CHEMICAL ENGINEERING TRANSACTIONS
VOL. 39, 2014
A publication of AIDIC
The Italian Association of Chemical Engineering
www.aidic.it/cet
Guest Editors: Petar Sabev Varbanov, Jiří Jaromír Klemeš, Peng Yen Liew, Jun Yow Yong

ISBN 978-88-95608-30-3; **ISSN** 2283-9216
DOI:10.3303/CET1439073

Scheduling of Automated Wet-Etch Stations

Máté Hegyháti, Olivér Ősz, Balázs Kovács, Ferenc Friedler*

Department of Computer Science and Systems Technology, University of Pannonia, Egyetem u. 10, H-8200, Veszprém, Hungary
friedler@dcs.uni-pannon.hu

Semiconductor manufacturing has recently attracted an increased attention in process optimization research. Various MILP and constraint programming methods were published to solve the scheduling problem arising in the wet-etching systems. In the present work, improvements are given to these state-of-the-art models, furthermore, the general purpose solver of the S-graph framework (Sanmartí et al. 2002) is applied to wet-etch station scheduling.

1. Introduction

Expansion of the semiconductor industry demands new planning and scheduling approaches to shorten production time. Fabrication of the silicon wafers is a critical stage of the manufacturing process with etching as its main step. There are two types of etching processes: wet-etching and plasma etching. During wet-etching, layers are chemically removed from the wafer surface. Several identical wafers are handled as one wafer lot by the automated wet-etch station (AWS). This system comprises several chemical and water baths and one or more transporting robots. To avoid damaging the wafers, the given etching times must be followed without deviation.

The aim of the scheduling is to minimize the makespan, the total production time of the given products. Most of the production recipes are permutation flow-shops: the lots go through the same stages in the same sequence without repetition and there is only one bath per stage. In this scenario, the order of the lots is the same in every bath. More general recipes allow products to have re-entrant paths, different task sequences (multipurpose) or multiple baths on stages (flexible flow-shop). Another parameter of the problem is the number of robots.

In the original problem, zero-wait (ZW) and unlimited wait (UW) stages alternate, representing the etching in chemical baths and the rinsing in water or deionizing baths. In a recent study, Aguirre et al. (2013) considered limited-wait (LW) policy for both stages to provide higher level of flexibility for storage policies.

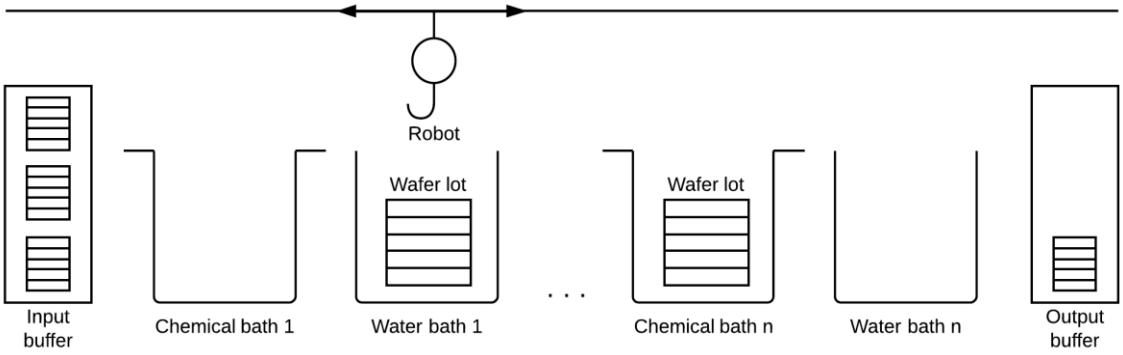

Figure 1: Structure of an AWS

Please cite this article as: Hegyháti M., Ősz O., Kovács B., Friedler F., 2014, Scheduling of automated wet-etch stations, Chemical Engineering Transactions, 39, 433-438 DOI:10.3303/CET1439073

