

Complex Job Shop Multiple Orders per Job Scheduling

Jagadish Jampani, Scott J. Mason

University of Arkansas, 4207 Bell Engineering Center, Fayetteville, AR 72701 USA,

{jjampan, mason}@uark.edu

Semiconductor manufacturing production scheduling is a challenging task due to process complexity that includes the assignment of customer orders to front opening unified pods (FOUPs), FOUP batch processing, parallel machines, and re-entrant flow. A network-based optimization model and a column generation (CG) heuristic are presented for this problem to minimize the weighted completion time of customer orders. Computational results show that the CG heuristic obtains solutions that are very close to/better than a mixed-integer program-based heuristic (MIP heuristic) for problem instances without order ready times. In the problem instances with order ready times, CG solutions are within 11% of MIP heuristic solutions, but are obtained in seconds rather than hours.

Keywords: semiconductor manufacturing, complex job shop, column generation

1. Introduction

Semiconductor manufacturing is a complex, expensive manufacturing process involving multiple process flows typically containing hundreds of process steps each. Wafer fabrication is the most complex stage in the integrated circuit (IC) manufacturing process [11]. Customers place orders for ICs, hundreds of which can be fabricated on a single silicon wafer. The newest wafer fabs manufacture ICs on silicon wafers 300-mm in diameter. Given the size/weight and value of a 300-mm wafer containing ICs, front opening unified pods (FOUPs) are used to store and to transport wafers between workstations in 300-mm wafer fabs in groups of 13 or 25. FOUPs are filled with inert gas to protect against contaminants/particulates coming into contact with the wafer surface. Customer IC orders are converted into equivalent 300-mm silicon wafer starts. As an order may require only 3-4 wafers to fill, multiple customer orders are combined into a single FOUP for fab processing and automated material handling system transport. Although each customer order has unique attributes (size, ready time, weight, promise date, etc.), 300-mm fab scheduling is performed at the FOUP (job) level. This is known as multiple orders per job (*moj*) scheduling [10].

A semiconductor wafer fab is a complex job shop (CJS) [5], as it contains parallel machines operating in *tool groups*, batch processes, and re-entrant process flows. In addition, tool groups may contain sequence-dependent setup times (e.g., ion implanters). Scheduling 300-mm wafer fabs adds an additional layer of complexity, as FOUPs contain multiple customer orders that may or may not remain together in the same FOUP throughout the entire manufacturing process flow. In this paper, we investigate the *moj* CJS scheduling problem (“*moj*-CJSSP”) to minimize total weighted order completion time for the case when a FOUP’s processing time is independent of its contents (i.e., *single lot* in [10]). In $\alpha | \beta | \gamma$ scheduling notation [3], the *moj*-CJSSP of interest can be denoted as $FJc | moj(lot), r_o, recrc | \sum w_o C_o$, where w_o (C_o) denotes order o ’s weight (completion time). The *moj*-CJSSP is NP hard, as the NP hard $Jm || C_{max}$ reduces to it.

Ovacik and Uzsoy [9] develop a CJS scheduling heuristic by using shop floor status information, but only consider re-entrant flows. Mason *et al.* [5] present a modified Shifting Bottleneck heuristic (MSBH) for minimizing total weighted tardiness (TWT) in a complex job shop, although a subsequent cycle-elimination procedure is required to guarantee feasibility [6]. Mason *et al.* [7] then present a mixed-integer program (MIP) for TWT minimization in a CJS and compare it to the MSBH and three dispatching rules. Moench and Driessel [8] present a distributed SBH to address minimizing TWT for the CJSSP. The overall problem is decomposed into two hierarchical stages: an upper stage decides start dates and planned due dates, while the lower stage uses upper stage inputs in a SBH. While Jampani and Mason [4] present column generation (CG) approaches for parallel machine *moj* scheduling problems, the *moj*-CJSSP has not been addressed in the literature.

2. Solution Approaches for *moj*-CJSSP

2.1. Network-Based MIP Formulation

Modelling MIPs using a network flow-based paradigm can be more computationally efficient than classical disjunctive approaches. The following steps create the network structure for the MIP formulation. Initially the steps followed to create a network representation for the *moj*-CJSSP corresponding to a single tool group is presented. This network structure is then extended to incorporate multiple tool groups and re-entrant flows.

