Int J Adv Manuf Technol (2005) 27: 145–151
DOI 10.1007/s00170-004-2152-1

ORIGINAL ARTICLE

Muh-Cherng Wu · Yai Hsiung · Hsi-Mei Hsu

Tool planning in the scenario of multiple existing semiconductor fabs

Received: 25 November 2003 / Accepted: 21 February 2004 / Published online: 3 August 2005

Abstract The tool planning problem involves determining how many tools should be allocated to each tool group to achieve a set of objectives. Most previous studies assume that a demand mix has been given for a new factory to be planned. However, when a semiconductor company has several existing fabs, the demand mix for the new fab is not explicitly known, and needs to be allocated from the demand mix of the whole company. This paper presents an integrated approach to determine the optimal demand mix and associated tool plan for the new fab that can minimize the tool cost of the new fab while each fab (new or existing) is requested to meet a predefined target in its mean cycle time. Simulation experiments indicate that the proposed solution is better than that obtained by a heuristic method used in industry. The saving in tool cost for a typical tool planning problem can be over US$ 70 million.

Keywords Capacity planning · Cycle time · Genetic algorithm · Tool planning

1 Introduction

Tool planning is a decision problem of determining how many tools to purchase for each tool group (workstation) in planning a new factory. The decision problem is very important in particular for the semiconductor industry, in which a typical factory (fab) costs over US $1 billion. Finding an optimal solution for the tool planning problem is therefore very important. The saving in tool cost can easily be over US $1 million. In the following presentation, a tool allocation plan in the tool planning problem is called a toolset, while some previous studies may call it a tool portfolio [1].

M.-C. Wu (✉) · Y. Hsiung · H.-M. Hsu
Department of Industrial Engineering and Management,
National Chiao Tung University,
Hsin-Chu, Taiwan, R. O. C.
E-mail: mcwu@cc.nctu.edu.tw
Tel.: +886-35-731913
Fax: +886-35-720610

Compared to other production systems, semiconductor manufacturing is relatively complex. The routing for manufacturing a typical wafer includes about 400–500 operations processed by about 100 tool groups. A collection of 25 wafers, called a lot, moves among tool groups with the re-entrant feature; that is, a lot may visit a tool group several times. Moreover, some random factors such as tool breakdown occur frequently and cannot be ignored in evaluating the performance of a toolset.

In solving tool-planning problems, most previous studies adopted or developed models to evaluate the performance of a toolset. Based on the evaluated performance of a toolset, they subsequently developed methods to search for a better toolset. The evaluation-and-search procedure is repeatedly performed until the desired toolset is found.

Computer simulation and queuing network models are two commonly used methods in previous literature to evaluate the performance of a toolset. Computer simulation provides precise modeling but requires lengthy computation. Queuing network models can rapidly give performance estimates but the estimates may be less accurate than those obtained by simulation.

Based on simulation models, some works regarding tool planning have been published. Grew et al. [2] presented a marginal allocation procedure that updates the candidate toolset by adding one tool at a time. Mollaghasemi and Evans [2] developed an interactive method to find a desired toolset to meet multiple objectives. Chen and Chen [4] presented an experimental design approach, combined with a response surface methodology, to identify a satisfactory toolset.

Various studies for tool planning adopting or developing queuing models have been proposed. Yoneda et al. [5] presented a simulated annealing approach. Bretthauer [6] developed a branch and bound algorithm. Connors et al. [7] proposed a marginal allocation algorithm. Kao et al. [1] developed equi-throughput curves to identify a toolset that meets the planner's criteria.

Some other studies on tool planning do not include either simulation or queuing models. These studies implicitly assume that a toolset is always acceptable as long as its available machine hours are higher than the aggregated demanding hours.

Based on this assumption, Swaminathan [8] formulated the tool planning problem in the context of multiple product mixes with associated occurrence probabilities by a mixed integer programming model. Such a model is fast in computation; however, it cannot guarantee to yield a toolset whose manufacturing cycle time is below a predefined target.

