



Simulation-based optimization vs. mathematical programming: A hybrid approach for optimizing scheduling problems

Andreas Klemmt*, Sven Horn, Gerald Weigert, Klaus-Jürgen Wolter

Electronics Packaging Laboratory, Faculty of Electrical Engineering and Information Technology, Technische Universität Dresden, 01062 Dresden, Germany

ARTICLE INFO

Article history:

Received 28 November 2008

Received in revised form

20 February 2009

Accepted 9 April 2009

Keywords:

Complex scheduling

Job Shop scheduling

Mixed integer programming

Simulation-based optimization

ABSTRACT

With the increasing computing power of modern processors, exact solution methods (solvers) for the optimization of scheduling problems become more and more important. Based on the mixed integer programming (MIP) formulation of a scheduling problem, it will be analyzed how powerful the present solvers of this problem class are and up to which complexity real scheduling problems are manageable. For this, initially some common benchmark problems are investigated to find out the boundaries for practical application. Then, the acquired results will be compared with the results of a conventional simulation-based optimization approach under comparable time restrictions. As a next step, the general advantages and disadvantages of both approaches were analyzed. As the result, a coupling of the discrete event simulation system and an MIP solver is presented. This coupling automatically generates an MIP-formulation for the present simulation model which can be solved externally by an MIP solver. After the external optimization process follows a backward transformation of the results into the simulation system. All features of the simulation system (like Gantt-Charts, etc.) could be used to check or to illustrate these results. To perform the coupling for a wide range of simulation models, it has to be defined which general constraints the model has to satisfy.

© 2009 Elsevier Ltd. All rights reserved.

1. Introduction

Mixed integer programming (MIP) formulations for solving scheduling problems can be traced back to 1959. At this time, Wagner [1] first formulated a Flow Shop problem as an integer program. In the following years, different types of MIP-formulations are defined for classical Flow Shop and Job Shop scheduling problems by Browman [2], Manne [3], Wilson [4], and Morten and Pentico [5]. But due to the fact that most scheduling problems are NP-hard (see e.g. [6] for a very detailed classification), computer capacity was very limited and the MIP algorithms were inefficient, there was no really practical use for them in the last century.

As an alternative to exact mathematical (branch and bound-based) approaches like MIP, disjunctive programming [7] or constrained programming (CP) [8], several heuristics were developed to find nearly optimal solutions for scheduling problems. Gupta and Sivakumar [9] group them into dispatching, search methods and artificial intelligence techniques. There is a lot of literature about this topic and different efficient heuristics were developed for special classes of scheduling problems (e.g. critical path methods for makespan optimization of Job Shop problems [7]).

One of these heuristic approaches, combining dispatching and heuristic search, is the method of simulation-based optimization (see Fig. 1). This method becomes, because of its flexibility, more and more important during the last years. Here, the manufacturing problem is described by a “object-based” simulation model instead of formulas or graphs, etc. Local rules (e.g. priority rules) in the simulation model allow a problem-specific dispatching. A global iterative operating (meta-)heuristic represents the actual simulation-based optimization approach. For this, the simulation model includes a control vector \mathbf{x} consisting of several variables x influencing the behavior of the simulation model (e.g. job permutations, buffer or machine capacities, release dates). After a simulation run is completed, the model responds with a scalar objective value C . A heuristic optimization algorithm, for example threshold acceptance, simulated annealing or genetic algorithm (e.g. see [7] for documentation), changes or adjusts the control variable after every optimization run depending on the objective function gradient [10]. The simulation-based optimization approach offers, next to the wide flexibility, high model accuracy. Also, some simulation systems allow an automated simulation model creation on the base of MRP and ERP systems in online mode [11]. The disadvantage of this approach is that there is no guaranty to find the global optimum or even to get good lower bounds for the objective. Also, the cyclic approach is very time-consuming

* Corresponding author. Tel.: +49 351 463 31696; fax: +49 351 463 37035.
E-mail address: klemmt@avt.et.tu-dresden.de (A. Klemmt).

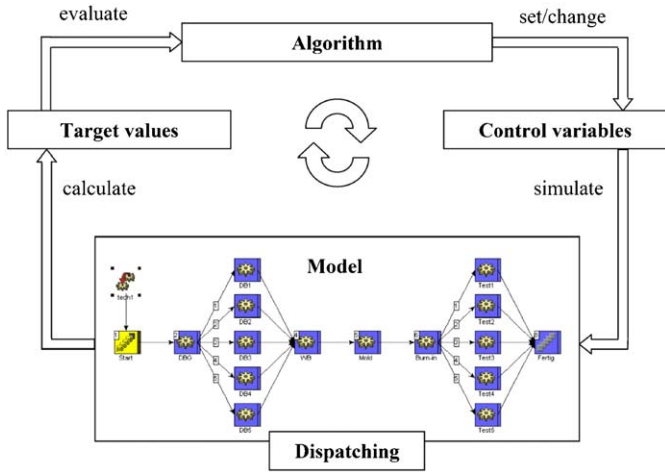


Fig. 1. Simulation-based optimization.

especially in the case of complex simulation models. So, it is usual to define a time limit for the optimization process.

A few years ago, mathematical programming-based scheduling approaches have been resurfaced. Today's computing power has increased and more efficient solver software, for example ILOG-CPLEX [12], is available. So first, the question up to which problem size classical scheduling problems (Job Shop benchmarks) are solvable today is investigated in Section 3. Also, a comparison to a heuristic simulation-based approach is shown in this section. The mathematical modeling of the Job Shop benchmarks is described in Section 2 following the Manne model [3]. The reason for this is a research of Pan [13], who investigated different MIP-formulations of scheduling problems where he concluded that Manne's model shows the best performance results for Job Shop problems. Another excellent survey of MIP-formulations for scheduling problems is given by Blazewicz [14]. Here primarily, single and parallel machine problems were studied extensively. These papers were used as the basis for the research in the later sections.

The investigation of benchmark problems offers the great advantage that different solution methodologies (exact and heuristic approaches) become comparable. For practical problems, benchmark models have no relevance (because there are much more constraints). Hence different types of problem-specific MIP-formulations for scheduling problems were developed in the last years. Here, especially in the field of electronics and semiconductor manufacturing, several interesting approaches exist. Pearn [15], for example developed an MIP-formulation for a wafer probe parallel machine scheduling problem with setup effects. In [16], an approach is described for minimizing the total processing time and the total setup time of a Die Bond assembly operation. Pan [17] formulated an MIP-formulation for re-entrance flow shops. Niemi [18] developed a worker allocation MIP model for an assembly problem.