- Step 1:* A dummy source node has a branch for each machine in the tool group. The upper bound of flow on corresponding arcs is equal to the number of orders and the lower bound is zero. This gives the flexibility of assigning any order to any machine in the tool group.
- Step 2:* Each of the nodes representing a machine branches out with arcs equal to the number of batches that can be processed on that machine. The upper bound of the flow on the corresponding arcs is equal to the number of orders and the lower bound is zero. Hence, any number of orders can be assigned to a batch subject to the capacity constraints.
- Step 3:* Each of the nodes representing a batch branches out into nodes equal to the batch size or the number of FOUPs that can be assigned to that batch in that tool group. The number of arcs connecting a head node to the tail node at this FOUP level is equal to the number of orders. The upper bound of flow on the corresponding arcs is equal to one and the lower bound is zero. By having the binary variables at this level, an order is either selected or not and orders will not be partially assigned to multiple FOUPs.
- Step 4:* All the nodes at the FOUP level are connected to one dummy sink node. Each FOUP node is connected to the sink node using only one arc. The upper bound on these arcs is equal to the number of orders and the lower bound is zero.

The path selected by an order gives all requisite assignment information for that order. Hence, all the arcs represent the decision variables in the formulation, which are non-continuous variables. With the information obtained regarding the path followed by an order in the network corresponding to a tool group, it can be traced back to find the order to FOUP, FOUP to batch, batch to machine, order to batch and order to machine assignments in that tool group. In this network, FOUP-to-batch and batch-to-machine assignments are done inherently. Hence, the overall assignments boil down to assigning orders to FOUPs subject to the network continuity constraints. This helps in cutting down significant number of constraints and binary variables in the formulation.

An example is presented to explain the steps involved in this network creation. A *moj* problem with ten orders and a tool group with two identical machines are considered. Each machine can accommodate two batches and two FOUPs can be assigned to each of the batches.

- Step 1:* As two machines are present in the tool group, the dummy source node has arcs 1 and 2 as shown in Figure 1. The upper bound on arcs 1 and 2 is 10 for the considered 10 orders problem instance and the corresponding lower bound is zero.

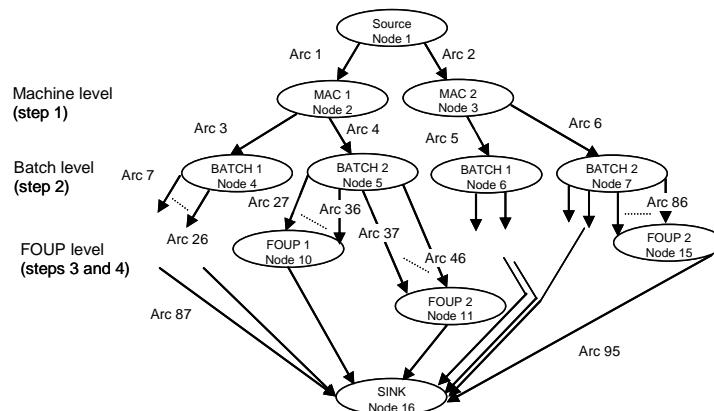


Figure 1. Example problem for network construction

- Step 2:* As two batches can be processed on each of the machines, the node representing machine 1 branches into 3 and 4, and similarly node 2 branches into arcs 5 and 6. The upper bound on arcs 3 through 6 is 10 in this example and the corresponding lower bound is zero.
- Step 3:* As two FOUPs can be assigned to a batch, FOUPs in batch 1 on machine 1 are represented by nodes 8 and 9. Similarly, FOUPs in batch 2 on machine 1 are represented by nodes 10 and 11. The number of arcs connecting node 5 and node 10 is equal to the number of orders (i.e., 10). The upper bound on each arcs at the FOUP level=1 and the lower bound=0.
- Step 4:* In this step, a new node 16 is created and the nodes from the FOUP level (nodes 8 through 15) are connected to sink node 16.

