Proceedings of the 2018 Winter Simulation Conference
M. Rabe, A.A. Juan, N. Mustafee, A. Skoogh, S. Jain, and B. Johansson, eds.

A CP MODEL FOR A SCHEDULING PROBLEM WITH LIMITED SECONDARY RESOURCES COMPARED TO A DES MODEL

Dirk Doleschal
Karlheinz Bock

Electronics Packaging Laboratory
Technische Universität Dresden
Dresden, D-01062, GERMANY

ABSTRACT

An efficient scheduling of bottleneck areas within the semiconductor manufactory gets more and more important. Due to the high complexity within the manufacturing area it is currently not possible to optimize the whole (or even a big part of the) factory at once. So mostly only work center specific optimization approaches are investigated. Typically scheduling problems only deal with two dimensions – jobs and equipment. But in some areas of semiconductor manufactory also a third dimension has to be considered – a limited secondary resource. In this paper a Constraint Programming model for such limited secondary resource problems is presented. Thereby the scheduling model also deals with setup matrices for the first and also secondary resource. The modeling of this CP model is shown in detail and the results are compared to a discrete event simulation using dispatching rules. The test data for the first tests are orientated on real production data.

1 INTRODUCTION

An efficient scheduling of bottleneck areas within the semiconductor manufactory gets more and more important. Within the semiconductor fab a huge variety of different scheduling problems exists. Also due to the fact that the semiconductor fab is known as the most complex manufactory worldwide (May and Spanos 2006), it is very challenging to optimize the production flow within this area. According to the high complexity it is currently not possible to optimize the whole (or even a big part) of the factory at once. So mostly only work center specific optimization approaches are investigated.

In this paper a limited secondary resource problem shall be solved by a constrained programming approach. In semiconductor industry only a few papers address scheduling with CP. In Ham et al. (2017) a diffusion processes area which batch processing and in Ham (2018) a Litho area with batch processing were investigated. Zeballos et al. (2011) presented a CP approach for automated wet etch stations within a semiconductor fab. Here a synchronization of a robot arm with the processing times within the wet etch bathes has to be performed. Scheduling problems with limited secondary resources occur in several processing areas of a semiconductor fab. One example is the lithography step, where masks – so called reticles – are needed to transfer the structure to the wafers. In Klemmt et al. (2010) an approach for optimizing this area of semiconductor manufactory was presented. Here a multi-stage optimization method based on mixed integer programming was applied. In summary four stages are implemented, whereby the granularity is increased from one stage to the next. A further area for such limited secondary resource problems within the semiconductor manufactory is the wafer test area. Here, also Klemmt et al. (2011) presented an approach for optimizing this area. In his paper he combined a simulation based optimization with a mixed integer based capacity planning. Here the needed secondary resources – the probe cards – also have different processing speeds. This heterogenic pool of probe cards also increases the complexity.

Doleschal et al. (2013) investigated a secondary resource planer based on a mixed integer capacity planning combined with a discrete event simulation system.

In this work an investigation of scheduling a work center with limited secondary resources, as well as setup times, release dates and due dates are presented. For this a constraint programming model is established and compared to a discrete event simulation model. A publication which is comparable to this paper has not been found in literature until now. Typically in scheduling investigations, only a two dimensional scheduling problem is investigated. These two dimensions are the jobs and the resources, where the jobs have to be scheduled. In this investigation an additional third dimension is included which is the limited secondary resource. This third dimension increases the scheduling complexity.

The paper is structured as follows. In the second section the underlying secondary resource problem is defined in detail. Section 3 described the modeling of the constraint programming model. In contrast to this in section 4 the buildup of the simulation model is presented. The results are shown in section 5 and in the last section a short conclusion and outlook is given.

2 PROBLEM DESCRIPTION

The underlying scheduling problem discussed in this paper can be found at several stations of a semiconductor fab. These are for example the lithography area and several test areas. In both parts of the semiconductor manufactory limited secondary resources are needed. In the lithography step these are for example reticle masks which are used to transfer the structure to the wafers. In the test areas these limited secondary resources may be probe cards which are needed to contact the electrical pads on the wafers or chips. Typically in these areas the secondary resources are very expensive and therefore only available in a limited number. Furthermore each product needs a separate secondary resource which may also depend on the current process step. This leads to a high number of different secondary resources but the number of secondary resources per type is relatively low.

In addition to the previous mentioned conditions in this investigation also sequence-dependent setup times exist. These setup times exist for the equipment as well as for the secondary resource. These times depend on several process parameters, i.e. the temperature.

The smallest unit to be processed within the factory is the lot. In this investigation this lot has to be processed on one equipment/secondary resource combination out of the allowed combinations. Thereby the secondary resource depends on the product and process step of the lot and furthermore, dedications exist which permit or forbid an equipment for processing the special lot or product. Typically in such scheduling problems, both resources – equipment and secondary resource – can be a bottleneck.