Motivated by these practical relevant problems, a more general MIP-formulation for advanced Job Shop problems is given in Section 4. To make the modeling more comfortable for the user, a coupling to a discrete event simulation system is designed which automatically parameterizes this advanced Job Shop. Its interface is described in Section 4 in detail. An illustrating example to them follows in the end of Section 4. Here, a SMT¹ manufacturing line is optimized.

In the last Section 5, a coupling between simulation-based and solver-based optimization is shown. This approach exploits the

advantages of both methods and leads to beneficial results. The communication between solver and simulation system for this coupling is also described in this section in detail.

2. Job Shop scheduling

First, the classical Job Shop scheduling problem $J||C_{\max}$ is investigated. For this, the MIP-formulation of Manne [3] is used. Classical Job Shop scheduling means in this context: there are n jobs which have to be scheduled on m machines. Every job has its own processing order and has to be processed in m steps on exactly m different machines. Every machine can only process one job at the same time. Initially, the machines are not loaded. Every job is immediately available and has no due date. The goal is to find an execution order for all jobs on all machines which minimizes the total makespan. The technological processing sequence of each job defines r_{ilk} which is 1 if job i requires machine k in Step l and 0 otherwise. The processing time of job i on machine k is given by p_{ik} . In the Manne model, there are now $m \cdot n \cdot (n-1)/2$ Boolean variables x_{ijk} ($i < j$) defined with

$$x_{ijk} = \begin{cases} 1, & \text{if job } i \text{ precedes job } j \text{ (not necessarily immediately)} \\ & \text{on machine } k \\ 0, & \text{otherwise} \end{cases} \quad (1.1)$$

There are $m \cdot n$ positive real valued unknowns $s_{ik} \geq 0$ specifying the starting time of job i on machine k , as well as the objective C_{\max} . The Manne model can now be written as follows:

$$C_{\max} \rightarrow \min \quad \text{subject to} \quad (1.2)$$

$$\sum_{k=1}^m r_{imk}(s_{ik} + p_{ik}) \leq C_{\max} \quad i = 1, \dots, n \quad (1.3)$$

$$\sum_{k=1}^m r_{ilk}(s_{ik} + p_{ik}) - \sum_{k=1}^m r_{i,l+1,k} s_{ik} \leq 0 \quad i = 1, \dots, n; l = 1, \dots, m-1 \quad (1.4)$$

$$K(1 - x_{ijk}) + s_{jk} - s_{ik} \geq p_{ik} \quad k = 1, \dots, m; 1 \leq i < j \leq n \quad (1.5)$$

$$Kx_{ijk} + s_{ik} - s_{jk} \geq p_{jk} \quad k = 1, \dots, m; 1 \leq i < j \leq n \quad (1.6)$$

thereby K is a large number (e.g. the sum of all process times). The constraints set (1.3) restricts the objective function C_{\max} . Eq. (1.4) restricts the starting time of job i in operation $l+1$ to be not earlier than its finish time in the predecessor operation l . Constraints sets (1.5) and (1.6) convert the "either-or" condition

$$s_{jk} - s_{ik} \geq p_{ik}, \quad \text{or else} \quad s_{ik} - s_{jk} \geq p_{jk} \quad (1.7)$$

into a linear inequality with Boolean unknowns. This disjunction ensures the requirement that only one job may be processed on a machine at any instant of time. Note that with the classical form of linear programming it is not possible to specify such an "either-or" condition as shown in Eq. (1.7). Hence, the Manne model describes the Job Shop scheduling problem with $m \cdot n \cdot (n-1)/2$ binary unknowns and $m \cdot n + 1$ continuous variables in $n^2 \cdot m$ inequality restrictions.

3. Comparison of solver-based and simulation-based optimization

In this section, the efficiency of solver-based and simulation-based optimization methods is compared on some common benchmarks of the Job Shop problem class. Thereby, the aim is

¹ Surface Mount Technology.

Table 1

Comparison of optimization results (CPLEX 11.0 settings: SUBALG = 2, MIPEMPHASIS = 0, DIVETYPE = 0, BRDIR = 0, BARALG = 3, BNDSTREIND = 0, cuts off).

Benchmark	Dimension		Optimum	MIP	Simulation-based optimization GA	
	J × M	Binary unknowns			Nondelay schedule	Delay schedule
ft06	6 × 6	90	55	55 (0.2s)	58	58
la01	10 × 5	225	666	666 (5.5s)	666	670
ft08 ^a	8 × 10	280	824	824 (4.7s)	856	899
abz5	10 × 10	450	1234	1234 (151s)	1266	1360
abz6	10 × 10	450	943	943 (15s)	973	1024
ft10	10 × 10	450	930	930	982	1049
la08	15 × 5	525	863	863 (4.7s)	863	882
la09	15 × 5	525	951	951 (50.6s)	951	972
la11	20 × 5	950	1222	1222 (50s)	1222	1242
la21	15 × 10	1050	1046	1102	1113	1305
la22	15 × 10	1050	927	942	986	1142
la26	20 × 10	1900	1218	1334	1296	1604
la27	20 × 10	1900	1235 ^{ub}	1439	1351	1642
abz7	20 × 15	2850	655	747	753	895
yn1	20 × 20	3800	886 ^{ub}	983	994	1206
la31	30 × 10	4350	1784	1912	1809	2200
swv20	50 × 10	12,250	2823 ^{ub}	3312	2824	3369
ta71	100 × 20	99,000	–	–	6144	7562

^a Modification of ft10, first 8 jobs only.

to estimate up to which problem dimension the MIP approach is superior to a heuristic simulation-based optimization. To approximate practical problems, a time limit of 5 min calculation time on a standard personal computer is set to both methods. All optimization runs are repeated 10 times to decrease stochastic influences by the heuristic methods on the listed results. The MATLAB toolbox TOMLAB containing CPLEX 11.0 is utilized for the solver of the MIP approach. The simulation-based optimization is performed by a genetic search algorithm (GA) combined with the discrete event simulation system *simcron* MODELLER. The influences of the used population size, mutation type and crossover parameters should be neglected here, because the given time limit is very tight. Table 1 shows the optimization results of both methods. If the optimum was found by the solver under the 5 min, the average calculation time is added in brackets. The problem dimension of the benchmark is marked as J × M (Jobs × Machines). To make different problem sizes comparable, also the number of binary unknowns (concerning Manne model) is listed which is a good indicator for problem complexity [13]. In the column, “Optimum” the optimum or the currently best known value (denoted by ^{ub} ... upper bound) from the literature is shown (Refs. [19,20]). A “–” is set if no upper bound was found in literature or if the solver could not generate a feasible solution in time.

