Proceedings of the 2013 Winter Simulation Conference
R. Pasupathy, S.-H. Kim, A. Tolk, R. Hill, and M. E. Kuhl, eds

STUDY ON MULTI-OBJECTIVE OPTIMIZATION FOR PARALLEL BATCH MACHINE SCHEDULING USING VARIABLE NEIGHBOURHOOD SEARCH

Robert Kohn
Oliver Rose

Universität der Bundeswehr München
Institut für Technische Informatik
Fakultät für Informatik
Neubiberg, 85577, GERMANY

Christoph Laroque

University of Paderborn
Heinz Nixdorf Institut
Business Computing, esp. CIM
Paderborn, 33102, GERMANY

ABSTRACT

Managing multiple objectives is a crucial issue coming up with scheduling solutions in wafer fabrication. This paper presents computational results for solving Parallel Batch Machine Problems (PBMSP) with Variable Neighborhood Search (VNS), enriched with experiences from industry. Based on experiments, we present correlation factors between most common Key Performance Indicators (KPI) considered as objectives, evaluating the strength and direction of their inter-relationships. We discuss experiments for pareto objective functions and weighted objective functions, composed of important KPIs. We place great importance on the specific role of critical constraints in a scheduling system empowered by optimization, e.g. time bounds and minimum batch sizes. The pure existence of critical constraints necessarily requires multi-objective function optimization. By experiments, this paper examines hierarchical objective functions managing maximum time bounds and minimum batch sizes, discussing solution strategies and pitfalls.

1 INTRODUCTION

Wafer fabrication as a complex job shop with numerous industry-specific characteristics and various area-specific constraints is widely considered to be one of the production systems most difficult to control. Since the very beginning, the prevailing system for operational shop floor control in this industrial sector is dispatching, which has been continuously improved in its ability to control work in progress (WIP). Even though today's dispatching systems have reached an impressive level of complexity, offering good working solutions for most control problems, the next-generation shop floor control system will be significantly characterized by powerful scheduling strategies. Several authors from industry present scheduling systems successfully implemented on operational level, reporting substantial benefits, even compared to highly developed dispatching systems (cf. Bixby et al. 2006; Yurtsever et al. 2009).

In the process of replacing dispatching by scheduling systems, usually the first step is to apply operational scheduling to a number of parallel machines that belong to the same area. By decomposition the entire job shop divides into disjunctive sets of parallel machines, which are typically connected to each other by re-entrant material flows. In general, reducing complexity by use of decomposition techniques enables us to partially tackle a problem considered before as too large, or too complex, or both. In particular, separately scheduling sets of parallel machines from the same type makes sense in minimum two directions, both reducing complexity. On the one hand, structural complexity is reduced by exclusively focusing on number of machines of a single process area, and that makes experts' life easier in the face of numerous area-specific characteristics that exist. Therefore, a solution for a certain Parallel Machine

978-1-4799-2076-1/13/$31.00 ©2013 IEEE

Scheduling Problem (PMSP) is intentionally developed to exclusively meet the area-specific requirements emerging in the focused area. On the other hand, we reduce computational complexity, which increases exponentially and results into billions of decision alternatives that finally form a schedule. Solving PMSPs is considered to be one of the most promising use cases in the area of scheduling on operational level. PMSPs seem to be technical feasible and promise to bring substantial benefit.

Especially PMSPs with batching are at the focus of researchers and experts from industry, in the further course of the work mentioned as Parallel Batch Machine Scheduling Problems (PBMSPs). PBMSPs are extensively studied because wafer fabrication batch machines, substantially contribute to lot cycle times, a key performance indicator (KPI) for manufacturing facilities. PBMSPs, usually located in the furnace area covering diffusion and oxidation processes, come with various constraints that are of high importance and need special treatment when developing a scheduling system. Some (critical) constraints may lead to infeasible schedules in certain situations, e.g. minimum batch sizes and maximum time bounds. Especially those critical constraints that could cause invalid solutions during the optimization process warrant attention in every scheduling system concept. Despite those challenges, the scheduling community expects remarkable improvements for KPIs when applying scheduling for PBMSPs; especially in combination with a simulation system that provides job arrival predictions.

Since most PMSPs are proofed to be NP-hard, metaheuristics examined for scheduling problems provide the greater part in research activities. Beside metaheuristics, Mixed Integer Programming (MIP) and Dynamic Programming (DP) that represent exact techniques play an important role - even if only to proof global optima for small problem instances. But, bearing in mind that technological development is not expected to slow down, exact methods combined with decomposition techniques have good chances to assert influence on problem instances with practical dimensions. In the field of population-based metaheuristics, a lot of work has been done to examine Evolutionary Algorithms (EA) and approaches based on Ant Colony Optimization (ACO). With respect to trajectory metaheuristics, we are seeing a considerable amount of literature concentrating around Variable Neighborhood Search (VNS) and Greedy Randomized Adaptive Search (GRASP). So far, the challenge of computational complexity is jointly responded by exact methods, decomposition techniques and powerful metaheuristics.

Researchers have been primarily focused on the issues feasibility and performance when developing optimization models, regardless whether they chose exact or heuristic approaches. Especially for PBMSPs, an optimization models' feasibility to be finally applied on the shop floor relies on various constraints (e.g. incompatible job families, process dedication, time bounds, minimum batch sizes, etc.) that need to be properly considered by appropriate implementations. With respect to performance of optimization methods, researches worked on scheduling procedures characterized by an acceptable ratio between solution quality and computation time. Thus, the community has been primarily fighting challenges related to performance and constraints, the majority in literature addresses scheduling problems with single component objectives. Nevertheless, there still exist countless open questions for scheduling problems with single objectives, the view slightly moves to Multi-Objective Optimization (MOO). For certain PBMSPs incorporating critical constraints that can cause infeasible solutions, MOO emerges as a necessity than an optional feature nice to have.

2 STATE-OF-THE-ART

As a lead-in to the vast variety of literature dealing with scheduling in the area of semiconductor manufacturing operations, we would like to refer to a recent survey given in (Mönch et al. 2011). In the following, we will give a short review of most important batching heuristics. For a more detailed review of batch scheduling (and dispatching), one will find an extensive overview in (Mathirajan et al. 2006).

Glassey et al. (1991) present the Dynamic Batching Heuristic (DBH) that for the first time incorporates information about future job arrivals, referred to as look-ahead in the following text. Fowler et al. (1992) introduce the more sophisticated Next Arrival Control Heuristic (NACH) that exclusively considers the next job arrival (cf. Fowler et al. 2000; Solomon et al. 2002). Van der Zee proposes the Dynamic

Job Assignment Heuristic (DJAH) that also considers look-ahead information (cf. van der Zee 2007). Habenicht and Mönch (2003) discuss batching heuristics combined with Genetic Algorithms (GA) in order to minimize tardiness measures. Balasubramanian et al. (2004) present the Batched Apparent Tardiness Cost (BATC) heuristic for weighted tardiness minimization, empowered by a search scheme based on GA. Gupta and Sivakumar (2006) propose a due-date oriented control strategy using look-ahead information in order to minimize maximum tardiness and number of tardy jobs. Sha also discusses look-ahead dispatching heuristics (cf. Sha et al. 2007). Moreover, several batching heuristics aim to minimize total weighted tardiness (TWT) are discussed in (Kim et al. 2010).

