*Proceedings of the 2009 Winter Simulation Conference*
*M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin and R. G. Ingalls, eds.*

# A COMPARISON OF MIP-BASED DECOMPOSITION TECHNIQUES AND VNS APPROACHES FOR BATCH SCHEDULING PROBLEMS

Andreas Klemmt
Gerald Weigert

Electronics Packaging Laboratory

Helmholtzstraße 18
Technische Universität Dresden
01062 Dresden, GERMANY

Christian Almeder

Dept. for Production and Logistics

Brünnertraße 72
University of Vienna
1210 Vienna, AUSTRIA

Lars Mönch

Dept. of Mathematics and Computer Science

Universitätsstr.1
University of Hagen
58097 Hagen, GERMANY

## ABSTRACT

This research is motivated by a scheduling problem found in the diffusion and oxidation areas of semiconductor wafer fabrication facilities. With respect to some practical motivated process constraints, like equipment dedication and unequal batch-sizes, we model the problem as unrelated parallel batch machines problem with incompatible job families and unequal ready times of the jobs. Our objective is to minimize the total weighted tardiness (TWT) of the jobs. Given that the problem is NP-hard, we propose two different solution approaches. The first approach works with a time window-based mixed integer programming (MIP) decomposition. The second approach uses a variable neighbourhood search (VNS). Using randomly generated test instances, we show that the proposed algorithms outperform common dispatching rules that cannot deal with the given constraints effectively.

## 1 INTRODUCTION

The planning and optimization of semiconductor manufacturing is a very complex task. Especially, in the field of wafer processing in the front-end a lot of different processing steps have to be performed. These steps are, for example, typical batch tool operations like oven processes and wet-etch processes, or typical cluster tool operations like dry-etch, implant or lithography processes having complex setup strategies. A batching machine allows that several jobs can be processed at the same time. Because of several specific constraints and dependencies, it is challenging to schedule the jobs. Moreover, meeting customer due dates is one of the important manufacturing objectives. Because of the complex nature of the process, the customer due date is set for each operation as operation due date (ODD) (cf. Rose 2003). The task to be solved consists in meeting these due dates for each job within each work center as good as possible with respect to different job priorities.

In this paper, we focus on diffusion and oxidation operations which are performed on batch machines, i.e. furnaces. Because of the long processing times of batch tools an effective scheduling of the furnace operation has an huge impact on global manufacturing objectives (cf. Mehta and Uzsoy 1998). Though several jobs can be processed simultaneously on these batch processing machines, the process restrictions require that only jobs belonging to the same family can be processed together at the same time. Further process restrictions specify that not all families can be processed by every machine (equipment dedication) and that machines can have different capacities. These equipment dedication constraints mainly represent different equipment qualifications. Especially for research and development wafer fabs with high product mixtures and constantly new developments, these constraints make the manufacturing control more sophisticated. In addition, the jobs to be processed have different priorities/weights, due dates, and ready times. So, in the case of unequal ready times, it is sometimes advantageous to form a non-full batch while in other situations it is a better strategy to wait for future job arrivals to increase the fullness of the batch.

In this research, we model the described problem as unrelated parallel batch machine scheduling problem with incompatible job families and unequal ready times of the jobs. The performance measure is TWT. Using the α | β | γ notation of Graham et al. (1979) our scheduling problem can be represented as $R_m \mid r_i,p\text{-}batch,incompatible \mid \Sigma w_i T_i$. This problem is NP-hard by reduction to $1 \| \Sigma w_i T_i$ (cf. Lawler 1977). Therefore, this research proposes two heuristic solution strategies that lead to good results in comparison to common dispatching approaches. The suggested heuristics are investigated with respect to dif-

978-1-4244-5771-7/09/$26.00 ©2009 IEEE

ferent application scenarios. We are especially interested in answering the question how the optimization potential depends on information for future job arrivals. This question is related to setting appropriate forecast horizons.

The paper is organized as follows. Related work concerning scheduling of batch machines is discussed in Section 2. We define the problem and provide some assumptions in Section 3. In Section 4, we explain several variants of dispatching approaches, the MIP decomposition scheme as well as the VNS approach. The test data generation scheme is introduced in Section 5. We present the results from our computational experiments in Section 6. Finally, Section 7 contains our conclusions and plans for future research.

## 2 PREVIOUS RELATED WORK

Approaches for scheduling batch processes are a matter of particular interest and are also recorded in several publications. The solution methodologies can be grouped into optimum approaches and heuristic approaches including simulation techniques. A recent collection of these approachess is provided by the survey of Mathirajan and Sivakumar (2003). Another review and classification concerning machine environments, constraints, and performance measures can be found in (Perez, Fowler, and Carlyle 2005). Mehta and Uzsoy (1998) describe dynamic programming and heuristic solution procedures for the single machine batch problem with incompatible job families and the total tardiness objective. This work was extended by Perez, Fowler, and Carlyle (2005) to the performance measure TWT. Balasubramanian et al. (2004) and Mönch et al. (2005) extend the existing solution methodologies from the single machine to the case of identical parallel batch machines. In (Mönch et al. 2005), dynamic job arrivals are taken into account which are essential for considering real-world problems. Fowler, Hogg, and Phillips (2000) illustrate the need to use information regarding future arrivals within different control strategies for batching tools.