It is recognizable that for smaller problems (<1000 binary unknowns) the MIP approach is superior to simulation-based optimization. A problem size of around 1000 binary unknowns offers similar results with both approaches. For larger problems, the simulation-based approach is predominant. Moreover, for problems with more than 10,000 binary unknowns it was difficult for the MIP solver even to find a valid solution. In contrast to this the simulation-based approach generates a feasible solution in every cyclic simulation run (a deadlock detection avoids invalid solutions). For the simulation-based optimization with GA two different approaches are investigated: For the simulation system, it is simple to limit the search space to nondelay² schedules. This allows a fast convergence of the search algorithm. But it is

possible that the optimum is not included in the reduced search space of nondelay schedules. For this, a second approach allowing delays is investigated. Then, it is theoretically possible to find the exact optimum. But the extremely enlarged search space makes the optimization much more complicated. So, on average, better results are reached with the nondelay approach, within the given time boundaries. Note: the MIP solver is also not limited in its search space, delay schedules can also be found.

In comparison to the state of the art solvers of constraint programming (e.g. see [20]) it can be said that the results are very similar to the reached MIP results (Table 1) for this problem class. However, for the heuristic optimization that does not apply. Here, more efficient heuristics for the Job Shop are described in the literature especially for the objective “makespan” (e.g. local search in combination with critical path method [7]). But these methods are often too problem-specific to allow a widely use of them. A critical path method, for example is closely linked with the objective makespan. For other objectives like cycle time or tardiness, it is not applicable. So, to receive flexibility concerning constraints and objectives, focusing on real world applications, we decided to use these (meta-)heuristic solution methodologies.

Next to the differences in optimization convergence, some further distinctions have to be emphasized between exact solver-based and heuristic simulation-based optimization approaches: a simulation system works almost time directed, so it normally produces nondelay schedules. As a consequence, time-related model constraints are hard to handle with a simulation system because delays in the schedule are often required in this case. So, these constraints (e.g. due dates, time coupling restrictions) only can be implemented by additional modeling efforts like scripts, dispatch rules or control variables. On the other hand, the modeling of complex resource constraints is very simple in a simulation system. Here, exact approaches reach boundaries very fast. The modeling effort increases because additional unknowns have to be defined (see later sections). Resulting from this the problem complexity grows on quickly. Table 2 summarizes the advantages and disadvantages of the investigated approaches.

In the following, a direct coupling of the simulation system and the MIP solver is described to exploit the advantage of both approaches. Therefore, an advanced Job Shop scheduling formulation has to be developed at first.

² A schedule is called nondelay if no machine is kept idle when there is an operation available for processing [21].

4. Advanced Job Shop scheduling

The investigation of benchmark problems has the advantage that different solution approaches become comparable. But for real world problems they have nearly no practical relevance. Here, often it is necessary to deal with a rolling forecast. This means that the manufacturing line is initially filled and the jobs have different counts of processing steps. Machines can be busy at the start; jobs can be supplied after the start or can be dependent on material or resources. Jobs can also have due dates or time constraints. Furthermore, there are parallel or alternative machines in some steps (branches). Also, sequence-dependent setups can be necessary.

A more general Job Shop MIP-formulation is presented in the following which allows a modeling of even such practical relevant features. This new formulation includes a lot of extensions to the classical Job Shop. Unfortunately, this results in a huge number of input data (e.g. sets of indices). To make the modeling more comfortable, a coupling to a discrete event simulation system is designed. It allows a direct derivation of an MIP-formulation from the simulation model to avoid manual input data extraction. To

explain this interface in detail, the simulation system has to be presented at first.

4.1. Simulation system

Although the simcron MODELLER is a powerful and practical simulation system, its box of simulation elements is very small. Almost all objects are described in Table 3. Some further elements for configuring time tables or stochastic behavior are also available but not discussed here. The simulation model can be extended with additional scripts, so it is possible to model complex manufacturing systems, too. The simulator works properly with as well as without graphical user interface. In the last case, the simulation engine is very fast; this makes the simulator also suitable for simulation-based scheduling systems with heuristic optimization algorithms. The simulation model gets all data via an ODBC-interface from the database of the corresponding ERP-system and generates the complete simulation model automatically [11]. In this case, the ERP-system controls exclusively the simulation run as well as the following data analysis.









As a next step, the MIP solver should be coupled with the simulator. The goal is to formulate a given problem as a simulation model and after that to export this model automatically into a data structure which can be interpreted by the solver. Of course, it is necessary to consider some additional constraints to the simulation model which have to be ensured to allow the described export, because not all features of the simulation system could be translated into formulas. So, for example, cumulative resource constraints (see [8] for mathematical modeling) or capacity constrained buffer sizes cannot be handled yet. Column three of Table 3 gives an overview about these constraints.

Also, the use of the here not described simulation objects: timetable (e.g. to model shift plans or down intervals), stochastic (to model non-deterministic behavior) as well as the use of additional scripts (program code which will be executed at selected events, i.e. for modeling complex/nonlinear constraints) is not possible for MIP-model generation.

Table 2
Comparison of solver-based and simulation-based optimization.

Method	Advantage	Disadvantage
Simulation-based optimization	<ul style="list-style-type: none"> Simple modeling (also for complex problems) Resource constraints modeling Problem specific extensible via additional scripts Automated model generation possible (from ERP system) 	<ul style="list-style-type: none"> Slow convergence Not necessarily optimal Implementation of time related constraints
Solver-based optimization	<ul style="list-style-type: none"> Fast convergence (for small problems) Time constraints modeling Exact solution possible (incl. proof) 	<ul style="list-style-type: none"> Only small problems exactly solvable High modeling effort (know how)

Table 3
Basic elements of the simulation system.

Element	Explanation	Additional constraints for the MIP-export
 Machine	This element describes in general limited resources which are needed for processing jobs, e.g. machine, operator or tool. Machines can be empty, busy, ready or down and can be marked with several setup states.	Machines can process only one job at the same time (only exclusive resource constraints).
 Queue	General storage place where jobs can be stored for an indefinite period.	There exists a (non-capacity constrained) queue in front of every machine to avoid blocking effects.
 Job	Represents real or virtual moveable consumers of resources in the manufacturing system, e.g. work piece, product, production lot or order. Usually jobs have an earliest supply and a latest due date as well as quantity properties and an optional product attribute.	
 Tech	Specifies the technological workflow (route) of a job by a sequence of processing steps. Each single processing step includes among others the processing time at a machine.	
 Branch	Consists of one or more arms and defines an alternative or parallel routing. In the latter case a machine group with m -of- n -decision can be free configured.	Only machines can be assigned to the arms of a branch. Branch cascading is not allowed.
 Need	This element describes dependencies between several processing steps of the same or distinguished technologies: a step may first start after one or more (need-) steps are finished. Need-objects can include Boolean expressions and can be cascaded, too. For example, they are required for modeling bill of materials.	Each need-object may include only one processing step. Boolean expressions and cascading are not allowed.
 Time coupling	Maximum acceptable waiting time of a job between two successive operations. (Forced by additional scripts)	
 Setup	Includes an (asymmetric) matrix of setup periods together with several setup conditions. A setup activity is necessary if the current machine setup state differs from the product attribute of the next job.	