In the area of scheduling empowered with optimization methods, there exists a variety of approaches to solve PBMS problems under consideration of various subsets of constraints. Reichelt and Mönch (2006) examine Multi-Population Genetic Algorithm (MPGA) and a Multi-Objective Genetic Algorithm (MOGA) for the PBMSP with the goal to minimize TWT and makespan at the same time. In (Klemmt et al. 2011, Wang et al. 2010, Klemmt et al. 2008) one will find MIP formulations for PBMSPs, which aim to minimize makespan or total weighted tardiness. Only for makespan optimization, Wang and Chou (2010) present a comparison between MIP and approaches based on Simulated Annealing (SA) and GAs, both enhanced with multi-stage dynamic programming (MSDP). Various evolutionary approaches, respectively nature-inspired search schemes based on the GA concept have also been published, e.g. Chiang et al. (2010), Kashan et al. (2008), Malve and Uzsoy (2007), Mönch et al. (2005), Balasubramanian et al. (2004) and Habenicht and Mönch (2003). Damodaran and Velez-Gallego (2011) present a GRASP and an implementation of the SA search in (Damodaran and Velez-Gallego 2012), both in order to minimize makespan. There exist approaches employing the ACO concept in order to minimize TWT for PBMSPs (cf. Mönch et al. 2009, Li et al. 2008, Raghavan and Venkataramana 2006). In Cakici et al. (2013) heuristic algorithms employing VNS schemes are discussed, and compared to a mathematical model developed for the PBMSP with dynamic arrivals and incompatible job families. Almeder and Mönch (2011) study popular metaheuristics applied to the PBMS with incompatible job families in order to minimize TWT; they examine variants of ACO, GA and VNS, and compared their performance. Klemmt et al. (2009) studied PBMS with incompatible job families and dynamic job arrivals, comparing MIP and VNS with respect to TWT minimization. Jula and Leachman (2010) present a algorithm based on Linear-Programming (LP), an approach based on Integer Programming (IP), and a heuristic-based algorithm to solve non-homogenous PBMSPs with non-identical job sizes and incompatible job families. Yugma et al. (2008) present a solution approach based on Simulating Annealing (SA) applied for PBMSPs improving throughput, batch efficiency and flow time factor.

3 DESIGN OF EXPERIMENTS

3.1 Experimental System

The experimental environment primarily consists of a simulation-based optimization framework developed to solve PBMSPs existing in wafer fabrication front-end facilities. The underlying system covers various PBMSPs that differ in their sets of constraints and objectives. For searching improved schedules, we implemented a generalized concept of VNS, offering numerous VNS variants, including most of those mentioned in related literature. The difference between those variants is basically the balance between exploitation and exploration of the search space, beside countless different parameter combinations to choose.

We consider the developed scheduling system as ready-for-pilot on operational level, meaning that we are able to load (and validate) a currently existing problem instance with actual data from fab databases. Followed by the scheduling procedure that creates an improved schedule, which is immediately written back to the manufacturing execution system (MES) for execution. In addition to implemented real-world and real-time features, a model generator offers to create user defined PMSP instances with specific characteristics. A model generating engine generates certain problem instances with specific characteristics basically described by. The model generator sets important model variables using random numbers following standard statistical distributions in order to generate a set of independent model instances that fundamentally show equal characteristics, but also differ slightly from each other. We use a database to establish the data management necessary to run the experimental system in an effective (and comfortable) manner. The entire experiment data representing the input parameters (including the model instances) as well as the and output results is accessible via database connections. The entire system is limited to PMSP instances that do not exceed 5 GB in their compressed size, which equals to the maximum size for a Character Large Object (CLOB). Since large simulation (optimization) experiments often suffer from the lack of computing power and time available, we delegate extensive studies to a High Performance Computing (HPC) cluster with 64 cores connected to the database. The entire system is written in C#, with more focus on code comprehensibility rather than speed in computation, so far.

3.2 Parallel Batch Machine Scheduling Problem (PBMSP)

We describe the scheduling problems examined in this study by use of the α|β|γ-classification scheme proposed in Graham (1979). The basic PBMSP under study incorporates unequal processing times, job specific machine dedications, parallel batching with incompatible job families and arbitrarily maximum batch sizes for a job family on a machine. According to the α|β|γ-notation, and considering the fact that the objective in our studies is changing and subject to analysis, thus simply denoted with *objective*, we describe the problem with *Rm|*M_j*,p-batch,incompatible,*$bmax_j$*|objective.* The set of implemented performance measures, which basically define objective functions, relates to common KPI, respectively throughput, flow time and tardiness; particularly extended with weights representing different job priority classes. The objective function compounds at least one of those performance measures, but principally numerous combined, with weights or hierarchical. The following list introduces the α|β|γ-notations used throughout the rest of the paper; that are

- *Rm:* unrelated parallel machines (with unequal processing times),
- M_j*:* machine dedications (a job is dedicated to a restricted set of machines),
- *p-batch:* parallel batching (a number of jobs is processed simultaneously on a machine),
- *incompatible:* incompatible job families (jobs of different families cannot be processed together),
- $bmax_j$*:* arbitrarily maximum batch size for a job family on a machine,
- $bmin_j$*:* arbitrarily minimum batch size for a job family on a machine,
- tb_j*:* arbitrary time bound constraint for a job.

The developed scheduling system allows us to examine PBMSPs with an arbitrary chosen subset of listed constraints, and with multi-component objective functions combining common KPIs together. Earlier in this paper we mentioned critical constraints, respectively minimum batch sizes ($bmin_j$) and maximal time bounds (tb_j). The basic model extending, we also examine the PBMSPs *Rm|*M_j*,p-batch,incompatible,*$bmax_j$*,*$bmin_j$*,|objective* and *Rm|*M_j*,p-batch,incompatible,*$bmax_j$*,*tb_{kj}*,|objective.* The special intentions of those critical constraints that may cause invalid solutions, and their implicit influence on the objective function that necessarily ends up in a multi-component objective function, is discussed in section 5.3.

Despite of the fact that we are able to compute PBMSPs with dynamic arrivals, respectively *Rm|*M_j*,*r_j*,p-batch,incompatible,*$bmax_j$*,|objective* where r_j stands for a non-zero release date of a job, and even though considering dynamic arrivals (look-ahead) remarkably contributes to optimization potential, we omitted them in this study. The reason is that we partially chose overall makespan minimization for objective, which would not make much sense when dealing with dynamic arrivals. To avoid any confusion and to keep the experiments in a clear and comparable way, we did not investigate dynamic arrivals in any experiment mentioned in this paper. Consequently, it is also superfluous to consider different utilization levels or minimum batch size constraints; this statement holds for PBMSPs without dynamic arrivals.

Within experiments, we consistently investigated three sizes of problem instances, denoted with S(5|150), M(10|300) and L(15|450), where the first parameter represents the number of machines and the second one the number of jobs. The default model settings are given in table 1.

