HEURISTIC AND METAHEURISTIC SIMULATION-BASED OPTIMIZATION FOR SOLVING A HYBRID FLOW SHOP SCHEDULING PROBLEM

A. Nahhas[a], P. Aurich[b], T. Reggelin[c], J. Tolujew[d]

[a]Department of Technical and Business Information Systems, Otto von Guericke University Magdeburg, Germany
[b],[c]Department of Logistics and Material Handling Systems, Otto von Guericke University Magdeburg, Germany
[d]Department of Logistics and Factory Systems, Fraunhofer Institute for Factory Operation and Automation, Germany

[a] abdulrahman.nahhas@hotmail.com, [b] paul.aurich@mail.de, [c] tobias.reggelin@ovgu.de
[d] juri.tolujew@iff.fraunhofer.de

ABSTRACT

This paper solves the hybrid flow shop (HFS) scheduling problem of a printed circuit board assembly. The production system investigated consists of four Surface-Mount Device (SMD) placement machines in the first production stage and five Automated Optical Inspection (AOI) machines in the second production stage. The objective is to minimize the makespan and to minimize the total tardiness. This paper describes and compares four approaches to solve the HFS scheduling problem: an integrated simulation-based optimization (ISBO) and three metaheuristics, simulated annealing, tabu search and genetic algorithm. All approaches lead to an improvement in terms of producing more jobs on time while minimizing the makespan compared to the decision rules used so far in the analyzed printed circuit board assembly. The integrated simulation-based optimization delivers results much faster than the metaheuristics. The metaheuristics lead to slightly better results in terms of total tardiness.

Keywords: Simulation-based optimization, hybrid flow scheduling problem, simulated annealing, tabu search, genetic algorithm, Meta-heuristics

1. INTRODUCTION

This paper describes the solution of a hybrid flow shop (HFS) scheduling problem with major and minor sequence-dependent setup times based on an industrial case of a printed circuit board (PCB) assembly. The objective was to minimize the makespan and the total tardiness. This paper is an extension and further analysis of the work of the authors, which will be presented in the Winter Simulation Conference 2016.

An HFS production environment consists of k production stages in series. Each production stage comprises m identical parallel machines. Each job j has to be processed on each production stage on one of the identical machines (Pinedo 2012). This problem is NP-hard (Lenstra et al. 1977). The paper proposes four different solutions to this HFS problem: an integrated simulation-based optimization algorithm (ISBO) developed by the authors and three widely used metaheuristics, simulated annealing, tabu search and genetic algorithm.

Scheduling is the deployment of resources in order to complete a set of tasks during a determined time span (Baker and Trietsch 2009). Scheduling problems have been extensively investigated in different fields of academia due to its essential role in manufacturing environments and different service sectors (Ruiz and Vázquez-Rodríguez 2010). Efficient allocation of resources supported by the appropriate sequencing is considered to be a major mathematical optimization problem (Lenstra et al. 1977). Johnson (1954) presented an optimal schedule for the two machine flow shop with sequence-dependent setup times, which is not as complex as an HFS problem. Direct optimization approaches have been previously implemented to solve HFS problems. Wittrock (1990) adopted a branch and bound algorithm to address the problem of identical parallel machines with major and minor sequence-dependent setup times, which can be considered as a simplified form of an HFS, and reported a near optimal solution. The branch and bound approach requires long computational time, even for small instances. Dynamic programming represents another direct optimization approach, which can be applied to solve HFS problems divided into smaller sub-problems (Baker and Trietsch 2009). The recursive behaviour of a dynamic programming approach facilitates the investigation of the whole solution space of a moderate size problem in reasonable computational time (Pinedo 2012).

Heuristics are used to obtain good solutions in reasonable computational time when the problem domain gets more complex (Allaoui and Artiba 2004). Priority Dispatching Rules (PDRs) are widely used in practice to define scheduling policies in manufacturing environments. PDRs are the simplest form of heuristics due to their ease of use and intuitive nature (Andersson et al. 2008). Shortest Production Time (SPT) and Earliest Due Date (EDD) are typical PDRs. They are often implemented to solve problems with a single objective function and they lack on solution quality as soon as the objective function gets more complex (Andersson et al. 2008). More sophisticated heuristics are adopted to deal with HFS scheduling problems. Voß

(1993) and Gupta (1988) used heuristics based on local search algorithms to solve a special case of an HFS with exactly one machine on the second stage and with the objective to minimize the makespan. This problem is still NP-hard (Gupta 1988). Local search algorithms are improvement procedures based on an initial feasible solution for the problem. They recursively search in the neighborhood of the initial solution for a better solution until a terminating condition is met.

Metaheuristics are often used to solve scheduling problems and are powerful solution approaches. Metaheuristics are guided local search algorithms. They are based on local search improvement algorithms and a general optimization or control strategy. The control strategy is used to guide the local search algorithms (Voudouris and Tsang 2003). The idea of metaheuristics is motivated by the fact that a local search algorithm often only obtains a local optimum from the solution space (Ross 2005). Simulated Annealing (Allaoui and Artiba 2004; Mirsanei et al. 2011), Tabu Search (Wang and Tang 2009) and Genetic algorithm are widely used metaheuristics.

