamic aspects

Based on the previously described examinations we decided to additionally take the dynamic complexity into account. After extending the available database by various aspects we face dynamic and stochastic settings including nonlinear cost functions in production. Since finding an optimal solution for a problem of such a complexity is not possible within reasonable computation times we used *SimOpt* for generating heuristic solutions and furthermore aimed at getting rid of negative effects, e.g. *Bullwhip* effect, while simultaneously reducing total operational cost and delayed deliveries. We finally compared the results of our combined approach to those of a comprise stand-alone method. This comparison was used to give further proof that applying *SimOpt* definitively makes sense even, or rather primarily, for highly complex networks. Especially the superiority of central planning instead of decentral planning is once more highlighted.

12.1. Data for dynamic-based analyses

Beside the fact that almost all parameters had to be redefined on a time-period-based level conducting experiments under consideration of dynamics meant to make a number of adaptions within our database. Not all of them are worth mentioning here but the most critical of them should be listed anyway:

- model extension by a set of (in the static model unconsidered) raw material suppliers aiming at a better alignment with the current *SimOpt* implementation
- calculation of reasonable initial inventory levels for each product type and each participant
- determination of cost and capacity values for each time period and each participant; this was primarily done by using estimations based on the given static data set
- generating the customer's demand rate within the planning horizon; here we used the delivery schedules provided by the sites' logistics authorities to estimate the points in time where customer demand occurs
- estimation of transportation time delays for all legs within the network based on the previously generated distance matrix; within the simulation these are extended by a stochastic component

- due to simplification reasons and in order to facilitate solution analysis we did not allow direct deliveries from manufacturing sites to customers for all dynamic-based calculations; thus, all transportation lots were forced to pass exactly one hub or port on their way to the customers; further adaptions were made to generate uniformly distributed demand occurrence
- we omitted the fixed product portfolios, i.e. we assumed that each product type can be manufactured at each of the four manufacturing sites
- as a matter of fact we estimated fixed production costs for each manufacturing sites

Clearly, due to the complexity of this model, especially while taking binary decisions into account, it was inevitable to apply a *SimLP* approach, i.e. to couple the simulation model with a pure linear optimization model. By applying *SimLP* we succeed in finding a solution for this complex planning problem within a computation time of approximately 10 minutes. However, we were not able to provide a valuable contribution here since there are no alternative solutions available. Even a comparison to the result obtained for the static case makes no sense. Too much adaptions and assumptions have already distorted the underlying data sets. Thus, we wanted to change the direction and conducted experiments used to quantify implications of changing customer behaviour on operational decisions of the supply chain participants. For this purpose we defined two different scenarios: (i) all customer's request products continuously over time and with no significant changes in their order amounts; (ii) customers' requests still occur continuous in time but the ordering amounts are highly fluctuating. In fact, for case (ii) demand is concentrated in the middle of the planning horizon where really large order amounts arise. Very small lots are requested before and after this peak in demand. However, in both situations we face the same total demand and average order size. The planning horizon is 50 time periods. In Appendix C the demand schedules for both ordering situations are displayed.

In order to generate significant statements on the dynamic experiments our study was twofold:

- we examined the implications of continuous and of bundled demand, i.e. situations (i) and (ii), on operational costs and decisions by means of a stand-alone simulation model; this model was mainly characterized by more sophisticated decisions rules compared to those usually used within our *SimOpt* applications
- we applied *SimLP* to the dynamic-based dataset and again figured out implications on operational costs and decisions when facing bundled demand instead of continuous one

Before going into depths with the stand-alone simulation model some reasonable innovations of the *SimOpt* implementation should be presented. These refer to the class *SCMContainerClass* which has been introduced briefly in Chapter 5. While for the quite small *SimOpt* experiments discussed so far there was no need to use container items, for the following they were essential. We incorporated them in order to generate a couple of submodels instead of a single unmanageable model, which would have lead to out-of-memory problems and an unoverviewable situation for sure. Figure 12.1 visualizes the main view on the simulation model where, apart of manufacturing sites and suppliers, all items are packed into containers. Again, we have dashed lines indicating information flow and solid lines indicating material flow. Container items (an exemplary one is illustrated in Figure 12.2) provide the possibility to implement an helpful aggregation of the model without any loss of precision. In our case, particularly the reduction of necessary transportation items helped a lot. The optimization model embedded within *SimOpt* did not need any adaptions at all since its general LP formulation covers the given situation sufficiently.

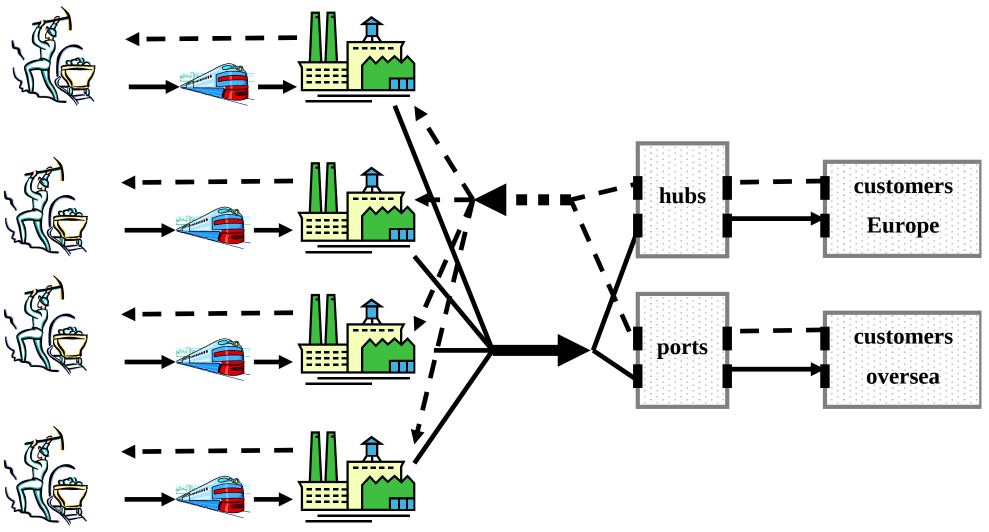

Figure 12.1.: Main view on the distribution network's simulation model using container items

Having adapted the *SimOpt* application which we implemented the stand-alone simulation method which was intended to provide us comparable reference solutions. So we again took our supply chain library and rebuilt the paper manufacturer's distribution network once more. This time we wanted to simulate the situation assuming there is no central planning but a decentral control of participants for themselves. This means that each intermediate object, i.e. manufacturing site, hub, or port, determines both their unique reorder point and order quantity due to predefined decision rules. These rules are more sophisticated than the simple (s.S)-policy used within the initial runs of *SimOpt* and are based on the reorder point policy presented by Ghiani, Laporte, and Musmanno (2004). Here the inventory is observed continuously and an order is sent as soon as the current inventory level

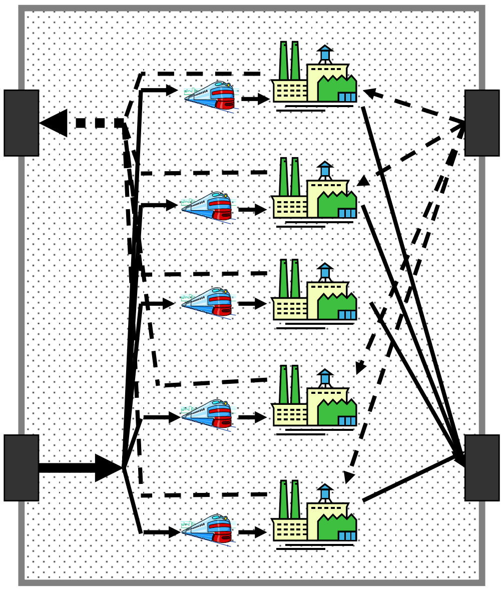

Figure 12.2.: Insight into a container item

reaches the reorder point l. Calculation of l is based on the assumption that the inventory level is nonnegative during lead time t_l with probability α and uses the following formula:

$$l = \bar{d}\bar{t}_l + z_\alpha \sqrt{\sigma_d^2 \bar{t}_l + \sigma_{t_l}^2 \bar{d}^2}, \tag{12.1}$$

where d is the normally distributed demand rate with expected value $\bar{d}$ and standard deviation σ_d, t_l the lead time with expected value $\bar{t}_l$ and standard deviation σ_{t_l}, and finally z_α the value under which a standard normal random variable falls with probability α. The latter denotes the desired service level. Since within *SimOpt* we usually assume a risk averse behaviour, i.e. we estimate time delays based on a quantile above 0.9% (cf. Chapter 3), we fixed z_α with value 3 which comes up with a service level of 99.987%. The expected lead time and standard deviation of lead times are logged during the simulation runs and used for a periodic update of each participant's unique reorder point. The order quantity $q^*$is determined according to the *economic order quantity (EOQ)*:

$$q^* = \sqrt{\frac{2k\bar{d}}{h}}, \tag{12.2}$$

with k being the fixed reorder cost and h the variable holding cost. Both reorder point and order quantity are usually determined on a product-based level, i.e. each participant calculates both of them for each type of product.