Table 1: Default model settings

Model (machines | jobs)	S (5|150)	M (10|300)	L (15|450)
# Machines	5	10	15
# Jobs	150	300	450
# Job families	5	10	15
Dedication density	0.7		
Process time [min]	U~(240,360)		
Initial tardiness [hrs]	N~(0,24)		
Job weights	U~(1,2,3,4,5)		
Maximum batch size [lot]	Const~(8)		

3.3 Variable Neighborhood Search (VNS)

Mladenović and Hansen (1997) proposed the VNS heuristic based on neighborhood structures used to solve large scale combinatorial problems. The simulation-based optimization framework that we use to solve PBMSPs employs VNS to create optimized schedules with respect to focused objectives. We implemented VNS as an abstraction of the proposed schemes, which allows us to freely configure two nested search levels. Both levels can be parameterized independently from each other, where each search level defines a set of neighborhood structures, the local search procedure (first improvement or best improvement), and the shaking policy managing the shaking range (either constant or increasing). This generalized implementation of VNS covers a wide range of VNS variants described in literature, namely Reduced VNS (RVNS), Variable Neighborhood Descent (VND), Generalized VNS (GVNS), Variable Neighborhood Decomposition Search (VNDS) and Skewed VNS (SVNS). For detailed descriptions see Hansen and Mladenović (2009). By combining strategies and parameter, we get hundreds of VNS search schemes, deterministic variants that only employ local search as well as stochastic variants that manage to escape from local optima. Those variants basically differ in their balance between exploring and exploiting search space. Additionally, the system supports MOO, whereas multiple objectives are combined hierarchically or weighted, or are equally combined in order to improve pareto fronts.

Six neighborhood structures create subspaces of the entire search space by encapsulating a certain set of operations used to modify the schedule. The implemented neighborhoods are defined as follows:

- Merge two batches: find two batches to merge to one of them (and new position).
- Split a batch: find a batch to split and insert the newly emerging batch at a new position.
- Swap two batches: find to batches and swap their positions.
- Move a batch: move a batch to another position.
- Swap two jobs: find two jobs out of different batches and swap them.
- Move a job: find a job and move it to another batch.

Since heuristic search procedures operate on given solutions, we need to provide an initial schedule as start solution for each problem instance. We use dispatching rules executed in a simulation system to generate initial schedules, which also provide reference objective measures for analyzing improvements gained by optimization. We use First-In-First-Out (FIFO) and Earliest Due Date (EDD) as simple dispatching rules. The dispatching interval is set to three minutes, which means that every three minutes the dispatching procedure is executed, which probably results in a new job/batch started. With respect to more sophisticated, optimizing dispatching heuristics (BATC, NACH) described in literature, we prefer to use simple dispatching rules providing, at least for the purpose of this paper. Because this papers' primary objective is neither to evaluate search method performance, nor to evaluate exact optimization potentials for certain model specifics; it is about MOO for PBMSPs.

For the optimization method we chose VND as appropriate search strategy to demonstrate the effects of MOO focused on two different objectives combined. In this regard, the only exceptional experiment is discussed in section 4 here we examine the correlation factors between a set of KPIs, where each is treated as single component objective. VND can be considered as a deterministic (best improvement) local search strategy operating on a limited set of neighborhoods specifically designed to solve the parallel batch machine problem. There are two reasons justifying our decision to apply a deterministic VNS variant, instead to choose a stochastic one, which has been proven to outperform deterministic approaches. On the one hand, the deterministic behavior of VND reduces the total number of experimental runs, since there is no need to run multiple replications that guarantee a certain level of statistical reliability, in contrast to stochastic VNS derivates. On the other hand, deterministic local search provides a better understanding of scheduling complexity with regard to the size of problem instances. Examination of measured computing times and/or number of search moves, combined with an analysis of improvements by optimization related to performance measures, makes clear whether local optima can be found for certain problems or not, in compliance to given computational deadlines. Additionally, in order to minimize the burden of analyzing averages and variances in experiments, at the same time increasing understandability and reliability of experimental results, we waived the use of stochastic search. In turn, deterministic search avoids analyzing variances resulting from stochastic effects, caused by multiple replications. We also like to point to the fact that potentials in optimization considerably depend on model characteristics. Based on extensive studies, we observed that there do exist non-negligible variances for simulation (optimization) results among a set of independent instances belonging to the same model type.

Table 2: Default VNS settings

VNS method	VND
Deadline [min]	10
Dispatching interval [min]	3
Dispatching rule	FIFO / EDD

4 CORRELATION OF OBJECTIVES

Performance measures are mutually interconnected, some are closely interlinked with each other, while others are not. We know about the existence of those relationships, but we do know very little about the magnitudes. In order to develop a better understanding, we designed an experiment to analyze the correlations between a selection of important KPIs, respectively performance measures or objectives. The objectives under study are throughput (THP) and makespan (Cmax), total cycle time (TCT), total weighted cycle time (TWCT), total queuing time (TQT), total weighted queuing time (TWQT), total tardiness (TT) and total weighted tardiness (TWT). The idea behind is to optimize for a specific objective, and then to calculate correlation factors after Pearson's method between the KPI as focused objective and the remaining KPIs. The resulting matrix of correlation factors provides exact values that indicate direction and strength of interdependencies between any combination of two KPIs. Figure 1 shows the results.

From experimental results, the calculated matrix of correlation factors according to Pearson, we derive a few observations that represent the strongest interconnections, given in decreasing order of their magnitudes. THP is an equivalent to Cmax, and both are, as expected, directly and strongly connected. The correlation factor equals to -1 and confirms by experiment that increasing THP as objective directly leads to decreased Cmax, and vice versa. We observed a very strong dependency (nearly 1) between QT and CT as well as between their non-weighted versions WQT and WCT. That makes sense, since both only differ in the portion of processing time. TWT and TT show high values around 0.8 for their correlation factors against each other, in both directions. Very similar, we observed correlation factors around 0.7 between TWCT and TCT, and between TQT and TWQT. Maximizing THP results into lower TCT and TQT, indicated by a correlation value around -0.5. We observed slightly lower impact on TWCT and TWQT, whereas the correlation factor is round about -0.4. Correspondingly, minimizing Cmax leads to lower TCT and TQT, and less directly to lower TWQT and TWCT.

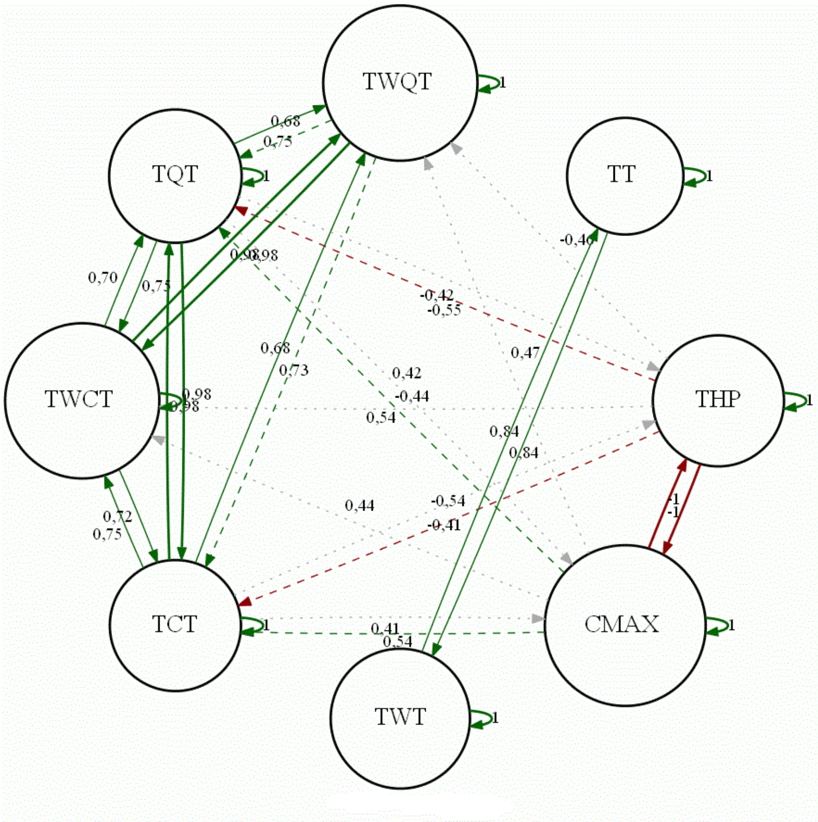

Figure 1: Correlation factors between performance measures

5 MULTIOBJECTIVE OPTIMIZATION

5.1 Pareto Objectives

For this experiment, we analyze bi-criteria objective functions, which combine two criteria equivalent to each other. The resulting bi-criteria pareto objective function considers solutions as improved if both measures show improvements, accepting that one of them remains unchanged. We consider three objective functions, each composed of two out of makespan (Cmax), total cycle time (TCT) and total tardiness (TT). Consequently, we examine the objective functions denoted with Cmax|TCT, Cmax|TT and TCT|TT. As initial solution and for reference, we use schedules obtained by simulating FIFO or EDD dispatching rules. The results shown in table 4 are given in relation to the initial solution.