To adapt the model to be applicable for using within a real environment, also the current states of the equipment, the secondary resources as well as the jobs have to be considered.

Summarized the input data from the real environment into the scheduling model is as follows:

- Lot
    - Release date
    - Priority / Weight
    - Due date
    - Dedications
        - Permitted equipment and secondary resource
        - Process time for each combination
        - Needed setup for each combination
- Equipment
    - Current setup (temperature, process program)
    - Current secondary resource
    - Current job
    - Remaining runtime
    - Current state

- Secondary resource
    - Current equipment
    - Current state
- Setup matrices for
    - Equipment
    - Secondary resource

With this input data a scheduling model can be buildup. Objectives within this investigation are:

- Tardiness – The summed amount of time, jobs are too late.
- Cycle time – The summarized time, the jobs spent within the work center
- Number of setups – The number of setups, the equipment have to perform
- Setup time – The summed amount of time, the equipment is in a setup state

3 CP MODEL

To build up a Constraint Programming (CP) model first the input data has to be defined. The underlying CP optimization system is the IBM ILOG CPLEX Optimization Studio. In this system a scripting language called OPL is implemented. Furthermore in contrast to Mixed Integer Programming the Constraint Programming is typically not build up with equation, rather with constraints which is like a verbal description of the problem. Due to this in this section also the constraints are used to describe the model and no mathematical description. In the further paper the lots which have to be processed on the equipment are called jobs. This is a convention which is often used for scheduling problems.

3.1 Input data

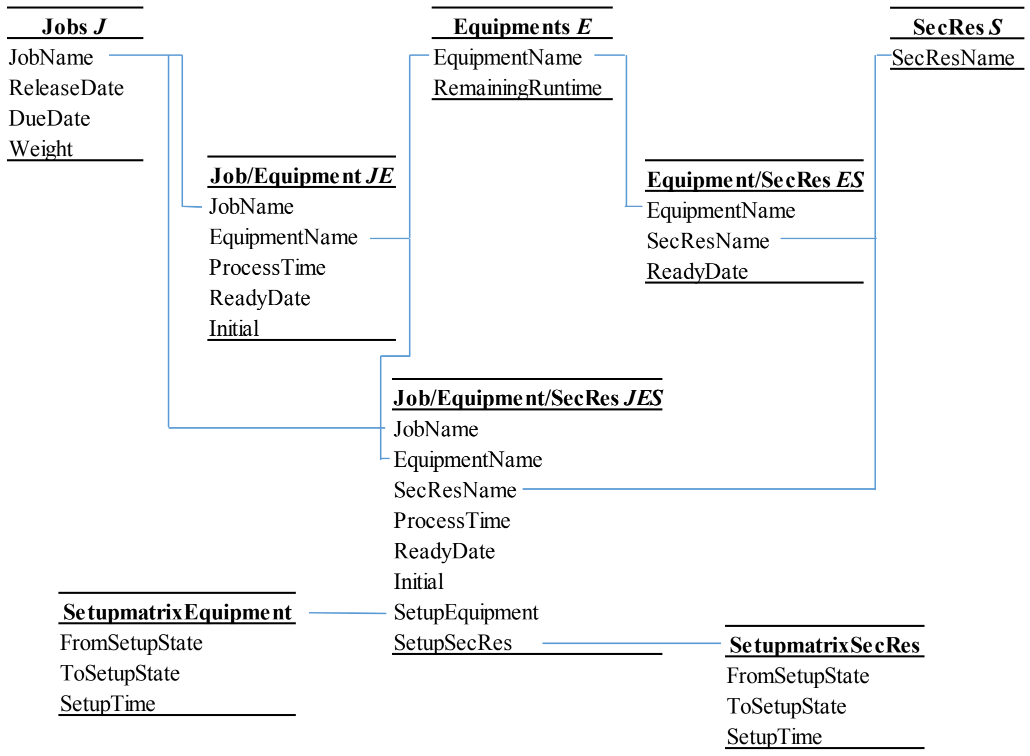

Figure 1: CP input data.

The input data for the CP model is formatted as sets of tuples within the OPL language. Thereby a tuple includes several elements/information for a tuple member. The tuple sets for the CP model orientate on the input data defined in section 2. Figure 1 shows the input data and the data connections between the tuples.

Here 8 different tuple sets are defined. The blue lines between these sets defines the connection between these sets. The secondary resource is shortened as "SecRes" in this picture and also in the further CP model description this abbreviation is used.