4.2. Advanced Job Shop MIP-formulation

The extraction of the MIP-formulation from the simulation model is done in 4 steps. Thereby, an ASCII-file is generated which is used later as input for the MIP solver. Depending on the underlying simulation model first some input data variables are specified. In step 2, several required index sets are calculated from the input data. The index sets are the basis for the MIP-formulation in step 4. They are required for the construction of the solution vector \mathbf{x} in step 3, too. With the help of the index sets it is possible only to encode information in the solution vector \mathbf{x} which are really essential for problem description. For example, there is no precedence variable specified between two jobs on a machine k if machine k is not included in both jobs technology. This reduces the problem dimension drastically. The construction of the solution vector \mathbf{x} and the MIP-formulation is built basing on a modular design principle. Depending on the used module types in the simulation model, equations and variables are full generically complied to an MIP. Variables and equations in step 3 and 4 only have to be defined if the shown module type exists in the simulation model. (The existence of jobs, technologies, queues and machines is assumed generally.)

Step 1: input data extraction

n	number of jobs
m	number of machines
i, j	job indices
k, k'	machine indices
l	operation index
J_i	job number i
M_k	machine number k
$Rest_i$	rest processing time of J_i
p_{ik}	processing time of J_i on M_k ; $p_{ik} = 0$ otherwise
I_k	rest processing time/down time on M_k
B_{il}	branch parallelism (J_i , operation l); $\{\}$ otherwise
Esd_i	earliest supply date of J_i ; $Esd_i = 0$ otherwise
Ldd_i	latest due date of J_i ; $Ldd_i = \inf$ otherwise
UM	all machines with an assigned setup object; $\{\}$ otherwise
u_{ijk}	setup time between J_i and J_j on M_k , $k \in UM$
lu_{ik}	initial setup time J_i on M_k , $k \in UM$
r_{ilk}	with $r_{ilk} = 1$ if the l th operation of J_i requires only M_k ; $r_{ilk} = 0$ otherwise ³
o_{ilk}	with $o_{ilk} = 1$ if the l th operation of J_i can be executed on M_k (in branch); $o_{ilk} = 0$ otherwise ³
$Need$	set of pairs (i, k) , with J_i has a need for M_k ; $\{\}$ otherwise
$N(i, k)$	with $N(i, k) = (j, k')$; J_j must be finished on $M_{k'}$ before J_i can start on M_k ; $\{\}$ otherwise
t_{il}	time coupling restriction; maximum waiting time of J_i between operation l and $l-1$; $t_{il} = \inf$ otherwise

Step 2: index set calculation

$$MS_i = \left\{ k \in \{1, \dots, m\} \mid \sum_{l=1}^m r_{ilk} = 1 \right\} \quad i = 1, \dots, n$$

single machines of J_i

$$AM_i = \left\{ l \in \{1, \dots, m\} \mid \sum_{k=1}^m r_{ilk} = 1 \right\} \quad i = 1, \dots, n$$

single machine operations of J_i

$$JO = \left\{ i \in \{1, \dots, n\} \mid \sum_{k=1}^m \sum_{l=1}^m o_{ilk} > 0 \right\}$$

all jobs with a branch in their technology

$$AO_i = \left\{ l \in \{1, \dots, m\} \mid \sum_{k=1}^m o_{ilk} > 0 \right\} \quad i \in JO$$

branch operations of J_i

$$MO_{il} = \{ k \in \{1, \dots, m\} \mid o_{ilk} = 1 \} \quad i \in JO; l \in AO_i$$

branch machines of J_i in operations l

$$MG_i = \left\{ k \mid \bigcup_{l \in AO_i} MO_{il} \cup MS_i \right\} \quad i = 1, \dots, n$$

referenced machines of J_i

$$LMM_i = \{ l \in AM_i \mid (l+1) \in AM_i \} \quad i = 1, \dots, n$$

single machine intersection of J_i

$$LMB_i = \{ l \in AM_i \mid (l+1) \in AO_i \} \quad i = 1, \dots, n$$

machine–branch intersection of J_i

$$LBB_i = \{ l \in AO_i \mid (l+1) \in AO_i \} \quad i = 1, \dots, n$$

branch–branch intersection of J_i

$$LBM_i = \{ l \in AO_i \mid (l+1) \in AM_i \} \quad i = 1, \dots, n$$

branch–machine intersection of J_i

$$UJ_k = \{ i \in \{1, \dots, n\} \mid k \in MG_i \} \quad k \in UM$$

setup jobs on M_k

$$PM_{il} = \{ (k, k') \mid k, k' \in MO_{il}, k \neq k' \} \quad i \in JO; l \in AO_i$$

all pair wise different machines of J_i in operation l

$$LastM_i = \{ k \in \{1, \dots, m\} \mid l = \max(AM_i, AO_i) : r_{ilk} = 1 \vee o_{ilk} = 1 \}$$

$i = 1, \dots, n$ last machine(s) of J_i

Step 3: construction of the solution vector $\mathbf{x} = (x_{ijk}, y_{ilk}, s_{ik}, z_{ijk}, C_{\max})^T$ with:

$$x_{ijk} = \begin{cases} 1, & \text{if } J_i \text{ precedes } J_j \text{ (notnecessarily immediately) on } M_k; \\ 0, & \text{otherwise} \end{cases}$$

$1 \leq i < j \leq n; k \in (MG_i \cap MG_j) \setminus UM$

$$y_{ilk} = \begin{cases} 1, & \text{if } J_i \text{ is planned in operation } l \text{ on } M_k; \\ 0, & \text{otherwise} \end{cases}$$

$i \in JO; l \in AO_i; k \in MO_{il}$ 🎨

$$z_{ijk} = \begin{cases} 1, & \text{if } J_i \text{ precedes } J_j \text{ directly on } M_k; \\ 0, & \text{otherwise} \end{cases}$$

$k \in UM; i, j \in UJ_k, i \neq j$ 🎨

$$s_{ik} \text{ starting time of } J_i \text{ on } M_k (s_{ik} \geq 0); i = 1, \dots, n; k \in MG_i$$

$$C_{\max} \text{ objective function } (C_{\max} \geq 0)$$

³ set for all $i = 1, \dots, n; k, l = 1, \dots, m$; every job can have maximum m operations (re-entrance handling is described later).

