SOLUTION APPROACHES FOR THE CLUSTER TOOL SCHEDULING PROBLEM IN SEMICONDUCTOR MANUFACTURING

Heiko Niedermayer
Computer Networks and Internet
Wilhelm-Schickard-Institute for Computer Science
University of Tübingen
D-72076 Tübingen, Germany
E-mail: niedermayer@informatik.uni-tuebingen.de

Oliver Rose
Distributed Systems
Department of Computer Science
University of Würzburg
D-97074 Würzburg, Germany
E-mail: rose@informatik.uni-wuerzburg.de

KEYWORDS

simulation, manufacturing, semiconductor, cluster tools, scheduling

ABSTRACT

With the increase in computer performance scheduling complex manufacturing plants with global heuristics is becoming a realistic option. The complete factory scheduling problem consists of the global problem and its subproblems for each work center and machine. In semiconductor manufacturing a lot of processing is done using cluster tools. Cluster tools are a special kind of machine that can be described as a small factory. We discuss solutions for the optimization of schedules for cluster tools which is a subproblem of the factory scheduling problem. Our main idea is to use rules based on slow-down factors or search approaches based on the use of slow-down factors for predicting the cycle times. We use simulation to evaluate the schedules.

INTRODUCTION

Semiconductor manufacturing plants are often considered to be the most complex factories that currently exist. Even ignoring the size of these manufacturing plants the scheduling problem is extremely complex. It is a job shop scheduling problem that includes batch machines, sequence-dependent setups, recirculating flows, and cluster tools. While batch machines and setups are widely studied in the literature (Pinedo 2001) the problem with cluster tools is not.
Cluster tools are machines that consist of loadlocks, handlers and other machines to process the wafers of the lots in the loadlocks. The advantage of cluster tools is that the processing of the wafers is pipelined. Thus compared with the cycle time of a lot using consecutive machines the cycle time of a lot in a cluster tool is reduced. As cluster tools are usually pumped to vacuum the yield may also be improved and less clean-room space may be required.

Since cluster tools are small factories themselves that may process more than one lot at a time, the behavior of cluster tools with respect to the cycle time is complex. In this paper we usually consider cluster tools to have two loadlocks which is a valid assumption for most cluster tools currently in use.

RELATED WORK

Typical work center problems that are addressed in the literature are single machine work centers, parallel machines, batch machines, and machines with sequence-dependent setups. Solutions and approaches for many of these problems can be found in the book by Pinedo (Pinedo 2001).
An important objective of the optimization problem is the total weighted tardiness (TWT). The FORCe scheduling project aims at globally scheduling a complete factory using the Shifting Bottleneck heuristic (Fowler et al. 2002a, Fowler et al. 2002b). For the solution of the work center problems they use the BATCS rule which is an adaptation of the ATC rule (Apparent Tardiness Cost) for batch machines and setups (Fowler et al. 2002b, Pabst et al. 2002).
For rather simple cluster tools Perkinson et al (Perkinson et al. 1994, Perkinson et al. 1996) analyzed their throughput and cycle time behavior analytically. Others used simulation and optimized schedules with genetic algorithms (Dümmler 1999). Dümmler already introduces the notion of a slow-down factor, but did not use it for scheduling. We analyzed and simulated cluster tools to study slow-down factors ourselves (Niedermayer and Rose 2003, Niedermayer and Rose 2004).

FACTORY SCHEDULING

Scheduling deals with allocating resources for tasks over time. The goal is to find a solution that is optimal or near-optimal with respect to given objectives. Initially makespan was the main objective. The focus of manufacturing today is often more on costumer orders and on-time delivery, thus it is most important to satisfy all or at least the most important due dates.
Finding an optimal schedule is not always easy. In fact, most scheduling problems are NP-hard.

Proceedings 16th European Simulation Symposium
György Lipovszki, István Molnár © SCS Press, 2004
ISBN 1-56555-286-5(book) / ISBN 1-84233-106-x(CD)

Definition: *Lateness*

Let c_i be the completion time of lot i and d_i its due date, then the lateness L_i of lot i is $L_i = c_i - d_i$.

Definition: *Tardiness*

Let c_i be the completion time of lot i and d_i its due date, then the tardiness T_i of lot i is $T_i = \max\{0, c_i - d_i\ ) = \max\{0, L_i\}$.