Most of the decomposition approaches for batch scheduling discussed in the literature are divide the overall scheduling problem into the phases batching the jobs and sequencing the batches on the machines (cf. Mehta and Uzsoy 1998, Balasubramanian et al. 2004, Mönch et al. 2005, and Raghavan and Venkataramana 2006 amongst others). Raghavan, and Venkataramana (2006) uses ant colony optimization to tackle the batching problem. In (Balasubramanian et al. 2004) and (Mönch et al. 2005), the decomposition is coupled with a genetic algorithm. Almeder and Mönch (2009) further enhance this approach by using VNS instead of the genetic algorithm. This especially leads to significantly reduced computation times. Unfortunately, most of the decomposition approaches cannot be used for machines with different batch sizes and dedications. The VNS approach of Almeder und Mönch (2009) is further extended in this paper to tackle these specific process constraints.

Klemmt et al. (2008) describe MIP models as well as a simulation-based optimization approach which was performed together with a genetic algorithm for unrelated parallel batch problems. Here, the objective was to minimize makespan and queuing time. In (Klemmt, Lange, and Weigert 2009) these methods where adapted to the objective TWT and tested using several benchmark instances. An online scheduling application using MIP and constraint programming techniques for the optimization of furnace operations in a 300mm wafer fab is described by Bixby, Burda, and Miller (2006). Unfortunately, further modeling details of this approach are not presented.

## 3 PROBLEM ASSUMPTIONS

In the remainder of this paper, different solution approaches for the unrelated batch scheduling problem are described. Related to Balasubramanian et al. (2004) the following assumptions are made.

1. The jobs fall into different incompatible families that cannot be processed together.
2. There exist $f$ job families. $F := \{1,\ldots,f\}$ represents the set of all families.
3. There are $n$ jobs to schedule. $J := \{1,\ldots,n\}$ represents the set of all jobs.
4. There are $m$ unrelated machines in parallel. $M := \{1,\ldots,m\}$ represents the set of all machines.
5. Machine preemption is not allowed.
6. The family of job $i$ is represented as $f_i$.
7. The priority weight for job $i$ is represented as $w_i$.
8. The (operation) due date of job $i$ is represented as $d_i$.
9. The processing time $p_i$ of job $i$ is assumed to be equal on all machines. Also it is equal for all jobs of the same family.
10. The ready time of job $i$ is represented as $r_i$. All jobs forming the batch have to be ready before the batch start.
11. The batch processing machine capacity of machine $k \in M$ measured in jobs is represented as $B_k$.
12. The dedication of machine $k$, i.e., the set of allowed families on machine $k$, is represented as a subset $D_k \subseteq F$.
13. The completion time of job $i$ is denoted by $C_i$.
14. The weighted tardiness of job $i$ is represented as $w_iT_i= w_i\,(C_i - d_i)^+$, where $x^+ := \max(x,0)$.

## 4 SOLUTION METHODOLOGIES

In this paper three, different solution strategies for the batch scheduling problems are investigated. The solution strategies are list-based scheduling approaches based on dispatching rules, MIP-based decomposition techniques, and a VNS scheme. All of them will be described in detail in separate sub sections.

### 4.1 Dispatching – BATC-II

The Apparent Tardiness Cost (ATC) dispatching rule suggested by Vepsalainen and Morton (1987) is important for obtaining schedules with small TWT. Some modifications of this rule (called BATC) for batch processes exists in literature (cf. Mönch et al. 2005 for more details on this batching rule). In its simplest version, the BATC rule ignores the ready times of the jobs. So, every time a machine becomes available a batch is started if at least one job is available. A more complex version of the rule, denoted as BATC-II, takes the ready times of the jobs into account. As a result, this rule is able to delay batch starts and to form more full batches if this is possible. So, for batch problems with ready times, this rule reaches best dispatch performance results (cf. Mönch et al. 2005 or Klemmt, Lange, and Weigert 2009).

In this paper, a further extension of this rule, denoted as BATC-IIi (insert), is discussed. Here, if a batch is delayed, the length of the resulting delay interval is checked whether there is an fitting batch that is available for processing or not. If several fitting batches are available, the batch with the highest BATC-II index is inserted. Within this research, the BATC, BATC-II, and BATC-IIi rules are used as reference heuristics to assess the performance of a MIP-based decomposition scheme and a VNS approach.

