

Dynamic Models of Production with Multiple Operations and General Processing Times



M. M. Dessouky; R. C. Leachman

The Journal of the Operational Research Society, Vol. 48, No. 6 (Jun., 1997), 647-654.

Stable URL:

<http://links.jstor.org/sici?sici=0160-5682%28199706%2948%3A6%3C647%3ADMOPWM%3E2.0.CO%3B2-E>

The Journal of the Operational Research Society is currently published by Operational Research Society.

Your use of the JSTOR archive indicates your acceptance of JSTOR's Terms and Conditions of Use, available at <http://www.jstor.org/about/terms.html>. JSTOR's Terms and Conditions of Use provides, in part, that unless you have obtained prior permission, you may not download an entire issue of a journal or multiple copies of articles, and you may use content in the JSTOR archive only for your personal, non-commercial use.

Please contact the publisher regarding any further use of this work. Publisher contact information may be obtained at <http://www.jstor.org/journals/ors.html>.

Each copy of any part of a JSTOR transmission must contain the same copyright notice that appears on the screen or printed page of such transmission.

JSTOR is an independent not-for-profit organization dedicated to creating and preserving a digital archive of scholarly journals. For more information regarding JSTOR, please contact support@jstor.org.



Dynamic models of production with multiple operations and general processing times

MM Dessouky¹ and RC Leachman²

¹ University of Southern California and ² University of California

Traditional integer programming model formulations for job-shops and flow-shops do not easily account for characteristics common to high-technology manufacturing such as high-volume semiconductor manufacturing. These characteristics are: (1) products (wafers) are processed by the same machine type more than once during the operation sequence, (2) many lots of similar type are run, and (3) there can be multiple machines of the same type. In this paper, we present two new integer programming formulations which easily account for these characteristics. The approach is based on restricting the allowed domain of events for the start of lot processing. The first model restricts production starts to the beginning of a planning time period. The second model uses a special time grid at each operation with width equal to the processing time, and allows starts to be scheduled at the grid points. In an example problem replicating a high-volume wafer fabrication process, it is shown that it is computationally practical to obtain solutions for the restricted start models where it is not computationally possible for the traditional integer programming model formulations.

Keywords: integer programming; job-shops; production; scheduling

Introduction

Developing realistic models of production that can be solved in a feasible amount of computer time is the goal of many schedulers. Typically, there is a trade-off between realism and speed of computation. Including more detail in a model tends to increase the amount of computation required to find an optimal solution or even a 'good solution. In contrast to other previously developed dynamic models of production (see Shephard¹ and Hackman and Leachman²), we present two new models that consider the processing time. Previous dynamic models of production that explicitly account for the processing time include the standard integer programming formulations for the job-shop and flow-shop scheduling problems (see Wagner,¹³ Manne⁴ and Pritsker *et al.*⁵) with later modifications by Selen and Hott,⁶ Stafford,⁷ Wilson,⁸ and Liao and You⁹ among others. As French¹⁰ points out, practical applications of these integer programming formulations have been disappointing because of the computational difficulty of obtaining optimal or near-optimal solutions and the simplifying assumptions of these models.

The focus of this paper is to develop alternative dynamic models of production that consider the processing time for classical job-shop or flow-shop environments characterized by jobs or production lots that must pass through a long

series of processing operations involving different kinds of equipment and labor. Changeovers of equipment to perform different operations are assumed to be negligible or lot-specific (not operation-specific), as is the case in many types of electronic and pharmaceutical factories and other high-technology manufacturing. Our particular motivation comes from high-volume semiconductor manufacturing. Some other characteristics that are common in semiconductor manufacturing are: (1) products (wafers) are processed by the same machine type more than once during the operation sequence, (2) many lots of similar type are run, and (3) there can be multiple machines of the same type. Note that the previously developed models (for example Manne's integer programming formulation) do not easily account for these characteristics. In this paper, we present two new integer programming formulations which easily account for these characteristics. The approach is based on restricting the allowed domain of events for the start of lot processing. We refer to the models as restricted start models.

The first restricted start model is a generalization of the model developed by Mizrach.¹¹ Our presentation of the model allows for processing times that are not integer multiples of one another and provides inventory variables for situations when products are not immediately applied to processing. The model restricts the possible epochs for production starts to correspond to the planning period time grid points. This model is referred to as Production Starts Set To Beginning of Period (Model PSTBP). This

Correspondence: Dr MM Dessouky, Department of Industrial and Systems Engineering, University of Southern California, Los Angeles, CA90089-0193, U.S.A.

approach results in some forced idle time of the resources when the processing times are not integer multiples of the period length.