2. SYSTEM DESCRIPTION

Any scheduling problem can be described and formulated based on the machine environment and configuration α, the job characteristics β and the objective function γ (Graham et al. 1979).

2.1. Machine environment and configuration

The analyzed production system is a hybrid flow shop, which consists of two production stages (see Figure 1). The first production stage contains four identical parallel surface mounting technology (SMT) assembly lines. The critical resource in the observed production lines is usually the surface mount device (SMD) placement machines (Csaszar et al. 2000). Consequently, we focused our analysis on the SMD placement machines. The second production stage contains five identical parallel automated optical inspection (AOI) machines. Each job j has to be processed on each production stage on one of the identical machines as it is shown in Figure 1.

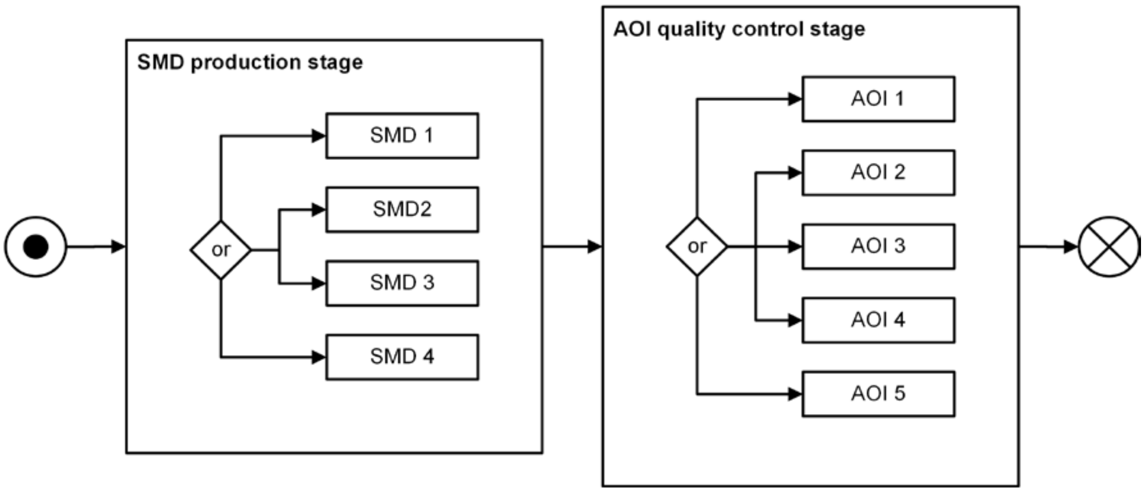

Figure 1: Two stages hybrid flow shop

2.2. Job characteristics

Jobs of the analyzed HFS scheduling problem can be characterized as follows:

- The number of jobs in a certain time period and the number of products per job are known and fix.
- Part types are very heterogeneous.
- The family type of a job depends on the used raw materials.
- The processing time $p_{j,m,s}$ of each job j on the machine m of stage s is known and fix.
- The priority of a job represents the delivery date to the customer.
- The sequence-dependent setup time $s_{j,k}$ is the time to setup the machine when changing from job j to job k.
- Machine breakdowns are not taken into consideration.
- Buffer size between production stages is unlimited.

In the first production stage (SMD), jobs are scheduled with sequence-dependent major and minor setup times on the machines. In the second production stage (AOI), jobs are scheduled incurring sequence-independent setup times on the machines. The concept of major and minor setup time was introduced by Wittrock (1990) as well as by So (1990) to describe sequence-dependency. As an illustration of this concept, jobs which share common raw materials, are grouped into families. On the one hand, a minor setup time will be inquired, if the machine switches from one part type to another inside the same family. On the other hand, a major setup time will be inquired, if the machine switches from one part type to another from a different family. In the first production stage job splitting is not permitted. More precisely, a production process of a job, once started, is not allowed to be interrupted for producing another job due to the fact that a major setup time is inquired to reconfigure the machine. Job splitting is allowed in the second production stage.

2.3. Objective functions

Accomplishing a balance between production system efficiency and the job's due-date is a trade-off decision. For this reason, tardiness has been frequently used as a major supplementary performance criterion along with the makespan (Lenstra et al. 1977). The objective functions of the analyzed HFS problem are to minimize the makespan C_{Max} and the total tardiness T. The makespan is the necessary time to complete all released jobs (Wittrock 1990). To minimize C_{Max} it is important to minimize the number of major setups. Tardiness is the difference between the completion time of a job C_j and its due date d_j as shown in (2).

$$C_{\max} = \max C_j \ \forall \ j = 1, \ ..., \ n \tag{1}$$

$$T = \sum_{j=1}^{n} T_j, \ T_j = \max \ (C_j - d_j, \ 0) \tag{2}$$

3. SOLUTION APPROACHES

The problem to minimize the makespan of a two stage hybrid flow shop is NP-hard (Gupta 1988). The development of a polynomial algorithm, which can provide an optimal solution in a reasonable time, is unlikely possible. Thus, breaking down the problem could be the key to obtain a near optimal solution by solving smaller sub-problems. It is often easier to solve the allocation and the sequencing independently (Baker and Trietsch 2009). Initially, jobs are allocated to the machines on each production stage. Four single machine problems with sequence-dependent setup times emerge on the first production stage and five single machine problems with sequence-independent setup times arise on the second production stage. A heuristic and metaheuristics were used to solve the allocation problem. A dynamic programming approach was used to develop a sequencing algorithm that builds a near optimal sequence of jobs on each machine.