In order to facilitate solution interpretations it might be useful to once more have a look on the network structure; this time on a higher aggregation level. In Figure 12.3 we see four layers each of them representing one group of supply chain participants, which are raw material providers, manufacturing sites, hubs and ports, and finally the customers at the downstream end of the chain. Each block of participants is connected by an arc, representing a group of transportation legs, to the subsequent layer within the network. Group T-I connecting suppliers with manufacturing sites, group T-II representing all legs between sites and hubs or ports, and group T-III for the connections between hubs or ports and the customer regions. It has already been mentioned that we considered fixed costs for these experiments, these exclusively occur at the manufacturing sites. Transportation times were deterministic for legs T-I, but are subject to stochasticity for legs T-II and T-III.

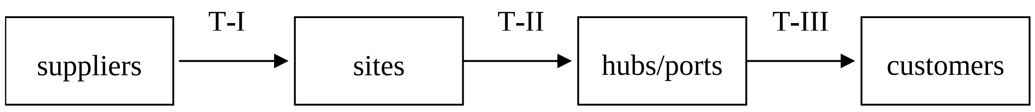

Figure 12.3.: Layers of the given distribution network

For the stand-alone simulation model we have customers sending requests to hubs or ports according to their given demand schedule; and hubs and ports ordering products from manufacturing sites according to their periodically revised reorder point policy. And finally manufacturing sites, also following an adaptive reorder point policy, requesting products at the raw material suppliers.

12.2. Results

Using both the stand-alone simulation with the adaptive ordering policy and the combined simulation-optimization approach we generated a solution for scenario (i), i.e. continuous demand, on the one hand and for scenario (ii), i.e. bundled demand, on the other hand. Thus, we tried to figure out the ability of each method to react adaptively on changing customer behaviour. Clearly, the combined approach, i.e. *SimLP*, having an advantage in information terminates with a solution that is out of reach for the autonomous simulation. This is neither surprising nor does it provide an interesting added value, hence, a direct comparison between these two methods does not make much sense. Table 12.1 shows the results for both scenarios and both methods. The given values in each case represent average values over the final 5 simulation runs.

We see that *SimLP*, i.e. the combined simulation-optimization method, can easily cope the changing demand behaviour. Both the continuous and the bundled demand situations lead to almost the same result showing no significant differences. For the latter *SimLP* even found in average for all parameters slightly better solutions than for the continuous case. This is due to the fact that the small ordering amounts before and after the few bundled orders in the middle of the time horizon lead to a lower capacity utilization than the continuously occurring medium sized requests.

Table 12.1.: Results (autonomous simulation and *SimLP*) for the continuous demand and the bundled demand scenario (costs in Euros, amounts in tons; hp - hubs and ports, s - sites)

	autonomous simulation		*SimLP*	
	scenario (i)	scenario (ii)	scenario (i)	scenario (ii)
total cost	34,505,450	38,257,760	9,256,401	9,194,846
invent. cost (s)	66,537	238,578	399	210
invent. cost (hp)	2,752,838	6,364,746	545,028	541,620
prod. volume	4,847	5,910	2,302	2,118
prod. orders	85	102	44	43
prod. cost	245,643	311,216	56,112	54,442
transp. cost T-I	5,050	6,435	2,250	2,066
transp. cost T-II	237,199	291,829	99,097	86,487
transp. cost T-III	233,397	244,287	151,155	166,290
delays (amount)	104	276	90	39
delays (lots)	6	9	10	5
penalties	103,800	276,400	90,400	39,400

Especially for the delayed deliveries the difference is significant. The autonomous simulation terminated with a different outcome. Here we were not able to adapt sufficiently on the changed demand situation and thus end up with an much worse result for the second situation than in the continuous setting. While the difference in total costs is just about 10% we have an increase of total inventory level at sites of 258% and for hubs/ports of 131%. All production related parameters went up around 20%. Transportation costs climbed up as well and total penalty costs and delayed deliveries' increase is far above 100%. Since we, due to the observation of the *SimLP* solution, know that the bundled demand scenario actually lead to a surplus of capacities it might be quite surprising that the autonomous simulation could not handle the bundled situation. On the contrary, it performed really worse compared to the balanced demand situation. Those bad results for the stand-alone simulation occured due to a kind of *Bullwhip* effect, i.e. a lack of information between all network actors (cf. Towill, Zhou, and Disney (2007)). This causes an increase in variability of inventory levels when traveling the supply chain in the upstream direction occurring due to anticipation of demand behaviour on various layers within a decentralized supply chain.

We already know that orders are sent based on the previously described adaptive reorder point policy. So each intermediate object estimates the expected lead time and the corresponding standard deviation for each product; this worked perfectly well in the first third of the time horizon. The situation changed shortly after the customers had requested their bundled orders. Hubs and ports were left with almost empty inventories and consequently started to send a number of requests in order to reach a level above the reorder point again. This high frequency of requests on return overstrained transaction, transportation and production capacities within the manufacturing sites' and suppliers' layers. Capacity shortages imply increased lead times and these again forced the subsequent participants to raise their specific reorder points. Thus, they sent even more requests and caused further increases in lead times. Although there were just a few bundled orders, the participants winded each other up; even though all customer orders in the third and final part of the planning were of very small size, this circle could not be broken. It is well known that this effect is the stronger the nearer a layer is to the final elements of a network. Thus, we had the already mentioned significant increases in inventory costs of 131% and 258% at hubs/ports and manufacturing sites, respectively. Central planning, of course, enables the ability to avoid such disadvantageous oscillations and is therefore preferable when trying to handle large supply chains efficiently. Managing them is a challenging task but by means of the presented real world case we state that applying the combined simulation-optimization approach *SimOpt* definitely is an attractive option for generating good solutions on an operational and also tactical level.

13. Conclusion and further research

Analyzing and improving supply chains are challenging tasks and several approaches have already been proposed in literature. In this thesis a new hybrid approach combining the advantages of complex simulation models and abstract optimization models has been presented. We implemented a toolbox allowing to easily create simulation models for any kind of supply chain network setting. A special technique enables the inclusion of optimization models into the simulation framework. Within this framework it is possible to test the optimal deterministic solution concerning it's feasibility in the context of a more realistic stochastic representation. By conducting an iterative procedure we are aiming on improved decisions on the tactical and operational level.

Based on a number of numerical experiments it has been shown that this method is able to generate competitive solutions quickly; even compared with traditional planning approaches that are much more time consuming. The main findings can be summarized as follows:

- In many cases the *SimLP* method seems to be a good trade-off between solution quality and computation time. If the nonlinear elements in the model are dominating it is better to apply the *SimMIP* approach and consider these nonlinearities in the optimization model as along as the computation time for solving the optimization model is acceptable.

- Furthermore, we investigated the impact of safety times for delays on the solution quality. If we use the 90%-quantile, we can generate robust plans, but for specific situations we might get better results with less safety time. Only for the case where stochastic is near the customer, the 90%-quantile is clearly the best. Nevertheless, the choice of the quantile depends on the structure of the supply chain and has to be fine-tuned in each case.