Generally we see nearly no difference between the results obtained for the small model S(5|150) and the medium model M(10|300), whereas for the large model L(15|450) considerably less improvements become obvious. The reason is that the deterministic local search scheme (VND) does not reach a local optimum within the given deadline, for the larger model. Interestingly, compared to FIFO and EDD, the results show 6% improvement in Cmax on average. For TCT, we observe 3% improvement, whereas the lowest improvements show up when focusing on TT at the same time. This effect supports the assumption that TCT and TT contradict. The improvements in TT lie around 2% on average compared to EDD, and up to 14% compared to FIFO. When optimizing for TT, beginning with an initial solution obtained with FIFO, we observe decreasing improvements with increasing model size. This observation is also explainable with the conceptual association between computational complexity caused by model size and the computational deadline limiting the search procedure. Figure 2 shows the results displayed in a scatter plot, where each point represents a single model instance.

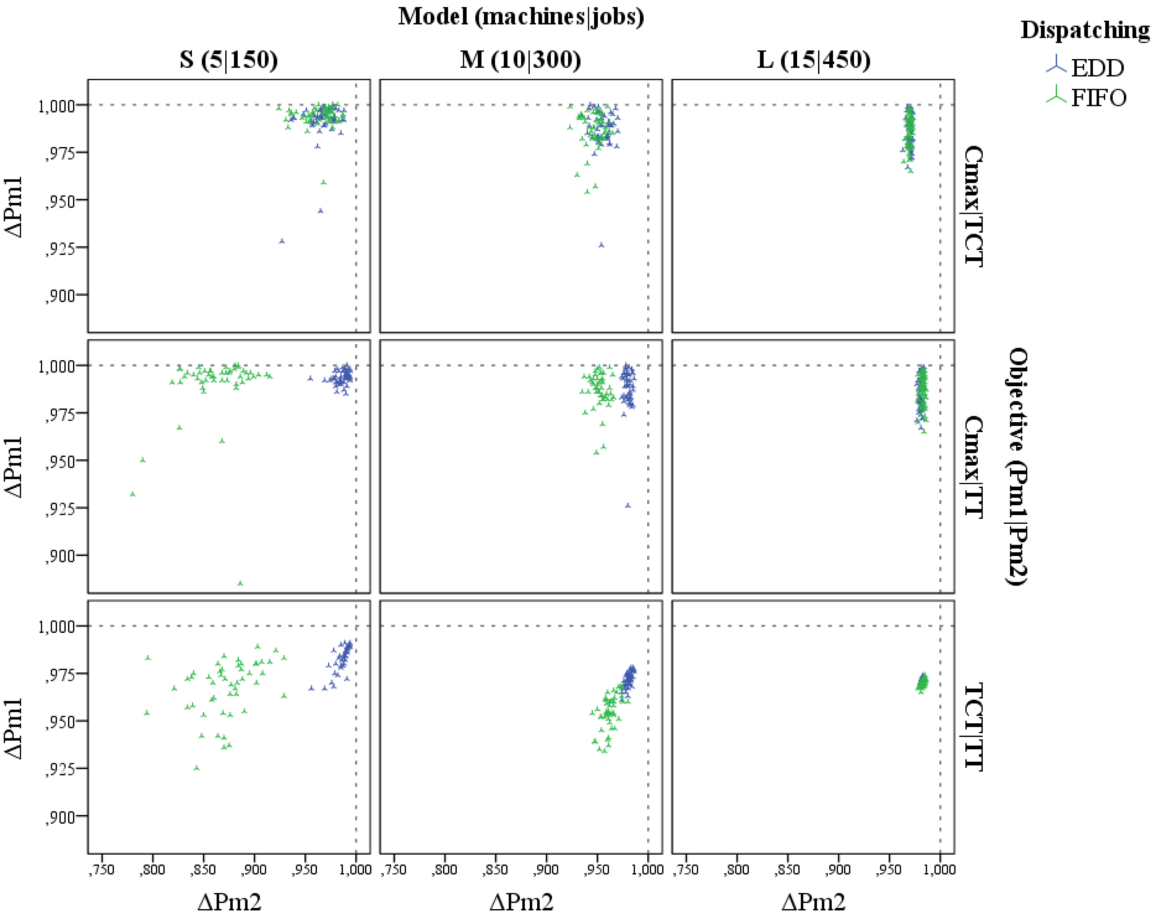

Figure 2: Computational results for bi-criteria pareto optimization

5.2 Weighted Objectives

Weighting objectives seems to be a proper strategy to combine multiple objectives. For the purpose of analyzing the effects of weights, we prepared the following experiment. Framed by this experiment, we examine a weighted multi-objective function, consisting of two measures, total cycle time (TCT) and total tardiness (TT). The weights range from zero to one, incremented in 0.2 steps, on the condition that the sum always equals to one. First we normalize both improvements in order to make them comparable. Then the normalized improvements are multiplied by their weights. The normalized and weighted improvements are summed up, and the result finally represents the decision value.

In accordance to the other experiments carried out, we see the worst results in improvements for the large models L(15|450), which leads us to the conclusion that the local search procedure probably does not reach a local optimum. In particular, in a line with the bi-criteria pareto optimization experiment mentioned before, we see identical results for the equally weighted case TCT(0,5)|TT(0,5). We observed maximum 5% improvement in CT compared to FIFO, and up to 2% improvement in TT compared to EDD. Generally we have to say, that for objective functions with weights higher than zero and lower than one, the results strongly resemble each other. Figure 3 shows the results displayed in a scatter plot, where each point represents a single model instance.

