Proceedings of the 2007 Winter Simulation Conference
S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, eds.

PREDICTING CLUSTER TOOL BEHAVIOR WITH SLOW DOWN FACTORS

Robert Unbehaun
Oliver Rose

Institute of Applied Computer Science
Dresden University of Technology
Dresden, 01062, GERMANY

ABSTRACT

Cluster tools are representatives of a special kind of tool where process times of jobs depend on the combination in which they are processed together on the tool and hence, depending on the sequence in which they are processed at a tool. To evaluate schedules of jobs to be processed at such a tool an estimation method is needed since a detailed simulation takes too long. In this paper, we present a method based on slow down factors which produces promising results and gives hints for the development of intelligent scheduling methods for this kind of tools.

1 INTRODUCTION

Cluster tools are special integrated tools for wafer processing in semiconductor manufacturing. They are used to maximize throughput and reduce lot cycle times at the cost of a very complex behavior. Since wafers with different types of process steps can circulate in a cluster tool simultaneously it can be regarded as a job shop environment. Cluster tools work under vacuum conditions inside the tool which means very little particles that could possibly contaminate wafers. As a consequence, the clean-room quality outside the tool is allowed to be lower than in traditional fabs. Cluster tools are the first representative we found so far for a special kind of scheduling problem which we refer to as sequence-dependent process times.

In the following sections, we will first describe how cluster tools look like and what specifics in behavior there are. Section 3 gives a short literature review. Section 4 will describe the problem of sequence-dependent setup times. Section 5 explains the idea of slow down factors. Section 6 shows the functionality of the matrix-prediction method with some experimental results in chapter 7. Chapter 8 presents the use of a slow down factor matrix for future scheduling approaches and finally chapter 9 will give a short conclusion.

2 COMPONENTS OF CLUSTER TOOLS

The basic components of a cluster tool are:

- A vacuum mainframe with one or two wafer handling robots
- Several processing chambers, where some of them can be dedicated to identical processes and hence used in parallel
- Two load locks to pump to vacuum or vent to atmos-pheric conditions
- Optionally there can be transfer chambers if there is more than one wafer handling robot
- An equipment front end module (EFEM) with an at-mospheric wafer handling robot and several load ports, which is attached to the load locks (see Figure 1)

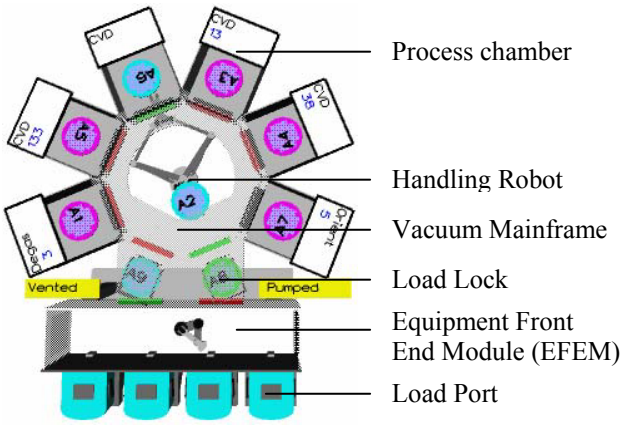

Figure 1: Example cluster tool

We assume that lots always contain 25 wafers. Each wafer of a lot has to take the same process steps in the cluster tool. This sequence of process steps is usually referred to as "recipe". In the following examples lots are denoted with capital letters which also represent their recipe. If two lots have the same letter, all wafers of these lots follow the same recipe.

A typical product flow in a cluster tool starts with loading lots into one of the load ports. After that single wafers are consecutively transferred from the load port to the load lock by the atmospheric robot. Then the load lock will pump to achieve vacuum conditions. Now the main frame robot can transfer the wafer to its destination chamber where it is supposed to be processed. The next step depends on whether the wafer shall leave the system or is requested to be processed in another chamber according to its recipe. After the last process step the wafer will be guided through the load lock back to the load port.