- Using the 50%-quantile, i.e. the expected values for the delays, always leads to pure results. If the uncertainty is concentrated far away from the customer, the cost increase by using the expected value is about 10% whereas the increase in case almost 50% if the uncertainty occurs close to the customer.

We used this combined simulation-optimization approach in the context of a real world case study where we have been asked to analyze the efficiency of a global distribution network. While the static deterministic consideration was tackled by the application of a pure linear network flow model, the stochastic and nonlinear case was far too complex to be solved by a stand-alone MIP model. Thus, we used *SimLP* to generate reasonable results quickly.

Further research for different aspects of this method is still possible and necessary. Especially the aggregation step and the generation of new decision rules offer open fields for further investigations. One possibility would be to interpret the solution of the optimization model only as a target strategy and use adaptive decision rules to approximate this target strategy in an uncertain environment. Thus, compensation of stochastic customer behaviour becomes possible. The use of sensitivity results of the optimization model might lead to improved decision rules as well. Investigations on preferable quantiles used for the parameter estimations may lead to general findings referring to safety times and safety stocks in specific stochastic environments. Further investigations are possible for the boundaries between the simulation and the optimization model. If more complex models are used, other fast solution methods (e.g. heuristics, metaheuristics, etc.) should be included in order to speed-up the determination of the MIP solutions. Another possible extension would be to take strategic decisions like, for example, location problems into account.

List of Tables

List of Figures

Bibliography

[1] Almeder C, Preusser M (2004) Mixed Analytical / DEVS Approach to ARGESIM Comparison C14 'Supply Chain Management' using Xpress-MP and Anylogic. SNE Simulation News Europe 41/42: 50

[2] Almeder C, Preusser M, Hartl RF (2008) Simulation and optimization of supply chains: alternative or complementary approaches? OR Spectrum (online available since Jan 12th, 2008), DOI: 10.1007/s00291-007-0118-z (original article on www.springerlink.com)

[3] Alonso-Ayuso A, Escudero LF, Garín A, Ortuño MT, Pérez G (2003) An Approach for Strategic Supply Chain Planning under Uncertainty based on Stochastic 0-1 Programming. Journal of Global Optimization 26: 97-124

[4] Arbib C, Marinelli F (2005) Integrating process optimization and inventory planning in cutting stock with skiving option: An optimization model and its application. European Journal of Operational Research 163: 617-630

[5] Biswas S, Narahari Y (2004) Object oriented modeling and decision support for supply chains. European Journal of Operational Research 153: 704-726

[6] Cachon HP, Lariviere MA (2001) Contracting to Assure Supply: How to Share Demand Forecasts in a Supply Chain. Management Science 47: 629-646

[7] Chen F(1999) Decentralized Supply Chains Subject to Information Delays. Management Science 45: 1076-1090

[8] Chen F, Federgruen A, Zhen Y (2001) Coordination Mechanisms for a Distribution System with One Supplier and Multiple Retailers. Management Science 47: pp. 693-708

[9] Dogan K, Goetschalckx M (1999) A primal decomposition method for the integrated design of multi-period production-distribution systems. IIE Transactions 31: 1027-1036

[10] Erengüc SS, Simpson NC, Vakharia AJ (1999) Integrated production/distribution planning in supply chains: an invited review. European Journal of Operational Research 115: 219-236

[11] Fleischmann B (1993) Designing distribution systems with transport economies of scale. European Journal of Operational Research 70: 31-42

[12] Fleischmann B, Meyr H, Wagner M (2005). Advanced Planning. In: Stadtler, Kilger (eds) Supply chain management and advanced planning - concepts, models, software and case studies, Springer: 62-77

[13] Fu MC (2002) Optimization for Simulation: theory vs practice. INFORMS Journal on Computing 14: 192-215

[14] Ghiani G , Laporte G, Musmanno M (2004) Introduction to Logistics Systems Planning and Control. John Wiley & Sons

[15] Glover F, Kelly JP, Laguna M (1999) New advances for wedding optimization and simulation. In: Farrington PA, Nembhrad HB, Sturrock DT, Evans GW (eds.) Proceedings of the 1999 Winter Simulation Conference, 255-260

[16] Gronalt M, Hartl, RF, Preusser M (2007) Design eines globalen Liefernetzes für Papier - Eine Fallstudie. In: Corsten H, Missbauer H (eds), Produktions- und Logistikmanagement, pp. 597-616. Vahlen Verlag

[17] Gunnarsson H, Rönnqvist M, Carlsson D (2007) Integrated production and distribution planning for Södra Cell AB. Journal of Mathematical Modelling and Algorithms 6, 25-45

[18] Jayaraman V, Pirkul H (2001) Planning and coordination of production and distribution facilities for multiple commodities. European Journal of Operational Research 133: 394-408

[19] Kelton WD, Sadowski RP, Sadowski DA (2002) Simulation with Arena, 2nd edn. McGraw-Hill

[20] Kleijnen JPC (2005) Supply chain simulation tools and techniques: a survey, International Journal of Simulation & Process Modelling 1: 82-89

[21] Kodnar M (2007) Cooperation models and Coordination in Supply networks. Diploma thesis, University of Vienna

[22] Kuhn A, Rabe M (1998) Simulation in Produktion und Logistik (Fallbeispielsammlung), Springer

[23] Lee YH, Kim SH (2002) Production-distribution in supply chain considering capacity constraints. Computers & Industrial Engineering 43: 169-190

[24] Lee HL, Ng SM (1998) Preface to global supply chain and technology management. In: Lee HL, Ng SM (eds) Global supply chain and technology management, POMS series in technology and operations management, Miami, Florida: 1-3

[25] Lee H, Wang S (1999) Decentralized Multi-Echelon Supply Chains: Incentives and Information. Management Science 45: 633-640

[26] Leung SCH, Tsang SOS, Ng WL, Wu Y (2007) A robust optimization model for multi-site production planning problem in an uncertain environment. European Journal of Operational Research 181: 224-238

[27] Meyr H (2002) Simultaneous lotsizing and scheduling on parallel machines. European Journal of Operational Research 139: 277-292

[28] Mitrovic D (2006) CPLEX, XPRESS-MP, GLPK Comparison by means of Supply Chain Network Models. Diploma thesis, Vienna University of Technology

[29] Pankaj C, Fisher ML (1994) Coordination of production and distribution planning. European Journal of Operational Research 72: 503-517

[30] Paraschis IN (1989) Optimale Gestaltung von Mehrproduktsystemen. Physica-Schriften zur Betriebswirtschaft 26, Physica-Verlag Heidelberg

[31] Preusser M, Almeder C, Hartl RF, Klug M (2005a) LP Modelling and simulation of supply chain networks. In: Günther HO, Mattfeld DC, Suhl L (eds) Supply Chain Managament und Logistik: Optimierung Simulation, Decision Support, pp 95-114. Physica-Verlag

[32] Preusser M, Almeder C, Hartl RF, Klug M (2005b) Hybrid Supply Chain Modelling - Combining LP-Models and Discrete-Event Simulation. In: Maroto Álvarez MC, Alcaraz Soria J, Ruiz GR, Crespo Abril F, Vallada R (eds) ORP3 - Operational Research Peripatetic Postgraduate Programme proceedings, pp 163-170. ESMAP, S.L., Universidad Politécnica de Valencia

[33] Santoso T, Ahmed S, Goetschalckx M, Shapiro A (2005) A stochastic programming approach for supply chain network design under uncertainty. European Journal of Operational Research 167: 96-115

[34] Schneeweiss C (2003) Distributed Decision Making, 2nd edn. Springer

[35] Shapiro JF (1993) Mathematical Programming Models and Methods for Production Planning and Scheduling, In: Graves SC, Rinnooy Kan AHG, Zipkin PH (eds) Handbooks in Operations Research and Management Science: Logistics of Production and Inventory, North-Holland

[36] Stadtler H (2005) Supply chain management and advanced planning - basics, overview and challenges. European Journal of Operational Research 163: 575-588