The second restricted start model assumes that the time between consecutive production starts is restricted to be a multiple of the processing time. This model is referred to as Production Starts Set to Processing Time (Model PSTPT). This assumption allows us to model the workload of an operation as a constant rate during intervals equal to the processing time. This model does not force as much idle time, in the case that the processing times are not integer multiples of one another. The version of Model PSTPT presented in this paper is a generalization of the model presented in Dessouky and Leachman¹² where it was assumed that all operations requiring the same resources had the same processing time. Our presentation of Model PSTPT in this paper allows for different operations requiring the same resource to have varying processing times.

The framework in which the models are based is first presented and then we present the restricted start models. Finally, in an example problem replicating a high-volume wafer fabrication process, it is shown that it is computationally practical to obtain solutions for the restricted start models where it is not computationally possible for the traditional models.

Framework

In the framework (adapted from Hackman and Leachman²), the production system is represented as a network where the nodes represent the *activities*. Directed arcs emanating from one activity to other activities indicate possible transfers of intermediate products which can be applied as inputs by other activities. Production at each activity requires intermediate product inputs as well as service resources which include labor, machines, and facilities.

An *activity* represents a unique operation in the production process. The number of individual product units produced by an activity per unit time is constrained by the resources available during processing. An activity may require the application of one or more resource type to produce its product. The duration of an activity (operation) performed on one unit of product is referred to as its *processing time*. It is assumed that the nature of each activity is such that this time is fixed. Products produced at each activity can be used to meet external demand for that product or can be used as input to follow-on activities.

The framework used here is a continuous time formulation admitting both rate-based and event-based flows. A rate-based flow, $x(t)$, represents the quantity per unit time of flow at time t while an event-based flow, $x(t)$, is the state of a variable at time t .

Previous dynamic models of production assume a single time grid for the entire model. The time grid points specify the points in time when the rate-based flows can change or

when the event-based flows can occur. The framework used in this paper defines two types of time grids. The *planning period time grid* includes the time points when any parameters of the production system can change such as demand and capacity. Typically, the planning period time grid is the set of integer numbers up to the planning horizon. The time points when the work rate of an activity can be planned to change is referred to as the *workload change time grid*.

The other assumptions of the framework are:

- (1) Each activity produces a single product type. This product is not produced by any other activity 'product-generated network'.
- (2) Activities do not receive products they cannot process, and do not transfer products they cannot produce 'normal production model'.
- (3) A product can be held in inventory only at the activity that produces it.
- (4) Activity input-output relationships are proportional and indexed by the activity's intensity function, measured in units of product output per unit time.
- (5) Each resource type consists of a bank of parallel servers that can be allocated among the activities.
- (6) The processing time of activity i , p_i , is deterministic.

The notation for parameters of our adaptation of the framework and further specializations of it to be presented is as follows:

- (1) Indices

t	Continuous time index, $t \leq T$, where T is the time horizon.
i, j	Activity index, $i, j = 1, \dots, N$, where N is the number of activities.
k	Resource type index, $k = 1, \dots, K$, where K is the number of resources.
- (2) Decision Variables

$z_i(t)$	The <i>intensity</i> of activity i at time t (that is the amount of work performed per unit time at time t), measured in units of completed product quantity per unit time.
$s_i(t)$	Number of production starts at activity i at time t . $s_i(t)$ measures the application of intermediate products at activity i , but expressed in units of activity i output.
$f_i(t)$	Number of units of intermediate products finished by activity i at time t (that is the number of production outs at time t).
$I_i(t)$	Inventory of product i at time t .
$B_i(t)$	Number of units backordered of product i at time t .
- (3) Parameters

$d_i(t)$	External demand for product i at time t .
----------	---

\bar{a}_{ij}	number of units produced by activity i required as input per unit intensity of activity j .
a_{ki}	Amount of resource k required as input by activity i per unit intensity of activity i .
$c_k(t)$	Capacity rate of resource k at time t .
p_i	Processing time of activity i .

An upper case of a decision variable or parameter is the notation used herein to refer to the cumulative flow. For rate-based variables the cumulative flow is an integral (for example $Z_i(t) = \int_0^t z_i(\tau)d\tau$). For event-based flows, the cumulative flow is simply the sum of all flows since time 0. We let x^+ denote the smallest integer not smaller than x , and let x^- be the greatest integer not greater than x .