The first MILP models in AWS scheduling used slot-based approaches to solve the one robot permutation flow-shop problem (Bhushan and Karimi 2003). Karimi et al. (2004) improved the model by redefining binary variables in a more efficient way. Aguirre et al. (2011) presented a multiple robot model for the problem based on general precedence.
Castro et al. (2012) developed a model based on hybrid time slots for one or more robots. This method drastically reduces computational need by cutting down the number of big-M constraints.
Constraint programming methods (Zeballos et al., 2011) were the first to consider the empty robot movement, and presented a solution for one robot. In addition to the offline case, an online scheduling approach is given by Novas and Henning (2012).
Neglecting the empty robot movement may result in an infeasible schedule. Aguirre et al. (2013) developed a novel precedence based model to solve more general hoist scheduling problems. Besides considering the empty movements even for multiple robots, the method included LW storage policies, robot zones, re-entrant flows, multiple identical baths and different recipes for products.
Section 2 of this paper presents some enhancements of the state-of-the-art MILP models. In Section 3, a novel S-graph based AWS scheduling method is introduced. The comparison and evaluation of the improved MILP model and the S-graph approach is carried out via an extensive empirical analysis, the results of which are summarized in Section 4. Section 5 introduces ideas to enhance the S-graph approach by exploiting AWS specific problem characteristics.

2. Enhancement of the mathematical programming methods

2.1 Improved precedence based formulation

The proposed model is based on the precedence based model of Aguirre et al. (2013), which is an extensively reformulated and extended version of the former model of Aguirre et al. (2011). To consider the empty robot movement, Aguirre et al. (2013) introduced immediate precedence sequencing variables along with some additional variables and constraints.
Empty robot movements can, however, be addressed appropriately without these additional variables as well. In our proposed model, constraints (11-20) of Aguirre et al. (2013) are replaced by the constraints (A11') and (A12') shown below, which are the modified versions of the original constraints (11) and (12). Moreover, the immediate precedence and some other variables that appeared only in the replaced constraints can be neglected as well.

$$\begin{aligned} &Ts_{i,s} \geq Ts_{i',s'} + \pi^{load}_{i,s} + \pi^{abs}_{j,j'} - M_T(1 - Y_{i,i',s,s'}) \\ &- M_T(2 - w_{i,s-1,j} - w_{i',s',j'}) - M_T(2 - q_{i,s,r} - q_{i',s',r}) \\ &\forall i, i' \in I : (i > i'), s \in S_i : (s > 1), s' \in S_{i'}, j \in J_{i,s-1}, j' \in J_{i',s'}, r \in R_{i,s} \cap R_{i',s'} \end{aligned} \tag{A11'}$$

$$\begin{aligned} &Ts_{i',s'} \geq Ts_{i,s} + \pi^{load}_{i',s'} + \pi^{abs}_{j,j'} - M_T(Y_{i,i',s,s'}) \\ &- M_T(2 - w_{i,s,j} - w_{i',s'-1,j'}) - M_T(2 - q_{i,s,r} - q_{i',s',r}) \\ &\forall i, i' \in I : (i > i'), s \in S_i, s' \in S_{i'} : (s' > 1), j \in J_{i,s}, j' \in J_{i',s'-1}, r \in R_{i,s} \cap R_{i',s'} \end{aligned} \tag{A12'}$$

These two constraints together with constraints (1-10) of Aguirre et al. (2013) provide a smaller and more efficient model to tackle the same set of problems as it has been shown in Section 1.

2.2 Extended hybrid time slot model for the AWS

The slot based method of Castro et al. (2012) performs considerably better than other investigated approaches for the permutation flow-shop problems. However, its incapability of handling empty robot movement limits its applicability. Ignoring the time needed for empty robot movement in the mathematical model may cause a practically infeasible schedule for the original problem. By a simple modification, the model can be extended to consider empty robot movement as well.
In their model, constraints (32), (33) and (35-38) addressed the timing of robots. The extension of the constraints are given in (C32'), (C33') and (C35-38') to solve the issue of empty robot movements.