[37] Stäblein T, Baumgärtel H, Wilke J (2007) The Supply Net Simulator SNS: An artificial intelligence approach for highly efficient supply network simulation. In: Günther HO, Mattfeld DC, Suhl L (eds) Management logistischer Netzwerke, pp 85-110. Physica-Verlag

[38] Swisher JR, Jacobson SH, Hyden PD, Schruben LW (2000) A survey of simulation and optimization techniques and procedures. In: Joines JA, Barton RR, Kang K, Fishwick PA, (eds) Proceedings of the 2000 Winter Simulation Conference, pp 119-128

[39] Towill DR, Zhou L, Disney SM (2007) Reducing the bullwhip effect: Looking through the appropriate lens. International Journal of Production Economics 108: 444-453

[40] Truong TH, Azadivar F (2003) Simulation based optimization for supply chain configuration design. In: Chick S, Sánchez PJ, Ferrin D, Morrice DJ (eds) Proceedings of the 2003 Winter Simulation Conference, pp 1268-1275

[41] Tsiakis P, Shah N, Pantelides CC (2001) Design of multi-echelon supply chains under demand uncertainty.Ind.Eng.Chem.Res

[42] Vidal CJ, Goetschalckx M (2001) A global supply chain model with transfer pricing and transportation cost allocation. European Journal of Operational Research 129: 134-158

[43] Weng ZK (1995) Channel Coordination and Quantity Discounts. Management Science 41: 1509-1522

[44] Werr H, Scheuerer S (2007) Reorganisation eines Distributions- und Beschaffungsnetzwerks. In: Günther HO, Mattfeld DC, Suhl L (eds) Management logistischer Netzwerke. Entscheidungsunterstützung, Informationssysteme und OR-Tools, pp 23-44. Physica-Verlag

[45] Yaged B (1971) Minimum cost routing for static network models. Networks 1:139-172

A. Notation for the optimization model

A.1. General network structure

$J = J_S \cup J_I \cup J_C$	set of supply chain participants
$j \in J_S$	suppliers
$j \in J_I$	intermediate nodes (production/transaction)
$j \in J_C$	customers
$p \in P$	products
$t \in T$	time periods
$v \in V$	available transportation modes

A.2. Decision variables

$m_i^p(t)$	amount of product p manufactured in period t at intermediate i
$u_i^p(t)$	amount of product p transacted in period t at intermediate i
${}^v x_{ij}^p(t)$	amount of product p transported with mode v on leg ij in period t

A.3. Costs, delays, and general parameters

a_i^p	capacity consumption factor for product p manufactured at intermediate i
$\alpha_i^p(p')$	amount of raw material p' required in order to produce one unit of product p at intermediate i
${}^v C_{ij}(t)$	maximum global transportation amount of mode v on leg ij in period t
$C_{ij}^p(\cdot)$	cost function for transporting product p with mode v on transportation leg ij
${}^{prod} C_i(t)$	maximum global production capacity at intermediate i in period t

${}^{v}Cap_{ij}^{p}(t)$	maximum amount of product p that can be transported with mode v in period t
${}^{invin}Cap_{i}^{p}(t)$	maximum inventory level of product p at the inbound inventory of intermediate i in period t
${}^{invout}Cap_{i}^{p}(t)$	maximum inventory level of product p at the outbound inventory of intermediate i in period t
${}^{prod}Cap_{i}^{p}(t)$	maximum production amount of product p in period t and at intermediate i
${}^{ta}C_{i}(t)$	maximum global transaction capacity at intermediate i in period t
${}^{ta}Cap_{i}^{p}(t)$	maximum transaction amount of product p in period t and at intermediate i
$D_{i}^{p}(t)$	demand for product p occurring at customer i in period t
d_{i}^{p}	capacity consumption factor for product p transacted at intermediate i
δ_{i}^{p}	production delay for product p at intermediate i
${}^{v}g^{p}$	capacity consumption factor for product p being transported with mode v
${}^{out}H_{i}^{p}(\cdot)$	cost function for storing product p at the outbound inventory of participant i
${}^{in}H_{i}^{p}(\cdot)$	cost function for storing product p at the inbound inventory of participant i
${}^{in}L_{i}(t)$	maximum global inventory level at the inbound inventory of intermediate i in period t
${}^{out}L_{i}(t)$	maximum global inventory level at the outbound inventory of intermediate i in period t
q_{i}^{p}	capacity consumption factor for product p being stored at intermediate i
$R_{i}^{p}(\cdot)$	cost function referring to backorders of product p at customer i
$r_{i}^{p}(t)$	extraordinary inflow of product p at the inbound inventory of participant i in period t
$S_{i}^{p}(t)$	supply of product p occurring at supplier i in period t
$s_{i}^{p}(t)$	extraordinary inflow of product p at the outbound inventory of intermediate i in period t
σi^{p}	transaction delay for product p at intermediate i
TC_{i}^{S}	total cost of supplier i
TC_{i}^{I}	total cost of intermediate i
TC_{i}^{C}	total cost of customer i
TC_{ij}^{F}	total cost on transportation leg itshape ij
τ_{ij}	transportation delay on leg ij
$W_{i}^{p}(\cdot)$	cost function referring to the production of product p at intermediate i

$Z_i^p(\cdot)$	cost function referring to the transaction of product p at intermediate i

A.4. Auxiliary variables

${}^{in}b_i{}^p(t)$	backorders product p at customer i in period t
${}^{in}f_i^p(t)$	amount of product p arriving at participant i in period t
${}^{out}f_i^p(t)$	amount of product p sent away of participant i in period t
${}^{out}l_i^p(t)$	outbound inventory level of product p at participant i in period t
${}^{in}l_i^p(t)$	inbound inventory level of product p at participant i in period t
$\chi_{t\geq\delta_i^p}$	indicator function which has value 1 if the production delay of product p at intermediate i is smaller or equal to the current period t and 0 otherwise
$\chi_{t\geq\sigma_i^p}$	indicator function which has value 1 if the transaction delay of product p at intermediate i is smaller or equal to the current period t and 0 otherwise

A.5. Parameters used for *SimLP* only

${}^{v}c_{ij}^p$	cost factor for transporting product p with mode v in leg ij
${}^{out}h_i^p$	cost factor for storing product p in the outbound inventory of participant i
${}^{in}h_i^p$	cost factor for storing product p in the inbound inventory of participant i
ρ_i^p	cost factor for backorders of product p at customer i
w_i^p	cost factor for producing product p at intermediate i
z_i^p	cost factor for transacting product p at intermediate i

A.6. Parameters used for *SimMIP* only

${}^{v}E_{ij}^p$	given bound for an interval of a step function referring to the transportation amount of product p on mode v and leg ij
${}^{v}e_{ij}^p$	fixed transportation cost on leg ij in case the load of product p on mode v is within interval $({}^{v}K_{ij}^p, {}^{v}E_{ij}^p]$
${}^{v}\epsilon_{ij}^p(t)$	binary decision variable indicating if the amount of product p transported with mode v on leg ij in period t is within the given limit $({}^{v}K_{ij}^p, {}^{v}E_{ij}^p]$ or not

${}^{v}\gamma_{ij}^{p}(t)$	binary decision variable indicating if the amount of product p transported with mode v on leg ij in period t is within the given limit $({}^{v}E_{ij}^{p}, {}^{v}Y_{ij}^{p}]$ or not
${}^{v}K_{ij}^{p}$	given bound for an interval of a step function referring to the transportation amount of product p on mode v and leg ij
${}^{v}k_{ij}^{p}$	fixed transportation cost on leg ij in case the load of product p on mode v is within interval $(0, {}^{v}K_{ij}^{p}]$
${}^{v}\kappa_{ij}^{p}(t)$	binary decision variable indicating if the amount of product p transported with mode v on leg ij in period t is within the given limit $(0, {}^{v}K_{ij}^{p}]$ or not
M	large number
N	large number
n_{i}^{p}	fixed production cost for product p at intermediate i
o_{i}^{p}	fixed transaction cost for product p at intermediate i
$\phi_{i}^{p}(t)$	binary variable indicating if product p is manufactured at intermediate i in period t
$\psi_{i}^{p}(t)$	binary variable indicating if product p is transacted at intermediate i in period t
${}^{v}Y_{ij}^{p}$	given upper bound for an interval of a step function referring to the transportation amount of product p on mode v and leg ij
${}^{v}y_{ij}^{p}$	fixed transportation cost on leg ij in case the load of product p on mode v is within interval $({}^{v}E_{ij}^{p}, {}^{v}Y_{ij}^{p}]$

B. Test instances

Data records appear according to their corresponding notation used for the optimization model.