Definition: *Total Weighted Tardiness (TWT)*

In addition to the due date each lot i is assigned a weight w_i that specifies its importance. Then the total weighted tardiness of all lots is defined as $TWT = \sum_i w_i T_i$.

In manufacturing it is common to use dispatching rules at the work centers or machines. Dispatching rules are myopic methods. Hence, they are usually not optimal for the solution of the global problem. It is likely that they are also not optimal regarding the local objective of interest.

PROBLEM GRAPH AND SCHEDULE GRAPH

Scheduling problems can be transformed into graph problems. Additionally, graphs give a visual representation of the problem.

The problem graph is a directed graph. It has a source (o) and sink (*) node, and there is a node for each operation. A detailed description of problem and schedule graphs can be found in the book by Ovacik and Uszoy (1997). Figure 1 shows the problem graph. It includes all potential arcs for determining a schedule. Operations at one machine or work center build a clique, except for operations of one job.

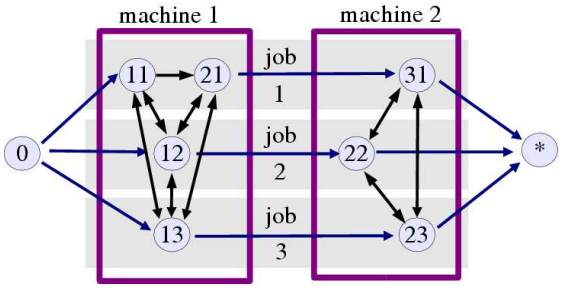

Figure 1: Problem graph

The schedule graph (Figure 2) is a directed acyclic graph. The arcs indicate the order of the operations. The longest path to an operation is the time at which the operation can start. The longest path from source to sink is the makespan. As one can see the scheduling problem quite naturally decomposes into subproblems at the work centers or machines. The graph specifies the release dates for the operations of the subproblem. When a one machine or work center is scheduled, these release dates for other work centers change. Hence, one solution for the scheduling problem can be to iteratively solve work center subproblems, adapt the graph and solve the next subproblem, and so on.

In this text we focus on the solution of such subproblems.

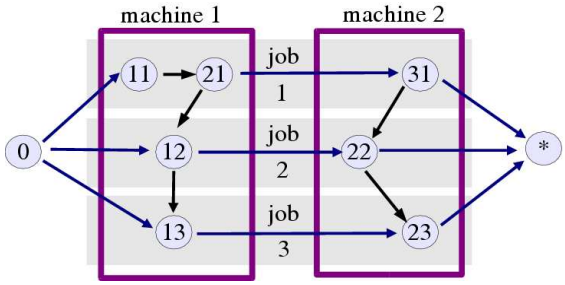

Figure 2: Schedule Graph

WORK CENTER SUB PROBLEMS

The jobs for each work center or machine have to be scheduled. A decomposition of the global scheduling problem usually consists of such problems. The same subproblem may be optimized several times with different release dates as arcs in the schedule graph change. The result is a sequence of operations in the input queue of the work center or machine. This may also include the assignment of an operation to a specific resource, e.g. to a particular machine in parallel machine problems.

Typical problems are single machine problems, parallel machine problems, batch machine problems, problems with sequence-dependent setups, and, as in our case, cluster tool problems.

CLUSTER TOOLS

Since the 1990s cluster tools are becoming a more and more integral element of wafer processing in semiconductor manufacturing. A cluster tool consists of a mainframe with several machines (chambers). A machine usually processes one wafer. A cluster tool has handlers to move the wafers from one chamber to another. The lots are stored in the loadlocks. A handler takes a wafer from a lot in the loadlock and moves it to the processing chamber as indicated by its recipe. A wafer may visit several machines until it is completed and brought back to its lot. A lot is completed and can leave the cluster tool when all its wafers are completed. Since there is vacuum inside the tool the loadlock has to be pumped after a lot enters and vented before the lot leaves the tool.

Figure 3 shows the model of an Endura cluster tool with 2 hand-over chambers. A typical configuration could be that the chambers in the section close to the loadlocks are used for pre-processing steps (alignment, heating) and post-processing steps (cooling) and the chambers of the other section are the machines for the long processing steps (Seidel 2001).