All of the aforementioned literature assumes that a certain demand mix or some probabilistic demand mixes for the new fab have been given. Here, a demand mix denotes the monthly quantity to be produced for each product. For example, the demand mix of three products for a new fab is known to be $(A,\ B,\ C) = (300$ lots, 200 lots, 250 lots). Given a demand mix, the previous studies propose methods to determine an optimum toolset to achieve a set of objectives. However, in the real world, the demand mix for planning a new fab may not be explicitly known, and one needs a method to identify which of a large number of options is the best for planning the new fab.

In the semiconductor industry, a well-established company usually has several existing fabs. Each existing fab was originally planned based on a particular demand mix. However, the demand mix for the whole company may have significantly changed by the time a new fab is to be planned. Each existing fab therefore cannot be operated with its optimal demand mix. To plan the new fab, the semiconductor company has to consider how to optimally allocate the whole demand mix to the new fab and other existing fabs. Such a demand allocation problem for tool planning indeed exists and is significant in the real world but has not been addressed in the literature.

This study presents a solution methodology for the tool planning of a new fab in the context of multiple existing fabs, where the mean cycle time of each fab must be under a predefined target value. The solution methodology comprises a genetic algorithm (GA) technique to identify a set of good demand allocation patterns. The set of good demand allocation patterns is then exhaustively evaluated in order to yield the optimal toolset for the new fab. A queuing model and its application to tool planning, developed by Connors et al. [7], are adopted to evaluate the quality of a demand allocation pattern. Test results show that the solution obtained by the proposed method is superior to that obtained by a heuristic method used in industry. The saving in tool cost can be over US $70 million in the testing examples.

The remainder of this paper is organized as follows: Sect. 2 introduces the input/output relationship of the adopted queuing model and its application to tool planning [7]. Section 3 presents the problem formulation. Section 4 describes the solution methodology based on the GA technique. Section 5 presents the results of some testing experiments, and concluding remarks are given in Sect. 6.

2 Review of tool panning for a given demand mix

Developing queuing models for estimating the performance of a toolset have been studied extensively. Some of them have been verified to be quite effective for tool planning in both computation time and solution quality. To avoid repeating earlier work, this paper focuses on developing a solution methodology that integrates a GA technique and existing queuing models to solve the tool planning problem in the scenario of multiple existing fabs. The queuing model specifically for semiconductor fabs, developed by Connors et al. [7], is adopted here as a vehicle for evaluating the mean cycle time of a toolset.

Inclusion of the queuing model in the proposed solution methodology is discussed below. Let $Q_i = [q_{i1}, \ldots, q_{ik}]$ represent the monthly quantity to be produced at a fab F_i, where q_{ij} denotes the production quantity of product j $(j = 1, \ldots, k)$ at fab F_i. Let $X_i = [x_{i1}, \ldots, x_{im}]^T$ represent a toolset of fab F_i, where x_{ij} denotes the number of tools in tool group j $(j = 1, \ldots, m)$; $C = [c_1, c_2, \ldots, c_m]$ denotes the tool cost vector, where c_i is the procurement cost per tool for tool group i $(i = 1, \ldots, m)$.

The input and output relationship of the queuing model in [7] can be formulated in Eq. 1. This equation shows that given a product demand mix (Q_i), the queuing model f_q can quickly compute the mean cycle time CT_i for a toolset X_i. The subscript q denotes that the function f_q is a queuing model.

$$CT_i = f_q(X_i, Q_i) \tag{1}$$

Based on the queuing model in Eq. 1, Connors et al. [7] formulated a single-site tool planning problem as follows, where Q_n is the given demand mix for the new fab F_n and CT_0 is the target of the mean cycle time of the fab.

Minimize $C \cdot X_n$

such that $CT_n = f_q(X_n, Q_n) \le CT_0$

$x_{nj} \in Z^+$

They subsequently proposed a marginal allocation algorithm [7] to find the optimum solution of X_n efficiently. The basic idea of the algorithm is to add one tool at a time to the tool group that yields greatest overall reduction in cycle time at minimal cost until the mean cycle time falls below the threshold. The marginal allocation algorithm is described below.