With more than one load port occupied the controller of the cluster tool will always process lots of the same recipe sequentially one after another and lots of different recipes in parallel.

Usually the main frame robot is a dualblade robot with the two blades either on the same side or opposite each other. Advantages compared to single blade robots are reduced wafer transfer times and with regard to multiple product flows a reduced amount of possible deadlocks as well.

3 RELATED WORK

A lot of cluster tool research is focused on issues of inside cluster tool scheduling and cluster tool controller dispatch rules as well as cluster tool simulation. Basic performance analysis and model development has been done in Perkinson et al. (1994) and Perkinson et al. (1996). Atherton (1995) gives a detailed introduction into cluster tools. Joo and Lee (1994) present a simulation framework with a virtual cluster tool controller included to reduce times for verfication of algorithms and behavior of real cluster tool controllers.

LeBaron and Domaschke (2005) compare different dispatch heuristics for the cluster tool controller on differently configured cluster tool models using the commercial simulator ToolSim for evaluation.

Niedermayer and Rose (2003) analyze cycle time delays occuring when lots of different recipes are processed in parallel in a cluster tool and present a method for lot cycle time prediction through slow down factors for cluster tools with two load locks. Our paper can be regarded as sequel to this work.

4 SEQUENCE-DEPENDENT PROCESS TIMES

To explain the problem of sequence-dependent process times, we assume a given set of lots. The only important parameter for each lot shall be its recipe denoted by capital letters. So a given sequence could be, e.g., ABBCBACA. There are different types of recipes. The recipe can require the wafers to be processed in all of the existing process chambers of a cluster tool or just in one of the chambers or a mixture of these types. As a consequence, each recipe requires a certain amount of resources in the cluster tool, where the set of required resources S of each recipe may overlap with another.

$$S_A \cap S_B \neq \varnothing \tag{1}$$

We process the queue on a cluster tool with two load ports. Here, it is possible to process two lots at the same time. There will be an improvement in overall makespan ($\Delta C_{\max} < 0$) through parallel processing but resource conflicts will extend the cycle time of each lot ($\Delta C_A, \Delta C_B > 0$) (see Figure 2).

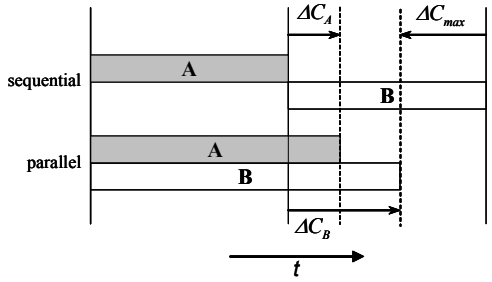

Figure 2: Comparison Single Mode vs. Parallel Mode

In contrast to problems of sequence dependent setup times, where the time for changing between two types of jobs can be exactly determined in advance, in case of cluster tools the problem is, that the process time is depending on the combination of recipes of the lots at the load ports. This combination will be referred to as load port recipe combination (LRC) in the following. Furthermore, the combination may change dynamically over time with lots slowed down to a different extend in the middle of their processing since resource conflicts may change depending on the recipes. Thus, solutions for problems of sequence dependent setup times can not easily be adapted for our case.

5 SLOW DOWN FACTORS

Slow down factors are one way to define the amount of time a lot of a certain recipe is slowed down when processed together with other recipes. The procedure is already known in the literature but only for cluster tools with a two

load lock configuration without EFEM. Our investigations include also cluster tools with EFEM and a number of load ports greater than two.

Definition 1 (Niedermayer and Rose 2003)

The slow down factor of lot A while processed in parallel with lot B is defined as

$$SDF(A, A+B) = \frac{CT(A, A+B)}{CT(A)} \tag{2}$$

where $CT(A, A+B)$ is the cycle time of lot A when processed in parallel with lot B and $CT(A)$ represents the cycle time of lot A when processed alone.