The data set is build up in this way, that each basic resource (jobs, equipment and secondary resources) has its own data tuple set (*J*, *E* or *S*) with basic information. Furthermore there exist combinations of these basic tuples. For example the job/equipment tuple *JE* implements all allowed job/equipment combinations and regards the information like the process time or the ready date which is – in this case – the maximum of the release date of the job and the remaining runtime of the equipment. The same structure was applied to the equipment/secondary resource combinations. The "main" tuple in this data input is the job/equipment/secondary resource tuple *JES*. Here all possible tasks are defined for our 3-dimensional scheduling problem. The three dimensions are the jobs, the equipment and the secondary resources. This tuple has the highest granularity compared to all other tuples.

With this input data the CP scheduling model can be build up. As already mentioned the CP model is presented in this investigation using the IBM ILOG OPL language. Further information to this scripting language can be found in IBM (2017).

To access an item in the tuple, the following is spelled out. To access for example the JobName of job $j \in J$ the following has to be written: $j^{JobName}$. With this convention the decision variables for building up the CP model can be defined.

3.2 Decision variables

To define a CP model first the decision variables have to be defined. Here the notation of the CP model is inspired by this one used in Ham (2017) and Ham (2018). Further information can also be found in the papers of Laborie and Rogerie (2008) and Laborie (2009). In this investigation first three interval variables for processing the jobs are defined:

$$\text{interval} \quad SpanJobs_j \qquad \forall j \in J \tag{1}$$

$$\text{interval} \quad EQjobs_t \text{ optional} \in [t.^{ReadyDate}, \infty) \text{ size } t.^{ProcessTime} \qquad \forall t \in JE \tag{2}$$

$$\text{interval} \quad SRjobs_o \text{ optional} \in [o.^{ReadyDate}, \infty) \text{ size } o.^{ProcessTime} \qquad \forall o \in JES \tag{3}$$

The first interval variable *SpanJobs* defines an interval for each job (1). The second interval variable (2) defines an interval for each allowed equipment/job combination. The earliest start date of this interval is the ready date defined within the tuple *JE*. This ready date is the maximum of the remaining runtime of the according equipment and the release date of the job. The length is defined by the process time. To allocate a secondary resource to the processing of a job on an equipment a third interval variable called *SRjobs* is needed. The allowed dedications of jobs to equipment and secondary resources are gained by the tuple input data. Here only allowed combinations exists. Here again the earliest start date of this job is defined by the ready date in the tuple set *JES*.

Furthermore two sequence variables are needed. One for the secondary resources and one for the equipment. With this sequence variable the regarding object gets the information, which task has to be processed at which time. A task thereby is the definition of processing a job on an equipment. Additionally this sequence variables can be used for the setup matrices:

$$\text{sequence } EQsequence_m \in SRjobs_o \text{ types } o.^{SetupEquipment} \quad \forall m \in E; o \in JES: o.^{EquipmentName} = m.^{EquipmentName} \tag{4}$$

$$\text{sequence } SRsequence_r \in SRjobs_o \text{ types } o.^{SetupSecRes} \quad \forall r \in S; o \in JES: o.^{SecResName} = r.^{SecResName} \tag{5}$$

The constraint (4) defines the sequence variable *EQsequence* for each equipment $m \in E$ where all tasks from the interval variable *SRjobs* with the same *EquipmentName* are used. In (5) the sequence *SRsequence* is defined in the same way, especially that this variable is defined for the secondary resources. The information after "types" defines the parameter which is used for the setup matrix. In this case for the equipment sequence a setup index called "*SetupEquipment*" is used. This index depends on the several process parameters as well as the chosen secondary resource. For the secondary resources also a setup matrix is buildup which also uses the process parameters and additionally the equipment.

Furthermore for the objective tardiness a further decision variable is needed:

$$\text{dvar } Tardiness_j \in \mathbb{R}^+ \qquad \forall j \in J \tag{6}$$

With these decision variables and sequences the constraint model can be defined.

3.3 Model

With the defined input data and decision variables, a CP model for the presented problem can be written as:

$$noOverlap(EQsequence_m, SetupmatrixEquipment) \qquad \forall m \in E \tag{7}$$
$$noOverlap(SRsequence_r, SetupmatrixSecRes) \qquad \forall r \in S \tag{8}$$
$$alternative(SpanJobs_j, \{EQjobs_t\}_{\forall t \in JE: j.^{JobName} = t.^{JobName}}) \qquad \forall j \in J \tag{9}$$
$$alternative(EQjobs_t, \{SRjobs_r\}_{\forall r \in JES: r.^{JobName} = t.^{JobName} \wedge r.^{EquipmentName} = t.^{EquipmentName}}) \qquad \forall t \in JE \tag{10}$$
$$presenceOf(EQjobs_t) = 1 \qquad \forall t \in JE : t^{Initial} \equiv 1 \tag{11}$$
$$startOf(EQjobs_t) = 0 \qquad \forall t \in JE : t^{Initial} \equiv 1 \tag{12}$$
$$presenceOf(SRjobs_o) = 1 \qquad \forall o \in JES : o^{Initial} \equiv 1 \tag{13}$$
$$startOf(SRjobs_o) = 0 \qquad \forall o \in JES : o^{Initial} \equiv 1 \tag{14}$$
$$Tardiness_j \geq endOf(SpanJobs_j) - j^{DueDate} \qquad \forall j \in J \tag{15}$$