The first solution strategy presented is an integrated simulation based optimization (ISBO). The ISBO integrates a heuristic and a sequencing algorithm into a simulation model. The second, the third and the fourth approach use metaheuristics: Simulated Annealing (SA), Tabu Search (TS), and Genetic Algorithm (GA) respectively. All metaheuristics are combined with a sequencing algorithm. Simulation models were used to assess the quality of the metaheuristics' solutions.

3.1. Integrated Simulation Based Optimization

In the integrated simulation Based Optimization (ISBO) (see Figure 2), the simulation is a part of the solution rather than an evaluation method for it. The allocation and sequencing algorithms are integrated in the simulation model. The simulation model was built with ExtendSim 9. The discrete-rate and discrete-event simulation-libraries were used to implement a hybrid mesoscopic simulation approach to avoid a long computation time (Reggelin and Tolujew 2011). The SMD and AOI production processes are modelled using the discrete-rate library. Flow rates differ depending on the current part type, being produced by the machines. The dispatching and decision making processes are modelled using the discrete-event library in order to ensure a high level of accuracy. The flow of a job is changed to a single object at decision points. When a job is released for processing, it is again modelled with a flow rate.

Product families and their jobs are initially allocated to the machines before the simulation starts. The shortest process time (SPT) discipline determines the initial allocation of the product families on the first production stage SMD. The earliest due date (EED) discipline initially allocates the jobs on the second production stage AOI. During the simulation, the interaction between the allocation and the sequencing algorithm leads to a sustainable production strategy with a near optimal sequence being continuously generated on each machine. The allocation algorithm ensure a balance of the production load between the machines.

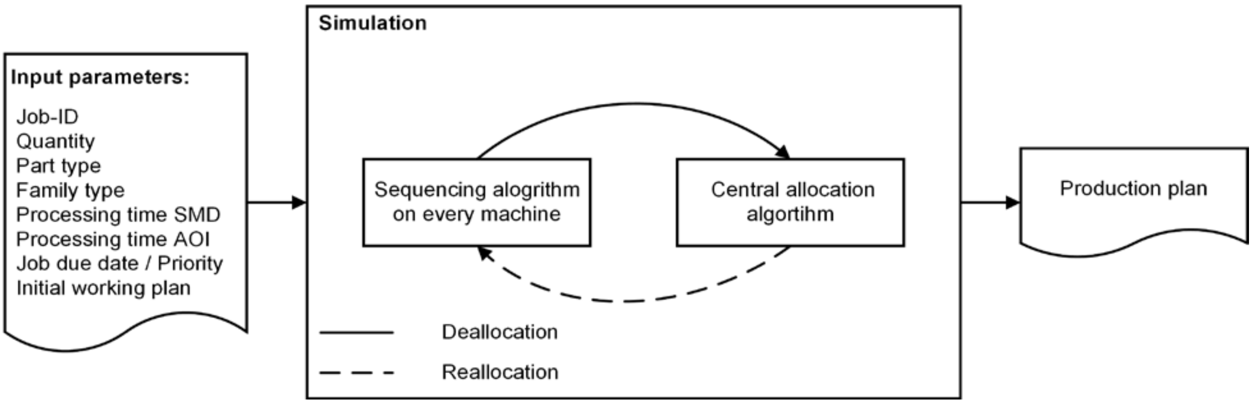

Figure 2 : Integrated Simulation Based Optimization

In order to minimize the makespan, all jobs of a product family should ideally be manufactured successively on the same SMD machine to avoid major setups. However, this would lead to delivery time violations of many jobs. The sequencing algorithm operates on two levels, the product family level and the job level (see Figure 3). On the family level, the smallest family, which contains at least one of the highest job priorities is chosen. Then, the sequencing algorithm switches to the job level. On the job level, the algorithm tends to dispatch jobs from the same family according to the priority of jobs using the EDD rule. The sequencing algorithm keeps operating on the job level until jobs of the family are completely produced or a critical point is met. The critical point describes a situation, when it is no longer possible to produce a job from the same product family without violating the delivery date of other jobs from different families. Choosing the smallest family increases the chance that the chosen family is completely produced before reaching a critical point. This behavior avoids a later major setup.