Step 4: MIP-formulation

$$C_{\max} \rightarrow \min \quad \text{subject to} \quad (4.1)$$

$$\sum_{k=1}^m r_{ilk}(s_{ik} + p_{ik}) - \sum_{k=1}^m r_{i,l+1,k} s_{ik} \leq 0 \quad i = 1, \dots, n; l \in LMM_i \quad (4.2)$$

$$t_{i,l+1} + \sum_{k=1}^m r_{ilk}(s_{ik} + p_{ik}) - \sum_{k=1}^m r_{i,l+1,k} s_{ik} \geq 0; \quad i = 1, \dots, n; l \in LMM_i \quad (4.3)$$

$$\sum_{k=1}^m r_{ilk}(s_{ik} + p_{ik}) - s_{ik'} - K(1 - y_{i,l+1,k'}) \leq 0; \quad i \in JO; l \in LMB_i; k' \in MO_{i,l+1} \quad (4.4)$$

$$t_{i,l+1} + \sum_{k=1}^m r_{ilk}(s_{ik} + p_{ik}) - s_{ik'} + K(1 - y_{i,l+1,k'}) \geq 0; \quad i \in JO; l \in LMB_i; k' \in MO_{i,l+1} \quad (4.5)$$

$$s_{ik'} + p_{ik'} - \sum_{k=1}^m r_{i,l+1,k} s_{ik} - K(1 - y_{ilk'}) \leq 0; \quad i \in JO; l \in LBM_i; k' \in MO_{il} \quad (4.6)$$

$$t_{i,l+1} + s_{ik'} + p_{ik'} - \sum_{k=1}^m r_{i,l+1,k} s_{ik} + K(1 - y_{ilk'}) \geq 0; \quad i \in JO; l \in LBM_i; k' \in MO_{il} \quad (4.7)$$

$$s_{ik} + p_{ik} - s_{ik'} - 2K + Ky_{ilk} + Ky_{i,l+1,k'} \leq 0 \quad i \in JO; l \in LBB_i; k \in MO_{il}; k' \in MO_{i,l+1} \quad (4.8)$$

$$t_{i,l+1} + s_{ik} + p_{ik} - s_{ik'} + 2K - Ky_{ilk} - Ky_{i,l+1,k'} \geq 0 \quad i \in JO; l \in LBB_i; k \in MO_{il}; k' \in MO_{i,l+1} \quad (4.9)$$

$$K(1 - y_{ilk}) + K(1 - y_{ilk'}) + s_{ik} - s_{ik'} \geq 0; \quad i \in JO; l \in AO_i; (k, k') \in PM_{il} \quad (4.10)$$

$$\sum_{k \in MO_{il}} y_{ilk} = B_{il} \quad i \in JO; l \in AO_i \quad (4.11)$$

$$Kx_{ijk} + s_{ik} - s_{jk} \geq p_{jk} \quad 1 \leq i < j \leq n; k \in (MG_i \cap MG_j) \setminus UM \quad (4.12)$$

$$K(1 - x_{ijk}) + s_{jk} - s_{ik} \geq p_{ik} \quad 1 \leq i < j \leq n; k \in (MG_i \cap MG_j) \setminus UM \quad (4.13)$$

$$0 \leq \sum_{j \in UJ_k(i \neq j)} z_{ijk} \leq 1 \quad k \in UM; i \in UJ_k \quad (4.14)$$

$$0 \leq \sum_{i \in UJ_k(i \neq j)} z_{ijk} \leq 1 \quad k \in UM; j \in UJ_k \quad (4.15)$$

$$nU_k - \sum_{i \in UJ_k} \sum_{j \in UJ_k(i \neq j)} z_{ijk} = 1 \quad k \in UM \quad (4.16)$$

$$\sum_{i \in UJ_k(i \neq j)} Kz_{ijk} + s_{jk} \geq lu_{jk} \quad k \in UM; j \in UJ_k \quad (4.17)$$

$$K(1 - z_{ijk}) + s_{jk} - s_{ik} - u_{ijk} \geq p_{ik} \quad k \in UM; i, j \in UJ_k (i \neq j) \quad (4.18)$$

$$s_{jk'} + p_{jk'} \leq s_{ik} \quad (i, k) \in \text{Need}; (j, k') = N(i, k) \quad (4.19)$$

$$\max(\text{Es}d_i, \text{Rest}_i, I_k + lu_{ik}) \leq s_{ik} \quad i = 1, \dots, n; k \in MG_i \quad (4.20)$$

$$s_{ik} + p_{ik} \leq \text{Ldd}_i \quad i = 1, \dots, n; k \in \text{Last}M_i \cap MS_i \quad (4.21)$$

$$s_{ik} + p_{ik} - K(1 - y_{ilk}) \leq \text{Ldd}_i; \quad i \in JO; l = \max(AO_i); k \in \text{Last}M_i \cap MO_{il} \quad (4.22)$$

$$s_{ik} + p_{ik} - K(1 - y_{ilk}) \leq C_{\max} \quad i = 1, \dots, n; k \in \text{Last}M_i \cap MS_i \quad (4.23)$$

$$s_{ik} + p_{ik} - K(1 - y_{ilk}) \leq C_{\max}; \quad i \in JO; l = \max(AO_i); k \in \text{Last}M_i \cap MO_{il} \quad (4.24)$$

Thereby, K is a high number and $nU_k = |UJ_k|$ $k \in UM$. So, this advanced Job Shop formulation can be interpreted in the following way: Eq. (4.1) forces the minimization of the objective function. Eqs. (4.2)–(4.9) restrict the starting time of J_i on the machine(s) in operation $l+1$ to be not earlier than its finish time in the predecessor operation l and not later than the given time coupling restriction (if those exists). Eq. (4.10) forces the same process start time of all parallel machines (if $B_{il} > 1$). Eq. (4.11) ensures the required branch parallelism. Eqs. (4.12) and (4.13) ensure the requirement that only one job may be processed on a non-setup machine at any instant of time. Eqs. (4.14) and (4.15) force that every job can have maximum one predecessor and can be predecessor of maximum one job (on a setup machine). Eq. (4.16) defines that there are exactly $nU_k - 1$ predecessor jobs on a setup machine. Constraint (4.17) regards the initial setup time. Eq. (4.18) ensures the requirement that only one job may be processed on a setup machine at any instant of time. Eq. (4.19) ensures the need-requirement. Eq. (4.20) guaranties the machine availability. (Note: the remaining processing time of a job is

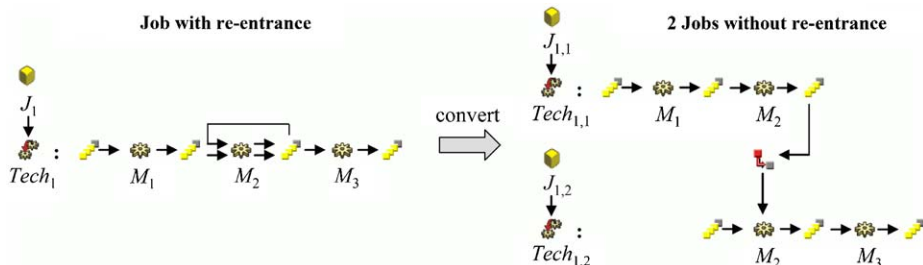


Fig. 2. Conversion of re-entrance loop.