Network creation steps 1-4 are repeated to create a network with multiple tool groups. The dummy source node for the succeeding tool group is the sink node of the previous tool group. In case of re-entrant flow, an order needs to be processed in the same tool group multiple times. This is accomplished by creating a duplicate tool group and restricting the machines of the tool group to process only one batch at a time. For example, if tool group 2 is a duplicate of tool group 1 containing a single machine, then the machine can process a batch either in tool group 1 or 2 at a time satisfying the machine capacity constraint. The following notation is used in our MIP formulation:

- o, j, b, m, p Index for orders, FOUP #, batch #, machine #, and process step #, respectively
- K FOUP capacity
- s_o, r_o, w_o Size, ready time and weight of order o , respectively
- ρ_p, c_p Processing time and tool group # for process step p , respectively
- $J(c_p)$ Set of FOUPs in tool group c_p
- $B(c_p)$ Set of batches in tool group c_p
- $M(c_p)$ Set of machines in tool group c_p
- $X_{o,j,b,m,p}$ Variable = 1 if order o is assigned to FOUP j that is present in batch b and processed on machine m at process step p ; otherwise, = 0.
- $Y_{b,m,p}$ Variable for completion time of batch b processed on machine m at process step p
- $Z_{b,m,p}$ Variable for ready time of batch b processed on machine m at process step p
- $R_{o,p}$ Variable for ready time of order o at process step p
- $F_{o,p}$ Variable for completion time of order o at process step p

The formulation for $FJc|moj(lot), r_o, recrc|\sum w_o C_o$ problem is as follows:

$$\text{Minimize } \sum_{o \in O} w_o F_{o,p=P} \quad (1)$$

$$\text{Subject to: } \sum_{m \in M(c_p)} \sum_{b \in B(c_p)} \sum_{j \in J(c_p)} X_{o,j,b,m,p} = 1 \quad \forall o \in O, p \in P \quad (2)$$

$$\sum_{o \in O} s_o X_{o,j,b,m,p} \leq K \quad \forall j \in J(c_p), b \in B(c_p), m \in M(c_p), p \in P \quad (3)$$

$$Z_{b,m,p} \geq r_o X_{o,j,b,m,p} \quad \forall o \in O, j \in J(c_p), b \in B(c_p) \quad (4)$$

$$\forall m \in M(c_p), p = 1$$

$$Y_{b,m,p} - Z_{b,m,p} \geq \rho_p \quad \forall b \in B(c_p), m \in M(c_p), p \in P \quad (5)$$

$$Z_{b,m,p} \geq F_{o,p-1} - M_{big} (1 - X_{o,j,b,m,p}) \quad \forall o \in O, j \in J(c_p), b \in B(c_p), \quad (6)$$

$$m \in M(c_p), p > 1, p < P$$

$$F_{o,p} \geq Y_{b,m,p} - M_{big} (1 - X_{o,j,b,m,p}) \quad \forall o \in O, j \in J(c_p), b \in B(c_p), \quad (7)$$

$$m \in M(c_p), p \in P$$

$$Z_{b,m,p} \geq Y_{b-1,m,p} \quad \forall b \in B(c_p), b > 1, m \in M(c_p), p \in P \quad (8)$$

$$X_{o,j,b,m,p} \in \{0,1\} \quad \forall o \in O, j \in J(c_p), b \in B(c_p), m \in M(c_p), p \in P \quad (9)$$

Objective function (1) minimizes the sum of weighted completion times of the orders during the last process step. Constraint set (2) makes sure that every order is assigned to a FOUP during every process step. Constraint set (3) corresponds to FOUP capacity constraints. Orders are assumed to have non-zero ready times. Constraint set (4) calculates batch ready times as a function of order ready times. Constraint set (5) takes care of the processing time of the orders in each of the tool groups during the process steps. During each process step, the ready time of the batches is dependent on the orders assigned to the batches and the orders completion time during the immediate preceding process step. Ready time of the batches is calculated at each process step by constraint set (6). Constraint set (7) calculates the completion time of the orders at each process step. Constraint set (8) restricts the starting time of batch b to be at least the completion time of batch $b - 1$. Hence, it also restricts only one batch to be processed on a machine at a time.