The framework enforces the following set of general constraints on the flows in the production system:

Material balance

$$B_i(t) - I_i(t) + I_i(0) + F_i(t) - \sum_{j=1}^N \bar{a}_{ij}S_j(t) - D_i(t) = 0, \quad \text{all } t \geq 0, \text{ all } i \quad (1)$$

Capacity

$$\sum_{i=1}^N a_{ki}z_i(t) \leq c_k(t), \quad \text{all } t \geq 0, \text{ all } k \quad (2)$$

Domain constraints linking

$$s_i(t), f_i(t) \quad \text{and} \quad z_i(t) \quad (3)$$

to technologically feasible patterns and requiring non-negativity throughout continuous time of all flows.

In the framework three types of constraints must hold at each instant of time. The first constraint type (1) is referred to as material balance, applicable to each intermediate product (activity) i . This constraint restricts the cumulative application of each intermediate product plus any external demand to be less than or equal to the cumulative amount produced plus initial inventory. The slack of this constraint at time t is the inventory or the number of units back-ordered of product i . The second set of constraints (2) restrict the total application of each resource to be less than or equal to the available capacity.

The third set of constraints (3) cannot be expressed generally, as they depend on specific technological characteristics and modeling assumptions about the production system. Such constraints insure that the application of resources and the application of intermediate products inputs are properly linked, and that activity output is properly linked to the input applications. These constraints also indicate whether flows are rate-based or event-based.

Models explicitly representing processing time

When the instantaneous processing time assumption is relaxed, the scheduling problem can be precisely formu-

lated only as an integer program. As processing times become more significant, production starts and completions are event that become less accurately represented as rates. In terms of the framework, $s_i(t)$ and $f_i(t)$ are event-based flows while $z_{i,t}$ is a rate-based, step function flow changing rates at only epochs of product starts and completions. Two alternative models representing this mixture of event-based and rate-based flows, differing in their allowed sets of epochs for production starts, are considered. Model PSTBP restricts production starts to the beginning of planning periods. Model PSTPT restricts the elapsed time between production starts to be multiples of the processing time. In addition to their computational advantages, the restricted start models as opposed to the traditional integer programming formulations of the job-shop can easily model the following situations common in semiconductor manufacturing without increasing the problem size:

- (1) Multiple machines of the same type,
- (2) Demand size greater than one unit for a particular product,
- (3) Repeat visits to the same machine type, and
- (4) Assembly operations which converge the outputs of several operations into a single product.

Consistent with previous studies, the planning periods are assumed to be unit-length. The capacity rates can change and the demand can occur only at the integer time points. The activity intensity can change only at the workload change time grid points. In model PSTBP, the workload change time grid is the same as the planning period time grid. In Model PSTPT, each activity has its own workload change time grid.

Production starts restricted to beginning of planning period

Mizrach¹¹ develops a model restricting the allowed epochs for product starts to the beginning points of planning periods. Mizrach's formulation assumes the processing time is an integer multiple of the planning period. In this section, the integer processing time assumption is relaxed and the formulation is revised in the context of the framework to consider inventory variables. As previously, each planning period is assumed to be unit-length ($t = 1, 2, \dots, T$).

In addition to the material balance and capacity constraints, Model PSTBP includes domain constraints relating activity intensity, output of finished products, and production starts at each activity i . To develop the domain constraints, we note that restricting starts to the beginning of a period is equivalent to not realizing outputs until the end of the period. The following equation defines the

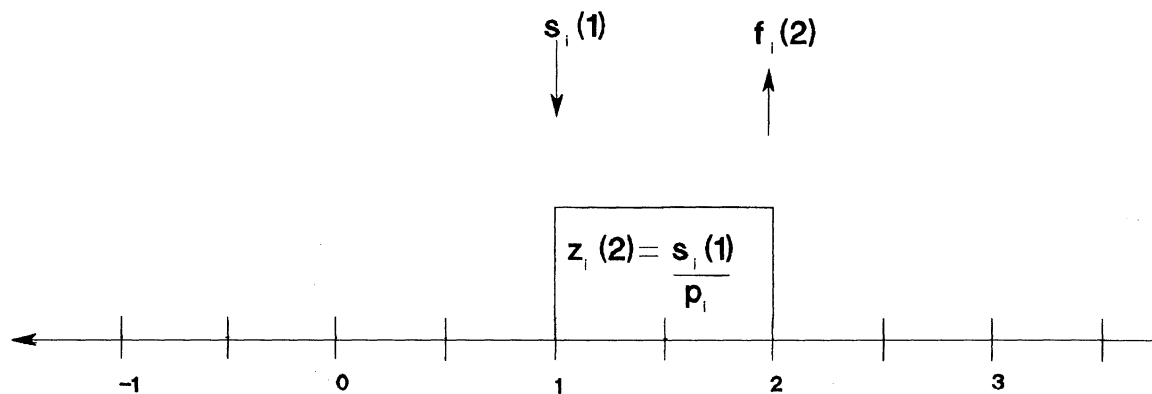


Figure 1 Relationship between the decision variables when processing time is integral.