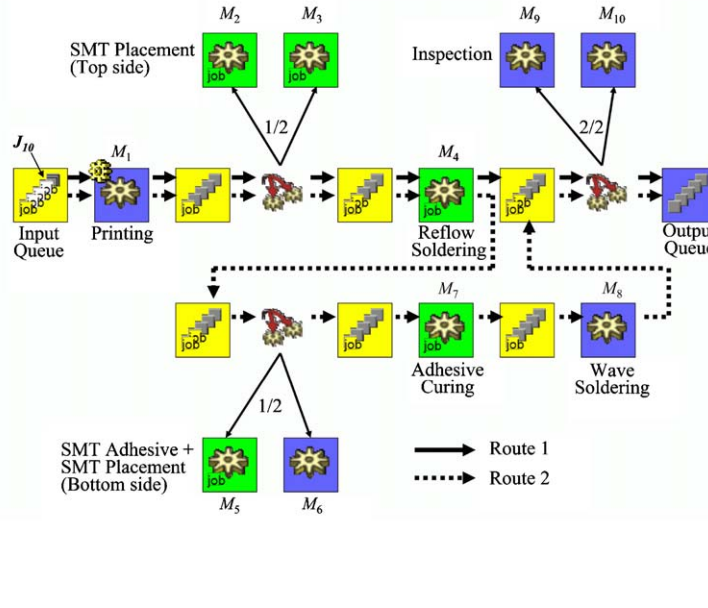
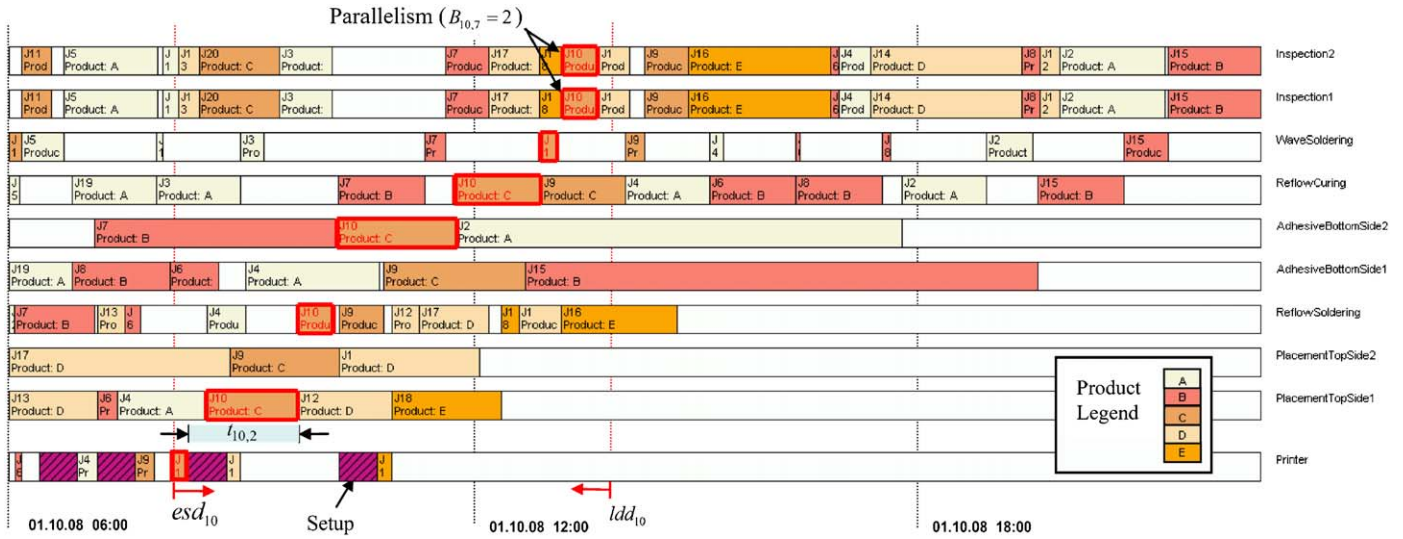


Fig. 3. Simulation model of a SMT-Line.

Example: Specification of J_{10}

$$\begin{aligned}
 r_{10,1,1} &= r_{10,3,4} = r_{10,5,7} = \\
 r_{10,6,8} &= o_{10,2,2} = o_{10,2,3} = \\
 o_{10,4,5} &= o_{10,4,6} = o_{10,7,9} = \\
 o_{10,7,10} &= 1 \\
 B_{10,2} &= B_{10,4} = 1; B_{10,7} = 2 \\
 t_{10,2} &= 3600; esd_{10} = 7800 \\
 ldd_{10} &= 28800 \\
 MS_{10} &= \{1, 4, 7, 8\} \\
 AM_{10} &= \{1, 3, 5, 6\} \\
 AO_{10} &= \{2, 4, 7\}; MB_{10,2} = \{2, 3\} \\
 MB_{10,4} &= \{5, 6\}; MB_{10,7} = \{9, 10\} \\
 MG_{10} &= \{1 \dots 10\}; LMM_{10} = \{5\} \\
 LMB_{10} &= \{1, 3, 6\}; LBM_{10} = \{2, 4\} \\
 PM_{10,2} &= \{(2, 3), (3, 2)\} \\
 PM_{10,4} &= \{(5, 6), (6, 5)\} \\
 PM_{10,7} &= \{(9, 10), (10, 9)\}
 \end{aligned}$$

Fig. 4. Gantt-chart of the optimized SMT-model (J_{10} emphasized).

booked as initial machine idle time I_k . Eqs. (4.21) and (4.22) force the adherence of job due dates (note: the MIP could be infeasible if there are due dates which cannot be met). Eqs. (4.23) and (4.24) restrict the objective function C_{max} . Furthermore, it is assumed that every job has at least one productive operation.