Thus, if there are more than two load ports available at the cluster tool slow down factors like $SDF(A, A+B+C+D)$ or $SDF(A, A+B+A)$ are possible as well. With the last example there is a problem occurring in terms of whether the slow down factors of both of the lots of type *A* are the same or not. As mentioned in Section 2, lots of the same recipe are processed sequentially and thus the second lot of *A* will stay longer in the system than the first and therefore have another slow down factor.

To determine the slow down factors, a cluster tool simulator is required. For our simulation experiments, we use a cluster tool simulator named ToolSim from Brooks Automation, Inc., which is already established and used by the industry for several years. With this simulator we are able to model most up-to-date cluster tool equipments in a sufficiently detailed manner.

Once the slow down factors are calculated, they can be used for the prediction of cycle times and completion dates for any queue of lots to be processed on a cluster tool with a given configuration.

6 MATRIX PREDICTION METHOD

In Unbehaun and Rose (2006) we already presented two methods for the prediction of cycle times and completion dates but stated that these were not sufficiently accurate und further investigation needed to be done.

The basis of the new method is a matrix containing the slow down factors of all possible recipe combinations (see Table 1).

Table 1: Example SDF-Matrix (3 Recipes, 3 Load Ports)

SFID\Recipe	A	B	C
A	1.0	0	0
B	0	1.0	0
C	0	0	1.0
AB	1.3	1.7	0
AC	1.5	0	1.6
BC	0	1.2	1.3
ABC	1.8	1.7	1.9

The slow down factors have to be determined for all load port recipe combinations which solely contain the recipes given with the corresponding SDF, e.g., SDF(AB) will be used for all LRCs just containing recipes *A* and *B* like (*A*+*A*+*B*) or (*B*+*A*+*B*) etc.. So, if the LRC is (*A*+*A*+*B*) each lot of *A* will be slowed down by the factor 1.3 and lot *B* by the factor 1.7. These values represent an average which is obtained by calculating the SDFs for each single possible LRC containing the corresponding recipes.

Given this matrix and the raw process times of each recipe, when processed alone in the cluster tool, the algorithm works as follows:

Step1:
Initially assign lots to load ports until all load ports are occupied (assuming that enough lots are in the queue).

Step 2:
Calculate the completion date of each assigned lot *i* by

$$CT_i = RPT_i \cdot SDF_i \,. \tag{3}$$

Step 3:
Set *T* equal to the smallest completion date. If there are still lots in the queue replace the corresponding completed lot with the next from the queue and decrease the queue size. Otherwise, remove the lot and continue.

Step 4:
If the recipe of the new lot is unique in the LRC then calculate its completion date through

$$CD_i = T + RPT_i \cdot SDF_i \tag{4}$$

Else, find the lot of this recipe in the LRC which is waiting the longest time and set this completion date according to Equation 4. If no lot was introduced in Step 3 check for unique recipes with unassigned completion dates and apply Equation 4 as well.

Step 5:
If the LRC changed adjust the remaining completion dates of lots with recipes different to the one of the new lot through

$$CD_{i,new} = T + \frac{CD_{i,old} - T}{SDF_{i,old}} \cdot SDF_{i,new} \tag{5}$$

with $CD_{i,old}$ as the former completion date used so far, $SDF_{i,old}$ as the former slow down factor and $SDF_{i,new}$ as the slow down factor for the new load port recipe combination.

Step 6:
If lots are still in progress repeat from Step 3. Stop otherwise. The makespan C_{max} is equal to the completion date of the latest finished lot.

With raw process times given in Table 2, Table 3 illustrates the computation of the makespan $C_{\max}$ of a schedule. The queue to be processed is *ABCCABCABCCB*.