Constraint sets (2) to (8) correspond to assignment and ready time constraints. As our *moj*-CJSSP formulation is network-based, the network is stored in terms of arc details, which are head/tail nodes and lower/upper bounds on the arcs. These arcs represent decision variables in the formulation. Network constraints pertain to upper and lower arc capacity restrictions, as well as arcs flow and flow continuity at nodes. The following notation is used in this part of the formulation:

g, n	Index over the arcs ($g \in G$) and nodes ($n \in N$), respectively
d, f	Index over the tail and head nodes, respectively
$\omega(n)$	Set of tail nodes of the arcs that have n as their head node
$\xi(n)$	Set of head nodes of the arcs that have n as their tail node
θ, ψ	Source and sink nodes of the network, respectively
$U_{d,f}$	Variable for amount of flow on an arc with tail node d and head node f

Additional network constraints for the $FJc|moj(lot), r_o, recrc|\sum w_o C_o$ formulation are as follows:

$$\sum_{f \in \xi(\theta)} U_{d=\theta, f} = O \quad (10)$$

$$\sum_{d \in \omega(\psi)} U_{d, f=\psi} = O \quad (11)$$

$$\sum_{d \in \omega(n)} U_{d, n} - \sum_{f \in \xi(n)} U_{n, f} = 0 \quad \forall n \in N - \{\psi, \theta\} \quad (12)$$

Constraint sets (10) and (11) require the sum of arc flows out of the source and into the sink equal the number of orders, respectively. Constraint set (12) maintains flow balance at each non-source and non-sink node. Depending on what an arc represents, its upper bound is set and the lower bound of all the arcs is zero. If an arc corresponds to the FOUP level (Figure 1), the upper bound on the corresponding binary decision variable is one unit. Otherwise, the upper bound on the decision variable is the number of orders. As the arcs at the FOUP level represent binary decision variables, considering flow continuity constraints results in the other arcs taking on only integer values. Therefore, they are declared as continuous variables to reduce MIP computation time. The following valid inequalities also help in improving the lower bound for this minimization problem and thereby reducing computation time.

$$F_{o,p} \geq \rho_p \quad o \in O, p = 1 \quad (13)$$

$$F_{o,p} \geq F_{o,p-1} + \rho_p \quad o \in O, p > 1 \quad (14)$$

$$G_{o,m,p} \geq X_{o,j,b,m,p} \quad \forall o \in O, j \in J, b \in B(c_p) \quad (15)$$

$$T_{m,p} \geq F_{o,p} - M(1 - G_{o,m,p}) \quad \forall o \in O, m \in M(c_p), p \in P \quad (16)$$

$$Z_{b,m,p} \geq T_{m,p} \quad \forall b \in B(c_p), m \in M(c_p), p \in P \quad (17)$$

Constraint set (13) restricts the completion time of an order during the first process step to be at least equal to its processing time. Completion time of an order in any process step ($p > 1$) is at least equal to the sum of processing time in the current process step and completion time of the order in the immediate previous process step. This condition is taken care of by constraint set (14). Constraint set (15) finds the machine on which an order is processed during a particular process step. Constraint set (16) calculates maximum completion time of processing all the orders on a particular machine during a process step. Constraint set (17) requires the ready time of an order at a tool group to be at least the completion time of the same order's previous visit to the same tool group, where p represents re-entrant flow steps only.

2.2 Column Generation Heuristic

It can be observed that the MIP can be decomposed into two problems, as in the case of CG. The column generation approach is based on the idea of *decentralized decision processes* [1]. In CG, the linear program is decomposed into multiple sub-problems and a master problem—the master has global visibility and coordinates linear transformations of vectors obtained from the sub-problems. The master problem (MP) selects the best feasible columns, while the subproblem generates new columns representing the assignment information corresponding to the process steps in the complex job shop. As the critical part of any CG approach is modeling the subproblem, a network-based subproblem is developed that is similar to the network created in Section 2.1.1. This results in more efficient subproblem computations that directly result in reduced overall computation time. The following notation is used in the CG formulation for the *moj*-CJSSP:

o	Index over the order number ($o \in O$)
k	Index over the slot number in a FOUP (K is FOUP capacity) ($k \in K$)
t_n	Index over the FOUP number during process step n ($t_n \in T_n$)
m_n	Index over the machine number during process step n ($m_n \in M_n$)
c	Index over the columns or patterns
n	Index over the process steps in the complex job shop problem ($n \in N$)
$b(n)$	Batch size during process step n
$C(o)$	Columns corresponding to order o ($c \in C(o)$)
$W_{o,c}$	Product of w_o and completion time of order o in the last process step, which is represented in column c
$A_{o,k,t_n,m_n,c}$	Variable = 1, if order o occupying slot k in FOUP t_n is processed on machine m_n and this information is represented in column c ; otherwise, = 0
E	Maximum number of FOUFs that can be used
M_{big}	Represents a large positive number (i.e., big M)
$X_{o,c}$	Variable = 1 if column c corresponding to order o is selected; otherwise, = 0
Z_{t_n,m_n}	Variable = 1 if FOUP t is occupied and processed on machine m_n ; otherwise, = 0
λ	Variable to find how many extra FOUFs are used than expected

Restricted Master Problem (MP)

$$\text{Minimize } M_{big} \lambda + \sum_{o \in O} \sum_{c \in C(o)} W_{o,c} X_{o,c} \quad (18)$$

$$\text{Subject to: } \sum_{n \in N} \sum_{m_n \in M_n} \sum_{t_n \in T_n} Z_{t_n,m_n} - \lambda \leq E \quad (19)$$

$$\sum_{o \in O} \sum_{c \in C(o)} \sum_{k \in K} A_{o,k,t_n,m_n,c} - M_{big} Z_{t_n,m_n} \leq 0 \quad \forall t_n \in T_n, m_n \in M_n, n \in N \quad (20)$$

$$\sum_{c \in C(o)} X_{o,c} = 1 \quad \forall o \in O \quad (21)$$

$$\sum_{o \in O} \sum_{c \in C(o)} A_{o,k,t_n,m_n,c} X_{o,c} \leq b(n) \quad \forall t_n \in T_n, k \in K, m_n \in M_n, n \in N \quad (22)$$

$$X_{o,c} \geq 0 \quad \forall o \in O, c \in C \quad (23)$$

The MP objective function (18) minimizes total weighted order completion time and penalizes the use of more FOUPs than are available. Constraint sets (19) and (20) assess whether the number of FOUPs used exceeds E , the maximum number allowed. Constraint set (21) selects a column corresponding to every order once. Constraint set (22) enforces batch capacity restrictions for each machine in the CJS. This constraint also controls FOUP to batch assignment. This formulation assumes batches are processed sequentially on a machine in increasing order of batch index without forced idle time. The following notation is used in the CG subproblem (SP) of the *moj*-CJSSP:

$\partial, \sigma_{t,m_n,n}$ Dual values from constraint (19)-(20), respectively

$\nu_o, \pi_{t,k,m_n,n}$ Dual values from constraint (21)-(22), respectively

$H(i), T(i)$ Head and tail node of arc i respectively

η_n, λ_n Source and sink node corresponding to process step n , respectively

ω Set of head nodes in the network

ψ Set of tail nodes in the network

φ Set of all the nodes in the network

$\ell_{o,n}$ Set of head nodes of the arcs that correspond to batches with processing start time less than the ready time of the order during process step n

$C_{i,j} = (((w_o / s_o)t) - \pi_{t,k,m_n,n}) - (\sigma_{t,m_n,n} / s_o)$, cost assigned to arc (i, j)

$Y_{i,j}$ Variable = 1 if an arc with tail node i and head node j is selected; otherwise, = 0

Subproblem (SP)

$$\text{Minimize} \quad \sum_{i \in \psi} \sum_{j \in \omega} C_{i,j} Y_{i,j} - \nu_o - \partial \quad (24)$$

$$\text{Subject to:} \quad \sum_{j \in H(\eta_n)} Y_{\eta_n,j} = 1 \quad (25)$$

$$\sum_{j \in T(\lambda_n)} Y_{j,\lambda_n} = 1 \quad (26)$$

$$\sum_{i \in T(j)} Y_{i,j} - \sum_{k \in H(j)} Y_{j,k} = 0 \quad \forall j \in \varphi - \{\eta_n, \lambda_n\} \quad (27)$$

$$\sum_{i \in T(j)} \sum_{j \in \ell_{o,n}} Y_{i,j} = 0 \quad (28)$$

$$Y_{i,j} \in \{0,1\} \quad \forall i, j \quad (29)$$

The network structure developed in Section 2.1 is used in SP. Constraint set (25) requires an outgoing arc from the source node be selected, while constraint set (26) requires the selection of an incoming arc into the sink node. Constraint set (27) maintains flow continuity by balancing the incoming and outgoing flow at all nodes in the network other than the source and sink. Finally, as 1) process starting time for batches is fixed and 2) order ready times at process steps are known, an order cannot be assigned to a batch that starts before the order is ready at a given step (28).