B.1. Data corresponding to Figure 6.2

B.1.1. General network structure and parameters

$J_S = \{1\}$
$J_I = \{1\}$
$J_C = \{1\}$
$P = \{1\}$
$T = \{1, \ldots, 6\}$
$V = \{2\}$

B.1.2. Costs and delays

description	estimated parameter	value
variable inventory cost (supplier)	${}^{out}h_1^1$	1
variable inbound inventory cost (intermediate)	${}^{in}h_1^1$	2
variable inbound inventory cost (intermediate)	${}^{out}h_1^1$	3
fixed transaction cost	z_1^1	10
variable transaction cost	z_1^1	15
transaction delay	σ_1^1	1
fixed transport cost (<5 units on mode 2)	${}^{2}c_{11}^1$	200
fixed transport cost (5 units on mode 2)	${}^{2}c_{11}^1$	800
variable transport cost for mode 1	${}^{1}c_{11}^1$	5

transport delay for mode 1	${}^{1}\tau_{11}$	2
transport delay for mode 2	${}^{2}\tau_{11}$	1
variable penalty cost	ρ_1^1	100

B.1.3. General parameters

Only index configurations leading to parameters larger than 0 are reported.

parameter	J_S	J_I	J_C	P	T	V	value
${}^{v}C_{ij}(t)$	1	1			[1;6]	1	10
${}^{v}C_{ij}(t)$		1	1		[1;6]	1	8
${}^{v}C_{ij}(t)$		1	1		[1;6]	2	5
${}^{v}Cap_{ij}^{p}(t)$	1	1		1	[1;6]	1	10
${}^{v}Cap_{ij}^{p}(t)$		1	1	1	[1;6]	1	8
${}^{v}Cap_{ij}^{p}(t)$		1	1	1	[1;6]	2	5
${}^{ta}C_{i}(t)$		1			[1;6]		100
${}^{ininv}Cap_{i}^{p}(t)$		1		1	[1;6]		100
${}^{outinv}Cap_{i}^{p}(t)$		1		1	[1;6]		100
${}^{ta}Cap_{i}^{p}(t)$		1		1	[1;6]		50
d_i^p		1		1			1
$D_i^p(t)$			1	1	5		10
${}^{v}g^{p}$				1		1	1
${}^{v}g^{p}$				1		2	1
${}^{in}L_{i}(t)$		1			[1;6]		1,000
${}^{out}L_{i}(t)$		1			[1;6]		1,000
q_i^p		1		1			1

B.2. D1-L to S6a-H

B.2.1. General network structure

$J_S = \{1\}$
$J_I = \{1\}$
$J_C = \{1\}$
$P = \{1, 2\}$
$T = \{1, \dots, 30\}$
$V = \{1\}$

B.2.2. Costs and delays

description	estimated parameter	value
variable inventory cost (supplier)	${}^{out}h_1^1$	1
variable inbound inventory cost (intermediate)	${}^{in}h_1^1)$	2
variable outbound inventory cost (intermediate)	${}^{out}h_1^p)$	2
production delay	δ_1^2	1
transaction delay	σ_1^1	1
deterministic transport delay	${}^1\tau_{11}$	3
stochastic transport delay towards intermediate	${}^1\tau_{11}$	u[1;9]
stochastic transport delay towards customer	${}^1\tau_{11}$	u[1;5]
variable penalty cost	ρ_1^p	100

B.2.3. Costs used for *SimLP* experiments only

description	estimated parameter	value
variable production cost	w_1^2	30
fixed production cost	w_1^2	50
increased fixed production cost (D1a-L, D6a-H, S1a-L, S6a-H)	w_1^2	1,000
variable transaction cost	z_1^1	15
fixed transaction cost	z_1^1	10
increased transaction cost (D1a-L, D6a-H, S1a-L, S6a-H)	z_1^1	500
fixed transport cost (low capacity consumption)	${}^1c_{11}^p$	100
fixed transport cost (medium capacity consumption)	${}^1c_{11}^p$	200

fixed transport cost (high capacity consumption)	${}^{1}c^{p}_{11}$	300

B.2.4. Costs used for *SimMIP* experiments only

Values marked by (*) refer to instances with increased fixed costs (S1a-L and S6a-H).

parameter	J_I	P	value
w^p_i	1	2	30
z^p_i	1	1	15
n^p_i	1	2	50
o^p_i	1	2	10
n^p_i	1	2	1,000*
o^p_i	1	2	500*

parameter	J_S	J_I	J_C	P	V	value
${}^{v}k^{p}_{ij}$	1	1		1	1	100
${}^{v}k^{p}_{ij}$		1	1	1	1	100
${}^{v}k^{p}_{ij}$		1	1	2	1	100
${}^{v}e^{p}_{ij}$	1	1		1	1	200
${}^{v}e^{p}_{ij}$		1	1	1	1	200
${}^{v}e^{p}_{ij}$		1	1	2	1	200
${}^{v}y^{p}_{ij}$	1	1		1	1	300
${}^{v}y^{p}_{ij}$		1	1	1	1	300
${}^{v}y^{p}_{ij}$		1	1	2	1	300

B.2.5. General parameters

Only index configurations leading to parameters larger than 0 are reported. For the customers' demand only exemplary realizations of the schemes for low demand and high cases are given.

parameter	J_S	J_I	J_C	P	T	V	value
a^p_i		1		2			1
$\alpha^p_i(p')$		1		2			1

${}^{v}C_{ij}(t)$	1	1			[1;30]	1	100
${}^{v}C_{ij}(t)$		1	1		[1;30]	1	60
${}^{prod}C_{i}(t)$		1			[1;30]		1,000
${}^{ta}C_{i}(t)$		1			[1;30]		100
${}^{ininv}Cap_{i}^{p}(t)$		1		1	[1;30]		100
${}^{outinv}Cap_{i}^{p}(t)$		1		1	[1;30]		100
${}^{outinv}Cap_{i}^{p}(t)$		1		2	[1;30]		100
${}^{v}Cap_{ij}^{p}(t)$	1	1		1	[1;30]	1	100
${}^{v}Cap_{ij}^{p}(t)$		1	1	1	[1;30]	1	60
${}^{v}Cap_{ij}^{p}(t)$		1	1	2	[1;30]	1	60
${}^{prod}Cap_{i}^{p}(t)$		1		2	[1;30]		40
${}^{ta}Cap_{i}^{p}(t)$		1		1	[1;30]		40
d_{i}^{p}		1		1			1
${}^{v}g^{p}$				1		1	1
${}^{v}g^{p}$				2		1	1
${}^{in}L_{i}(t)$		1			[1;30]		1,000
${}^{out}L_{i}(t)$		1			[1;30]		1,000
q_{i}^{p}		1		1			1
q_{i}^{p}		1		2			1
$S_{i}^{p}(t)$	1			1	[1;30]		100

parameter	J_C	P	T	value
$D_{i}^{p}(t)$	1	1	13	55
$D_{i}^{p}(t)$	1	1	16	50
$D_{i}^{p}(t)$	1	1	20	35
$D_{i}^{p}(t)$	1	1	27	40
$D_{i}^{p}(t)$	1	1	30	40
$D_{i}^{p}(t)$	1	2	14	55
$D_{i}^{p}(t)$	1	2	17	50
$D_{i}^{p}(t)$	1	2	21	35
$D_{i}^{p}(t)$	1	2	23	40
$D_{i}^{p}(t)$	1	2	28	40

parameter	J_C	P	T	value
$D_{i}^{p}(t)$	1	1	13	59
$D_{i}^{p}(t)$	1	1	16	57
$D_{i}^{p}(t)$	1	1	17	57
$D_{i}^{p}(t)$	1	1	27	40
$D_{i}^{p}(t)$	1	1	20	33
$D_{i}^{p}(t)$	1	1	23	59
$D_{i}^{p}(t)$	1	1	24	31
$D_{i}^{p}(t)$	1	1	27	31
$D_{i}^{p}(t)$	1	1	28	33