With the constraint (7) it is ensured, that all tasks performed on an equipment cannot overlap another task performed on the same equipment. Furthermore the setup matrix SetupmatrixEquipment is embedded and executed by the CP solver. This means if two adjacent tasks on an equipment does not have the same setup type (defined in (4)) a setup time which is defined in the setup matrix has to be respected. The constraint (8) does the same thing as constraint (7), but this time for the secondary resources. Constraint (9) ensures that each job is assigned to exactly one equipment out of the allowed equipment. This is, because the interval variables *EQjobs* and *SRjobs* are optional, which means that the tasks in these variables can be scheduled but do not have to be scheduled. In contrast to this, the interval variable *SpanJobs* is not optional, which means, that each task/job within this variable has to be scheduled. With constraint (10) it is ensured – in the same way as in constraint (9) – that each task of *EQjobs*, which is active, has to be assigned to exactly one task out of the possible set of *SRjobs*. With the *alternative* constraint the tasks are also "connected". This means the start time and end time of the tasks are synchronized. Because in this investigation the work center is not empty in the beginning of scheduling, also "initial" constraints have to be buildup. These constraint are the constraints (11) - (14). Here for each task which is defined as "initial" it is ensured, that the associated task is active (for the *EQjobs* and *SRjobs*) and the start time of this task is at time 0, which is the beginning of the scheduling model. The last constraint (15) is used to calculate the tardiness for each job by comparing the end date of the scheduled task with the due date of the job. The variable *Tardiness* is chosen out of the set of positive real numbers, which ensures that the variable cannot be negative.

With this model a permissible schedule can be calculated. To gain a good schedule regarding the objectives defined in section 2 a minimization objective has to be defined.

3.4 Objectives

For defining the optimization goal for the CP model first the calculation of each objective is shown. For the tardiness already the auxiliary variable *Tardiness$_j$* is defined and calculated in the CP model. To calculate the overall tardiness this variables must be summed up:

$$TotalTardiness = \sum_{j \in J} Tardiness_j \tag{16}$$

The cycle time can be calculated by the following equation:

$$TotalCycleTime = \sum_{j \in J} endOf(Spanjobs_j) - j^{\cdot ReleaseDate} \tag{17}$$

To calculate the number of setups and setup time a much more complicated equation have to build up.

$$NrSetups_m = \sum_{\substack{o \in JES: \\ o^{EquipmentName} = m^{EquipmentName}}} o^{SetupEquipment} \neq typeOfNext(EQsequence_m, SRjobs_o, o^{SetupEquipment}) \tag{18}$$

$$TotalNrSetups = \sum_{m \in E} NrSetups_m \tag{19}$$

Equation (18) calculates the number of setups for each equipment and in equation (19) this is summed up over all equipment. The setup time can be calculated using a similar equation like shown in (18):

$$SetupTime_m = \sum_{\substack{o \in JES: \\ o^{EquipmentName} = m^{EquipmentName}}} \begin{aligned} & o^{SetupEquipment} \neq typeOfNext(EQsequence_m, SRjobs_o, o^{SetupEquipment}) \\ & \cdot SetupmatrixEquipment(o^{SetupEquipment}, typeOfNext(\cdots)) \end{aligned} \tag{20}$$

$$TotallSetupTime = \sum_{m \in E} SetupTime_m \tag{21}$$

Equation (20) also includes the *SetupmatrixEquipment*(). This function returns the setup time which is needed for a setup change from state $o^{SetupEquipment}$ to the new setup state *typeOfNext*(*EQsequence$_m$*,*SRjobs$_o$*, $o^{SetupEquipment}$). Equation (21) sum up the calculated *SetupTime$_m$* for all equipment $m \in E$.

To control the importance of the four different goals additional weighting parameters ω_T, ω_C, ω_{NS} and ω_{ST} are added for the four objectives tardiness, cycle time, number of setups and setup time. Now the overall optimization goal for CP model can be written as:

$$\text{minimize} \quad \begin{aligned} & \omega_T \cdot TotalTardiness + \omega_C \cdot TotalCycleTime \\ & + \omega_{NS} \cdot TotalNrSetups + \omega_{ST} \cdot TotalSetupTime \end{aligned} \tag{22}$$

The CP optimization build up in this section will be compared to a simulation based optimization. The buildup of the simulation system is presented in the next section.