### 4.2 MIP Approach

The suggested MIP is similar to the MIP suggested by Klemmt et al. (2008). Some modifications concerning constraints and objective become necessary. The parameter $b_k$ restricts the number of batches on each machine $k \in M$. The subset $M_i \subseteq M$ includes all machines which are allowed to process job $i$. The subset $J_k \subseteq J$ contains all jobs which are allowed for processing on machine $k$. Note that both subsets can be directly derived from the set $D_k$. $K$ is a very large positive number. The following decision variables have to be defined:

$s_{jk} \in \mathbb{R}_+$ starting time of batch $j$ on machine $k$; $(k \in M;\ j = 1,\ldots,b_k)$,
$x_{ijk} \in \{0,1\}$ job $i$ is scheduled in batch $j$ on machine $k$, 0 otherwise; $(i \in J;\ k \in M_i;\ j = 1,\ldots,b_k)$,
$y_{jkl} \in \{0,1\}$ family $l$ is scheduled in batch $j$ on machine $k$, 0 otherwise; $(k \in M;\ j = 1,\ldots,b_k;\ l \in D_k \subseteq F)$,
$C_i \in \mathbb{R}_+$ completion time of job $i$,
$T_i \in \mathbb{R}_+$ tardiness of job $i$.

The remaining parameters have been already introduced in Section 3. In a next step, the batch process MIP model has to be formulated.

*Batch process MIP model:*

$$\sum_{i\in J} w_i T_i \to \min \qquad \text{subject to} \tag{1}$$

$$\sum\nolimits_{k\in M_i} \sum\nolimits_{j=1}^{b_k} x_{ijk} = 1 \qquad i \in J \tag{2}$$

$$\sum\nolimits_{i\in J_k} x_{ijk} \le B_k \qquad k \in M;\, j = 1,\ldots,b_k \tag{3}$$

$$\sum\nolimits_{l\in D_k} y_{jkl} = 1 \qquad k \in M;\, j = 1,\ldots,b_k \tag{4}$$

$$y_{jkl} - x_{ijk} \ge 0 \qquad i \in J; l = f_i; k \in M_i;\, j = 1,\ldots,b_k \tag{5}$$

$$x_{ijk} r_i \le s_{jk} \qquad i \in J; k \in M_i;\, j = 1,\ldots,b_k \tag{6}$$

$$s_{jk} + p_i x_{ijk} \le s_{j+1,k} \qquad i \in J; k \in M_i;\, j = 1,\ldots,b_k - 1 \tag{7}$$

$$K\left(1 - x_{ijk}\right) + C_i \ge s_{jk} + p_i \qquad i \in J; k \in M_i;\, j = 1,\ldots,b_k \tag{8}$$

$$C_i - T_i \le d_i \qquad i \in J\ . \tag{9}$$

Here, the constraints set (9) restricts the objective function (1). Constraints set (2) ensures that each job can only be processed in one batch on one machine. The maximum batch size is restricted by (3). Equations (4) and (5) make sure that only jobs of the same family can form a batch. Constraints set (6) guarantees that a job can only be processed after it is ready. Equation (7) forces that a batch can be started only after the previous one is finished. Constraints set (8) restricts the completion time of each job.

Because of the NP-hardness of the researched problem, it has been computationally shown that even state of the art MIP solver can only handle relatively small problems for the TWT objective (cf. Klemmt, Lange, and Weigert 2009). Therefore, in this section several decomposition approaches are described.

*Time window decomposition:* Every time $t$ a machine becomes available, a MIP is formulated and solved for the sub-problem. Only jobs with ready times smaller than $t + \Delta t$ are considered. We use the set $\tilde{J} := \{ i \in J | r_i < t + \Delta t \}$ instead of $J$. As a result, we obtain sub-problems of smaller size. The timestamp $t' \leq t + \Delta t$ of the next event is calculated (i.e., a machine becomes available, a new job is arrived, etc.). All jobs that have been completed until $t'$ are further disregarded. All batches which are currently, i.e., at time $t'$, processed define new time periods $R_k$ (denoted as $R_k'$ in Figure 2) where jobs are processed on the machines. To consider these periods of time in each MIP sub-problem a further constraint set (10) has to be defined:

$$R_k \leq s_{1,k} \quad k \in M \; . \tag{10}$$

The jobs that form these batches are also disregarded further. All jobs which have not been started until $t'$ as well as all jobs which enter oven operation in the time period $t' + \Delta t$ form the new job pool. The MIP which represents the new sub-problem is solved. The process is repeated until all jobs are scheduled. For the cyclic solving of the MIP sub-problems (1) is replaced by

$$K \cdot \sum\nolimits_{i \in \tilde{J}} w_i T_i + \sum\nolimits_{i \in \tilde{J}} C_i \rightarrow \min \; . \tag{11}$$

The advantage of (11) is that jobs without tardiness cost are additionally optimized with respect to their completion time. This leads to minimal unavailability times $R_k$' for the machines in the subsequent MIP sub-problem. Figure 1 illustrates the MIP time window decomposition. Note that the number of jobs involved in a sub-problem is usually much smaller than $n$.