Table 2: Example raw process times

Recipe r_i	Raw Process Time RPT(r_i)
A	10
B	20
C	30

Table 3: Example calculation

Time stamp T	Load port assignment		
	A	B	C
$T = 0$	18	34	57
	C_2	B	C_1
$T = 18$	x	29,3	44,7
	C_2	A	C_1
$T = 29,3$	x	44,3	48,3
	C_2	B	C_1
$T = 44,3$	x	68,3	47,6
	C_2	B	C_3
$T = 47,6$	86,6	68,3	x
	C_2	A	C_3
$T = 68,3$	90,8	83,3	x
	C_2	B	C_3
$T = 83,3$	89,4	107,3	x
	C_4	B	C_3
$T = 89,4$	x	107,3	128,4
	C_4	C_5	C_3
$T = 107,3$	x	x	123,5
	C_4	C_5	B
$T = 123,5$	162,5	x	147,5
	C_4	C_5	x
$T = 147,5$	159	x	x
	x	C_5	x
$T = 159$	x	189	x
	x	x	x
$T = 189$	x	x	x

Initially, we take the first three lots, assign them to the three load ports and calculate their completion date (3). Then, we proceed to the next completion date $T = 18$ and replace lot A with lot C_2 from the queue. Since there already exists a lot of recipe C the new lot will have to wait for processing until the completion of this lot (C_1). Due to the recipe combination changes in this step, the completion dates of all lots in progress have to be adjusted (5). When the queue is empty the remaining lots in the load ports will be finished and the algorithms terminates. We obtain makespan of $C_{max} = 189$.

7 EXPERIMENTAL RESULTS

In this section, we present a variety of results comparing our new prediction approach to simulation.

Two configurations were chosen. For both a cluster tool with three load ports, three process chambers and a single handling robot is used, which is only able to transfer one wafer at a time. In the first test case, the process chambers are dedicated to different tasks and the wafers of all recipes need to pass all chambers in the same order (sequential configuration). In the second case, the process chambers are used in parallel and the wafers of each recipe only need to be processed in one of the chambers (parallel configuration). For both configurations the recipes differ by the time they need to be processed in a chamber. A queue of 40 lots is processed with ten independent runs for each configuration.

Table 4: Results

	Sequential	**Parallel**
Avg. $C_{\max}$ Error	8.8%	7.3%
Avg. CD Error	2.7%	4.0%
Avg. CT Error	10.3%	13.9%