Theoretically, it is possible to couple this MIP-export with nearly every simulation or scheduling system. For this, only a short program for the system specific data input extraction (step 1) has to be written. Also, this program has to detect re-entrance loops of a job and to convert them into n new jobs and technologies without re-entrance loops to allow the MIP modeling shown above. Fig. 2 should illustrate this procedure.

Here, the example of a job (J_1) is exemplarily shown passing a machine (M_2) two times. To allow the shown MIP modeling this cycle has to be removed. For this, two new Jobs ($J_{1,1}$ and $J_{1,2}$) were created. Thereby, $J_{1,1}$ is passing machine M_1 and M_2 only one time and job $J_{1,2}$ is passing machine M_2 and M_3 only one time. An

additional need object ensures that the processing of $J_{1,2}$ can be started on machine M_2 when job $J_{1,1}$ has finished this station. So, in step 1 of data input extraction the two new jobs and the need object have to be initialized instead of job J_1 .

A practical manufacturing problem should illustrate the coupled optimization approach shown in this section. Here, a simulation model representing a SMT manufacturing line should be optimized (see Fig. 3). The model includes 20 jobs which have to be scheduled. Each job consists of a defined quantity of PCBs.⁴ The quantity represents the volume of the job which again influences the length of the processing times on the machines. The manufacturing system is initially filled. That means, there are jobs which have already finished some operations or are being currently in process. Also, there are jobs which have not entered

⁴ Printed circuit board.

the system until now. For those a supply time is given. Some jobs have a due date which has to be met. For all jobs, it is assumed that there is no preemption and no advanced transfer possible. The technological manufacturing process can be described in the following way: there are two different types of boards which have to be manufactured. Boards of type 1 have electronic devices only on the top side (route 1), type 2 has components also on the bottom side (route 2) of the board. The first technological process step is the printing of the soldering paste (M_1). Thereafter, the devices are placed on the boards (M_2 , M_3). Between these two steps a time coupling restriction is set to avoid drying out of soldering paste. After the placement the process step reflow soldering fixes the devices on the boards (M_4). For all jobs on route 2, the next technological process step is the SMT adhesive and placement (M_5 , M_6). Here, the devices were pasted on the bottom side of the board. A process for curing the adhesive follows (M_7). Then, a wave soldering operation fixes the electronic devices on the bottom side (M_8). Finally, the jobs of both routes perform an inspection step (M_9 , M_{10}). Here, the PCBs were tested. The SMT placement and SMT adhesive is done by one of two available machines. For the inspection step 2, parallel machines were used. Overall five different products have to be manufactured. Because of the different layout of the products, different printing masks in the first process step have to be used. So, here a product depending setup is necessary.

The aim is now to calculate a feasible schedule which minimizes the total makespan by keeping all described technological process constraints. Other objective functions (e.g. cycle time or tardiness) are also available and easily to adapt in the MIP model but not discussed here. To illustrate the MIP extraction of this model, we calculate exemplarily some index sets for job J_{10} (route 2) entering the system in approximately 2 h and having a due date in 8 h (Fig. 3 right).

With the help of the index sets it is possible to describe the whole problem with an MIP of only 614 unknowns (binary and continuous) and 845 constraints. So, this problem is solvable using TOMLAB CPLEX in near real-time (note: the complexity of the mathematical model strongly depends on the number of jobs and machines in the model as well as on the used objective function). Now, a special dispatch rule in the simulation model allows enforcing this exact solution during a simulation run. All features of the simulation system (like Gantt-Charts, etc.) could be used to check or to illustrate the results (see Fig. 4).

5. Coupling of simulation-based and solver-based optimization methods

As shown in Section 3, exact solution approaches (e.g. MIP) are beneficial especially for small problem instances. Complex manufacturing problems could be handled effectively by the method of simulation-based optimization. With the theoretical background of Section 4, a direct coupling and interaction of both methods is possible now. For this, a more complex manufacturing problem (i.e., higher number of jobs, machines) is investigated in this section (see Fig. 5). So, it is a usual way to attack this problem with a simulation-based approach. As a first step, the simulation model of the manufacturing line is automatically build up from the underlying ERP-system. Here the concept described in [11] is used. As a second step, this complex model is heuristically optimized by the method of simulation-based optimization. In this step, the current bottleneck of the manufacturing process is detected. Here different characteristics like machine utilizations or lot stocks can be used as indicators for bottlenecks. Because of general short planning horizons, shifting bottlenecks can be neglected. Now, a simulation model which represents this

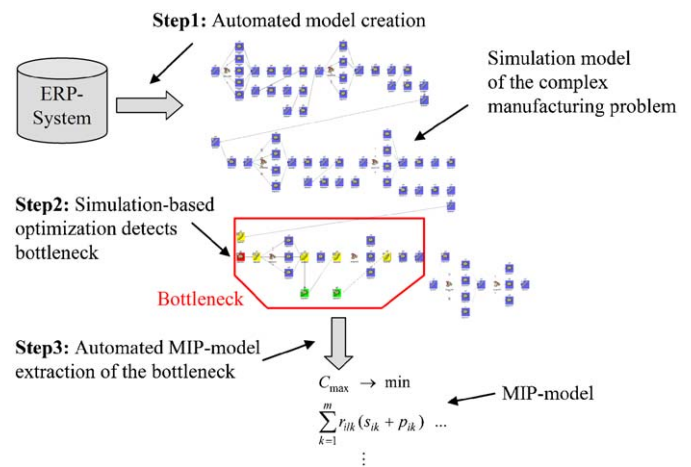


Fig. 5. Optimization of a complex manufacturing problem.

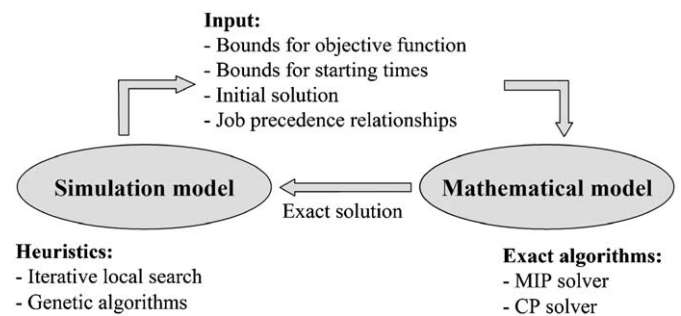


Fig. 6. MIP-model upgrade with results of the simulation-based optimization approach.

bottleneck could be extracted. Therefore, a multi-step optimization concept as shown in [22] can be used in most of the practical cases. The problem dimension of the bottleneck model (jobs, machines, etc.) is much lower than the original problem size. If it is low enough, an exact mathematical modeling, for example by an MIP solver, becomes possible. Now, for deriving an MIP from the bottleneck model, the method is used as shown in Section 4.