As an order's ready time during a process step depends on its completion time in the immediate predecessor step, the SP is solved once for each process step. Hence, during every iteration of the MP, the subproblem is solved the number of times that is equal to the product of the number of orders and the number of process steps. The assumption that the batches are processed on the machines in increasing order of their index also avoids non-linearity in the formulation and makes a

difference only when the orders have non-zero ready times. As the start time of processing the batches is fixed, it may introduce some unnecessary delay in processing a batch. Hence, a post-process step is applied after CG terminates. This post process step adjusts the processing start times of the batches by identifying and eliminating any unnecessary delays and not violating any constraints. In this CG approach, multiple subproblems are solved to generate the complete information required to create a new column and add to the MP rather generating the whole column information in one subproblem. This may be resulting in a degeneracy situation in some instances, where no more improving new columns are created but still some of the columns have negative reduced costs. In such situations, the CG is allowed to run for ten more additional iterations, where the objective value of the restricted MP does not improve, before terminating.

3. Experimental Results

A mini-fab model that represents a complex job shop environment with the characteristics of multiple tool groups, batching, parallel machines, and re-entrant flow is used for our experimentation purposes [2]. Arc labels in Figure 2 indicate step number/process time/batch capacity (in FOUPs).

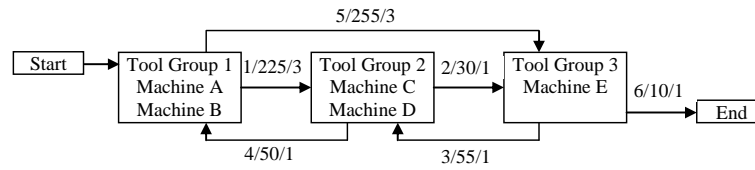


Figure 2. The mini-fab model [2]

The performance of CG-based heuristic for the complex job shop problem is evaluated using the experimental design in Table 1. Many hundreds of integrated circuits (ICs) can be fabricated on a single 300-mm wafer. All experimental parameters are modeled to mimic realistic values in practice. Number of orders is varied at three levels and FOUP capacity is varied at two levels. Order sizes, weights, and ready times are generated from discrete uniform distributions, with each being varied at two levels. For each combination of experimental factors, 10 replications are performed.

Table 1. Experimental Design

Factor	Level Description	# Levels
Number of orders	10, 15, 20	3
Order size (s_o)	$DU[v - (v + 1)/2, v + (v + 1)/2]$, $v \in \{3, 5\}$	2
FOUP capacity (K)	$12\beta + 1$, $\beta \in \{1, 2\}$	2
Weights (w_o)	1, $DU[1, 15]$	2
Ready times (r_o)	0, $DU[1, 300]$	2
Replications per level combination Total number of instances		10 480

CG-based heuristic solutions are compared with the optimization model's solution after a maximum of six hours of computation time ("MIP heuristic"). CPLEX 10.0 is used for solving the optimization models and MIPs in the CG-based heuristics on a 3.40 GHz PC with 2 GB RAM. Computational results are reported in Table 2 in terms of performance ratio (PR), which is defined as the ratio of CG-based heuristic solution to the MIP heuristic solution. Results are given as a/b denoting instances *without*/*with* order ready times present. It can be observed that the CG solution is close to the MIP heuristic solution for problem instances without ready times. For problem instances with ready times of the orders, CG obtains solutions that are within 11% of the MIP heuristic, on average. In the problem instances with higher number of orders and order weights, PR decreases. This may be because the MIP could not find the optimal or near optimal solution within six hours of computation time for these large problem instances. The computation time limit terminates branch-and-cut before memory limits are exceeded, especially in problems with many orders. This translates into better average performance ratios in 20 order problem instances.