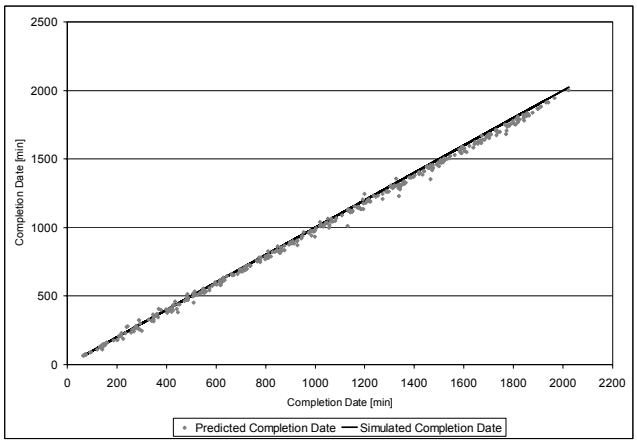

Figure 3: Predicted completion dates - sequential configuration

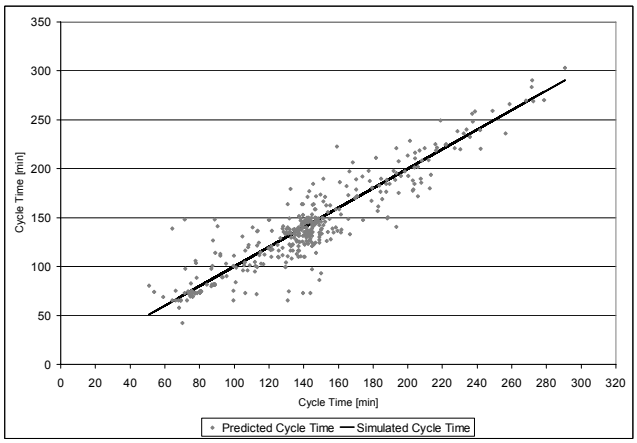

Figure 4: Predicted cycle times - sequential configuration

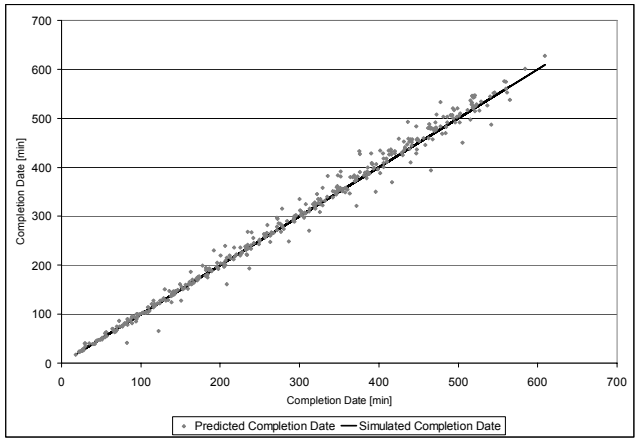

Figure 5: Predicted completion dates - parallel configuration

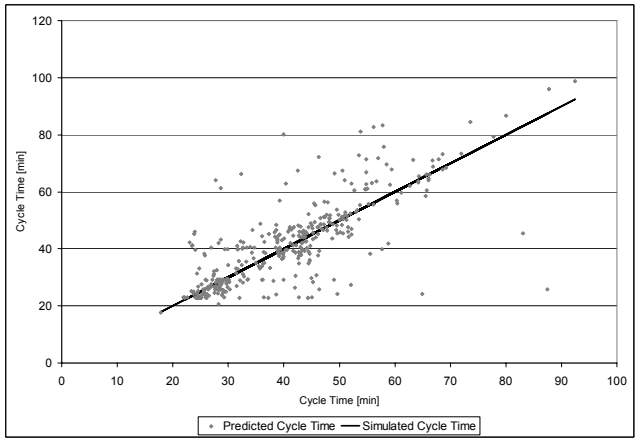

Figure 6: Predicted cycle times - parallel configuration

In Table 4, we present the prediction results for the sequential as well as the parallel test configuration. The two configurations were compared by means of the average deviation from the objective value C_{max} and the average deviation of the completion date (CD) and cycle time (CT) of every single lot from the simulated values.

The average estimation errors of makespan and completion dates are below ten percent in both cases and the average cycle time deviation is below fifteen percent. We intend to use our new approach for further analysis of the problem described in Section 4, because the results are considerably better than those of the methods presented in Unbehaun and Rose (2006). The reason for the improved performance is mainly the introduction of additional system knowledge into the prediction method. The consideration of same lots being processed sequentially and different lots being processed in parallel leads to better results. Nonetheless more test runs need to be made to show the robustness of the approach with respect to configuration changes.

As depicted in Figures 3 and 5 , approximated completion dates can be very close to the simulated values, while Figures 4 and 6 show that cycle times are not estimated very accurately for all lots. Since positive and negative deviations compensate each other in both configurations, completion dates are effected to a smaller extend. Still, the precision of completion date prediction suffers if raw process times of different recipes vary considerably. But this is only an academic case, since differences of raw process times of, e.g., several hours between recipes would lead to unnecessary long waiting times for certain lots and a processing on separate machines should be considered then. However, the impact of cycle time deviations on completion date estimates needs to be investigated further.

8 USING SLOW DOWN FACTORS FOR SCHEDULING

Given a matrix of slow down factors, an analysis of useful load port recipe combinations - based on the assumption that reduced cycle times will lead to a reduced makespan - can be applied.

The basic idea is that, any order of processing will not yield a makespan worse than sequential processing of the lots one after another, because in this case parallel processing will not provide any benefit. Thus, this particular makespan represents an upper bound which can be used as reference to implement, e.g., a Branch and Bound algorithm. In addition, it is possible to evaluate single LRCs of a slow down factor matrix. Similar to the approach for a whole schedule a certain lot combination is useful if parallel processing of the corresponding recipes is faster than sequential processing. The cycle time for sequential processing CT_s is defined as

$$CT_s = \sum_{j \in SFID} \left(RPT\left(r_j\right) \right) \tag{6}$$

where *SFID* represents the recipes contained in the slow down factor combination under consideration and $RPT\left(r_j\right)$ the raw process time of recipe *j* when processed alone in the cluster tool.