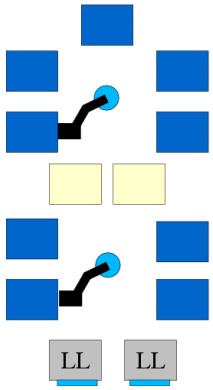

Figure 3: Model of an Endura cluster tool

For our scheduling problem cluster tools with one loadlock behave exactly like single machines (with lot-dependent processing times). Thus, we focus on parallel mode cluster tools with two loadlocks. Here, lots can overlap and since they use common resources they slow each other down during their overlap. Depending on the compatibility of the recipes the slow-downs can range from small (roughly 1) to large (more than 2). The result is a rather complex behavior with respect to the lot cycle time.

Definition: *Slow-down factor*

Let $CT(A, f)$ be the time needed for the fraction f of the work for lot A in the cluster tool when lot A is alone. Let f be the fraction of the work done for lot A during its overlap with lot B and $CT(A, A+B, f)$ be the length of this overlap. Then the slow-down factor is

$$SDF_{AB} = \frac{CT(A, A+B, f)}{CT(A, f)}$$

In previous papers we studied this slow-down factor and studied how to predict it (Niedermayer and Rose 2003, Niedermayer and Rose 2004). The slow-down factor may be different for different kind of overlaps, e.g. our simulator tends to prefer the lot that entered the tool first.

The particular tool cycle times for lots in a schedule can only be predicted for the complete schedule from the beginning to the time of interest. This can be done using simulation or approximation. An approximation can be computed as follows. We assume that we know the cycle times of all lots while they use the cluster tool exclusively (in single mode). This can be estimated once for each recipe by simulation or analytically. Additionally, we assume that we know the slow-down factors for the recipe combinations of interest. If a lot is processed completely during an overlap with another lot its cycle time can be approximated by

$$CT[A, A+B, 100\%] = SDF_{AB} CT[A, 100\%]$$

For lots that overlap with more than one lot or only partially overlap with one lot, the amount of work that was done during each overlap has to be determined and the cycle time can then be computed accordingly. Figure 4 shows the basic algorithm in pseudo code. The function TimeToNextEvent and the estimation of the amount of work done during an overlap need the slow-down factors and the single mode cycle time predictions for the lots.

```
ALGORITHM
INPUT: Queue with lots in the order given by the schedule, Time
WHILE Queue.notEmpty() and Clustertool.notEmpty() DO
  FOR all empty loadlocks DO
    IF Queue.nextLotReady(Time) THEN
      Add Q.next() to loadlock and set its start time.
    ENDIF
  ENDFOR
  LengthOfOverlap = TimeToNextEvent(all lots in
                                    clustertool);
   Time = Time + LengthOfOverlap
   FOR all lots in cluster tool DO
    Determine the amount of work done for lot during
      the overlap.
    IF lot is completed THEN Remove lot from
                loadlock and set its completion time.
  ENDFOR
ENDWHILE
```

Figure 4: Basic algorithm for computing lot cycle times of schedule for cluster tools

Compared to simulation this algorithm is very efficient because it only needs a few floating-point operations per lot.

From this description of cluster tools in parallel mode it becomes obvious that parallel mode cluster tool scheduling is a problem that is different from standard scheduling problems with batch machines or sequence-dependent setups, etc.

CLUSTER TOOL SIMULATION

For our studies we used the cluster tool simulator CluSim that was developed by Dümmler et al. (Dümmler 1999, Schmid 1999) at the University of Würzburg and is also used at Infineon Technologies for cluster tool optimization.

We used simulation for two purposes. The first one was to determine slow-down factors for all combinations of lots. Since the slow-down factor may vary depending on how lots overlap and which lot is processed first, we simulated different overlaps. We discussed this in more detail in a previous paper (Niedermayer and Rose 2003). This is important because in the next section we will use these slow-down factors for scheduling.

The second purpose of simulation is to evaluate the schedules computed by the different optimization approaches. For the evaluation scenarios were created, for each test set 20 scenarios with 20 lots, each lot with

weight, release date and due date. The lots were released in a way that the cluster tool did not run out of work. The results of the simulation runs are used to determine cycle times, makespan and tardiness. The cluster tool input queue in CluSim is a FIFO queue (First In First Out). To evaluate a particular schedule the release dates in the input file of the simulator have to be adapted to ensure the correct order. The release date of a lot following another lot has to be higher than the release date of its predecessor even though it is actually released before its predecessor.