$$\begin{aligned} &T_{t,m} \geq T_{t',m'} + \pi_m - H(1 - Y_{t,m,t',m'}) - H(2 - W_{t,m,r} - W_{t',m',r}) \\ &\forall t, t' \in T : (t' > t), m, m' \in M : (m \neq m'), r \in R \end{aligned} \tag{C32'}$$

$$\begin{aligned} &T_{t',m'} \geq T_{t,m} + \pi_{m'} - H(Y_{t,m,t',m'}) - H(2 - W_{t,m,r} - W_{t',m',r}) \\ &\forall t, t' \in T : (t' > t), m, m' \in M : (m \neq m'), r \in R \end{aligned} \tag{C33'}$$

$$T_{t+1,m} \geq T_{t,m} + \sum_{i \in I} N_{i,t} p_{,i,m} + \pi_{m+1} + \pi_m \quad \forall m \in ZW, t \in T : (t \neq |T|) \tag{C35'}$$

$$T_{t,m} + \pi_{m+1} + \pi_m \quad \forall m \in LS, t \in T : (t \neq |T|) \tag{C36'}$$

$$T_{t,m} \geq T_{t',m'} + \pi_m - [H(1 - Y_{t,m,t',m'})]|_{(t,m,t',m') \notin YE1} \quad \forall (t,m,t',m') \in YD0 \tag{C37'}$$

$$T_{t',m'} \geq T_{t,m} + \pi_m - [H(Y_{t,m,t',m'})]|_{(t,m,t',m') \notin YE0} \quad \forall (t,m,t',m') \in YD1 \tag{C38'}$$

3. S-graph based approach for the AWS scheduling problem

The S-graph framework was developed for scheduling multipurpose batch processes (Sanmartí et al., 2002). The approach relies on a directed graph based mathematical model and a branch-and-bound optimization procedure. In the model, called the S-graph, nodes represent tasks and finished products. The weights of the arcs between tasks are lower bounds of the time difference between their starting times. The branch-and-bound algorithm assigns tasks to equipment units and determines their order by adding so-called scheduling arcs to the graph. In the resultant graph, the weight of the longest path gives the makespan of the schedule.

3.1 Representing AWS scheduling problems with S-graphs

In order to solve AWS scheduling problems with the S-graph framework, an analogy is given between the components of the AWS and batch processes:

- Wafer lots → products
- Etching steps → tasks
- Cleaning steps → tasks
- Transfers → tasks
- Baths → equipment units
- Robots → equipment units

Transfer tasks can be performed by the robot units with the processing times equal to the corresponding transfer times. In addition to the aforementioned reformulation, the S-graph algorithms had to be extended to tackle the ZW or LW policies of etching steps. This was implemented by adding arcs in the opposite direction of the recipe arcs with the negative weight of the sum of the processing and maximal waiting time. The zero-weighted cycles arising from these arcs are permitted unlike the ones with only scheduling arcs which indicate cross-transfer (Hegyháti et al., 2009).

Using the described analogy, the S-graph algorithms from the literature are capable of solving wide-range of AWS scheduling problems, even the ones featuring multiple baths per stage, re-entrant flow, LW policy, and robot movement zones.

3.2 Illustrative example

The described S-graph representation of AWS problems is illustrated by a permutation flow-shop example for 3 products with a single ZW chemical stage, an UW water stage, and one robot. Figure 2 shows the structure of the example along with processing and transfer times. The S-graph model of this illustrative example is shown in Figure 3.