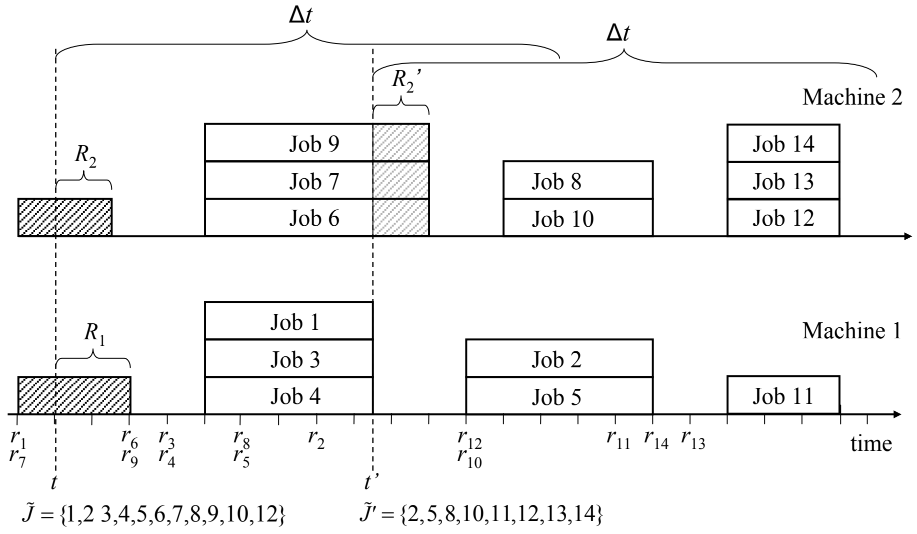

Figure 1: Illustration of time window decomposition for two batch tools

*Machine pool decomposition:* The MIP sub-problem complexity can be further reduced. Therefore, only machines (in a time window) with availability times lower than a time horizon $t + \Delta t_2$ are regarded. So, $\tilde{M} := \{ k \in M \mid R_k < t + \Delta t_2 \}$ is used instead of $M$. This decomposition is effective for highly distributed processing times (like found in the benchmark test instances in Section 5) with $\Delta t \leq \Delta t_2 \leq \max_i (p_i)$.

*Job list reduction:* As shown in Figure 1, the time window approach only fixes the jobs of the next batch on each machine (first batch in the time window). All following batches are recalculated in a later time window. So, if the number of jobs in a time window is still too large, the current job list is further reduced. Therefore, for all jobs the ATC priority index (cf. Mönch et al. 2005) is calculated. Each job *i* has a priority index $I_i$. The jobs are sorted concerning this index. Then only the first $\bar{n}$ jobs are taken into account. That means that $\tilde{J} := \{ i \in J \,|\, r_i < t + \Delta t, I_i < \bar{n} \}$ is used in the time window. Based on the results found in (Klemmt, Lange, and Weigert 2009) $\bar{n} \in [20, 30]$ has to be chosen.

*Strong upper bounds:* To accelerate the optimization process, for every MIP sub-problem an initial feasible solution is calculated using the BATC rule. So the optimization is started already from an good initial point in the search space with strong upper bounds for the objective function.

### 4.3 VNS Approach

Variable neighborhood search is a local-search based metaheuristic (cf. Hansen and Mladenovic 2001). The main idea is to enrich a simple local search method in order to enable it escaping from local optima. If the local search gets stuck in a local optimum, it is restarted from a randomly chosen neighbor of a large neighborhood (larger than those used during the local search step). This restarting step is called shaking and is performed with different neighborhoods of increasing sizes. A basic VNS consists of the following steps:

***Initialization*:**

```
1.    Define $K$ different neighborhoods $N_k$
2.    Generate an initial solution $x$
3.    Set $k$=1
```

***Algorithm*:**

```
4.    Repeat until stopping criterion is met
a.    Shaking: choose randomly $x' \in N_k(x)$
b.    Local search: Improve $x'$ by a local search method
c.    Accept? If $x'$ is better than $x$, then $x=x'$ and $k$=1 otherwise $k=(k \bmod K)+1$.
```

The VNS algorithm designed for the batching problem operates on the final solution representation, i.e. each job is assigned to a batch and each batch is assigned to a certain position on a machine. Its design is similar to the VNS for the batching problem of parallel machines with jobs with zero release times (cf. Almeder and Mönch 2009). The proposed local search method used for the VNS consists of two different phases. During the first phase the workload of the machines is balanced. If the last batch of the machine with the maximum completion time starts later than the completion time of another machine which is suitable for that batch, the batch is moved to that machine. This step is repeated until no batch can be moved. During the second phase of the local search the following three steps are applied iteratively:

1. job insert (remove a job from a batch and insert it into another batch),
2. job swap (swap two jobs of different batches),
3. batch swap (swap two batches of the same family).