relationship between the number of units of finished product and production starts:

$$f_i(t) = s_i(t - p_i^+)$$

Figure 1 shows the relationship between $s_i(t)$, $z_i(t)$, and $f_i(t)$ when the processing time is an integral. In this model, when p_i is not a multiple of the planning period length, the workload at an activity, $z_i(t)$, during a planning period is not uniform. Let $z_{i,t}$ be the maximum of $z_i(\tau)$ where $\tau \in (t - 1, t]$ for $t = 1, 2, \dots, T$. A diagram of the relationship between work and production starts when $p_i = 1.5$ is depicted in Figure 2. The figure illustrates the value of $z_{i,3}$ as determined by the $z_i(t)$ function. As the diagram shows, the production starts at time 2 are worked on during the entire time period 3 while the production starts at time 1 are worked on during the first half of the period. In general, during period t , the workload at activity i includes the starts made at times $t - 1, t - 2, \dots, t - p_i^+$ during $(t - 1, t - 1 + p_i - p_i^-]$, and falls off to include only the starts made at times $t - 1, t - 2, \dots, t - p_i^+ + 1$ during $(t - 1 + p_i - p_i^-, t]$.

Since capacity is constant during each planning period, activities must be constrained so that resource allocation rates supporting the workload rate at the start of each planning period do not exceed resource capacities. Accordingly, $z_{i,t}$ in this model is defined to be the intensity of activity i at the start of period t . Let P denote the set of time epochs marking the end points of the planning periods.

The complete Model PSTBP is:

Material balance

$$B_i(t) - I_i(t) + I_i(0) + F_i(t) - \sum_{j=1}^N \bar{a}_{ij} S_j(t) - D_i(t) = 0, \quad \text{all } t \in P, \text{ all } i \quad (4)$$

Capacity

$$\sum_{i=1}^N a_{ki} z_{i,t} \leq c_{k,t}, \quad \text{all } t \in P, \text{ all } k \quad (5)$$

Domain Constraint Linking Outs to Starts

$$f_i(t) = s_i(t - p_i^+), \quad \text{all } t \in P, \text{ all } i \quad (6)$$

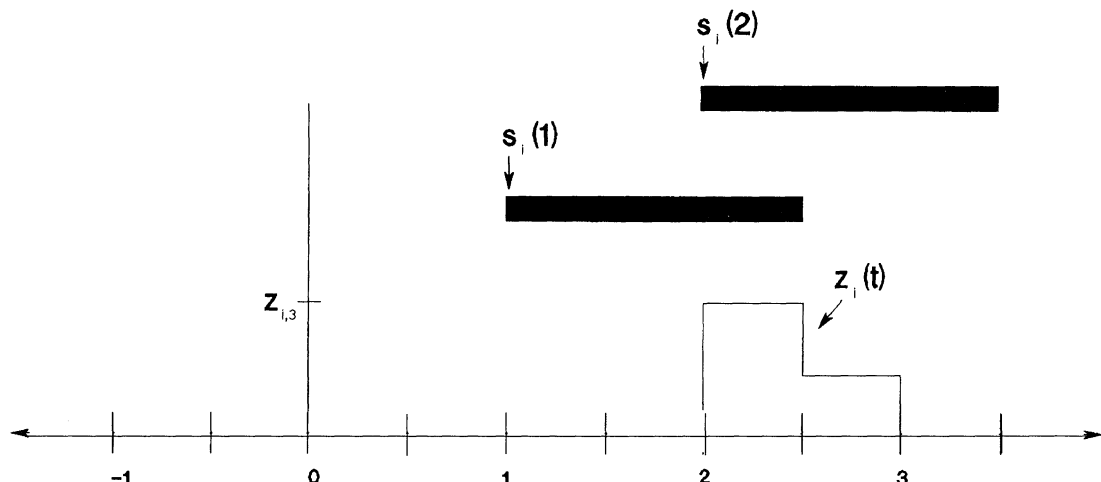


Figure 2 Relationship between starts and work for model PSTBP.