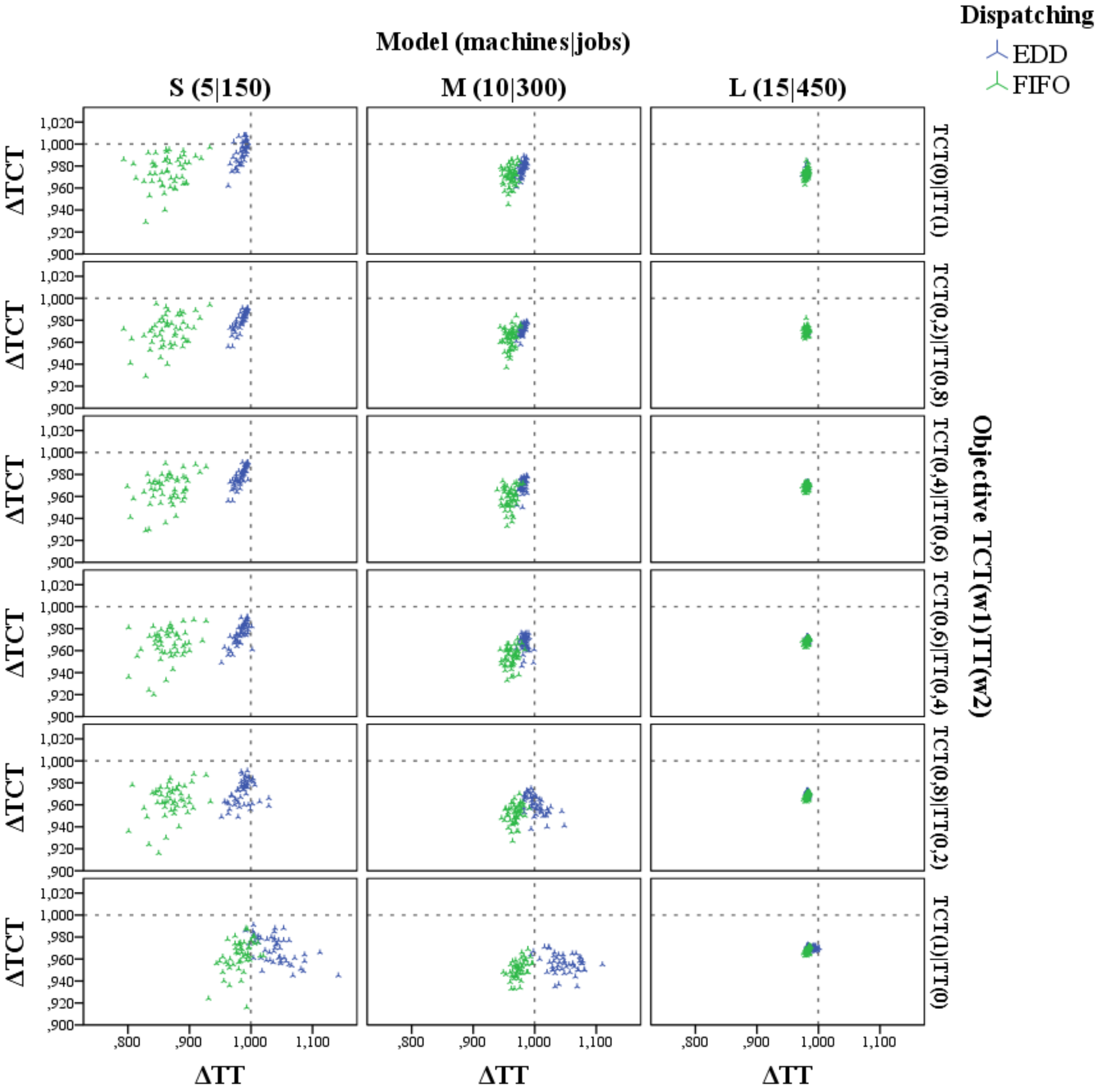

Figure 3: Computational results for optimization with weighted objectives

5.3 Hierarchical Objectives

It is most likely, that that there exists no hierarchy comprising the KPIs of interest as objectives, which would sufficiently represent the needs of shop floor control. But there exist special use cases justifying hierarchical structures for objective functions. Such a use case, for example, deals with critical constraints. Critical constraints may cause corrupted schedules, which contain unscheduled jobs, which are treated as undesired violations in our concept. Let us define the terms used in this context to provide a clear understanding. When solving PBMSPs, we may have to consider various constraints, and some of them are considered to be critical. Critical means in this context, that for certain problem instances, some (critical) constraints may cause corrupted schedules, here defined as schedules that contain a number of unscheduled jobs. This means, there might exist an amount of jobs left unscheduled due to one or more non-satisfied constraints, respectively critical constraints. During our studies, we identified two important critical constraints: job specific time bounds and minimum batch sizes.

When solving problem instances with critical constraints, we face the challenge of corrupted schedules, which contain a number of unscheduled jobs due to violated constraints. In case of an exact optimization method used to create the schedule, we have the proof that there exists no schedule that contains the entire amount of jobs fulfilling required conditions. In case of a heuristic scheduling procedure, we do only know that the method does not lead to a schedule that contains all jobs with respect to constraints. But both approaches, exact and heuristic, suffer from the same problem: the potential risk of a situation in which we do not have properly scheduled all the jobs.

Usually scheduling problems are defined in such a way that the entire amount of available jobs is required to be scheduled properly with respect to given constraints, without any exceptions. This requirement (to not allow any unscheduled jobs) combined with critical constraints consequently leads to the risk of invalid solutions. Put another way, we face the danger to be confronted with situations where no valid solution/schedule is available. That risk is not acceptable to shop floor control systems.

Strictly speaking, although it is highly desired to avoid violations in terms of unscheduled jobs, it is not necessarily required. Practitioners would prefer bad schedules instead of no schedules. This brings us to relax the requirement that requires all jobs to be properly scheduled. At that point we discover a new problem with respect to optimization. Imagine a scheduling problem with critical constraints and an objective that is meant to minimize a performance measure related flow time or on-time delivery. By optimization, no matter which kind of method applied, the resulting schedule would be most likely corrupted to a large extent. As a result, the corrupted schedule would contain a remarkable number of unscheduled jobs due to violated constraints, and a very low objective value as targeted by the objective function.

It comes down to the fact that it becomes unavoidable to somehow punish the optimizer for creating violations. Introducing a penalty system into the optimization process is an option. A similar approach to overcome that problem is to treat the number of violations, respectively the amount of jobs that remain unscheduled caused by constraints, as an objective. For this paper, we introduced a performance measure that equals to the number of job violations. During the optimization process, we treat any unscheduled job as undesired violation. Since avoiding violations is our top priority, we studied a combined objective function that hierarchically connects two objectives, first the number of violations and secondly an arbitrarily chosen objective measure.

5.3.1 Time Bound Constraints

Time bounds are common practice in wafer fabrication front-ends, driven by quality issues. We most often observe time bounds between operations in the wet chemistry area and furnace operations. Time bounds basically intend to limit unwelcome oxidation processes of the substrate, which happens while the job/lot is waiting for the next process under atmospheric conditions. Once a time bound is violated, the job needs separate treatment by experts, and this additional procedure is unwanted. When creating and improving schedules for the furnace area, it is top priority to avoid time bound violations: First of all because time bound violations increase the risk of quality problems, secondly because violated time bounds require separate treatment, which in particular ties up human resources. We like to refer to dispatching and scheduling approaches that also deal with time bound management (Ham and Fowler 2008; Klemmt et al. 2012; Mason et al. 2007).

For this experiment, we defined an objective function that contains two objectives in hierarchy, primarily the number or violations and secondarily total cycle time. We examined ten different time bound schemes, which represent uniformly distributed time bounds between zero and a multiple of the expected makespan (EM = 1800 min). The EM is calculated by use of the theoretical throughput of the system, mainly defined by the number of machines and their distribution of processing times, in relation to the number of jobs. FIFO dispatching serves as initial solution and reference for improvements. Table 6 shows experimental results, where V denotes the total number of unscheduled jobs in cause of violated time bounds, ΔV describes the improvement in terms of prevented time bound violations compared to FIFO dispatching, CT stands for the average cycle time, and ΔCT represents the improvement in average cycle time compared to FIFO. Figure 4 shows a boxplot visualizing the total number of time bound violations depending on model size and time bound schemes. Figure 5 shows the prevented time bound violations displayed in a scatter plot, where each point represents a single model instance.