Therefore the design of the neighborhood structure for the shaking step of the VNS considers manipulations of whole batches across different machines. We define five classes of neighborhoods:

- ***splitBatch(l)***: Select randomly a batch and split it into two batches. One remains on that position, the other one is inserted into the sequence on a different machine. Repeat this step *l* times.
- ***moveBatch(l):*** Select randomly a batch from a machine and remove it. Insert it on a random position on a randomly selected machine. Repeat this step *l* times.
- ***moveSeq(l):*** Randomly select a position on a machine. Remove a sequence of at most *l* (*l* or all remaining batches) and insert either this partial sequence or the reversed one on another machine at a randomly selected position.
- ***swapBatch(l)***: Select randomly two batches from different machines which are both capable of handling those batches and exchange their positions. Repeat this step *l* times.
- ***swapSeq(l):*** Randomly select two positions on different machines and exchange the batch sequences starting from that position of at most length *l* (*l* or all remaining batches).

Note that in each of these neighborhoods only splits, moves, and swaps are considered which result in a feasible solution, i.e. restrictions of job families and batch sizes are considered. The neighborhoods are applied in the order given above.

## 5 TEST DATA GENERATION

For testing our solution methodologies, test instances are generated randomly. A benchmark scheme is used which is investigated in different variants by many researchers (see Mehta and Uzsoy 1998, Mönch et al. 2005 amongst others). In Table 1, this benchmark is extended to the researched scheduling problem. Some modifications have to be made to allow the modeling of some real-world conditions (cf. also Klemmt, Lange, and Weigert 2009). To represent the high product mixture, the number of families is between four and eight. Furthermore, the utilization $u$ of the parallel machine group is varied from 0.7 to 0.9. Here, an average batch size coefficient $\bar{b}$ = 0.75 is assumed to represent capacity limitations resulting from product mixture and equipment dedication. The expected makespan of each test instance can be computed as follows:

$$makespan = \frac{n \cdot \bar{p}}{\bar{b} \cdot u \cdot \sum_{k=1}^{m} B_k}, \tag{12}$$

where $\bar{p}$ denote the average processing time of the jobs.

The ready times of each job are uniformly distributed in [0, *makespan*]. It is assumed that all jobs are controlled by ODD in pre-operations, the due-date spreads around the ready date. Here, a deviation of five ($v = 5$) average process times is set. The weight of the jobs is not uniformly distributed to represent the different priority items. The batch size is not variable but fixed at different capacities for the machines. Also, additional equipment dedication constraints are considered. It allows the modeling of different machine qualifications. The number of allowed machines per families spreads from one to five. Family 1 can be regarded as a standard process which is executable on each machine (see Table 1). In contrast, for family 2 only one qualified machine exists because of new products or engineering tasks. Altogether 162 different test instances (every possible factor-combination is generated two times) with a number of jobs ranges from 80 to 320 are considered within the computational experiments. The used design is summarized in Table 1.

Table 1: Design of experiments

| Factors | Values used | Total values |
|---|---|---|
| Number of machines | 3, 4, 5 | 3 |
| Number of jobs/family | 20, 30, 40 | 3 |
| Number of families | 4, 6, 8 | 3 |
| Utilization | 0.7, 0.8, 0.9 | 4 |
| Family processing time (probability) | 2 (20%), 4 (20%), 10 (30%), 16 (20%), 20 (10%) | 1 |
| Weight per job (probability) | 0.3 (75%), 0.6 (20%), 1 (5%) | 1 |
| Release date | $r_i$~ Uniform (0, *makespan*) | 1 |
| Due date | $d_i$~ Uniform ( $r_i - (v-1)\cdot\bar{p}$ , $r_i + (v+1)\cdot\bar{p}$ ) | 1 |
| Batch size | $B_1 = 3, B_2 = 4, B_3 = 6, B_4 = 4, B_5 = 2$ | 1 |
| Dedication (allowed families/machine) | $D_1 = \{1,2,3,4,5,6\}, D_2 = \{1,3,7,8\}, D_3 = \{1,4,6,7\}, D_4 = \{1,4,5,8\}, D_5 = \{1,3,4\}$ | 1 |
| | Number of independent instances | 2 |
| | Total problems | 162 |

## 6 RESULTS

In this section, we present the optimization results to the 162 test instances achieved with the different proposed solution methodologies. Two different application scenarios are investigated:

1. The entire problem is tackled. All information are available for the optimization method.
2. Only jobs within a forecast horizon $\Delta t$ are available/known at time $t$. This scenario arises if only a small number of future job arrivals is known or the arrivals are very uncertain.