Simulation is also important for the overall scheduling problem. Once the operations for a cluster tool are scheduled, the schedule can be simulated and the results can be used to specify the processing times in the schedule graph.

CLUSTER TOOL SCHEDULING

In this section we describe two approaches to schedule cluster tools. We assume that we have one input queue and that we can modify the order of the lots in the queue.

Our first approach is a dispatching rule based on the slow-down factors introduced in the last two sections.

The dispatching rule SDFavg works as follows. Let lot A be in the cluster tool. From all lots that are currently released and available in the input queue take the lot B that minimizes $\frac{SDF_{AB} + SDF_{BA}}{2}$.

To implement this rule we need to predict the cycle times of the lots already scheduled to determine the current time. This is necessary to identify the lots that are released, but not yet scheduled.

Our second approach is a Random Search approach. It is actually a variation of a genetic algorithm. It uses a population of individuals, elitism, and the mutation is based on permutation of lots. Schedules are evaluated with cycle time predictions using slow-down factors for the lot combination and single mode cycle time predictions for the lots. We did not search for the fastest search algorithm. We simply wanted to study which solution such a search algorithm can find given a considerable amount of time. Since the predictions differ from the actual cycle times an optimum of the Random Search is not necessarily an optimum of the real scheduling problem.

COMPARISON

In this section we compare SDFavg, the Random Search based on predictions with slow-down factors, FIFO and EDD (Earliest Due Date).

First, we examined the quality of the schedulers for the objective makespan. Table 1 shows that SDFavg outperforms FIFO and that SDFavg is roughly as good as the Random Search.

Table 1: Objective Makespan

Optimizer	Test set	Normalized Makespan
FIFO	Dresden 1	1.152
SDFavg	Dresden 1	1.036
Random Search	Dresden 1	1.032
FIFO	Dresden 2	1.129
SDFavg	Dresden 2	1.015
Random Search	Dresden 2	1.033
FIFO	Villach	1.135
SDFavg	Villach	1.045
Random Search	Villach	1.037
FIFO	Simple	1.293
SDFavg	Simple	1.031
Random Search	Simple	1.060

For the test sets Simple and Villach we also computed lower bounds for the makespan for each scenario using a work distribution algorithm. Since precedence constraints are ignored these bounds are true lower bounds. Any schedule has to be worse. Table 2 gives the details. On average SDFavg is rather close to this lower bound.

Table 2: Comparing the schedules with a true lower bound

Test set	Lower Bound (unrealistic)	SDFavg	FIFO
Villach	47396s	54749s	61822s
Simple	67169s	78687s	101641s

As second objective we used the total weighted tardiness (TWT). Table 3 shows that the cluster tool throughput is sequence-dependent and that in the schedules produced by EDD the throughput is reduced. As a consequence, a lot of lots are not completed on time. Our rule SDFavg on the other hand does not know anything about due dates, but since its schedules result in high throughput most lots are completed in time. The Random Search with objective TWT was usually slightly better.

We can conclude that dispatching rules that ignore cluster tool behavior produce poor throughput and may therefore fail to achieve their primary objective. Thus, combining them with SDFavg or similar approaches is necessary.

Table 3: Objective Total Weighted Tardiness

Optimizer	Test set	Normalized TWT
EDD	Dresden 1	1.70
SDFavg	Dresden 1	1.18
Random Search	Dresden 1	1.28
EDD	Dresden 2	1.53
SDFavg	Dresden 2	1.19
Random Search	Dresden 2	1.08
EDD	Villach	1.35
SDFavg	Villach	1.32
Random Search	Villach	1.06
EDD	Simple	1.46
SDFavg	Simple	1.15
Random Search	Simple	1.10

LOCAL IMPROVEMENTS

To avoid long individual cycle times or to avoid poor overall throughput it is useful to have the option to let one lot wait and thus to avoid certain combinations of lots.

To do this it is necessary to have combination characteristics that indicate for the scheduler whether a combination should be avoided. Again, slow-down factors are a solution. To avoid long individual cycle times one could avoid lot combinations for which the slow-down factor of one lot is larger than a given threshold, say 2.5. To avoid poor overall performance one could avoid a lot combination when the average slow-down factors are larger than a particular threshold, say 2.0.