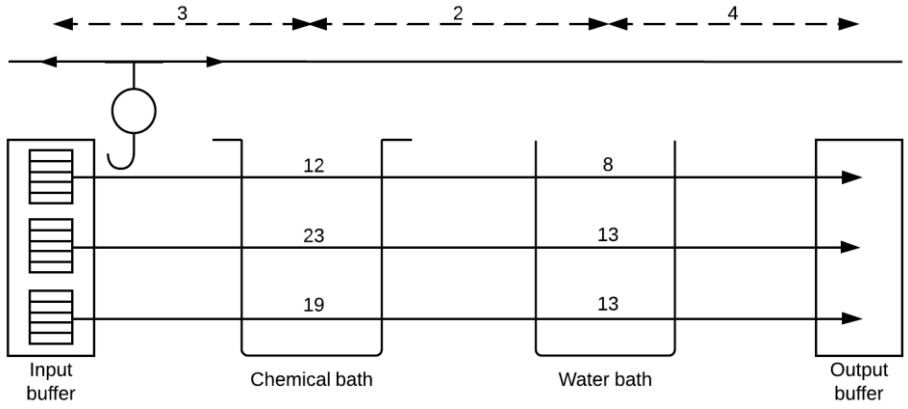

Figure 2: Structure of the illustrative example

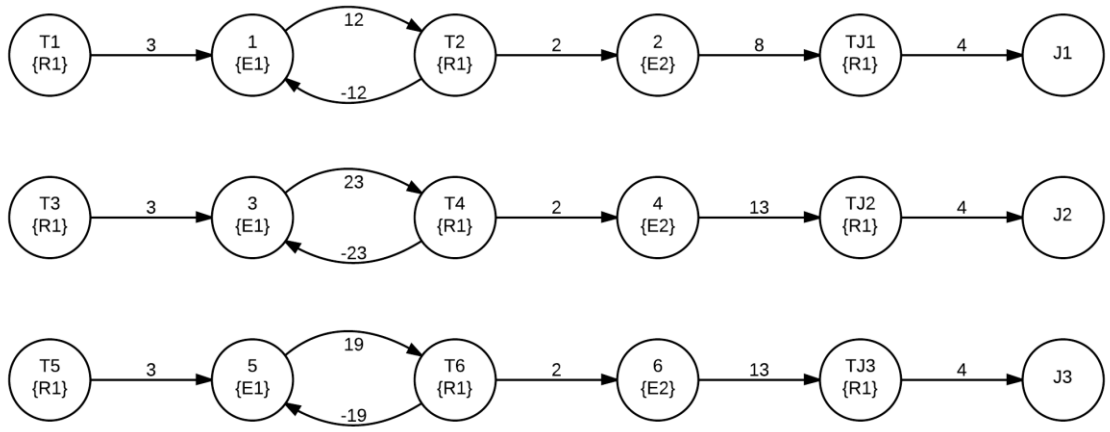

Figure 3: Recipe graph of the example

The makespan minimization algorithm of the S-graph framework (Sanmartí et al., 2002) generates the optimal schedule shown in Figure 4. The highlighted longest path indicates the optimal makespan of 83 time units. This graph can be converted to a commonly used Gantt-chart representation, see Figure 5.