$D_i^p(t)$	1	1	30	31
$D_i^p(t)$	1	2	14	59
$D_i^p(t)$	1	2	15	31
$D_i^p(t)$	1	2	18	57
$D_i^p(t)$	1	2	19	57
$D_i^p(t)$	1	2	21	33
$D_i^p(t)$	1	2	22	31
$D_i^p(t)$	1	2	26	59
$D_i^p(t)$	1	2	29	33
$D_i^p(t)$	1	2	28	33
$D_i^p(t)$	1	2	30	31

B.3. L1-L-S to L6a-H-C

B.3.1. General network structure

$J_S = \{3\}$
$J_I = \{4\}$
$J_C = \{3\}$
$P = \{1, 2, 3, 4\}$
$T = \{1, \dots, 30\}$
$V = \{1\}$

B.3.2. Costs and delays

description	estimated parameter	value
variable inventory cost (supplier)	${}^{out}h_i^p$	1
variable inbound inventory cost (intermediate)	${}^{in}h_i^p$	2
variable outbound inventory cost (intermediate)	${}^{out}h_i^p$	2
production delay	δ_i^p	2
transaction delay	σ_i^p	1
deterministic transport delay towards layer 1	${}^1\tau_{ij}$	1
stochastic transport delay towards layer 2	${}^1\tau_{ij}$	u[1;3]
stochastic transport delay towards customer	${}^1\tau_{ij}$	u[1;5]
variable penalty cost	ρ_i^p	100

B.3.3. Costs used for *SimLP* experiments only

description	estimated parameter	value
variable production cost	w_i^p	1
fixed production cost	w_i^p	50

increased fixed production cost (L1a-L-S, L6a-H-S, L-1a-L-C, L6a-H-C)	w_i^p	1,000
variable transaction cost	z_i^p	1
fixed transaction cost	z_i^p	10
increased transaction cost (L1a-L-S, L6a-H-S, L-1a-L-C, L6a-H-C)	z_i^p	500
fixed transport cost	${}^1c_{ij}^p$	50

B.3.4. Costs used for *SimMIP* experiments only

Values marked by (*) refer to instances with increased fixed costs (L-1a-L-C and L6a-H-C).

parameter	J_I	P	value
w_i^p	1	3	1
w_i^p	2	3	1
w_i^p	3	4	1
w_i^p	4	4	1
z_i^p	$\forall i$	1	1
z_i^p	$\forall i$	2	1
z_i^p	3	3	1
z_i^p	4	3	1
n_i^p	1	3	50
n_i^p	2	3	50
n_i^p	3	4	50
n_i^p	4	4	50
o_i^p	$\forall i$	1	10
o_i^p	$\forall i$	2	10
o_i^p	3	3	10
o_i^p	4	3	10
n_i^p	1	3	1,000*
n_i^p	2	3	1,000*
n_i^p	3	4	1,000*
n_i^p	4	4	1,000*
o_i^p	$\forall i$	1	500*
o_i^p	$\forall i$	2	500*
o_i^p	3	3	500*
o_i^p	4	3	500*

parameter	J_S	J_I	J_I	J_C	P	V	value
${}^vk_{ij}^p$	$\forall s$	1			1	1	50
${}^vk_{ij}^p$	$\forall s$	2			1	1	50

${}^{v}k_{ij}^{p}$	$\forall s$	1			2	1	50
${}^{v}k_{ij}^{p}$	$\forall s$	2			2	1	50
${}^{v}k_{ij}^{p}$		1	3		1	1	50
${}^{v}k_{ij}^{p}$		1	3		2	1	50
${}^{v}k_{ij}^{p}$		1	3		3	1	50
${}^{v}k_{ij}^{p}$		1	4		1	1	50
${}^{v}k_{ij}^{p}$		1	4		2	1	50
${}^{v}k_{ij}^{p}$		1	4		3	1	50
${}^{v}k_{ij}^{p}$		2	3		1	1	50
${}^{v}k_{ij}^{p}$		2	3		2	1	50
${}^{v}k_{ij}^{p}$		2	3		3	1	50
${}^{v}k_{ij}^{p}$		2	4		1	1	50
${}^{v}k_{ij}^{p}$		2	4		2	1	50
${}^{v}k_{ij}^{p}$		2	4		3	1	50
${}^{v}k_{ij}^{p}$			3	$\forall c$	$\forall p$	1	50
${}^{v}k_{ij}^{p}$			4	$\forall c$	$\forall p$	1	50

B.3.5. General parameters

Only index configurations leading to parameters larger than 0 are reported. For the customers' demand only exemplary realizations of the schemes for low demand and high cases are given.

parameter	J_S	J_I	J_I	J_C	P	T	V	value
a_i^p		$\forall i$			$\forall p$			1
$\alpha_i^p(p')$		$\forall i$			3 (p'=1)			1
$\alpha_i^p(p')$		$\forall i$			4 (p'=2)			1

${}^{v}C_{ij}(t)$	$\forall s$	1				[1;30]	1	60
${}^{v}C_{ij}(t)$	$\forall s$	2				[1;30]	1	60
${}^{prod}C_{i}(t)$		$\forall i$				[1;30]		1,000
${}^{ta}C_{i}(t)$		$\forall i$				[1;30]		1,000
${}^{ininv}Cap_{i}^{p}(t)$		$\forall i$			$\forall p$	[1;30]		1,000
${}^{outinv}Cap_{i}^{p}(t)$		$\forall i$			$\forall p$	[1;30]		1,000
${}^{v}Cap_{ij}^{p}(t)$	$\forall s$	1			1	[1;30]	1	60
${}^{v}Cap_{ij}^{p}(t)$	$\forall s$	2			1	[1;30]	1	60
${}^{v}Cap_{ij}^{p}(t)$	$\forall s$	1			2	[1;30]	1	60
${}^{v}Cap_{ij}^{p}(t)$	$\forall s$	2			2	[1;30]	1	60
${}^{v}Cap_{ij}^{p}(t)$		1	3		1	[1;30]	1	60
${}^{v}Cap_{ij}^{p}(t)$		1	3		2	[1;30]	1	60
${}^{v}Cap_{ij}^{p}(t)$		1	3		3	[1;30]	1	60
${}^{v}Cap_{ij}^{p}(t)$		1	4		1	[1;30]	1	60
${}^{v}Cap_{ij}^{p}(t)$		1	4		2	[1;30]	1	60
${}^{v}Cap_{ij}^{p}(t)$		1	4		3	[1;30]	1	60

parameter	J_S	J_I	J_I	J_C	P	T	V	value
${}^{v}Cap_{ij}^{p}(t)$		2	3		1	[1;30]	1	60
${}^{v}Cap_{ij}^{p}(t)$		2	3		2	[1;30]	1	60
${}^{v}Cap_{ij}^{p}(t)$		2	3		3	[1;30]	1	60
${}^{v}Cap_{ij}^{p}(t)$		2	4		1	[1;30]	1	60
${}^{v}Cap_{ij}^{p}(t)$		2	4		2	[1;30]	1	60
${}^{v}Cap_{ij}^{p}(t)$		2	4		3	[1;30]	1	60
${}^{v}Cap_{ij}^{p}(t)$		1	3		1	[1;30]	1	60
${}^{v}Cap_{ij}^{p}(t)$		3		$\forall c$	$\forall p$	[1;30]	1	60
${}^{v}Cap_{ij}^{p}(t)$		4		$\forall c$	$\forall p$	[1;30]	1	60
${}^{prod}Cap_{i}^{p}(t)$		1			3	[1;30]		40
${}^{prod}Cap_{i}^{p}(t)$		2			3	[1;30]		40
${}^{prod}Cap_{i}^{p}(t)$		3			4	[1;30]		40
${}^{prod}Cap_{i}^{p}(t)$		4			4	[1;30]		40
${}^{ta}Cap_{i}^{p}(t)$		$\forall i$			1	[1;30]		40
${}^{ta}Cap_{i}^{p}(t)$		$\forall i$			2	[1;30]		40
${}^{ta}Cap_{i}^{p}(t)$		3			3	[1;30]		40
${}^{ta}Cap_{i}^{p}(t)$		4			3	[1;30]		40
d_{i}^{p}		$\forall i$			$\forall p$			1
${}^{v}g^{p}$					$\forall c$		$\forall v$	1
${}^{in}L_{i}(t)$		$\forall i$				[1;30]		1,000
${}^{out}L_{i}(t)$		$\forall i$				[1;30]		1,000
q_{i}^{p}		$\forall i$			$\forall p$			1
$S_{i}^{p}(t)$	$\forall i$				1	[1;30]		100
$S_{i}^{p}(t)$	$\forall i$				2	[1;30]		100