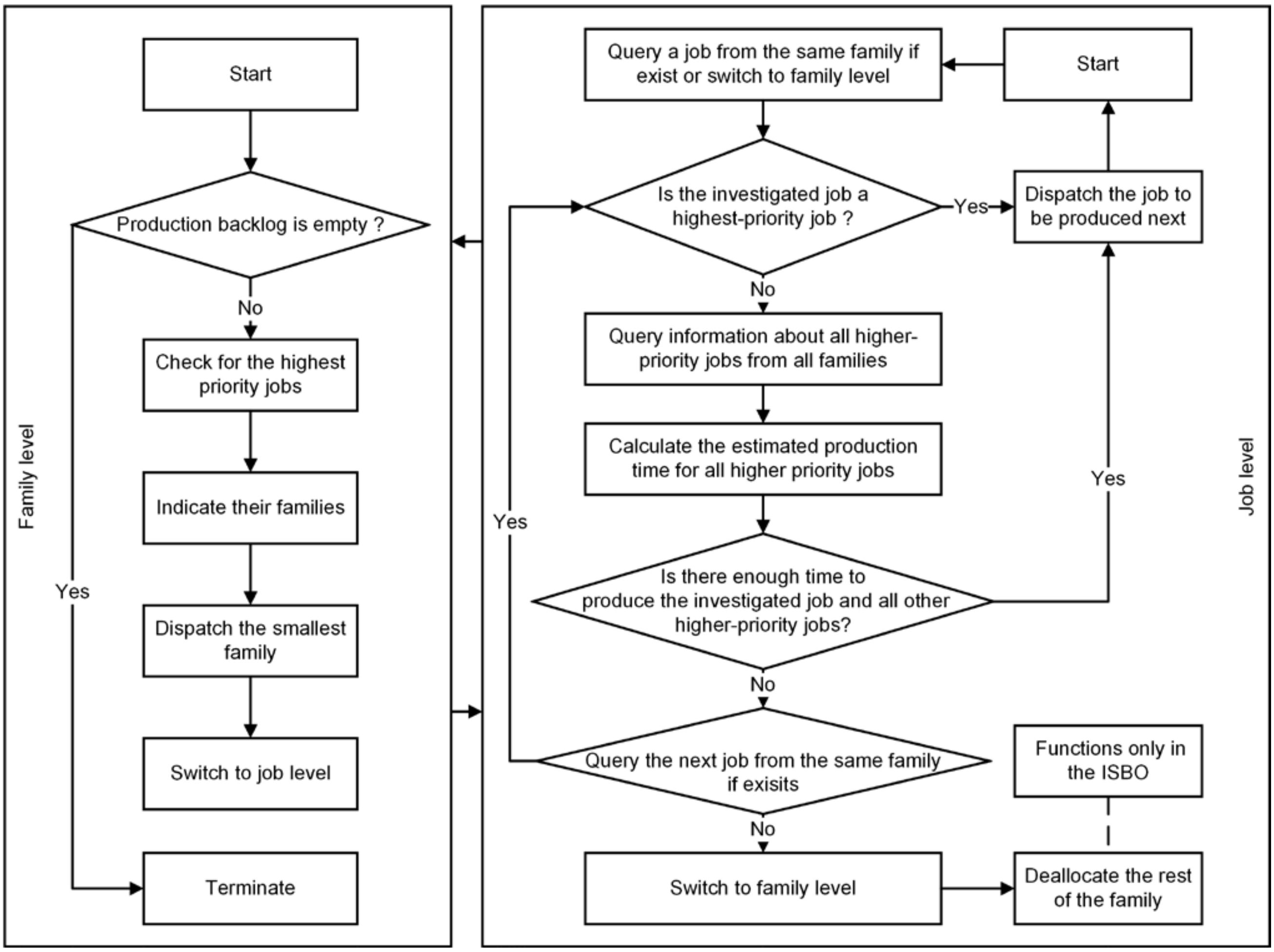

Figure 3: Sequencing Algorithm

The allocation algorithm tries to sustain a balance of the production load between the machines on each production stage. It performs two types of allocation, event-based allocation and pre-defined allocation. The event-based allocation is triggered by the sequencing algorithm (see Figure 2 and Figure 3) when critical points are reached. It checks for the least loaded machine and reallocates the remaining jobs of the family to this machine. The pre-defined allocation is performed each day to balance the production load of the next highest three priorities. All families except the one in production are deallocated. The allocation algorithm starts reallocating families to the least loaded machines during the next three simulated working days. It tends to balance the amount of must-be-produced jobs in the next three days according to their delivery date between the machines. The pre-defined allocation processes tries to avoid major setups by sustaining a balance of the must-be-produced jobs between the machines by avoiding critical points. Manipulating the allocation of families during the simulation better explores the solution space of the problem after significant changes in the production load. Producing from different families changes the form of the production load and therefore, finding an enhancement in the allocation is possible during the simulation despite a perfect initial allocation.

3.2. Simulated Annealing

Simulated annealing (SA) was combined with a discrete-event simulation model to solve the allocation. For the sequencing, the algorithm shown in Figure 3 was used again. Simulated annealing is derived from the concept of physical annealing of a solid substance. It was first introduced in the early eighties by Kirkpatrick et al. (1983) to solve combinatorial optimization problems. Annealing is the process of melting a solid substance and cooling it slowly down until the particles arrange themselves in the solid state (Aarts et al. 2005; Kirkpatrick et al. 1983; Mirsanei et al. 2011). When the temperature is high the particles are free to move randomly since they hold a high energy. In this state, the simulated annealing shows a very random behavior and is more likely to accept a worse solution than the current best solution (Mirsanei et al. 2011). When the cooling process starts, the solid state reduces the random behavior of the simulated annealing. The algorithm starts to search for a better solution in the same region of the solution space, rather than jumping from one region to another region.