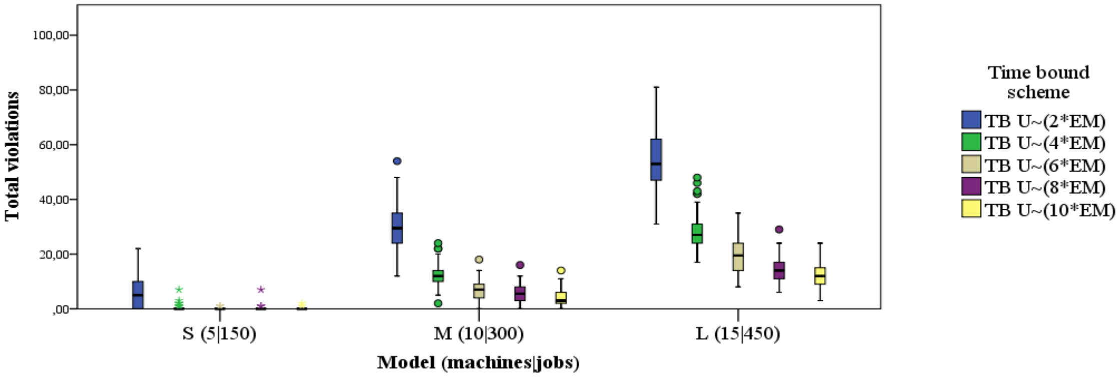

Figure 4: Computational results for hierarchical MOO with time bounds (1)

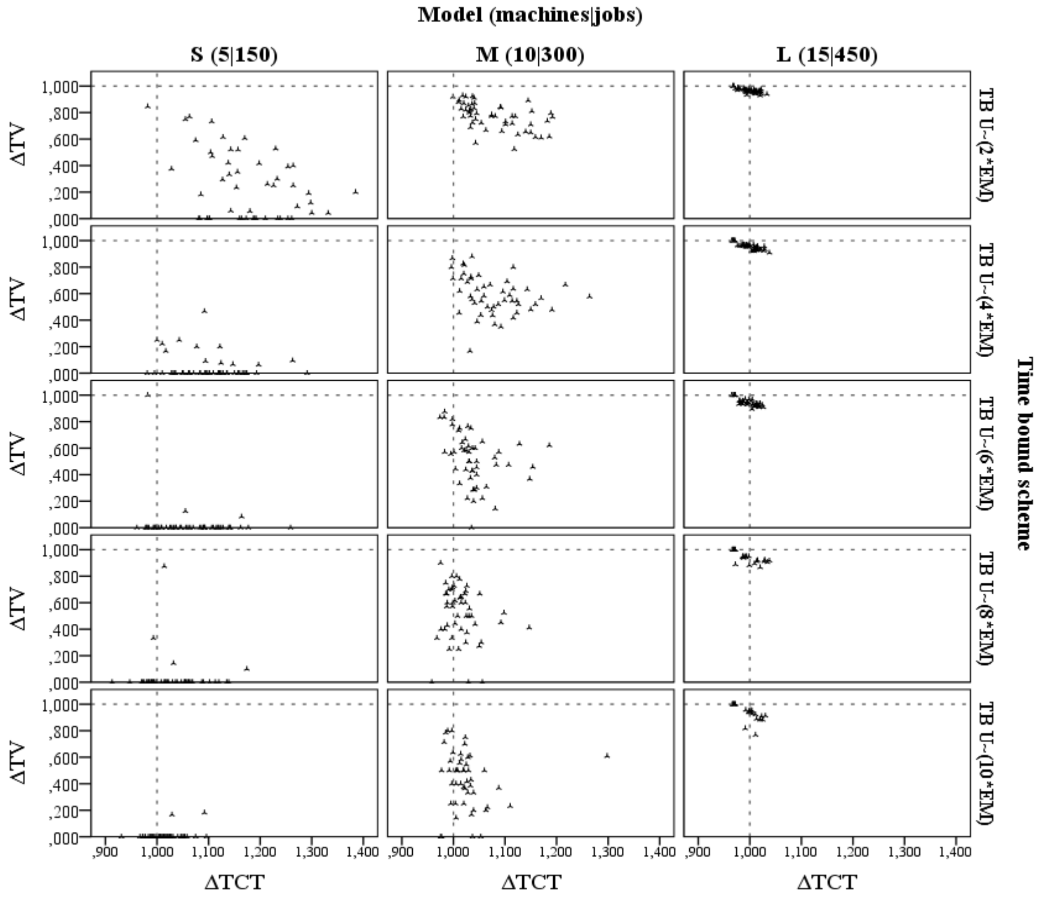

Figure 5: Computational results for hierarchical MOO with time bounds (2)

At this point we like to reflect the necessity of considering violations as a part of an objective function, and then present an idea for an improvement. First, we would like to make clear why violations need to be a part of an objective function for scheduling problems characterized by time bounds. Imagine a system with a single objective function aimed to minimize makespan, queuing times, cycle times or tardiness, and characterized by time bounds. An optimization system, regardless if it is of exact or heuristic type, would seek for solutions with increased number of violations that consequently results in lower objective values caused by a minimized number of scheduled jobs contributing to above mentioned performance measures. In this study, we managed violations by minimizing the total number of violations.

From our point of view, this will not entirely satisfy our needs. We propose, instead of counting the total number of violations as objective, a more sophisticated objective measure; total violation time. We define the total violation time is calculated as the sum of all violation timeouts. The idea behind is to distinguish between *soft* and *hard* violations, those jobs which slightly missed the deadline and those which are clearly behind. We like to illustrate the desired effect with a simple example. Assuming we have to choose between two solutions, the first shows two *soft* time bound violations and the second suffers from a single job with a *hard* time bound violation. From practitioners point of view, we clearly would choose the solution with two *soft* violations, because the more the time bound is exceeded, the more trouble with quality issues emerge. In an immediate sense, such a distinction becomes possible with total violation time as an objective, in contrast to the total number of violations. Consistently pursuing this idea, and considering the fact that we often deal with priorities in terms of weighted jobs, we come to the point that total weighted violation time should be an important part of an objective function.

5.3.2 Minimum Batch Size Constraints

Minimum batch size constraints represent a lower threshold for the number of wafers or jobs within a batch. They are usually set due to three reasons: a) as a requirement that guarantees process stability, b) as a lower threshold that prevents too small batches, driven by economic concerns c) as planning value that aims on factory performance according to theory of operation curve management (OCM).

In order to guarantee process stability, it is often required to keep the amount of substrates (wafers) inside the reactor constant, across all runs. Usually the reactor is completely filled at maximum capacity with wafers. In those cases in which the proposed batch counts less wafers than reactor's maximum, the reactor's wafer-slots left free are filled with non-productive wafer. In this case, those non-productive wafer do only have one intention: to guarantee the required heat capacity inside the reactor during the process. A certain amount of non-productive wafers are always stored inside the machine, for fast access without any considerable delay. If those are not sufficient, it is required to additionally request non-productive wafers, to be delivered as non-productive lots from somewhere in the fab.

In this case, a minimum batch size as a requirement that intends to prevent process runs with an unwanted ratio between productive and non-productive wafers, driven by economic concerns. Another reason could lie in an insufficient supply system for non-productive wafers. Summarized, for some reasons, it may necessarily be required to create batches that count a minimum number of wafers in sum provided by their batch partners.