4 SIMULATION MODEL

For testing and comparing purposes a simulation model for the same problem was build up. The used software is the simcron MODELLER 3.3. This is a discrete event based simulation system. Here several main elements exists:

- Queues and machines – In this stations typically processes are done and process time is consumed.
- Jobs – This is the smallest unit which has to be processed by the stations.
- Routes – A route defines, in which order the associated jobs have to be processed on which machines.
- Branches – branches are similar to routes, except that branches only consists of one process step where several stations can be used for this step. Here it is possible to assign the job to only one station of the branch, to all stations or to a subset of the stations within this branch.
- Setup objects – the setup objects are assigned to stations and uses a job property for choosing the right setup state. A setup matrix can be assigned to the setup object for the setup times.
- Event handler – the event handler are very powerful tools to get a manual intervention to the simulation system. The event handler can be placed on nearly all objectives of the system and will be executed at special defined time (for example if a job is allocated to a machine or even if it wants to be allocated).

With this simulation system it is possible to depict the problem defined in section 2.

4.1 Structure of the simulation model

The simulation model is build up in this way, that each job gets its own route. Furthermore all machines and secondary resources within the associated work center are depicted as machine stations.

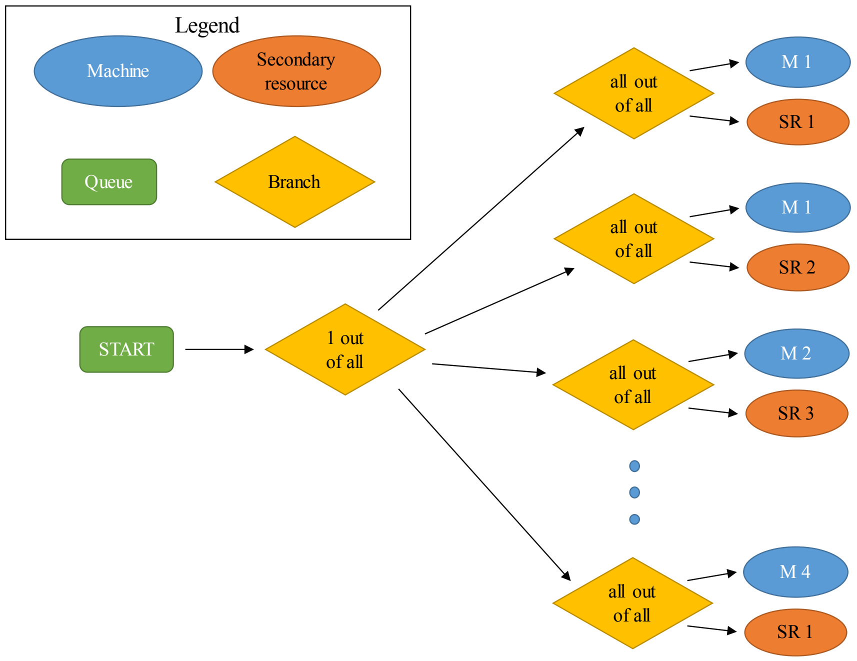

Figure 2: Schematic view of simulation model.

The schematic buildup of the simulation model is presented in Figure 2. Here an exemplary route for one job is shown.

Here a job is first located in a start queue. From this start queue the job is “moved” to the first branch. This branch includes all allowed machine/secondary resource combinations. Therefore, here in this branch the job has to use exactly one path out of all allowed paths. The elements in this branch are again branches. These (second) branches includes the special machines and secondary elements. In this branch the job has to be placed on both elements (machine and secondary resource) at the same time. With this buildup it is ensured, that a job uses a machine and secondary resource at the same time and the simulation model only place a job to a combination if both machine types are available.

The ordering of the jobs can be done in the START queue by using job properties. These could be for example the due date, priority, waiting time etc.. To gain relatively good results furthermore event handler are placed on the machines to ensure for example a setup minimization.

4.2 Dispatching rule

The implemented dispatching rule is described in this section. As already mentioned the simulation model has two types of job ordering/dispatching. The first type is the job ordering in the START queue. In this testing model the ordering of the jobs is done by the due date. This means that jobs with a small due date are ordered first and jobs with a large due date are ordered at the end of the queue.

The second implementation of a dispatching rule is a setup minimization. Here each machine gets an event handler which is activated if a job wants to enter the machine. In this simulation model two different approaches for the setup minimization are implemented. The first approach is from the machine viewpoint and the second approach is from a job viewpoint.

The setup minimization of the machine view is called *SetupMinMachine* and works as following:

1. Checks if the setup state of the machine and secondary resource is identically to the needed setup state
    a. If this is true, than the job is allowed to enter the machine/secondary resource combination
2. If the setup state does not match, the rule tests all other jobs within the START queue, if the needed setup state of another job is identically, than the asking job is denied.

The setup minimization of the job view is called *SetupMinJob* and works as follows. Here an additional factor ω is implemented to adjust the dispatching rule:

1. If a job wants to enter a machine, all possible branches of this job are tested and the minimum setup time $t_{\min}$ is noticed.
2. Then only these branches with a setup time $\leq t_{\min} \cdot \omega$ are allowed.