Domain Constraint Linking Resource Applications to Starts

$$z_{i,t} = \frac{1}{p_i} [S_i(t - 1) - S_i(t - p_i^+ - 1)] \quad \text{all } t \in P, \text{ all } i \quad (7)$$

Integer Domains

$$s_i(t), f_i(t) \geq 0, \quad \text{all } t \in P, \text{ all } i \quad (8)$$

Non-negative Domains

$$I_i(t), B_i(t), z_{i,t} \geq 0, \quad \text{all } t \in P, \text{ all } i \quad (9)$$

Note that the decision variables $f_i(t)$ and $z_{i,t}$ can be eliminated from the above formulation by substituting the expressions (6) and (7) into (4) and (5), respectively. In addition, by restricting the start and output variables to be integers. The inventory and backorders variables will automatically be integer variables if the demand is integral.

If each activity has a unique resource, if all end products have no intermediate products in common, and if all processing times are integer multiples of the planning period time length, the above formulation reduces to a separate minimum cost network flow problem for each end product. The integer variable constraints may be changed to simple non-negativity constraints.

A disadvantage of this model is that products are forced to wait whenever p_i is non-integer. In essence, the model artificially creates the idle time for the resources. For activity i , the proportion of forced idle time (L_i) is:

$$L_i = \frac{p_i^+ - p_i}{p_i^+}$$

The proportion of forced idle time approaches zero as the processing time increases ($\lim_{p_i \rightarrow \infty} L_i = 0$). The forced idle time is exactly zero when the processing time is an integer.

Production starts set to processing time multiples

Another disadvantage of Model PSTBP is that a good integer solution may be difficult to obtain when the processing times are non-integer because it is difficult to find a period length that is a multiple of all the processing times in the network. Model PSTPT alleviates these problems by using a grid of possible epochs for production starts equal to integer multiples of the processing time. This assumption allows modeling the workload at an activity as a constant

rate during intervals equal to the processing time. As general notation, let W_i denote the set of time epochs marking the end of a time period for activity i .

Each activity has its own workload change time grid which is the set of integer multiples of the processing time. The workload change time grid for activity i , the time points in set W_i , is shown in Figure 3. Since the time grid length is equal to the processing time for an activity, the workload is constant during any period and is denoted by $Z_{i,t/p_i}$. The cumulative number of finished units of a product is a step function with the jumps occurring at the workload change time grid points. Note that by setting the workload change time grid to these points in time, the cumulative workload will equal the cumulative outs at these time points ($S_i(t - p_i) = Z_i(t) = F_i(t)$ where $t \in W_i$).

As notation, let P_i denote the subset of the set P including only the points at which there is external demand for product i . To determine the inventory level of product i throughout continuous time, the material balance constraint for product i only needs to be evaluated at points in time where inventory is withdrawn or added. When non-integer processing times are included in the model, the workload is not constant across unit intervals. However, the points in time when production can start and the inventory at activity i increases or decreases can be identified. This set of time points is referred to as GI_i :

$$GI_i = \begin{cases} \text{all } t \in W_i & \text{flows into inventory of product } i \\ \text{all } t \in \bigcup_{\{j|\bar{a}_{ij} \neq 0\}} W_j & \text{intermediate product input by} \\ & \text{follow-on activities} \\ \text{all } t \in P_i & \text{external demand for product } i \end{cases}$$

To guarantee mass conservation of product i throughout continuous time, the inventory balance constraint need only be enforced at time points where inventory is withdrawn. That is, the set GI_i could be reduced to the set $\{(\bigcup_{\{j|\bar{a}_{ij} \neq 0\}} W_j) \cup P_i\}$. If several consecutive points in this set fall between points in W_i , only the last such point needs a material balance constraint, as material balance constraints for the preceding points lying in the same workload period for activity i are redundant. However, if such redundant constraints are eliminated from the formulation, it becomes more awkward to precisely formulate inventory costs in the objective function. The set GI_i must

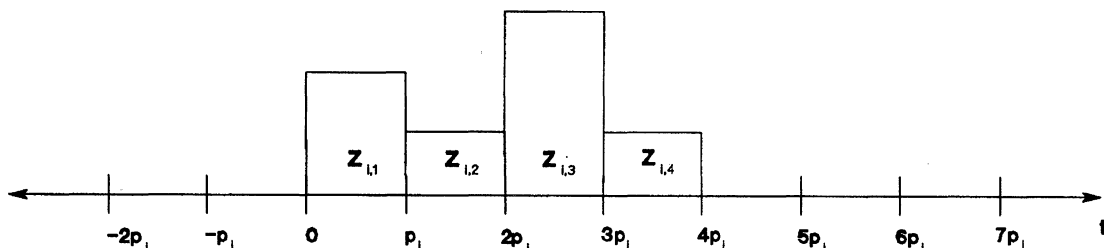


Figure 3 Time grid for activity i .

include W_i if there are upper bounds on the inventory of product i that need to be considered.