CONCLUSIONS

We demonstrated that dispatching rules based on slow-down factors are an interesting and promising approach for scheduling cluster tools. This approach takes the special characteristics of cluster tools into account and avoids poor throughput and long cycle times. It is also efficient and schedules can be computed quickly.

Larger studies are to be made. In particular, the impact of prediction errors has to be analyzed more deeply. Future work may also include the adaptation of the ATC rule for the use with cluster tools and to include slow-down factors. One might also think of combining EDD or Critical Ratio with a rule based on slow-down factors for local optimization. Another option is to analyze and use other, maybe simpler, lot compatibility measures than slow-down factors. Another field of research could be to find other problems where such an approach might be useful.

REFERENCES

Dümmler, M. 1999. “Using simulation and genetic algorithms to improve cluster tool performance.” In Proceedings of the 1999 Winter Simulation Conference. 875-879.

Fowler, J., Carlyle, M., Runger, G., Gel, E., Mason, S., Rose, O. “A New Approach for Scheduling Semiconductor Wafer Fabs.” Semiconductor Fabtech, 15th Edition, pp. 39-41, 2002.

Fowler, J., Brown, S., Carlyle, M., Gel, E., Mason, S., Mönch, L., Rose, O., Runger, G., Sturm, R. ”A Modified Shifting Bottleneck Heuristic for Scheduling Wafer Fabrication Facilities.” In Proceedings of the FAIM 2002, July 15-17, Dresden, Germany, pp. 1231-1236, 2002.

Niedermayer, H. and Rose, O. “A Simulation-based Analysis of the Cycle Time of Cluster Tools in Semiconductor Manufacturing” In Proceedings of the 15th European Simulation Symposium, Delft, Netherlands, 2003.

Niedermayer, H. and Rose, O. “Approximation of Cycle Time of Cluster Tools in Semiconductor Manufacturing” In Proceedings of the Annual IIE Industrial Engineering Research Conference, Houston, Texas, 2004.

Ovacik, I.M. and Uzsoy, R. 1997. “Decomposition Methods for Complex Factory Scheduling Problems” Kluwer Academic Publishers.

Pabst, D., Fowler, J., Pfund, M., Mason, S., Rose, O., Mönch, L., Sturm, R. “Deterministic Scheduling of Wafer Fab Operations.” In Proceedings of the Brooks Worldwide Automation Symposium 2003, Oct 2003.

Perkinson, T.L., McLarty P.K., Gyurcsik, R.S., Cavin III., R.K. 1994. "Single-Wafer Cluster Tool Performance: An Analysis of Throughput." In IEEE Transactions on Semiconductor Manufacturing Vol. 7, August 1994.

Perkinson, T.L., McLarty P.K., Gyurcsik, R.S., Cavin III., R.K. 1996. ""Single-Wafer Cluster Tool Performance: An Analysis of the Effects of Redundant Chambers and Revisitation Sequences on Throughput." In IEEE Transactions on Semiconductor Manufacturing Vol. 9, August 1996.

Pinedo, M. 2001. “Scheduling. Theory, Algorithms, and Systems. 2nd edition. Prentice-Hall.

Schmid, M. 1999. “Modellierung und Simulation von Cluster Tools in der Halbleiterfertigung.” Master’s thesis. Department of Computer Science. University of Würzburg, Germany.

Seidel, G. 2001. “Simulation und Optimierung von Cluster Tools in der Halbleiterfertigung”. Master’s thesis, Institute of Mathematics. Technical University of Graz, Austria.

AUTHOR BIOGRAPHIES

HEIKO NIEDERMAYER is Ph.D. student at the Department for Computer Networks and Internet at the University of Tübingen. He received an M.S. degree in Computer Science from the University of Würzburg. His e-mail address is: `niedermayer@informatik.uni-tuebingen.de`.

OLIVER ROSE is assistant professor in the Department of Computer Science at the University of Würzburg, Germany. He received an M.S. degree in applied mathematics and a Ph.D. degree in computer science from the same university. He has a strong background in the modeling and performance evaluation of high-speed communication networks. Currently, his research focuses on the analysis of semiconductor and car manufacturing facilities. He is a member of IEEE, ASIM, and SCS. His web address is: `www3.informatik.uni-wuerzburg.de/~rose`.