Now, the results from these simulation models are compared to the results gained by the constraint programming approach. To test both methods test data from real industry environment is used. Due to confidentiality this data is anonymized.

5 RESULTS

In this first investigation three different test sets are used. The test sets are build up artificially but the structure is orientated on real data.

Table 1: Overview of used test sets.

Parameter	Test set 1	Test set 2	Test set 3
Number of jobs	140	200	80
Number of equipment	25	25	25
Number of secondary resources	40	50	30

All additional data are the same as shown in section 2 and 3.1. Parameters like the machine utilization, range of job release dates or due date range is tried to keep equal for each test sets. Due to the near to real environment a detailed information of these test sets could not be given.

For the CP optimization an optimization time of two minutes is allowed. This time is chosen due to the fact that the method should be used in an real fab environment where the calculation time is very limited. The optimization uses only one thread and is done on a Notebook with an Intel i7-4600U CPU and 12 GB RAM. In summary four different CP optimization parameters are used:

Table 2: Different parameters for CP approach.

Parameter	**CP 1**	**CP 2**	**CP 3**	**CP 4**
Tardiness ω_T	100	1000	1000	100
Cycle time ω_C	1	1	1	1
Number of setups ω_{NS}	1000	1000	1000	1000
Setup time ω_{ST}	0	0	10	10

The parameters are chosen in this way, that the dimensions of the single result values are nearly the same. So in average the cycle time is about 1000 time the number of setups or ten times of the setup time.

All results are normalized. This means for each test set and objective the minimal result was calculated and then all values from this result was divided by the minimum value. With this normalization the smallest value can only be 1.

For each test set an own result figure was generated. In Figure 3 the result for the first test set is shown. Here it can be seen, that the CP approach generates good results for the cycle time, and depending on the parameters for the number of setups (CP 2) and tardiness (CP 3). Overall CP 2 seems a good compromise for this test set compared to the simulation. For the simulation it can be seen, that set *SetupMinJob* dispatching rule generates a low setup time.

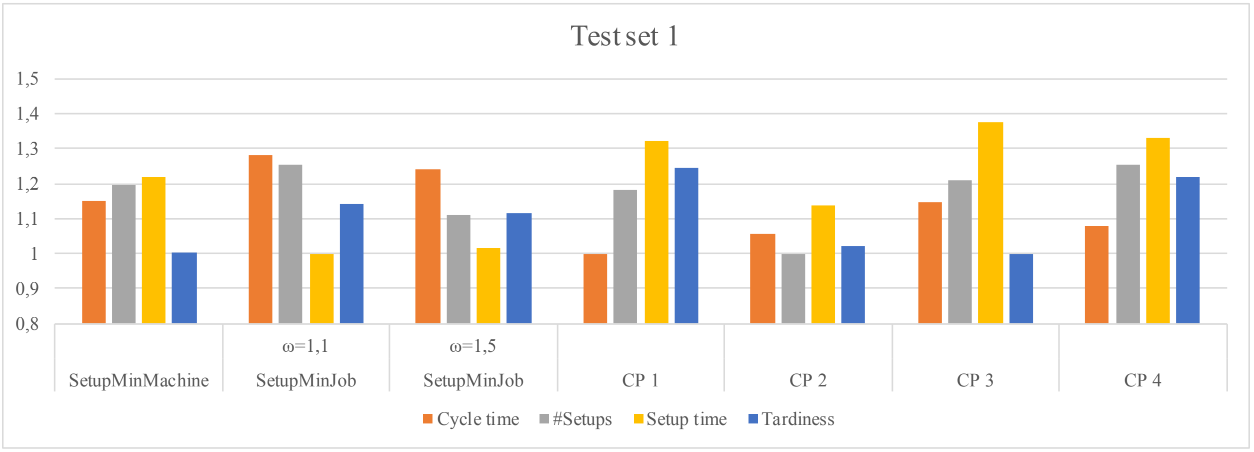

Figure 3: Result for test set 1.

In Figure 4 the results for test set 2 are presented. This is the biggest test set in this investigation. The high number of jobs in this test set leads to more worse results compared to the simulation model. If the setup time is also used in the constraint programming approach, the number of additional conditions rises dramatically and leads to bad results – especially for bigger problems. Overall the *SetupMinMachine* simulation model seems to be best for this test set. The highest value for the cycle time was calculated by CP4 with 4.5% above the best calculated cycle time. CP2 calculated the best value for the tardiness. This example shows, that the constraint programming also highly depends on the chosen parameters.