Simulated annealing is used to solve the allocation problem on the SMD placement machines. The approach starts with a feasible solution to the problem as depicted in Figure 4. Then, the neighborhood search of the simulated annealing tends to find randomly a better solution in the current region of the solution space. The neighborhood search is based on a random single point operator (Naderi et al. 2009), in which a random family is picked and reallocated randomly to a different SMD placement machine. The number of changes (number of reallocated families) was restricted to one to avoid the simulated annealing behaving like a random search. After all families being allocated, the sequencing algorithm starts to build the production schedule of each SMD placement machine. After that, the jobs are allocated to the AOI machines based on their expected finishing time on the SMD placement machines. The allocation to the AOI machines tries to achieve a balanced production load between the machines and tries to consider the priorities of the jobs (due dates). The generated schedule is evaluated by using the discrete-event simulation model. The production sequences on the AOI machines are determined with the help of the EDD rule during the simulation run.

After passing the result of the simulation run back to the simulated annealing algorithm, three cases can be differentiated:

1. The new schedule dominates the old one in both objective values. The solution is accepted and used as the next start solution.
2. The old schedule dominates the new one.
3. Neither the old schedule nor the new one dominates.

For case two and case three, the Boltzmann distribution is used to decide whether to accept a new solution or not (Naderi et al. 2009). A weighted sum of the observed objective values was used since the Boltzmann distribution contains only one value. The probability of accepting a worse solution depends on the current temperature of the simulated annealing. The setup of the parameters of the simulated annealing strongly impacts its quality (Pirlot 1996). The parameters are initial temperature, the number of iterations before changing the temperature and the cooling rate. In this implementation, the simulated annealing starts with an initial temperature between 20 and 30 degrees. Each temperature contains 10 to 20 iterations. The implemented cooling schedule is linear and the cooling rate deviates between 0.1 and 0.25 degrees.

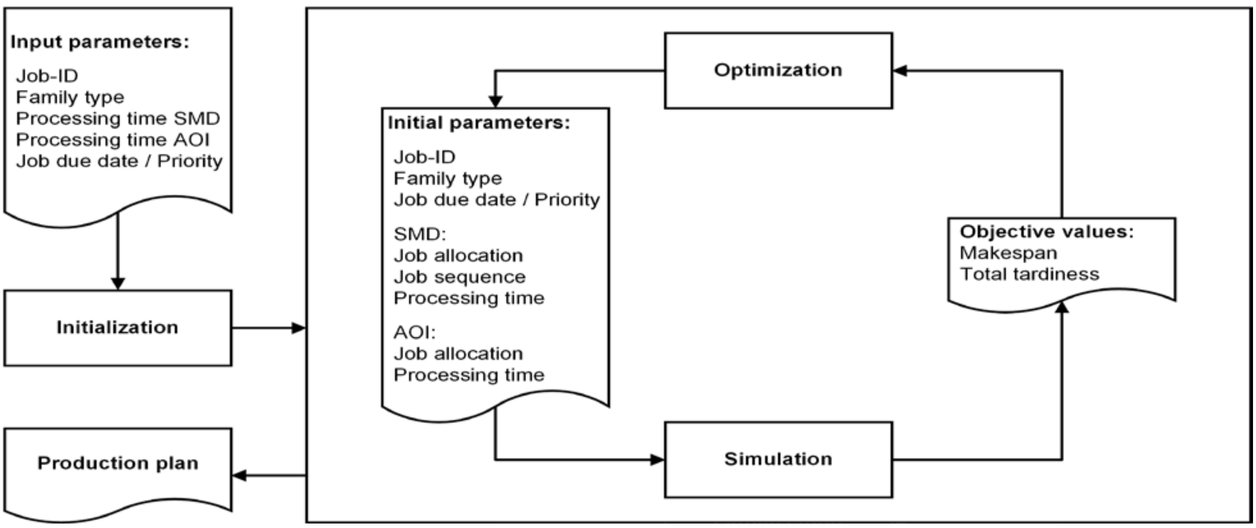

Figure 4: Metaheuristic simulation based optimization

3.3. Tabu Search

The Tabu Search (TS) algorithm was combined with the same discrete-event simulation model, which was used for the simulated annealing. For the sequencing, the algorithm shown in Figure 3 was used again. Tabu search is one of the oldest metaheuristic approaches, which was introduced by Glover (1986) to solve combinatorial optimization problems. In contrast to simulated annealing, tabu search is based on a deterministic solution mechanism, in which the neighborhood of the initial solution is build based on a set of specific moves, which are conducted on the initial solution to obtain new solutions. The current neighborhood is then investigated to identify the best solution. The initial solution is replaced by the best solution found, before starting the next iteration. The move which led to the current solution is stored in the tabu list and cannot be used again in order to avoid cycling. The aspiration function of the tabu search is used independently to evaluate the quality of the generated solutions of the moves from the tabu list to decide they can be used again (Nowicki and Smutnicki 1996).