The heuristic based on the BATC rule operates without any job forecast. The approaches based on BATC-II and BATC-IIi are restricted to scenario 2. They can handle job forecasts but as shown in (Klemmt, Lange, and Weigert 2009) they are only applicable for very short horizons $\Delta t$. For this reason, these methods are only investigated for $\Delta t = \overline{p}/4$. The MIP approach can be applied to solve the problems in both scenarios. But with regard to the strongly increasing computational burden only the investigation of scenario 2 is reasonable. The assessment is performed using different forecast horizons $\Delta t$ ($\overline{p}/4, \overline{p}/2, \overline{p}$), and different optimization time limits (5s, 30s) per time window. The VNS approach can be used in both application scenarios. However, in this research, it is only investigated for scenario 1 (full information). This will show how much more beneficial the global knowledge of future job arrivals in contrast to local time window information is. The VNS scheme is tested for 60 seconds runtime applied to all 162 test instances. For each test instance and this runtime three runs with different seeds are performed and the average values for TWT are used for comparison. All the tests for VNS are performed on an Intel QuadCore 2.4 GHz, 4GB computer with Linux Suse 10.1 as operating system.

In Table 2, we present the results for our test instances. Altogether 1944 different schedules are created with the different solution methods. The average TWT value of all test instances with their respective method and their denoted compare item (for example, $m = 3$) is shown relative to the TWT value obtained by the BATC rule. All MIP approaches are implemented with the parameter settings ($\Delta t_2 = \overline{p}$, $\overline{n} = 25$) using the CPLEX 11.2 library of TOMLAB (CPLEX-Settings: NODEALG = 1; SUBALG = 1; PREIND = 0; AGGIND = 0; all cuts disabled). All the tests for MIP are performed on an Intel Core 2 Duo 2.54 GHz, 4GB computer with Windows Vista as operating system.

Table 2: Comparison of the dispatching approaches, MIP approaches, and VNS approach for all 162 test instances. The results are aggregated according to the different instance properties. All values are relative to the TWT values found by the BATC rule.

| **Compare** | **Machines** | | | **Jobs per family** | | | **Families** | | | **Utilization** | | | **Overall** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $m = 3$ | $m = 4$ | $m = 5$ | $j = 20$ | $j = 30$ | $j = 40$ | $f = 5$ | $f = 6$ | $f = 7$ | $u = 0.7$ | $u = 0.8$ | $u = 0.9$ | |
| **BATC** | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 |
| **BATC-II** | | | | | | | | | | | | | |
| $\Delta t = \overline{p}/4$ | 1.005 | 1.005 | 0.989 | 0.993 | 1.008 | 0.997 | 0.986 | 1.016 | 0.997 | 0.987 | 0.993 | 1.018 | 0.999 |
| **BATC-IIi** | | | | | | | | | | | | | |
| $\Delta t = \overline{p}/4$ | 0.984 | 0.990 | 0.986 | 0.982 | 0.993 | 0.984 | 0.982 | 0.994 | 0.984 | 0.984 | 0.977 | 0.999 | 0.986 |
| **MIP (5s)** | | | | | | | | | | | | | |
| $\Delta t = \overline{p}/4$ | 0.915 | 0.946 | 0.954 | 0.949 | 0.936 | 0.930 | 0.940 | 0.931 | 0.944 | 0.939 | 0.929 | 0.947 | 0.938 |
| $\Delta t = \overline{p}/2$ | 0.908 | 0.938 | 0.942 | 0.936 | 0.926 | 0.926 | 0.926 | 0.919 | 0.943 | 0.925 | 0.924 | 0.940 | 0.929 |
| $\Delta t = \overline{p}$ | 0.922 | 0.976 | 0.975 | 0.942 | 0.954 | 0.977 | 0.928 | 0.945 | 1.001 | 0.930 | 0.941 | 1.002 | 0.957 |
| **MIP (30s)** | | | | | | | | | | | | | |
| $\Delta t = \overline{p}/4$ | 0.914 | 0.942 | 0.955 | 0.947 | 0.933 | 0.930 | 0.937 | 0.928 | 0.945 | 0.938 | 0.928 | 0.944 | 0.936 |
| $\Delta t = \overline{p}/2$ | 0.902 | 0.933 | 0.936 | 0.932 | 0.920 | 0.918 | 0.923 | 0.914 | 0.933 | 0.922 | 0.920 | 0.929 | 0.923 |
| $\Delta t = \overline{p}$ | 0.917 | 0.944 | 0.959 | 0.929 | 0.941 | 0.951 | 0.915 | 0.927 | 0.979 | 0.921 | 0.923 | 0.977 | 0.940 |
| **VNS** | | | | | | | | | | | | | |
| (total 60 s) | 0.899 | 0.925 | 0.924 | 0.910 | 0.913 | 0.924 | 0.900 | 0.909 | 0.938 | 0.917 | 0.908 | 0.922 | 0.916 |

As Table 2 shows, the presented MIP approach and VNS scheme clearly outperform the traditionally BATC type rules. More interesting is the result that the local information of $\Delta t = \overline{p}/2$ leads to solutions with an objective function difference smaller than 1% compared to the best (global) VNS results. Also it is interesting that almost the MIP (5s) approach nearly reaches the same solution quality. However, we expect that a more carful and refined design of the neighborhoods for the VNS scheme will further improve the results obtained by this metaheuristic.