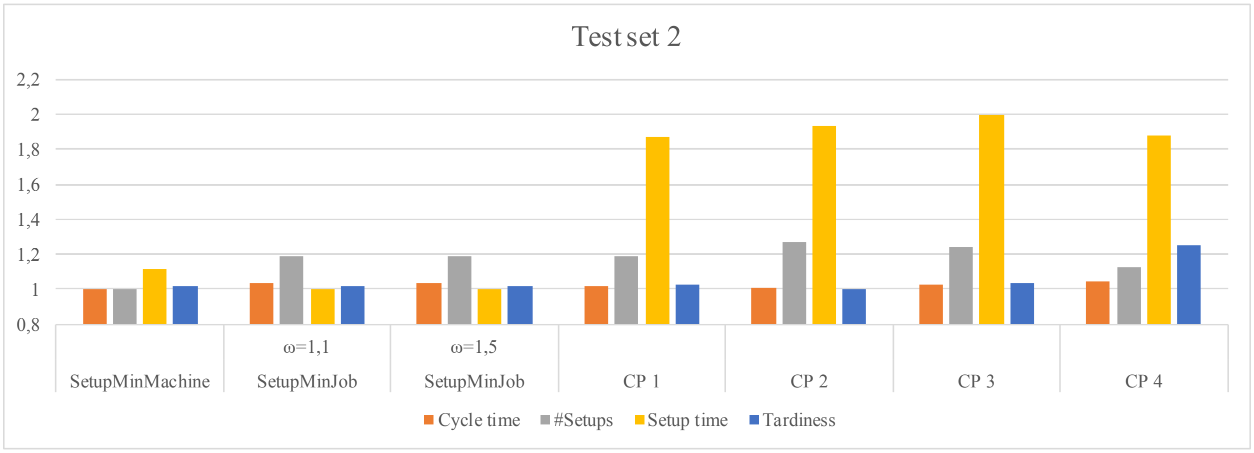

Figure 4: Result for test set 2.

The results for the last test set are shown in Figure 5. This is a much smaller test set compared to the first and second test set. Due to the smaller amount of constraints and conditions the results for the CP model are much better than the results from the simulation model. Depending on the priority of the objectives, CP 2 or CP 4 seems to perform good for this test set.

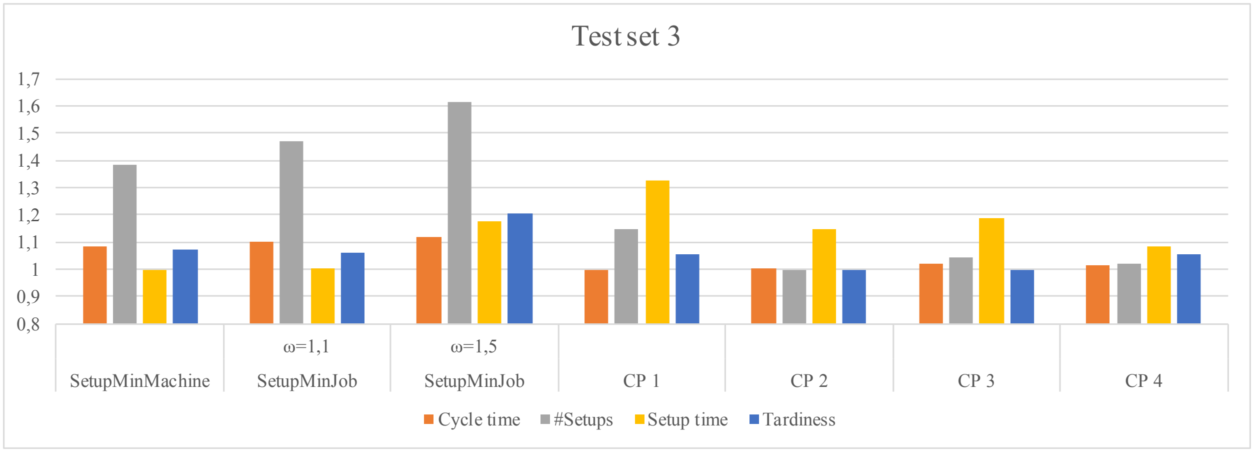

Figure 5: Result for test set 3.

6 CONCLUSION AND OUTLOOK

In this investigation a constraint programming approach for a limited secondary resource problem with setup matrices for a work center scheduling model is presented. This model is build-up using the IBM ILOG CPLEX Optimization Studio. The formulation in this work orientates on the IBM OPL language using the special CP constraints. The propagated CP model is further compared to a simulation based approach. Here two different dispatching rules are tested. Objectives in this investigation are the cycle time, tardiness, number of setups as well as the setup time. The results show, that CP can outperform the simulation model if the underlying scheduling problem is not too big. Otherwise the CP model gets too complex, which ends in worse results.