The loading rate of resource k by activity i at time t is $a_{ki} * z_{i,(t/p_i)^+}$. The available capacity at time t is c_{k,t^+} , assuming that capacities change only at the planning period time grid points. Let P_k denote the subset of P including only the points at which the capacity of resource k is changed. The capacity constraint for resource k needs to be enforced for each subinterval between the points in time when the rates $z_{i,\tau}$ and $c_{k,\tau}$ can change. This set of time points marking off these subintervals is referred to as GC_k :

$$GC_k = \begin{cases} \text{all } t \in \bigcup_{\{i|a_{ki} \neq 0\}} W_i & \text{change in application rates by} \\ & \text{activities using resource } k \\ \text{all } t \in P_k & \text{change in capacity of resource } k \end{cases}$$

Enforcing the capacity constraint for $t \in GC_k$ guarantees mass conservation of resource k throughout continuous time.

The domain constraints for Model PSTPT are simpler than for Model PSTBP. For example, the starts at the beginning of the period equal the outs at the end of the period for Model PSTPT. The workload of an activity in any period is simply the starts at the beginning of the period divided by the processing time. The complete Model PSTPT is:

Material balance

$$B_i(t) - I_i(t) + I_i(0) + F_i(t) - \sum_{j=1}^N \bar{a}_{ij} s_j(t) - D_i(t) = 0, \quad \text{all } t \in GI_i, \text{ all } i \quad (10)$$

Capacity

$$\sum_{i=1}^N a_{ki} z_{i,(t/p_i)^+} \leq c_{k,t^+}, \quad \text{all } t \in GC_k, \text{ all } k \quad (11)$$

Domain Constraint Linking Outs to Starts

$$f_i(t) = s_i(t - p_i), \quad \text{all } t \in W_i, \text{ all } i \quad (12)$$

Domain Constraints Linking Resource Applications to Starts

$$z_{i,t/p_i} = \frac{1}{p_i} s_i(t - p_i) \quad \text{all } t \in W_i, \text{ all } i \quad (13)$$

Integer Domains

$$s_i(t), f_i(t) \geq 0, \quad \text{all } t \in W_i, \text{ all } i \quad (14)$$

Non-negative Domains

$$I_i(t), B_i(t), z_{i,t/p_i} \geq 0, \quad \text{all } t \in W_i, \text{ all } i \quad (15)$$

The cumulative flows $F_i(t)$ and $S_j(t)$ are expressed in terms of $z_{i,\tau}$ and $z_{j,\tau}$, respectively. The cumulative output of activity i can be expressed as:

$$F_i(t) = p_i \left[\sum_{\tau=1}^{(t/p_i)^-} z_{i,\tau} \right]$$

That is, the cumulative output at any time is given by the cumulative workload at the next earlier workload change time grid point. The summation includes the workload for all units that were completed by time t at activity i .

The intermediate product applications term can be expressed as:

$$S_j(t) = p_j \left[\sum_{\tau=1}^{(t+p_j/p_j)^-} z_{j,\tau} \right]$$

That is, the cumulative starts at any time is given by the cumulative workload at the next higher workload change time grid point.

Assuming external demand flows are event-based flows occurring at the integer time points, the external demand term can be expressed as:

$$D_i(t) = \sum_{\tau=1}^t d_i(\tau)$$

Example problem

For comparison, the 2 processing time models are tested on a data set that is illustrative of a semiconductor manufacturing process. The example problem is defined in Tables 1 and 2. Table 1 contains data on the machines while Table 2 shows the routing data. In this example, there are 4 machine types and the number of identical machines of each type can be more than 1. The number of end products is 2, and the number of operations (activity) for each end product is 6. As the routes show, each end product can be processed by the same machine type more than once. The demand for end product 1 is 50 units and for end product 2 is 25 units. All the demand is due at the end of the planning horizon ($T=60$ h).