The implementation of the tabu search in this paper is based on a single point operator neighborhood search. A single move is committed by picking a family and reallocating it to another machine. Each family is associated with three moves that generate three different solutions in the neighborhood. The solutions represent the possible allocations of each family to all considered SMD placement machines. For each generated new allocation, the sequencing algorithm is used to build the new production schedule on each SMD placement machine. Then, the jobs are allocated to the AOI machines, based on their expected finishing times on the SMD placement machines. Finally, the quality of each production schedule is evaluated using simulation. The results of the simulation runs are stored to identify the best solution and add its SMD allocation to the tabu list. The length of the tabu list is limited either to 10 or 15 solutions. Since two objective functions (makespan, tardiness) are considered, a weighted sum was used to identify the best solution before starting the next iteration. The forbidden schedules from the tabu list are evaluated using the aspiration function. If a dominant solution is found, it is used to start a new iteration.

3.4. Genetic Algorithms

The Genetic Algorithm (GA) was used to deal with the allocation problem. The sequencing part of the problem was solved also with the same algorithm shown in Figure 3. Genetic algorithms are guided random search techniques that are often used to solve scheduling problems (Andersson et al. 2008). They are categorized under evolutionary algorithms. The concept of genetic algorithm is based on mimicking the process of natural evolution (Ross 2005). Natural selection as well as genetic inheritance are fundamental element of a genetic algorithm. It maintains a population of candidate feasible solutions of the problem. A problem is first encoded in a genetic representation (Cheng et al. 1996). Each candidate solution is represented by a genome. Then, it simulates an evolution process of the candidate set of solutions to choose the best set. After the evaluation process, the best solutions are selected to evolve a new generation (offspring) of candidate solutions. The evolution process is usually performed by crossing-over the genomes of parents and/or combining them (Mutation), to indicated the genomes of the children solutions (Zheng and Wang 2003). It keeps iteratively performing these steps until some stopping criterion is met.

The genetic algorithm was implemented using ExtendSim 9. A discrete-event model was built and the optimizer of ExtendSim 9 was used. This optimizer is based on genetic algorithm. The allocation problem was encoded and passed to the optimizer as a genome. Then the optimizer starts generating candidate solution, which represents different allocation possibilities of families to the SMD placement machines. For each generated new allocation, the sequencing algorithm is used to build the new production schedule on each SMD placement machine. Then, the jobs are produced on AOI machines according to the priority of jobs using the EDD rule. By the end of a simulation run, the objective values are send back to the optimizer to continue the evolvement process of the next generation of solutions. The general functionality of genetic algorithm can be also derived from Figure 4. In this implementation, the population size of the candidate solutions is 50.

4. COMPUTATIONAL RESULTS

The experiments have been performed on four different datasets, which were provided by the company. The datasets are heterogeneous in terms of the considered number of families, number of jobs and their associated part types as shown in table 1. A large number of families and over 650 different product types have been taken into consideration in all datasets. The major setup time averages 65 minutes and the minor setup time 20 minutes.

Table 1: Input datasets

	Dataset 1	**Dataset 2**	**Dataset 3**	**Dataset 4**
Number of jobs	164	170	175	143
Number of families	41	37	36	35
SMD processing time interval (min)	4 - 3,142	2 - 3,736	4 - 3,293	4 - 3,209
Accumulated SMD processing time (min)	54,685	62,345	61,274	56,250
AOI processing time interval (min)	4 - 4,351	3 - 5,590	5 - 3,528	3 - 4,300
Accumulated AOI processing time (min)	72,528	88,702	74,738	79,294
Quantity of PCB (parts)	40 - 109,920	20 - 143,040	21 - 186,960	20 - 216,000

Table 2 shows the computational results of the approaches used to solve the HFS problem. The Family Production (FP) scenario is a batch production strategy, which has been so far adapted by the company to set scheduling policies for their production. In the FP, a machine, once started producing jobs from a family, is not allowed to be switched to another family until the family is fully produced. Consequently, this scenario point out the minimum number of inquired major setup times to produce all jobs of any data set presented. As it was expected, the reported results of the standard priority dispatching rules didn't meet the required performance criteria. Although, the shortest production time rule is often used to minimize the makespan, the indicated sequence-dependent setup times radically impacted their results and reported inefficient results for both objective values. The earliest due date rule also didn't meet the requirements of minimizing the total tardiness and reported, in addition to the high number of inquired setup times, a violation in the delivery dates of jobs in most of the datasets.