However, during our studies we came to the point that a minimum batch size constraint is rarely a necessary constraint due to the facts mentioned above, rather it is used as a planning and control value. We strongly recommend to distinguish between minimum batch size thresholds considered as necessary constraints and planned batch sizes resulting from performance considerations. We made the experience that planned batch sizes are often declared as minimum batch sizes, non-beneficial for scheduling problems. It is true, that dispatching systems and even scheduling systems without any look-ahead information about incoming jobs from upstream, do need a planned batch size considered as a minimum threshold. Depending on the utilization level, there exists an optimal batch size value, which should not be exceeded by dispatching actions. In contrast to that, if a dispatching or scheduling system has information about future job arrivals, there is no need for planned batch sizes anymore.

At this point, we like to discuss a problematic situation that can be described as the minimum batch size dilemma for batch schedule optimization. We take it as a fact, that there exists a minimum batch size constraint, which is intended to promote or prevent a certain situation. Taking for granted that the entire amount of jobs need to be scheduled without exceptions, and taking minimum batch size constraints into account, there exist scheduling problems for which no valid solution exists. Such problem instances suffer from a number of unscheduled jobs, which neither fit into one of existing batches nor fulfill the minimum batch size constraint. In order to guarantee a valid solution at any time, and for any problem instance, it is necessary to relax the premise that requires finding a schedule without any unscheduled jobs. Consequently we are forced to allow jobs that remain unscheduled. And as a consequence of this, it is necessary to consider the amount of unscheduled as an additional objective. Because if not, the optimizing procedure would seek for solutions with an increased number of unscheduled jobs in order to reduce common objectives like cycle time or tardiness. And that is the point where the dilemma arises. When taking the number of unscheduled jobs as part of an objective function, the optimization system primarily strives for a low number of unscheduled jobs, and secondarily improves objectives we actually focus on.

6 CONCLUSIONS

Our experiments show that the complexity border for a deterministic local search procedure, which is allowed to proceed 10 minutes, lies beyond problem instances with 10 machines and 300 jobs. What we want to say is, that our implementation of VND most often reached an optimum within 10 minutes for the M(10|300) problem instance. As shown in previous studies, the presented experiments confirm that PBMS without dynamic job arrivals only shows very slightly better results compared to dispatching, unfortunately. This is due to the fact that batch scheduling benefits to a large extend rely on job arrival predictions. As a general statement, we can say that we gain up 5% improvement for common performance measures. We observed 6% improvement in makespan, 4% improvement in average cycle time, and 2% improvement in average tardiness. The existence of critical constraints, time bounds and/or minimum batch size, requires multi-objective functions appropriately considering situations with unscheduled jobs caused by violated constraints. In this case, hierarchical structures for multi-objective functions lend themselves.

With respect to minimum batch size, we state that batch scheduling usually needs minimum batch size constraints, if no look-ahead information is available. In order to guarantee a valid schedule at any time, minimum batch size constraints result in the permission to allow a number of unscheduled jobs. The permission for unscheduled jobs requires an objective function that considers unscheduled jobs to a large extent. Finally, unscheduled jobs as a part of an objective function potentially reduces performance measure improvements we are actually interested in. This is what we call the minimum batch size dilemma for scheduling.

We also emphasize that beside performance issues, an appropriate objective function carefully considering important needs, will determine the success of an operational scheduling system in the long run. For PBMSPs, this means that we have combined various measures into an objective function. Beside common local scope objectives usually focused on weighted cycle time and weighted tardiness, it is important to consider the effect of locally optimized decisions on downstream operations. That is of considerable importance in order to keep improvements remaining throughout the following operation(s). Since PBMSPs under optimal conditions rely on job arrival predictions, which always come with errors, it appears necessary to incorporate prediction errors into the decision process as a part of a multi-objective function.

ACKNOWLEDGEMENTS

We like to thank our colleges from Infineon Technologies for their support at any time. Special thanks go to Jens Doleschal and Manfred Benesch from Dresden University of Sciences who maintain server and systems essential for our experiments with all their expertise and commitment. This work was supported by Grant 13N11588 of the Federal Ministry of Education and Research Germany.

REFERENCES

Almeder, C., and L. Mönch. 2011. "Metaheuristics for scheduling jobs with incompatible families on parallel batching machines." *JORS* 62(12):2083–96.

Balasubramanian, H., L. Mönch, J.W. Fowler, and M.E. Pfund. 2004. "Genetic algorithm based scheduling of parallel batch machines with incompatible job families to minimize total weighted tardiness." *International Journal of Production Research* 42(8):1621–38.

Bixby, R., R. Burda, and D. Miller. 2006. "Short-Interval Detailed Production Scheduling in 300mm Semiconductor Manufacturing using Mixed Integer and Constraint Programming." In *The 17th Annual SEMI/IEEE Advanced Semiconductor Manufacturing Conference (ASMC 2006),* 148–54.

Cakici, E., S.J. Mason, J.W. Fowler, and H.N. Geismar. 2013. “Batch scheduling on parallel machines with dynamic job arrivals and incompatible job families.” *International Journal of Production Research* 51(8):2462–77.

Chiang, T.-C., H.-C. Cheng, and L.-C. Fu. 2010. “A memetic algorithm for minimizing total weighted tardiness on parallel batch machines with incompatible job families and dynamic job arrival.” *Computers & Operations Research* 37(12):2257–69.

Damodaran, P., and M.C. Vélez-Gallego. 2012. “A simulated annealing algorithm to minimize makespan of parallel batch processing machines with unequal job ready times.” *Expert Systems with Applications* 39(1):1451–8.

Damodaran, P., M.C. Vélez-Gallego, and J. Maya. 2011. “A GRASP approach for makespan minimization on parallel batch processing machines.” *Journal of Intelligent Manufacturing* 22(5):767–77.

Fowler, J., D. Phillips, and G. Hogg. 1992. “Real-time control of multiproduct bulk-service semiconductor manufacturing processes.” *Semiconductor Manufacturing, IEEE Transactions on* 5(2):158–63.

Fowler, J.W., G.L. Hogg, and D.T. PHILLIPS. 2000. “Control of multiproduct bulk server diffusion/oxidation processes. Part 2: multiple servers.” *IIE Transactions* 32(2):167–76.

Glassey, C., and W. Weng. 1991. “Dynamic batching heuristic for simultaneous processing.” *Semiconductor Manufacturing, IEEE Transactions on* 4(2):77–82.

Graham, R., E. Lawler, J. Lenstra, and A. Rinnooy Kan. 1979. “Optimization and Approximation in Deterministic Sequencing and Scheduling: a Survey.” In *Discrete Optimization II Proceedings of the Advanced Research Institute on Discrete Optimization and*. Annals of Discrete Mathematics. Elsevier Science.

Gupta, A.K., and A.I. Sivakumar. 2006. “Optimization of due-date objectives in scheduling semiconductor batch manufacturing.” *International Journal of Machine Tools and Manufacture* 46(12–13):1671–9.

Habenicht, I., and L. Mönch. 2003. “Simulation-based assessment of batching heuristics in semiconductor manufacturing.” In *Proceedings of the 35th Winter Simulation Conference*, edited by S.E. Chick, P.J. Sanchez, D.M. Ferrin and D.J. Morrice, 1338–45. WSC '03. ACM.

Hansen, P., N. Mladenovi´c, J. Brimberg, and J.A.M. P´erez. 2009. “Variable Neighborhood Search.” In *Handbook of metaheuristics*, edited by M. Gendreau, and J.-Y. Potvin. International Series in Operations Research & Management Science. New York. Springer.