parameter	J_C	P	T	value
$D_i^p(t)$	$\forall c$	1	16	40
$D_i^p(t)$	$\forall c$	1	17	30
$D_i^p(t)$	$\forall c$	1	21	25
$D_i^p(t)$	$\forall c$	1	26	45
$D_i^p(t)$	$\forall c$	1	28	40
$D_i^p(t)$	$\forall c$	2	22	40
$D_i^p(t)$	$\forall c$	2	23	30
$D_i^p(t)$	$\forall c$	2	25	45
$D_i^p(t)$	$\forall c$	2	30	25
$D_i^p(t)$	$\forall c$	2	16	40
$D_i^p(t)$	$\forall c$	3	17	45
$D_i^p(t)$	$\forall c$	3	20	25
$D_i^p(t)$	$\forall c$	3	24	45
$D_i^p(t)$	$\forall c$	3	30	30
$D_i^p(t)$	$\forall c$	4	21	25
$D_i^p(t)$	$\forall c$	4	25	40
$D_i^p(t)$	$\forall c$	4	26	25
$D_i^p(t)$	$\forall c$	4	28	30

parameter	J_C	P	T	value
$D_i^p(t)$	$\forall c$	1	15	49
$D_i^p(t)$	$\forall c$	1	17	47
$D_i^p(t)$	$\forall c$	1	20	47
$D_i^p(t)$	$\forall c$	1	23	49
$D_i^p(t)$	$\forall c$	1	25	47
$D_i^p(t)$	$\forall c$	1	27	21
$D_i^p(t)$	$\forall c$	1	30	21
$D_i^p(t)$	$\forall c$	2	16	21
$D_i^p(t)$	$\forall c$	2	19	47
$D_i^p(t)$	$\forall c$	2	22	21
$D_i^p(t)$	$\forall c$	2	24	21
$D_i^p(t)$	$\forall c$	2	26	21
$D_i^p(t)$	$\forall c$	2	27	21
$D_i^p(t)$	$\forall c$	2	29	23
$D_i^p(t)$	$\forall c$	3	14	21
$D_i^p(t)$	$\forall c$	3	17	47
$D_i^p(t)$	$\forall c$	3	18	21
$D_i^p(t)$	$\forall c$	3	21	23
$D_i^p(t)$	$\forall c$	3	24	10
$D_i^p(t)$	$\forall c$	3	27	10
$D_i^p(t)$	$\forall c$	3	28	23
$D_i^p(t)$	$\forall c$	4	17	10

$D_i^p(t)$	$\forall c$	4	20	10
$D_i^p(t)$	$\forall c$	4	21	10
$D_i^p(t)$	$\forall c$	4	23	10
$D_i^p(t)$	$\forall c$	4	25	10
$D_i^p(t)$	$\forall c$	4	27	10
$D_i^p(t)$	$\forall c$	4	29	10

B.3.6. S1-L to S6a-H solution analysis

Minimum (*min*) and maximum (*max*) of total cost as well as the coefficient of variance (*coeff*) are reported.

instance		MIP	SimLP	SimMIP
S1-L	min	61,477	55,615	53,900
	max	73,197	74,995	74,860
	coeff	4.57%	7.45%	9.03%
S2-L	min	56,271	56,550	54,619
	max	67,954	69,794	68,315
	coeff	5.67%	5.52%	5.89%
S3-L	min	54,769	55,030	53,822
	max	77,051	74,478	83,628
	coeff	9.13%	8.45%	11.48%
S4-L	min	56,664	56,548	57,076
	max	76,156	72,408	76,744
	coeff	8.64%	7.23%	8.00%
S5-L	min	55,043	56,069	57,376
	max	69,121	69,071	69,721
	coeff	5.29%	6.66%	5.91%
S6-H	min	65,117	66,095	62,768
	max	84,860	81,935	84,527
	coeff	7.75%	7.57%	6.99%
S7-H	min	63,451	63,578	64,246
	max	81,557	82,120	81,446
	coeff	7.22%	7.11%	6.37%
S8-H	min	66,950	66,338	63,523
	max	89,720	100,694	93,541
	coeff	6.84%	10.60%	9.31%
S9-H	min	65,014	67,399	63,916
	max	84,574	83,571	83,248
	coeff	7.78%	6.58%	7.78%
S10-H	min	65,623	64,709	67,264
	max	79,415	75,381	85,520
	coeff	5.30%	4.42%	6.03%
S1a-L	min	64,970	65,045	62,497

	max	79,037	79,985	77,510
	coeff	5.81%	5.82%	6.73%
S6a-H	min	73,017	82,245	73,017
	max	102,937	99,475	102,937
	coeff	8.03%	5.44%	8.03%
Average	coeff	6.83%	6.91%	7.63%

C. Continuous and bundled demand

period	demand (contiuous)
8	63
9	25
10	25
11	36
12	90
13	50
14	51
15	63
17	70
18	43
19	34
20	26
21	38
22	42
24	70
25	24
26	74
27	73
28	65
29	95
30	35
31	10
32	28
33	129
34	188
35	169
36	161
37	70
38	48
39	25
40	100
41	13

42	124
43	123
44	97
45	46
46	24
47	129
48	227
49	86
50	115

period	demand (bundled)
8	53
9	55
10	55
11	56
12	40
13	40
14	41
15	53
17	50
18	53
19	64
20	56
21	68
22	62
24	60
25	54
26	54
27	53
28	55
29	64
30	52
31	41
32	41
33	45
34	48
35	1,250
36	47
37	53
38	32
39	31
40	35

41	51
42	48
43	35
44	44
45	26
46	24
47	29
48	27
49	26
50	15

D. Abstract

Supply Chain Management is a popular keyword appearing in both management science and practice regularly. Globalization and other economical developments lead to a significant increase of large and complex supply chains spanning production sites and distribution networks all over the world. Managing supply chains becomes a more and more important but also a highly challenging task. The scientific world knows a huge number of optimization methods trying to provide reasonable decision support within large networks.

Leaving strategical decisions, like network design, aside we remain on the mid- or short-term decision level and focus on supply chain planning. Discrete-event simulation and (mixed-integer) linear programming are widely used for supply chain planning. Simulation models are mostly applied in order to mimic a real system including all necessary stochastic and nonlinear elements. They are used as a playground for analyzing and improving a real situation on a trial-and-error basis. A systematic optimization method on top of a simulation model has two disadvantages: The optimization method uses the simulation model as a black-box. Information about the structure of the problem is not available and cannot be used for an intelligent optimization strategy. On the other hand pure optimization models used for planning scenarios are usually built on a very abstract level including many assumptions and simplifications. This is necessary, because otherwise we would end up with complex optimization models which cannot be solved anymore. One possible solution out of this dilemma is to use a simple optimization model within the framework of a complex simulation model in order to improve the overall performance by adapting decision rules. We present a general framework to support the operational decisions for supply chain networks using a combination of an optimization model and discrete-event simulation. The simulation model includes nonlinear and stochastic elements, whereas the optimization model represents a simplified version. We developed a supply chain network library for the simulation software AnyLogic (© XJ Technologies) and a linearized version as an optimization model implemented using XpressMP (© Dash Optimization). Based on initial simulation runs cost parameters, production, and transportation times are estimated for the optimization model. The solution of the optimization model is translated into decision rules for the discrete-event simulation. This procedure is applied iteratively until the difference between subsequent solutions is small enough. This method is applied successfully to several test examples and is shown to deliver competitive results much faster compared to conventional mixed-integer models in a stochastic environment. It provides the possibility to model and solve more realistic problems (incorporating dynamism and uncertainty) in an acceptable way and it enriches the simulation framework by a powerful tool to improve the supply chain by simultaneously optimizing a large number of possible decisions.