The primary objective of the shop is to meet all the demands by the end of the planning horizon. This objective is enforced by setting the backorder cost to a large number

Table 1 Machine data

Machine type	Capacity
1	4
2	3
3	1
4	4

Table 2 Route data

End product	Activity	Machine type	Processing time (hrs)
1	1	2	0.875
1	2	1	1.000
1	3	4	3.000
1	4	1	1.000
1	5	2	0.875
1	6	1	1.000
2	7	3	1.125
2	8	1	1.000
2	9	2	0.875
2	10	1	1.000
2	11	3	1.125
2	12	1	1.000

in the objective function. To discourage any leftover unmet demand at the end of the planning horizon, a backorder cost of 50,000 dollars per unit per hour is used for the last period. A secondary objective of the shop is to minimize the number of units in work-in-process and finished goods inventory. To minimize the inventory, an inventory carrying cost of 1.00 dollar per unit per hour is included in the objective function for all inventory variables. Assuming a period length of one hour, the objective function can be formally stated as follows.

$$\min \sum_{i=1}^2 \left\{ 50,000B_{i,60} + \sum_{t=1}^{60} I_{i,t} \right\}$$

Using Manne's model formulation for job-shops with the assumption that the lot size is one unit, each unit of demand has to be formulated as a separate job which leads to a large number of integer variables. In addition, the number of machines of the same type for this data set is greater than 1 which cannot be easily handled using Manne's model. An approximate means of using Manne's Model to model situations in which $c_k > 1$ is to set the machine capacity to 1 and divide the processing time by the capacity. Even with this approximate method, the resultant number of integer variables for this data set using Manne's Model is 35,400 while Model PSTPT has 695 integer variables.

To find a solution to the example problem, a restricted start model is needed since Manne's Model (unrestricted start) contains too many integer variables. We next

compare Model PSTPT with Model PSTBP for the example problem. Note that the formulation for Model PSTPT is independent of the planning period length since for this model the time grid for each activity is a function of the activity's processing time. For Model PSTPT, the number of integer variables is given by $\sum_{i=1}^N \lceil T/p_i \rceil$. However, the formulation for Model PSTBP is clearly a function of the selected planning period length. For Model PSTBP, the number of integer variables is given by $N\lceil T/g \rceil$, where g is the length of the planning grid. A smaller planning time grid length, g , improves the quality of the solution given by the model at the expense of increasing the problem size. We compare three different planning period lengths for Model PSTBP on the sample problem. They are 1, 0.25, and 0.125 hour, respectively. An optimal solution to Model PSTBP using an interval of 0.125 is the only model of the four tested models (the three PSTBP models and one PSTPT model) in which it is ensured that it will also be an optimal solution to an unrestricted start model, such as Manne's model formulation, for this sample problem because the period length is an integer multiple of all the processing times.

The experiments are run using CPLEX¹³ on an IBM RS/6000 Workstation. CPLEX uses a branch-and-bound procedure to find an optimal solution to the problem. The stopping criteria used in the search is 2 CPU hours. The results are summarized in Table 3. For each model, the best integer solution found, the tightest lower bound, the number of integer variables, and the number of inventory constraints are recorded. As the table shows, an optimal solution is found to the restricted start Model PSTPT. For all the PSTBP models the 2 CPU hours time limit is reached before finding an optimal solution. To compare the quality of the best integer solution CPLEX found, we also show the tightest lower bound found by CPLEX. Model PSTBP with period lengths of 1 and 0.25 have a large total cost because for these models not all the demand could be met by the end of the planning horizon resulting in excessive backorder costs.

The total cost from the optimal solution given by Model PSTPT is \$1991.10. Using a period length of 1 hour, the number of integer variables in Model PSTBP is 720 about the same amount as in Model PSTPT. However, the resultant forced idle time in the schedule generated from Model PSTBP is too large when the period length is 1 hour

Table 3 Results of example problem

Model	Period length	Best integer solution	Tightest lower bound	No. of integer variables	No. of inventory constraints
PSTPT	1.000	1991.1	1991.1	695	1350
PSTBP	1.000	551366.0	526390.0	720	720
PSTBP	0.250	26451.0	20214.4	2880	2880
PSTBP	0.125	1561.7	1527.5	5760	5760

so that the entire demand cannot be met within the planning horizon. The solution in Model PSTBP cannot fulfill 11 units of demand for end product 2 while the entire demand is satisfied in the solution of Model PSTPT. Using a period length of 0.25 h, the number of units of unmet demand in the solution of Model PSTBP is 2. The number of integer variables using an interval of 0.25 h in Model PSTBP is 2880 which is more than four times the number in Model PSTPT and yet yielding an inferior solution to Model PSTPT. The solution from Model PSTBP using an interval of 0.125 can meet the entire demand. The resultant total cost for this model is the smallest with a cost of \$1561.70. However, the number of integer variables in this formulation is 5760. These results show that for the same problem size Model PSTPT outperforms Model PSTBP but the solution quality for Model PSTBP may be improved by increasing its problem size through using a smaller planning time grid. The tradeoff is solution quality versus problem size.