The reasons why the BATC-based dispatch rules does not provide comparable good scheduling results is stated below and also depicted in Figure 2:

- In the MIP solution, some batches are retarded and resulting less batches are formed.

- The BATC-rule does not treat the equipment dedication constraints in an appropriate manner, i.e., for example, family 3 and 4 is scheduled on machine 1 in Figure 2.

The last mentioned point leads to a significantly increased makespan because of unbalanced workloads. Because BATC-II and BATC-IIi type rules have similar problems, their TWT value improvements compared to the BATC rule are small.

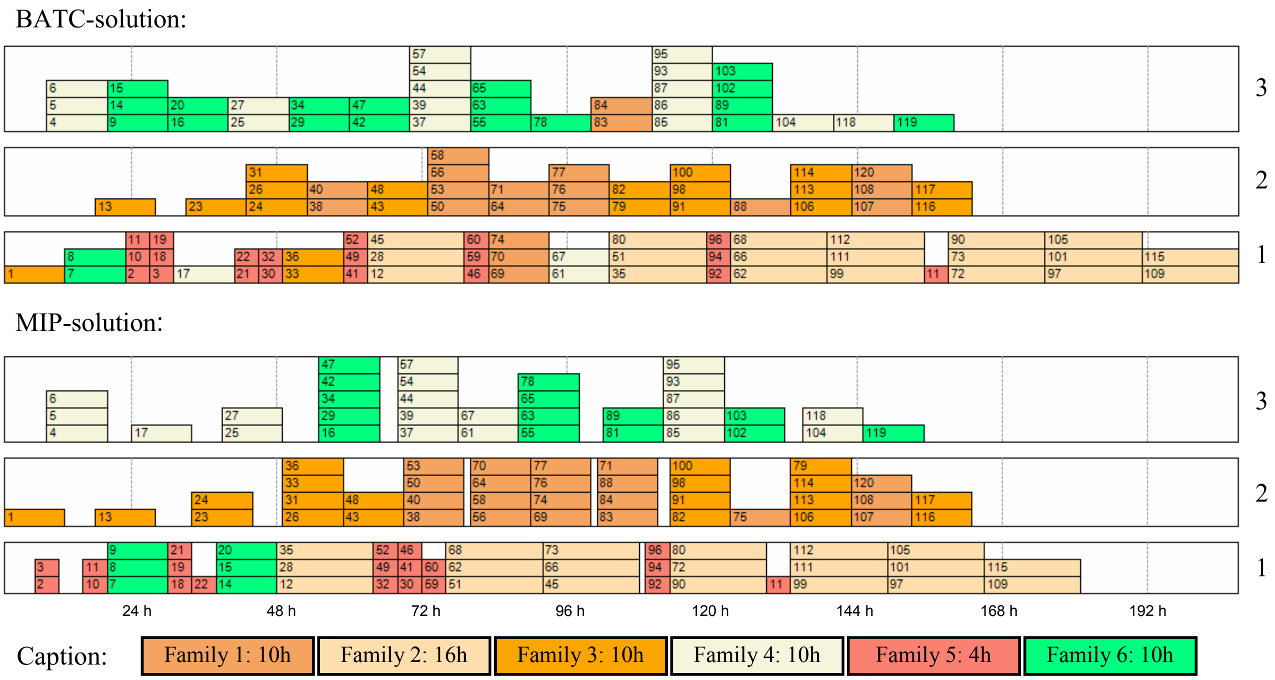

Figure 2: Comparison of BATC and MIP-solution for a test instance ($m$ = 3, $j$ = 20, $f$ = 6, $u$ = 0.9, $\overline{b}$ = 0.75, $\overline{p}$ = 10)

## 7 CONCLUSIONS AND FUTURE RESEARCH

In this paper, we discussed heuristics that are motivated by real-world scheduling problems in the semiconductor manufacturing domain. The researched scheduling problem contains unrelated parallel batch machines with different batch sizes, machine dedications, incompatible job families, and ready times of the jobs. We suggested a MIP-based approach that is based on a time window decomposition. Furthermore, we also proposed a VNS scheme that takes the various constraints into account. It turned out that the VNS approach slightly outperforms the MIP approach with respect to solution quality and time needed for computation. However, the MIP approach seems to offer some advantage when more additional constraints like, for example, time constraints between consecutive process steps have to be taken into account. The design of appropriate neighborhoods for VNS is more sophisticated in this situation.

There are some directions for future research. First of all, additional computational experiments with more test instances are highly desirable to assess the performance of the suggested heuristics. We are also interested to enrich the problem by additional, more real-world like, constraints. Furthermore, it is planned to use our VNS scheme for scenario 2. VNS seems to be faster than the MIP approach, but it provides at the same time high quality solutions.