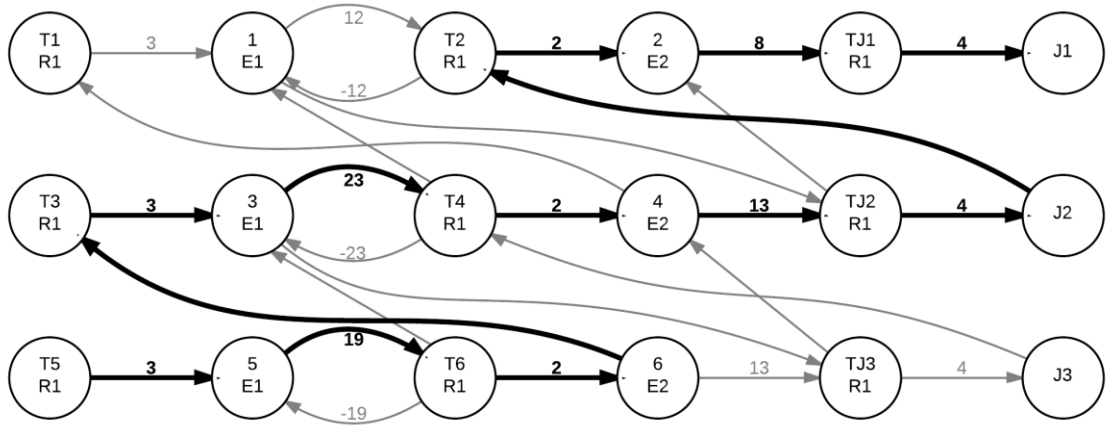

Figure 4: S-graph of the optimal schedule

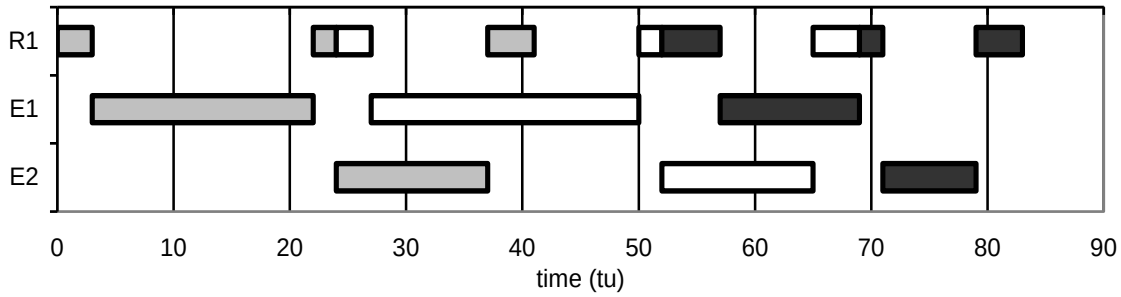

Figure 5: Gantt-chart showing the schedule of Figure 4

4. Empirical analysis

In order to measure the improved precedence based MILP model and evaluate the S-graph framework for AWS problems, the approaches were compared on literature examples. According to the most common approach among the related papers, the instances used for our comparison were derived from the industrial size AWS scheduling problem by Bhushan and Karimi (2004). This benchmark example has 25 products. Each of them is produced through a 12 stage process containing 6 chemical and 6 water baths. The test cases consider a certain subset of products and a restricted number of production steps. In this way, a broad range of test instances were generated with different sizes which made them expedient to analyze the scalability of the approaches.
For each test instance, the CPU time and the objective value of the reported solution is presented. If the time required for solving the actual instance exceeded 1,000 seconds, the optimization procedure was stopped and the best solution found was reported.
The tests were run on an Intel Xeon E5504 (4 cores, 2 GHz) CPU with 8 GiB RAM. For solving the MILP models, the Gurobi Optimizer 5.5.0 was used.

4.1 Efficacy of the improved precedence based model

The set of test cases ranges from 5-13 products (N) with 4-12 stages (M), and a single robot with UW policy for the transfers. Some of the results are presented in Table 1.

Table 1: Results of the improved precedence based model (Intel Xeon E5504 2 GHz quad-core CPU, 8 GiB RAM)

	Aguirre et al. (2013)		Improved model		
MxN	Time (s)	Makespan (tu)	Time (s)	Makespan (tu)	Speedup
4x5	14.98	62	0.7	62	21,4x
4x6	19.5	69.15	1.61	69.15	12,11x
4x7	75.81	76.18	8.32	76.18	9,11x
6x4	3.56	61.08	0.38	61.08	9,37x
6x5	15.67	72.41	1.74	72.41	9,01x
6x7	158.41	90.83	14.94	90.83	10,6x
6x11	1,000	128.81	1,000	123.87	-
6x13	1,000	311.15	1,000	139.02	-
12x5	187.79	128.9	7.82	128.9	12,57x

The results clearly show that the acceleration of the MILP method decreased the CPU time requirements by one order of magnitude in average.

4.2 Evaluation of the S-graph based approach

Although the S-graph algorithms were developed for general scheduling problems, not only for AWS scheduling, the evaluation of the S-graph solver provided valuable information about the potentials of this approach. The same set of problems was solved as in Section 4.1 using the algorithm by Sanmartí et al. (2002). The results are shown in Table 2 in comparison to the precedence based approach of Aguirre et al. (2013).