Apart from this derived MIP-formulation, different further useful information from simulation-based optimization of the bottleneck model upgrades the MIP model. These are explicit: upper and lower bound for the objective function, lower bounds for starting times as well as an initial feasible solution (see Fig. 6). This initial solution avoids time-consuming solver runs for finding a valid point in the search space. Because most of the simulation systems operate nondelay, good and strong bounds for objective functions and starting times become available. So, these additional information reduce the search space notably which provides a faster MIP-solver convergence.

If the problem dimension of the mathematical model is still too high, it can be reduced by some further job precedence or rolling forecast heuristics. For example, one powerful heuristic is the reduction of x_{ijk} . These unknowns and the resulting constraints (4.12) and (4.13) normally have the greatest influence on optimization convergence. Reduction means that some job precedence relationships were fixed. Concerning the SMT manufacturing line it is, for example, not really essential to define x_{ijk} between a job J_i entering the system and a job J_j which is currently processed by the wave soldering equipment for the step inspection (machine M_k). Obviously x_{ijk} will be 1 in most of the cases

then. If the relation is fixed, this x_{ijk} can be removed from the solution vector \mathbf{x} . Also, the Eqs. (4.12) and (4.13) were replaced with Eq. (4.25) for this specific i, j, k constellation.

$$s_{jk} - s_{ik} \geq p_{ik} \quad (4.25)$$

If there is more than one relation fixed, like $x_{ijk} = 1$ and $x_{ij'k} = 1$ for the three jobs J_i, J_j and $J_{j'}$, on the same machine M_k , then the definition of the constraints (4.12) and (4.13) become completely redundant for i, j', k . For the detection of potential fixations and for the definition of job precedence heuristic a simulation system can be used excellently. Next to process step differences, also relevant starting time differences between two jobs (recognized as a result of a simulation run) can be an indicator for fixing relationships.

After solving the (perhaps reduced by additional heuristics) optimization problem, the solution is transferred back to the simulation model. Now, the simulation of the global model with the optimized bottleneck allocation follows. All features of the simulation system like Gantt-Charts, diagrams or reports could be used to execute the optimized manufacturing process.

6. Summary

In the past, MIP solvers were only suitable for solving small – mostly not really practically relevant – scheduling problems. The number of Boolean and continuous variables in an MIP of a scheduling problem increases very fast with every object (job, machine), so neither the progress in computer technology nor in MIP solvers (e.g. the powerful CPLEX library) could really improve this situation. Our approach aims to a reduction in the number of variables by definition of special index sets as well as definition of job precedence relationship heuristics. We have developed an advanced Job Shop MIP model with a lot of additional restrictions, like release dates, due dates, branches, setups, etc. which allows a modeling of practical relevant scheduling problems. Thereby, this MIP-formulation is automatically generated from a simulation model. At one hand, this is comfortable for the user on the other hand it is also the basis for a coupling between an MIP solver and simulation-based optimization methods. We think this could be a key for solving a lot of real scheduling problems, especially if real-time conditions are requested. In general, for a successful optimization strategy of complex manufacturing systems, it seems to be necessary to find the specific bottlenecks at first. These

subsystems could be optimized by an MIP solver whereas heuristic algorithms are still used for the remaining part of the system.

References

- [1] Wagner HM. An integer linear-programming model for machine scheduling. *Naval Research Logistics Quarterly* 1959;6:131–40.
- [2] Browman EH. The scheduling-sequence problem. *Operations Research* 1959;7:621–4.
- [3] Manne AS. On the job shop scheduling problem. *Operations Research* 1960;8(2):219–23.
- [4] Wilson HM. Alternative formulations of a flow-shop scheduling problem. *Journal of Operational Research Society* 1989;40:395–9.
- [5] Morton TE, Pentico DW. *Heuristic Scheduling Systems*. New York: Wiley; 1993.
- [6] Brucker P. *Scheduling Algorithms*. Springer: Berlin; 2004.
- [7] Pinedo ML. *Planning and scheduling in Manufacturing and Services*. Springer Series in Operations Research; 2005.
- [8] Baptiste P, Le Pape C, Nuijten W. *Constrained-Based Scheduling*. Boston: Kluwer Academic Publishers; 2001.
- [9] Gupta AK, Sivakumar AL. Job shop scheduling techniques in semiconductor manufacturing. *International Journal of Advanced Manufacturing Technology* 2006;27:1163–9.
- [10] Klemmt A, Horn S, Beier E, Weigert G. Investigation of Modified Heuristic Algorithms for Simulation-Based Optimization. *Papers of the 30th International Spring Seminar on Electronics Technology* 2007, p. 24–29.
- [11] Weigert G, Horn S, Jähnig T, Werner S. Automated creation of DES models in an industrial environment. *16th International Conference FAIM, Flexible Automation & Intelligent Manufacturing* 2006;1:311–8.
- [12] <http://www.ilog.com/>.
- [13] Pan CH. A study of mixed integer programming formulation for scheduling problems. *Journal of System Science* 1997;28(1):33–41.
- [14] Blazewicz J, Dror M, Weglarz J. Mathematical programming formulations for machine scheduling: a survey. *European Journal of Operational Research* 1991;51:283–300.
- [15] Pearn WL, Chung SH, Yang MH, Chen YH. Algorithms for the wafer probing scheduling problem with sequence-dependent setup time and due date restrictions. *Journal of Operational Research Society* 2004;55(11):1194–207.
- [16] Pearn WL, Chung SH, Lai CM. Scheduling integrated circuit assembly operations on die bonder. *Electronics Packaging Manufacturing* 2007;30(2):97–105.
- [17] Pan CH, Chen JS. Minimizing makespan in re-entrant permutation flow-shops. *Journal of Operational Research Society* 2003;54:642–53.
- [18] Niemi E. Worker allocation in make-to-order assembly cells. *Proceedings of the 18th International Conference on Flexible Automation & Intelligent Manufacturing* 2008:1169–76.
- [19] Aydin ME, Fogarty T. A distributed evolutionary simulated annealing algorithm for combinatorial optimization problems. *Journal of Heuristics* 2004;10:269–92.
- [20] <http://bach.istc.kobe-u.ac.jp/csp2sat/jss/>.
- [21] Pinedo ML. *Scheduling: Theory, Algorithms and Systems*. Prentice Hall International Series in Industrial and System Engineering; 1995.
- [22] Weigert G, Klemmt A, Horn S. A multi-stage optimization approach for a semiconductor facility. *19th International Conference on Production Research*; 2007.