## REFERENCES

Almeder, C., and L. Mönch. 2009. Variable neighborhood search for parallel batch machine scheduling. In *Preprints of the VIII Metaheuristics International Conference*.

Balasubramanian H., L. Mönch, J. Fowler, and M. Pfund. 2004. Genetic algorithm based scheduling of parallel batch machines with incompatible job families to minimize total weighted tardiness. *International Journal of Production Research* 42:1621-1638.

Bixby, R., R. Burda, and D. Miller. 2006. Short-interval detailed production scheduling in 300mm semiconductor manufacturing using mixed integer and constraint programming, In *Proceedings of the Advanced Semiconductor Manufacturing Conference*, 148-154.

Fowler, J. W., G. L. Hogg, and D. T. Phillips. 2000. Control of multi-product bulk-server diffusion/oxidation processes part two: multiple servers. *IIE Transactions on Scheduling and Logistics* 32(2):167–76.

Graham R. L., E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan. 1979. Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of Discrete Mathematics* 5:287-326.

Hansen, P., and N. Mladenovic. 2001. Variable neighborhood search: Principles and applications. *European Journal of Operational Research* 130:449-467.

Klemmt A., S. Horn, G. Weigert, and T. Hielscher. 2008. Simulation-based and solver-based optimization approaches for batch processes in semiconductor manufacturing. In *Proceedings of the 2008 Winter Simulation Conference*, eds. S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, J. W. Fowler, 2041-2049. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Klemmt A., J. Lange, and G. Weigert. 2009. Work center optimization methods for semiconductor manufacturing. In *Proceedings of the 19th International Conference on Flexible Automation & Intelligent Manufacturing,* 1325-1332.

Lawler E. L. 1977. A ‘pseudopolynomial’ time algorithm for sequencing jobs to minimize total tardiness. *Annals of Discrete Mathematics* 2:75–90.

Mathirajan M., and A. I. Sivakumar. 2003. Scheduling of batch processors in semiconductor manufacturing – a review. *Innovation in Manufacturing Systems and Technology (IMST).*

Mehta, S. V., and R. Uzsoy. 1998. Minimizing total tardiness on a batch processing machine with incompatible job families, *IIE Transactions* 30:165-178.

Mönch L., H. Balasubramanian, J. Fowler, and M. Pfund. 2005. Heuristic scheduling of jobs on parallel batch machines with incompatible job families and unequal ready times. *Computers & Operations Research* 32:2731-2750.

Perez I. C., J. Fowler, and W. M. Carlyle. 2005. Minimizing total weighted tardiness on a single batch process machine with incompatible job families. *Computers & Operations Research* 32:327-341.

Rose, O. 2003. Accelerating products under due-date oriented dispatching rules in semiconductor manufacturing. In *Proceedings of the 2003 Winter Simulation Conference*, eds. S. E. Chick, P. J. Sanchez, D. M. Ferrin, D. J. Morrice, 1346-1350. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Raghavan, N. R. S., and M. Venkataramana. 2006. Scheduling parallel batch processors with incompatible job families using ant colony optimization. In *Proceedings International Conference on Automation Science and Engineering (CASE 2006)*, 507-512.

Vepsalainen, A. P. J., T. E. Morton. 1987. Priority rules for job shops with weighted tardiness costs. *Management Science* 33:1035–1047.

## AUTHOR BIOGRAPHIES

**ANDREAS KLEMMT** studied mathematics at Dresden University of Technology, Germany. He obtained his degree in 2005 in the field of optimization and data classification. He has been a Research Assistant at Electronics Packaging Laboratory of the Dresden University of Technology since 2006 and works on the field of production control, simulation & optimization of manufacturing processes, especially in the field of electronics and semiconductor industry. His email is <klemmt@avt.et.tu-dresden>.

**GERALD WEIGERT** is an Assistant Professor at Electronics Packaging Laboratory of the Dresden University of Technology. Dr. Weigert works on the field of production control, simulation & optimization of manufacturing processes, especially in electronics and semiconductor industry. His email is <Gerald.Weigert@tu-dresden.de>.

**CHRISTIAN ALMEDER** is Assistant Professor at the Department of Business Administration of the University of Vienna. His research interests are discrete-event simulation, combinatorial optimization, metaheurstics, and optimal control models in the context of production and logistic systems. His email is <christian.almeder@univie.ac.at>.

**LARS MÖNCH** is Professor in the Department of Mathematics and Computer Science at the University of Hagen, Germany. He received a master’s degree in applied mathematics and a Ph.D. in the same subject from the University of Göttingen, Germany. His current research interests are in simulation-based production control of semiconductor wafer fabrication facilities, applied optimization and artificial intelligence applications in manufacturing, logistics, and service operations. He is a member of GI (German Chapter of the ACM), GOR (German Operations Research Society), SCS, INFORMS, and IIE. His email address is <Lars.Moench@fernuni-hagen.de>.