Table 2: Computational results of the different solution approaches

	Makespan (minutes)	**Major-Setup (number)**	**Penalty (number)**	**AVG Tardiness (minutes)**
Dataset 1				
FP	23,513	37	39	5,097
SPT	23,586	126	30	2,883
EDD	21,154	104	0	0
ISBO	19,354	43	1	148
SA	21,930	45	0	0
TS	19,669	45	0	0
GA	17,786	43	0	0
Dataset 2				
FP	25,447	33	52	5,225
SPT	26,662	135	31	4,811
EDD	26,226	136	9	537
ISBO	21,819	53	0	0
SA	23,108	55	0	0
TS	25,142	55	0	0
GA	22,458	53	0	0
Dataset 3				
FP	23,626	32	62	5,060
SPT	25,756	131	36	4,143
EDD	22,603	139	8	750
ISBO	19,979	56	2	268
SA	23,059	59	0	0
TS	22,507	60	0	0
GA	22,671	53	1	28
Dataset 4				
FP	23,539	31	48	5,362
SPT	20,507	113	26	4,430
EDD	21,145	113	2	482
ISBO	18,806	42	3	213
SA	20,562	58	0	0
TS	21,610	57	0	0
GA	22,569	45	0	0

The ISBO, the Simulated Annealing (SA), the Tabu Search (TS) and the Genetic Algorithm (GA) have reported significant improvements in terms of the makespan and the total tardiness in comparison to the currently conducted scheduling polices in the company. The ISBO delivers for all datasets an improved production schedules, which are more concentrated on the efficiency of the production system. The ISBO slightly outperformed TS, SA and GA in terms of the makespan. This is caused by the dynamic behavior of the allocation algorithm implemented in the ISBO, which tends to balance the critical jobs and their families instantly during the simulation. The behaviors of the TS and the SA are relatively identical towards the considered system. Both solutions strategies reported production schedules without recording any penalties and slightly outperformed the ISBO in terms of total tardiness. GA slightly outperformed both SA and TS in terms of the number of major setup times.

The makespan optimization with sequence-dependent setup time requires reducing the inquired setup times as well as the achieving a balance in the production load between machines. Those two goals tend to be often conflicting goals in any system and optimizing one of them would lead to deteriorating the other (Wittrock 1990). For this reason, the obtained results from the SA as well as from the TS optimized the total tardiness, whereby the makespan witnessed an increase in all datasets. However, minimizing the inquired number of major setup times in a dynamic system, where the production backlog is never empty, leads to minimizing the makespan.

The reported results from the ISBO have been obtained from a single simulation run, which required approximately 1 minute computation time. The SA has been configured to run 1500 simulation runs and reported the results for the datasets after approximately 3 hours computation time. The TS was configured to run between 20 to 30 TA iterations, which roughly correspond to 2100 and 3150 simulation runs respectively. The number of simulation runs in the TS depends on the number of the considered families in the dataset. The genetic algorithm is configured to run 6000 to 8000 simulation runs. The experiments have been conducted on a computer with the following characteristics: CPU 4 x 2.6 GHz, RAM 8 GB and windows operating system.

5. CONCLUSION

The paper has shown that the four applied solution approaches integrated simulation-based optimization (ISBO), simulated annealing (SA), tabu search (TS) and genetic algorithm (GA) could solve the hybrid flow shop (HFS) scheduling problem better than the decision rules used very often in practice in the printed circuit board assembly. All approaches lead to an improvement in terms of minimizing the makespan and producing more jobs on time. The ISBO delivers results much faster than the three metaheuristics SA, TS and GA. The metaheuristics lead to slightly better results in terms of total tardiness.

The dynamic allocation as applied in the suggested ISBO allows for a very deep investigation of the solution space during the simulation and thus achieves very good results in terms of minimizing the makespan compared to the metaheuristics SA, TS and GA. The experiments with four real data sets have revealed one major challenge of solving HFS scheduling problems: Big jobs can lead to difficulties in finding a good solution.

REFERENCES

Aarts, E., J. Korst, and W. Michiels. 2005. "Simulated annealing." In *Search Methodologies,* edited by E. K. Burke and G. Kendall, 187–210, Boston, MA: Springer US.

Allaoui, H., and A. Artiba. 2004. "Integrating simulation and optimization to schedule a hybrid flow shop with maintenance constraints." Computers & Industrial Engineering, 47 4, 431–450.

Andersson, M., A. H. C. Ng, and H. Grimm. 2008. "Simulation optimization for industrial scheduling using hybrid genetic representation." In *Proceedings of the 2008 Winter Simulation Conference,* 2004–2011.

Baker, K. R., and D. Trietsch. 2009, Principles of Sequencing and Scheduling, Wiley Publishing.

Cheng, R., M. Gen, and Y. Tsujimura. 1996. "A tutorial survey of job-shop scheduling problems using genetic algorithms—I. representation." Computers & Industrial Engineering, 30 4, 983–997.

Csaszar, P., T. M. Tirpak, and P. C. Nelson. 2000. "Optimization of a high-speed placement machine using tabu search algorithms." Annals of Operations Research, 96 (1), 125–147.

Glover, F. 1986. "Future paths for integer programming and links to artificial intelligence." Computers & Operations Research, 13 5, 533–549.

Graham, R. L., E. L. Lawler, J. K. Lenstra, and A. R. Kan. 1979. "Optimization and approximation in deterministic sequencing and scheduling: a survey." Annals of discrete mathematics, 5, 287–326.

Gupta, J. N. D. 1988. "Two-stage, hybrid flowshop scheduling problem." Journal of the Operational Research Society, 359–364.