Jula, P., and R.C. Leachman. 2010. “Coordinated Multistage Scheduling of Parallel Batch-Processing Machines Under Multiresource Constraints.” *Operations Research* 58(4-Part-1):933–47.

Kashan, A.H., B. Karimi, and M. Jenabi. 2008. “A hybrid genetic heuristic for scheduling parallel batch processing machines with arbitrary job sizes.” *Computers & Operations Research* 35(4):1084–98.

Kim, Y.-D., B.-J. Joo, and S.-Y. Choi. 2010. “Scheduling Wafer Lots on Diffusion Machines in a Semiconductor Wafer Fabrication Facility.” *Semiconductor Manufacturing, IEEE Transactions on* 23(2):246–54.

Klemmt, A., S. Horn, G. Weigert, and T. Hielscher. 2008. “Simulations-based and solver-based optimization approaches for batch processes in semiconductor manufacturing.” In *Simulation Conference, 2008. WSC 2008. Winter,* 2041–9.

Klemmt, A., and L. Mönch. 2012. “Scheduling jobs with time constraints between consecutive process steps in semiconductor manufacturing.” In *Proceedings of the Winter Simulation Conference,* 194:1□194:10. WSC ’12. Winter Simulation Conference.

Klemmt, A., G. Weigert, C. Almeder, and L. Mönch. 2009. “A comparison of MIP-based decomposition techniques and VNS approaches for batch scheduling problems.” In *Proceedings of the Winter Simulation Conference*, edited by M.D. Rossetti, R.R. Hill, B. Johansson, A. Dunkin and R.G. Ingalls, 1686–94. WSC '09.

Klemmt, A., G. Weigert, and S. Werner. 2011. “Optimisation approaches for batch scheduling in semiconductor manufacturing.” *European Journal of Industrial Engineering* 5(3):338–59.

Li, L., F. Qiao, and Q. Wu. 2008. “ACO-Based Scheduling of Parallel Batch Processing Machines with Incompatible Job Families to Minimize Total Weighted Tardiness.” In *Ant Colony Optimization and Swarm Intelligence*, edited by M. Dorigo, M. Birattari, C. Blum, M. Clerc, T. Stützle and A. Winfield, 219–26. Lecture Notes in Computer Science. Springer Berlin / Heidelberg.

Malve, S., and R. Uzsoy. 2007. “A genetic algorithm for minimizing maximum lateness on parallel identical batch processing machines with dynamic job arrivals and incompatible job families.” *Computers & Operations Research* 34(10):3016–28.

Mason, S.J., M. Kurz, M.E. Pfund, J.W. Fowler, and L. Pohl. 2007. “Multi-Objective Semiconductor Manufacturing Scheduling: A Random Keys Implementation of NSGA-II.” In *Proceedings of the IEEE Symposium on Computational Intelligence in Scheduling (SCIS ’07),* 159–64.

Mathirajan, M., and A.I. Sivakumar. 2006. “A literature review, classification and simple meta-analysis on scheduling of batch processors in semiconductor.” *The International Journal of Advanced Manufacturing Technology* 29(9):990–1001.

Mladenović, N., and P. Hansen. 1997. “Variable neighborhood search.” *Computers & Operations Research* 24(11):1097□1100.

Mönch, L., and C. Almeder. 2009. “Ant Colony Optimization for Scheduling Jobs with Incompatible Families on Parallel Batch Machines.” In *Proceedings of the 4th Multidisciplinary International Scheduling Conference (MISTA),* 106–14.

Mönch, L., H. Balasubramanian, J.W. Fowler, and M.E. Pfund. 2005. “Heuristic scheduling of jobs on parallel batch machines with incompatible job families and unequal ready times.” *Computers & Operations Research* 32(11):2731–50.

Mönch, L., J.W. Fowler, S. Dauzère-Pérès, S.J. Mason, and O. Rose. 2011. “A survey of problems, solution techniques, and future challenges in scheduling semiconductor manufacturing operations.” *Journal of Scheduling* 14(6):583–99.

Raghavan, N.S., and M. Venkataramana. 2006. “Scheduling Parallel Batch Processors with Incompatible Job Families Using Ant Colony Optimization.” In *Proceedings of the International Conference on Automation Science and Engineering,* 507–12. Shanghai.

Reichelt, D., and L. Mönch. 2006. “Multiobjective Scheduling of Jobs with Incompatible Families on Parallel Batch Machines.” In *Evolutionary Computation in Combinatorial Optimization*, edited by J. Gottlieb, and G.R. Raidl, 209–21. Lecture Notes in Computer Science. Springer Berlin / Heidelberg.

Sha, D., S.-Y. Hsu, and X. Lai. 2007. “Design of due-date oriented look-ahead batching rule in wafer fabrication.” *The International Journal of Advanced Manufacturing Technology* 35(5-6):596–609.

SOLOMON, L., J.W. Fowler, M. Pfund, and P.H. JENSEN. 2002. “THE INCLUSION OF FUTURE ARRIVALS AND DOWNSTREAM SETUPS INTO WAFER FABRICATION BATCH PROCESSING DECISIONS.” *Journal of Electronics Manufacturing* 11(02):149–59.

van der Zee, D.J. 2007. “Dynamic scheduling of batch-processing machines with non-identical product sizes.” *International Journal of Production Research* 45(10):2327–49.

Wang, H.-M., and F.-D. Chou. 2010. “Solving the parallel batch-processing machines with different release times, job sizes, and capacity limits by metaheuristics.” *Expert Systems with Applications* 37(2):1510–21.

Yugma, C., S. DAUZERE-PERES, A. Derreumaux, and O. Sibille. 2008. “A Batch Optimization Solver for diffusion area scheduling in semiconductor manufacturing.” In *Advanced Semiconductor Manufacturing Conference, 2008. ASMC 2008. IEEE/SEMI,* 327–32.

Yurtsever, T., E. Kutanoglu, and J. Johns. 2009. “Heuristic based scheduling system for diffusion in semiconductor manufacturing.” In *Proceedings of the Winter Simulation Conference*, edited by M.D. Rossetti, R.R. Hill, B. Johansson, A. Dunkin and R.G. Ingalls, 1677–85. WSC '09.

AUTHOR BIOGRAPHIES

ROBERT KOHN is a member of the scientific staff of Prof. Dr. Oliver Rose at the Chair of Modelling and Simulation at the University of the German Federal Armed Forces Munich, Germany. His focus is on simulation based scheduling in semiconductor manufacturing. He received his M.S. degree in computer science from University of Applied Sciences Stralsund, Germany. His e-mail address is robert.kohn@unibw.de.

OLIVER ROSE holds the Chair for Modelling and Simulation at the University of the German Federal Armed Forces Munich, Germany. He received an M.S. degree in applied mathematics and a Ph.D. degree in computer science from Würzburg University, Germany. His research focuses on the operational modelling, analysis and material flow control of complex manufacturing facilities, in particular, semiconductor factories. He is a member of IEEE, INFORMS Simulation Society, ASIM, and GI. His e-mail is oliver.rose@unibw.de.

CHRISTOPH LAROQUE studied business computing at the University of Paderborn, Germany. From 2003 to 2007 he has been a PhD student at the graduate school of dynamic intelligent systems and, in 2007, received his PhD for his work on multi-user simulation. He is team leader of the “simulation & digital factory” at the chair of Business Computing, esp. CIM. He is mainly interested in the simulation-based decision support for operational production and logistic processes. His email address is laroque@upb.de.