This complexity problem has to be researched more in detail and also it seems to be important to find methods to reduce problem complexity or to divide the problems in smaller disjunctive units which could be solved independently. Furthermore, to make the problem more realistic also reentrances of jobs in the same work center should be considered. Such reentrances typically occur in the semiconductor manufactory. For example in the wafer test in the frontend or in the backend test the jobs typically have several test steps where the same pool of machines can be used. For such problems it would also be interesting, how good the simulation model can perform with that, in comparison to a constraint programming based approach. With such reentrance problems the complexity also increases. In a further stage this approach can be implemented in a rolling horizon manner. This would give a better assessment of the impact.

ACKNOWLEDGEMENT

This project has received funding from the Electronic Component Systems for European Leadership Joint Undertaking under grant agreement No 737459. This Joint Undertaking receives support from the European Union's Horizon 2020 research and innovation program and Germany, Austria, France, Czech Republic, Netherlands, Belgium, Spain, Greece, Sweden, Italy, Ireland, Poland, Hungary, Portugal, Denmark, Finland, Luxembourg, Norway, Turkey.

REFERENCES

Doleschal, D., G. Weigert, A. Klemmt, and F. Lehmann. 2013. "Advanced Secondary Resource Control in Semiconductor Lithography Areas: From Theory to Practice". In *Proceedings of the 2013 Winter Simulation Conference*, edited by R. Pasupathy et al., 3879-3890. Piscataway, New Jersey: IEEE.

Ham, A. 2018. "Scheduling of Dual Resource Constrained Lithography Production: Using CP and MIP/CP". *IEEE Transactions on Semiconductor Manufacturing* 31(1):52–61.

Ham, A., J. Fowler, and E. Cakici. 2017. "Constraint Programming Approach for Scheduling Jobs with Release Times, Non-identical Sizes, and Incompatible Families on Parallel Batching Machines". *IEEE Transactions on Semiconductor Manufacturing* 30(4):500-507.

IBM 2017. *IBM ILOG CPLEX Optimization Studio OPL Language Reference Manual*. Version 12, Release 8 ed. IBM, USA.

Klemmt, A., J. Lange, G. Weigert, F. Lehmann, and J. Seyfert. 2010. "A Multistage Mathematical Programming-based Scheduling Approach for the Photolithography Area in Semiconductor Manufacturing". In *Proceedings of the Winter Simulation Conference,* edited by B. Johansson et al., pp. 2474-2485. Piscataway, New Jersey: IEEE.

Klemmt, A., J. Lange, G. Weigert, E. Beier, and S. Werner. 2011. "Combination of simulation and capacity optimization for detailed production scheduling in semiconductor manufacturing". In *Proceedings of the 21th International Conference on Flexible Automation and Intelligent Manufacturing,* June 26th – 29th, Taichung, Taiwan, 603-610.

Laborie, P. 2009. "IBM ILOG CP Optimizer for Detailed Scheduling Illustrated on Three Problems". In *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, edited by W.-J. van Hoeve and J. N. Hooker, 148–162. Berlin, Heidelberg: Springer.

Laborie, P. and J. Rogerie. 2008. "Reasoning with Conditional Time-Intervals.". In *Proceedings of the 21st FLAIRS Conference*, edited by David C. Wilson and H. Chad Lane, May 15th – 17th, Florida, USA, 555–560.

May, G. S. and C. J. Spanos. 2006. *Fundamentals of semiconductor manufacturing and process control*. New Jersey: John Wiley & Sons, Inc.

Zeballos, L. J., P. M. Castro, and C. A. Méndez. 2010. "Integrated Constraint Programming Scheduling Approach for Automated Wet-etch Stations in Semiconductor Manufacturing". *Industrial & Engineering Chemistry Research 50*(3):1705-1715.

AUTHOR BIOGRAPHIES

DIRK DOLESCHAL studied Mathematics at Dresden University of Technology, Germany. He obtained his degree in 2010 in the field of optimization. He has been a Research Assistant at Electronics Packaging Laboratory of the Dresden University of Technology since 2010 and works in the field of production control, simulation & optimization of manufacturing processes. His email is dirk.doleschal@tu-dresden.de.

KARLHEINZ BOCK, Senior Member IEEE, achieved the Dr.-Ing. degree in RF Microelectronics from the University of Darmstadt, Germany. During his scientific life he has been with Tokoku University in Sendai Japan 1995-96, Imec vzw. in Leuven Belgium 1996-1999 and Fraunhofer IZM and EMFT in Munich Germany 1999-2014. Since March 2008 until September 2014 he also served as professor of Polytronic Microsystems at the University of Berlin (TU Berlin). He received in 2012 the Dr. h. c. from Polytechnical University of Bukarest in Romania. Since October 2014 he serves as professor of electronics packaging and director of the Institute for Electronics Packaging (IAVT) at the Dresden university of technology (TU Dresden), at present he serves also as vice dean of the faculty of electrical and computer engineering and as IEEE EPS region 8 representative in the board of governors. His email address is karlheinz.bock@tu-dresden.de.