Table 2: S-graph based results (Intel Xeon E5504 2 GHz quad-core CPU, 8 GiB RAM)

	S-graph approach		Aguirre et al. (2013)	
MxN	Time (s)	Makespan (tu)	Time (s)	Makespan (tu)
4x5	12.5	62	14.98	62
4x6	159.85	69.15	19.5	69.15
4x7	1000	76.18	75.81	76.18
6x4	2.48	61.08	3.56	61.08
6x5	1,000	72.41	15.67	72.41
6x7	1,000	97.98	158.41	90.83
6x11	1,000	137.64	1,000	128.81
6x13	1,000	159.2	1,000	311.15
12x5	1,000	132.48	187.79	128.9

The results show that the general purpose S-graph algorithms are capable of solving AWS scheduling problems to optimality. However, for larger problems the solution exceeded the time limit. In some cases, the optimal solution was found but proving the optimality required more time. The higher solution times of

the S-graph solver were expected, since the used algorithms were developed for general scheduling problems, and the AWS characteristics were not exploited.

5. Acceleration opportunities for the S-graph based approach

There are several opportunities to exploit the problem specific properties of AWSes in order to accelerate the S-graph framework for this set of problems. For example, with the most common, permutation flow-shop recipes, the product sequence is the same on all stages. Thus, when the precedence between two products is decided for one bath, additional scheduling arcs may set it on other stages as well. In this way, not only the search space is reduced, but the bounds for the partial schedules also become sharper.
Another example is for the case of single robot flow-shop AWS problems where the schedule of the baths can be derived from the schedule of the robot. By eliminating all the baths from the scheduling problem, the number of branches in the search tree can be drastically reduced.

6. Conclusions

The formerly developed MILP approaches for the scheduling of wet-etch stations were enhanced and an analogy were given between AWS problems and batch processes to make the problem solvable by the general purpose algorithms of the S-graph framework.
The more specific, time slot based model of Castro et al. (2012) was extended to address empty movement times. The time needed for the solution of the more general, precedence based MILP model of Aguirre et al. (2013) was drastically reduced. Although the preliminary results of the general purpose S-graph based method show higher computational need than the formerly mentioned specialized approaches, the graph-theoretic approach has its potentials.

Acknowledgments

This research has been supported by the European Union and Hungary and co-financed by the European Social Fund through the project TÁMOP-4.2.2.C-11/1/KONV-2012-0004 - National Research Center for Development and Market Introduction of Advanced Information and Communication Technologies.

References

Aguirre A.M., Méndez C.A., Castro P.M., 2011, A novel optimization method to automated wet-etch station scheduling in semiconductor manufacturing systems, Comput. Chem. Eng., 35, 2960-2972.
Aguirre A.M., Méndez C.A., Garcia Sánchez Á., Ortega-Mier M., Castro P.M., 2013, General framework for automated manufacturing systems: multiple hoists scheduling solution, Chemical Engineering Transactions, 32, 1381-1386.
Bhushan S., Karimi I.A., 2003, An MILP approach to automated wet-etch station scheduling, Ind. Eng. Chem. Res., 42, 1391-1399.
Bhushan S., Karimi I.A., 2004, Heuristic algorithms for scheduling an automated wet-etch station Comput. Chem. Eng., 28, 363-379.
Castro P.M., Zeballos L.J., Méndez C.A., 2012, Hybrid time slots sequencing model for a class of scheduling problems, AIChE Journal, 58, 789-800.
Hegyháti M., Majozi T., Holczinger T., Friedler F., 2009, Practical infeasibility of cross-transfer in batch plants with complex recipes: S-graph vs MILP methods, Chemical Engineering Science, 64, 605-610.
Karimi I.A., Tan Z.Y., Bhushan S., 2004, An improved formulation for scheduling an automated wet-etch station, Comput. Chem. Eng., 29, 217-224.
Novas J.M., Henning G.P., 2012, A comprehensive constraint programming approach for the rolling horizon-based scheduling of automated wet-etch stations, Comput. Chem. Eng., 42, 189-205.
Sanmartí E., Holczinger T., Puigjaner L., Friedler F., 2002, Combinatorial framework for effective scheduling of multipurpose batch plants, AIChE Journal, 48, 2557-2570.
Zeballos L.J., Castro P.M., Méndez C.A., 2011, Integrated constraint programming scheduling approach for automated wet-etch stations in semiconductor manufacturing, Ind. Eng. Chem. Res., 50, 1705-1715.