Johnson, S. M. 1954. "Optimal two-and three-stage production schedules with setup times included." Naval research logistics quarterly, 1 (1), 61–68.

Kirkpatrick, S., C. D. Gelatt, and M. P. Vecchi. 1983. "Optimization by Simulated Annealing." Science, 220 4598, 671–680.

Lenstra, J. K., A. R. Kan, and P. Brucker. 1977. "Complexity of machine scheduling problems." Annals of discrete mathematics, 1, 343–362.

Mirsanei, H. S., M. Zandieh, M. J. Moayed, and M. R. Khabbazi. 2011. "A simulated annealing algorithm approach to hybrid flow shop scheduling with

sequence-dependent setup times." Journal of Intelligent Manufacturing, 22 6, 965–978.
Naderi, B., M. Zandieh, and V. Roshanaei. 2009. "Scheduling hybrid flowshops with sequence dependent setup times to minimize makespan and maximum tardiness." The International Journal of Advanced Manufacturing Technology, 41 11-12, 1186–1198.
Nowicki, E., and C. Smutnicki. 1996. "A fast tabu search algorithm for the permutation flow-shop problem." European Journal of Operational Research, 91 (1), 160–175.
Pinedo, M. L. 2012, Scheduling: theory, algorithms, and systems, Springer Science & Business Media.
Pirlot, M. 1996. "General local search methods." European Journal of Operational Research, 92 3, 493–511.
Reggelin, T., and J. Tolujew. 2011. "A mesoscopic approach to modeling and simulation of logistics processes." In *Proceedings of the 2011 Winter Simulation Conference*, 1508–1518.
Ross, P. 2005. "Hyper-Heuristics: Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques." In *Search Methodologies*, edited by E. K. Burke and G. Kendall, 529–556, Boston, MA: Springer US.
Ruiz, R., and J. A. Vázquez-Rodríguez. 2010. "The hybrid flow shop scheduling problem." European Journal of Operational Research, 205 (1), 1–18.
So, K. C. 1990. "Some heuristics for scheduling jobs on parallel machines with setups." Management Science, 36 4, 467–475.
Voß, S. 1993. "The Two - Stage Hybrid - Flowshop Scheduling Problem with Sequence - Dependent Setup Times." In *New Directions for Operations Research in Manufacturing*, edited by G. Fandel, et al., 336–352, Berlin, Heidelberg: Springer Berlin Heidelberg.
Voudouris, C., and E. P. K. Tsang. 2003. "Guided Local Search." In *Handbook of Metaheuristics*, edited by F. Glover and G. A. Kochenberger, 185–218, Boston, MA: Springer US.
Wang, X., and L. Tang. 2009. "A tabu search heuristic for the hybrid flowshop scheduling with finite intermediate buffers." Computers & Operations Research, 36 3, 907–918.
Wittrock, R. J. 1990. "Scheduling parallel machines with major and minor setup times." International Journal of Flexible Manufacturing Systems, 2 4, 329–341.
Zheng, D.-Z., and L. Wang. 2003. "An Effective Hybrid Heuristic for Flow Shop Scheduling." The International Journal of Advanced Manufacturing Technology, 21 (1), 38–44.

AUTHORS BIOGRAPHY

ABDULRAHMAN NAHHAS is a master student of Business Informatics at the Otto von Guericke University Magdeburg. His main work interests are scheduling problems and simulation based optimization solution strategies. Abdulrahman Nahhas holds a bachelor degree in Business Informatics from Otto von Guericke University Magdeburg. His email address is abdulrahman.nahhas@hotmail.com.

PAUL AURICH is a master student of Industrial Engineering with specialization in production systems at the Otto von Guericke University Magdeburg. His main work interests are scheduling problems and simulation based optimization solution strategies. Paul Aurich holds a bachelor degree in Industrial Engineering with specializing in logistics from Otto von Guericke University Magdeburg. His email address is paul.aurich@mail.de.

TOBIAS REGGELIN is a project manager at the Fraunhofer Institute for Factory Operation and Automation IFF and Otto von Guericke University Magdeburg. His main research and work interests are logistics system modeling and simulation and the development and conduction of logistics management games. Tobias Reggelin received a doctoral degree in engineering from Otto von University Magdeburg. Furthermore, he holds a diploma degree in industrial engineering in logistics from Otto von Guericke University Magdeburg and a master's degree in Engineering Management from Rose-Hulman Institute of Technology in Terre Haute, IN. His email address is tobias.reggelin@ovgu.de.

JURI TOLUJEW is a project manager in the Department for Logistic Systems at the Fraunhofer Institute for Factory Operation and Automation IFF in Magdeburg, Germany. He received a Ph.D. degree in automation engineering in 1979 from the University of Riga. He also received a habil. degree in computer science from the University of Magdeburg in 2001. His research interests include the simulation based analysis of production and logistics systems, protocol based methods for analyzing processes in real and simulated system as well as mesoscopic approaches in the area of simulation. He is an active member in the ASIM, the German organization of simulation. His email address is juri.tolujew@iff.fraunhofer.de.