Table 2. Average PR for $FJc|moj(lot), r_o, recrc|\sum w_o C_o$ problem instances

FOUP Size	Order Size	Order Weight	Average Performance Ratios		
			10 Orders $r_o = 0 r_o > 0$	15 Orders $r_o = 0 r_o > 0$	20 Orders $r_o = 0 r_o > 0$
13	DU[1,5]	1	1.010 1.178	1.017 1.088	1.020 1.112
		DU[1,15]	0.988 1.170	0.978 1.123	0.958 1.049
25	DU[1,5]	1	1.000 1.143	0.997 1.132	1.003 1.092
		DU[1,15]	1.000 1.132	0.998 1.138	0.973 1.051
13	DU[2,8]	1	1.039 1.079	1.073 1.070	1.017 0.970
		DU[1,15]	0.991 1.026	1.003 1.062	0.931 1.030
25	DU[2,8]	1	1.003 1.182	1.015 1.104	1.020 1.121
		DU[1,15]	0.991 1.187	0.982 1.134	0.978 1.087
Average			1.003 1.137	1.008 1.106	0.988 1.064

In terms of computation time, a 20-order problem with $r_o > 0$ and $K=25$ required the largest average computation time of 133 seconds. As the number of orders and FOUP capacity increase, this results in more constraints and decision variables in the CG's MP and SP, which results in increased computation time.

4. Conclusions and Future Work

In this paper, a network-based MIP formulation and a CG-based heuristic approach are presented for the *moj*-CJSSP. A mini-fab model and a representative range of problem instances are selected for experimentation to analyze the effectiveness of the CG approach compared to a time-limited MIP heuristic. From computational results, it can be observed that the CG approach obtains solutions that are very close to the MIP heuristic in the problem instances without ready times. But in the problem instances with ready times, CG obtains solutions within 11% of the MIP heuristic solution. Future research will investigate *moj(item)* processing wherein FOUP processing time is dependent on its contents. Lower bounds estimates for minimizing total weighted order completion time in *moj*-CJSSPs will also help to further assess heuristic solution quality.

References

- [1] G.B. Dantzig, P. Wolfe (1960), Decomposition principle for linear programs, *Ops Rsch*, **8**(1), 101 – 111.
- [2] M.K. Ed Adl, A.A. Rodriguez, K.S. Tsakalis (1996), Hierarchical modeling and control of Re-entrant semiconductor manufacturing facilities, *Proc 35th Conf Decision and Control*, Japan, 1736 – 1742.
- [3] Graham, R.L., Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G. (1979), Optimization and approximation in deterministic sequencing and scheduling theory: a survey, *Annals of Discrete Math* **5**, 287 – 326.
- [4] J. Jampani, S.J. Mason (to appear), Column generation heuristics for multiple machines, multiple orders per job scheduling problems, *Annals of Operations Research*.
- [5] S.J. Mason, J.W. Fowler, W.M. Carlyle (2002), A modified shifting bottleneck heuristic for minimizing the total weighted tardiness, *Journal of Scheduling* **5**(3), 247 – 262.
- [6] S.J. Mason, K. Oey (2003), Scheduling complex job shops using disjunctive graphs: A cycle elimination procedure, *Int. Journal of Productions Research* **41**(5), 981 – 994.
- [7] S.J. Mason, J.W. Fowler, W.M. Carlyle, D.C. Montgomery (2005), Heuristics for minimizing total weighted tardiness in complex job shops, *Int. Journal of Production Research* **43**(10), 1943 – 1963.
- [8] L. Moench, R. Driessel (2005), A distributed shifting bottleneck heuristic for complex job shops, *Computers and Industrial Engineering* **49**(3), 363 – 380.

- [9] M.I. Ovacik, R. Uzsoy (1994), Exploiting shop floor status information to schedule, *Journal of Manufacturing Systems* **13**(2), 73 – 84.
- [10] P. Qu, S.J. Mason (2005), Metaheuristic scheduling of 300mm jobs containing multiple orders, *IEEE Trans on Semiconductor Manufacturing* **18**(4), 633 – 643.
- [11] R. Uzsoy, C.Y. Lee, L.A. Martin-Vega (1992), A Review of Production Planning and Scheduling Models in the Semiconductor Industry Part I: System Characteristics, Performance Evaluation and Production Planning, *IIE Transactions* **24**(4), 47 – 60.