The cycle time for parallel processing CT_p is defined as

$$CT_p = \max_{j \in SFID} \left(SDF\left(r_j \mid SFID\right) \cdot RPT\left(r_j\right) \right) \tag{7}$$

where $SDF\left(r_j \mid SFID\right)$ is the slow down factor of recipe *j* in the corresponding combination *SFID* and $RPT\left(r_j\right)$ the raw process time of recipe *j*.

Now, for example, with the slow down factor matrix given in Table 1 and raw process times given in Table 2, we obtain the following result (Table 5).

As a result, only some of the given combinations can be expected to reduce cycle time through parallel lot processing. It also turns out, that with more different recipes processed together the parallel processing tends to be fa-

vorable. But this also depends on the increase of the slow down factors.

Table 5: Example Calculation

SFID	CT_s	CT_p
A	10	10
B	20	20
C	30	30
AB	30	34
AC	40	48
BC	50	39
ABC	60	57

9 CONCLUSIONS

In this paper, we presented the problem of sequence-dependent process times and cluster tools as a possible application area. Moreover, we provided a method to estimate makespans, completion dates and cycle times for these kind of problems. Our approach can also be used to evaluate schedules and, hence, provides the basis for developing scheduling algorithms.

We also presented first thoughts which might help to find an intelligent scheduling strategy. In this context, the knowledge about load port recipe combinations which are useful for parallel processing allows us to save cycle time and helps us to reduce the makespan.

REFERENCES

Atherton, L.F., and R.W. Atherton. 1995. *Wafer Fabrication: Factory Performance and Analysis*. Kluwer.

Joo, Y.-J., and T.-E. Lee. 2004. Virtual Control: A Virtual Cluster Tool for Testing and Verifying a Cluster Tool Controller and a Scheduler. *IEEE Robotics & Automation Magazine*, 11(3), 33-49.

LeBaron, T., and J. Domaschke. 2005. Optimizing Robot Algorithms with Simulation. In *Proceedings of the 2005 Winter Simulation Conference*. 2211-2217.

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

Perkinson, T.L., P.K. McLarty, R.S. Gyurcsik, and R.K. Cavin III. 1994. Single-Wafer Cluster Tool Performance: An Analysis of Throughput. *IEEE Transactions on Semiconductor Manufacturing*, 7 (3), 369-373.

Perkinson, T.L., P.K. McLarty, R.S. Gyurcsik, and R.K. Cavin III. 1996. Single-Wafer Cluster Tool Performance: An Analysis of the Effects of Redundant Chambers and Revisitation Sequences on Throughput. *IEEE Transactions on Semiconductor Manufacturing*, 9 (3), 384-400.

Unbehaun, R., and O. Rose. 2006. The Use of Slow Down Factors for the Analysis and Development of Scheduling Algorithms for Parallel Cluster Tools. In *Proceedings of the 2006 Winter Simulation Conference*. 1840-1847.

AUTHOR BIOGRAPHIES

ROBERT UNBEHAUN is a PhD student and member of the scientific staff of Prof. Dr. Oliver Rose at the Chair of Modeling and Simulation of the Dresden University of Technology, Germany. He received a M.S. degree in computer science from Ilmenau University of Technology, Germany. His research interests include optimization approaches for complex manufacturing environments through development and analysis of scheduling algorithms for material flow control. His e-mail address is <robert.unbehaun@inf.tu-dresden.de>.

OLIVER ROSE holds the Chair for Modeling and Simulation at the Institute of Applied Computer Science of the Dresden University of Technology, Germany. He received an M.S. degree in applied mathematics and a Ph.D. degree in computer science from Würzburg University, Germany. His research focuses on the operational modeling, analysis and material flow control of complex manufacturing facilities, in particular, semiconductor factories. He is a member of IEEE, INFORMS Simulation Society, ASIM, and GI. His web address is <http://www.iai.inf.tu-dresden.de/ms>