To compare the difference between the first integer solution and the best integer solution found by CPLEX, Table 4 displays for each model the first integer solution found, the number of nodes in the branch-and-bound procedure evaluated to find the first integer solution, and the total number of simplex iterations. As the problem size increases the more iterations are required to find the first integer solution. The interesting aspect of the table is that the first integer solution found is very close to the best integer solution found from Table 3, and the computational requirements to find the first integer solution is significantly less. This result suggests that for practical applications it may be sufficient to stop the branch-and-bound procedure after the first integer solution is found.

In general, schedules generated by Models PSTBP and PSTPT are suboptimal schedules to an unrestricted start system because both models hold products in inventory longer than necessary, but Model PSTPT is typically less restrictive in this sense. The solution to Model PSTBP is optimal to the unrestricted start system when all the processing times are integer multiples of all the processing times of the activities. In practice, it is difficult to find a period length that is an integer multiple of all the processing times while still being computationally feasible to solve. Thus, the selection of the period length is very important in determining good schedules. Model PSTPT

Table 4 Summary of first integer solution found

Model	Period length	First integer solution	Number of nodes	Number of iterations
PSTPT	1.000	2006.2	25	2892
PSTBP	1.000	551379.0	19	2395
PSTBP	0.250	26451.0	33	42451
PSTBP	0.125	1561.7	186	266378

remedies this problem by defining a separate time grid for each activity.

Conclusion

Traditional integer programming models for job-shops and flow-shops do not easily account for characteristics common in semiconductor manufacturing such as multiple machines of the same type, demand size greater than one unit for a particular product type, and repeat visits to the same machine type. Two alternative integer programming model formulations are presented that easily account for these characteristics. They are especially useful for modeling high production volume facilities since the problem size of the presented models is not a function of the demand size. The models differ in their assumption for the allowed epochs for production starts at an operation. The drawback with the restricted start models is that the optimal solutions from these models may be suboptimal when applied to an unrestricted start system such as job-shops. For the restricted start models, there is typically a trade-off between the quality of the solution and the problem size of the model, especially for Model PSTBP.

References

- Shephard RW (1970). *Theory of Cost and Production Functions*. Princeton University Press: Princeton, New Jersey.
- Hackman ST and Leachman RC (1989). A general framework for modeling production. *Mgmt Sci* 35: 478–495.
- Wagner HM (1959). An integer programming model for machine scheduling. *Naval Res Logist Q* 6: 131–140.
- Manne AS (1960). On the job-shop scheduling problem. *Opns Res* 8: 219–223.
- Pritsker AAB, Watters LJ and Wolfe PM (1969). Multiproject scheduling with limited resources: a zero-one programming approach. *Mgmt Sci* 16: 93–108.
- Selen WJ and Hott DD (1986). A mixed-integer goal-programming formulation of the standard flow-shop scheduling problem. *J Opl Res Soc* 37: 1121–1128.
- Stafford EF (1988). On the development of a mixed-integer linear programming model for the flowshop sequencing problem. *J Opl Res Soc* 39: 1163–1174.
- Wilson JM (1989). Alternative formulations of a flow-shop scheduling problem. *J Opl Res Soc* 40: 395–399.
- Liao C-J and You C-T (1992). An improved formulation for the job-shop scheduling problem. *J Opl Res Soc* 43: 1047–1054.
- French S (1987). *Sequencing and Scheduling: An Introduction to the Mathematics of the Job-Shop*. Ellis Horwood: England.
- Mizrach RR (1985). Dynamic models of detailed and aggregate production networks. Unpublished PhD dissertation, University of California: Berkeley, California.
- Dessouky MM and Leachman RC (1994). An optimization-based methodology for release scheduling. *Prod Ops Mgmt* 3: 276–295.
- CPLEX Optimization (1996). 930 Tahoe Blvd # 802-279, Incline Village, NV, 89451.

Received June 1996;
accepted January 1997 after one revision