The combined simulation-optimization approach, which we call *SimOpt* is applied to a real-world case study as well. We show that even in case of realistic sized supply networks referring to dynamic and nonlinear aspects this new solution method is able to cope with the given complexity and finds solutions with reasonable computational effort.

E. Zusammenfassung

Die Verschärfung der Wettbewerbssituation und die fortschreitende Globalisierung zwingen die Unternehmen zunehmend zu einer kontinuierlichen Optimierung der Kosten entlang der gesamten Wertschöpfungskette. Supply Chain Management ist in diesem Zusammenhang als Lösungsansatz, sowohl in der Literatur als auch in der Praxis, nicht mehr wegzudenken. Simulationsmodelle und (gemischt ganzzahlige) Optimierungsmodelle werden häufig als Lösungsmethoden herangezogen. Beide Methoden haben ihre Vor- aber auch wesentliche Nachteile. So werden Simulationsmodelle eher als Versuchsumgebung für die Evaluation bestimmter Szenarien verwendet; eine sinnvolle Optimierung im Rahmen der Simulation ist nur bedingt möglich. Im Gegensatz dazu verlangen Optimierungsmodelle nach wesentlichen Vereinfachungen der realen Gegebenheiten, da Nichtlinearitäten oder dynamische Aspekte die benötigte Zeit der Lösungsfindung massiv erhöhen. Der von uns entwickelte Ansatz kombiniert die beiden Methoden und zielt darauf ab die jeweiligen Vorteile zu verbinden. Somit handelt es sich dabei um eine Kombination aus einem möglichst realitätsnahen, ereignisorientierten Simulationsmodell und einem vereinfachten (teilweise linearen) Optimierungsmodell. Um diese Methode auch auf andere Lieferketten anwenden zu können, haben wir ein Modellkonzept basierend auf einer sehr allgemeinen Definition eines Supply Chain Netzwerkes erstellt. Das Simulationsmodell besteht aus mehreren Modulen, welche die einzelnen Teilnehmer einer Lieferkette (vom Zulieferer bis zum Kunden) sowie deren Transportverbindungen darstellen. Diese Module bilden die wesentlichen Vorgänge (Lagerhaltung, Produktion, Umschlag) mit all ihren Parametern und stochastischen Eigenschaften ab. Das Simulationsmodell wurde in AnyLogic (© XJ Technologies), einem Java-basierten Simulationstool, implementiert.

Das Optimierungsmodell ist implementiert in Xpress-MP (© Dash Optimization) und entspricht einer (teilweise) linearen Beschreibung der Netzwerkaktivitäten. Die Kopplung zwischen den beiden Modellen erfolgt über eine Datenbank, die zum Informationsaustausch zwischen Simulation und Optimierung dient. Auch die globalen Parameter, die von beiden Modellen zur Netzwerkinitialisierung verwendet werden, werden in dieser Datenbank hinterlegt. Als Initialisierungsschritt werden einige Simulationsläufen durchgeführt um eine erste Schätzung der Kosten und Zeiten zu erhalten. Diese Werte werden dann mit Hilfe der Datenbank an das Optimierungsmodell übergeben. Die Lösung des Optimierungsmodells wird dann in Form von Entscheidungsregeln in das Simulationsmodell rückgeführt. Dann können weitere Simulationsexperimente durchgeführt werden und die Parameter gegebenenfalls angepasst werden. Wie Tests an kleineren Beispielen gezeigt haben, reichen insgesamt 4-5 Iterationen aus um zu einer verbesserten Lösung im Simulationsmodell zu konvergieren.

Weitere Tests haben gezeigt, dass dieser Ansatz eine raschere Berechnung von robusten Lösungen erlaubt als gängige Methoden unter Verwendung exakter gemischt ganzzahliger Formulierung.

Neben der Evaluierung anhand kleinerer Testinstanzen wurde das Verfahren auch zur Lösung eines komplexen Realitätsfalls herangezogen. Das weltweite Distributionsnetz eines Papierproduzenten wurde mitsamt der wichtigsten nichtlinearen und dynamischen Aspekten abgebildet und für die weitere Evaluierung der simulationsbasierten Optimierung herangezogen. Auch dieser, realtitätsnahe, Komplexitätsgrad konnte von unserem Verfahren in zufriedenstellender Zeit gelöst werden.

F. Curriculum Vitae

Ausbildung

1985-1989	Volksschule Wien 23, Prückelmayrgasse
1989-1997	GRG Wien 12, Singrienergasse 19 - 21
Juni 1997	Matura mit Gutem Erfolg
1997-1999	Kolleg für Tourismus und Freizeitwirtschaft, HLF Krems
Mai 1999	Abschluß des Kollegs mit Diplomprüfung
Okt. 1999-Sept.2003	Studium Internationale Betriebswirtschaft an der Universität Wien
Seit Okt. 2003	Doktoratsstudiums der Wirtschaftswissenschaften

Berufliche Tätigkeiten

Feb.-Juli 2003	Studienassistentin am Lehrstuhl für Produktion und Logistik der Universität Wien
Okt. 2003-Feb.2004	Lektorin an der Fachhochschule des bfi Wien
Okt. 2003-Sept. 2006	Freier Dienstnehmer bei ARC Seibersdorf, Projekt: Optimization in Supply Chain Management
Nov. 2003-Dez. 2006	Projektassistentin bei Paradigma Consulting GmbH
Okt. 2003-Jänner 2008	Projektassistentin/Wissenschaftliche Assistentin und Lehrbeauftragte am Lehrstuhl für Produktion und Logistik der Universität Wien
Seit Feb. 2008	Kotányi GmbH, Mitarbeiterin Supply Chain Management

Publikationen

- Almeder C, Preusser M (2004) Mixed Analytical / DEVS Approach to ARGESIM Comparison C14 'Supply Chain Management' using Xpress-MP and Anylogic. SNE Simulation News Europe 41/42: 50

- Almeder C, Preusser M, Hartl RF (2008) Simulation and optimization of supply chains: alternative or complementary approaches? OR Spectrum (online available since Jan 12th, 2008), DOI: 10.1007/s00291-007-0118-z (original article on www.springerlink.com)

- Gronalt M, Hartl, RF, Preusser M (2007) Design eines globalen Liefernetzes für Papier - Eine Fallstudie. In: Corsten H, Missbauer H (eds), Produktions- und Logistikmanagement, pp. 597-616. Vahlen Verlag

- Preusser M, Almeder C, Hartl RF, Klug M (2005a) LP Modelling and simulation of supply chain networks. In: Günther HO, Mattfeld DC, Suhl L (eds) Supply Chain Managament und Logistik: Optimierung Simulation, Decision Support, pp 95-114. Physica-Verlag

- Preusser M, Hartl RF, Dörner K, (2004) Column Generation for Bin Packing Problems, Proceedings of the 5^{th} EuroSim Congress on Modelling and Simulation, Paris, Frankreich

- Preusser M, Almeder C, Hartl RF, Klug M (2005b) Hybrid Supply Chain Modelling - Combining LP-Models and Discrete-Event Simulation. In: Maroto Álvarez MC, Alcaraz Soria J, Ruiz GR, Crespo Abril F, Vallada R (eds) ORP3 - Operational Research Peripatetic Postgraduate Programme proceedings, pp 163-170. ESMAP, S.L., Universidad Politécnica de Valencia