Step 1: Set an initial toolset for the starting search
$Y_0 = (1, 1, \ldots, 1);\ X_n = Y_0$

Step 2: Compute the mean cycle time of X_i
$CT_n = f_q(X_n, Q_n)$

Step 3: Check whether X_n is the solution
If $CT(X_n) \le CT_0$ then X_n is the solution, stop.
Else, continue

Step 4: Update X_n
Determine the tool group that reduces cycle time most effectively:

$$k = \arg\left\{\max_j \left(\frac{f_q(X_n, Q_n - f_q(X_n + E_j, Q_n))}{c_j}\right)\right\},$$
$$j = 1, \ldots, m$$

where E_j is the unit vector in the jth direction.
Let $X_n = X_n + E_k$. Go to step 3

The input/output relationship of the tool planning problem stated above can be formulated as $X_n = f_{qm}(Q_n, CT_0)$. That is, given a demand mix Q_n and a target mean cycle time CT_0 for a new fab F_n, the method proposed by Connors et al. [7] can yield a toolset X_n that minimizes tool cost. The two subscripts of f_{qm} denote that the solution method involves a queuing model and a marginal allocation algorithm.

3 Problem formulation

The tool planning problem investigated in this research is formulated under the following assumptions: the semiconductor company produces k products whose routings require at most m tool groups. The company currently has $n-1$ existing fab sites, F_i $(i = 1, \ldots, n-1)$. Suppose the future monthly demand pattern of the company is forecasted to be $D = (d_1, d_2, \ldots, d_k)$, where d_j denotes the demand quantity of product j. The future demand is so large that a new fab F_n needs to be constructed.

Each existing fab F_i, when constructed, was optimally designed for a given demand mix $P_i = (p_{i1}, p_{i2}, \ldots, p_{ik})$, where p_{ij} represents the monthly throughput of product j in fab F_i. Due to a change in the market at the time when a new fab is to be established, each existing fab F_i may have to produce a demand mix other than P_i. Let $Q_i = (q_{i1}, q_{i2}, \ldots, q_{ik})$ represent the demand quantity allocated to fab F_i when the demand for the whole company is D; that is, $D = \sum_{i=1}^{n} Q_i$ and $d_j = \sum_{i=1}^{n} q_{ij}$ $(j = 1, \ldots, k)$.

Let $C = (c_1, \ldots, c_j, \ldots, c_m)$ represent the tool unit cost vector, where c_j is the cost per tool for tool group j. Suppose $X_i = (x_{i1}, \ldots, x_{im})^T$ represent the toolset of fab F_i, where x_{ij} represents the number of tools in tool group j of fab F_i. The company requests that the mean cycle time of each fab should be operated under a target value CT_0. The decision problem involves determining how to allocate the company demand D to the fab demand Q_i $(i = 1, \ldots, n)$ so that the tool cost of the new fab X_n is minimized while each fab meets the cycle time constraint.

The tool-planning problem for constructing a new fab in a company with multiple existing fab sites therefore can be formulated as follows:

$$\text{Minimize } C \cdot X_n$$

$$\text{such that } D = \sum_{i=1}^{n} Q_i \tag{2}$$

$$CT(Q_i, X_i) \le CT_0, \quad i = 1, \ldots, n \tag{3}$$

$$x_{nj} \in Z^+, \quad j = 1, \ldots, m \tag{4}$$

In the above formulation, D and X_i $(i = 1, \ldots, n-1)$ are given, while X_n and Q_i $(i = 1, \ldots, n)$ denote the decision variables. Let $Q = [Q_1, \ldots, Q_n]^T$ represent a candidate solution for the demand allocation to each fab. Notice that Q is a $n \times k$ matrix, where the ith row $Q_i = [q_{i1}, q_{i2}, \ldots, q_{ik}]$ represents the allocated demand of fab F_i. The decision variables of the problem can therefore be denoted by X_n and Q. The objective is to minimize the tool cost of X_n.

4 Solution method

The investigated tool-planning problem as formulated in Sect. 3 is much more complicated than that in the previous literature, formulated in Sect. 2. It requires the development of a method to effectively allocate the company demand D to fab demand Q_i. This study develops a genetic algorithm (GA) approach to solve the problem, which comprises the following two procedures.

The first procedure is to select a quantity M "good" Q matrices and put them in a set, called the X-space. A good Q matrix has two properties. First, Q_i of each existing fab F_i $(i = 1, \ldots, n-1)$ has to meet the cycle time constraint formulated in Eq. 3, yet Q_n of the new fab F_n does not need to meet this constraint. Second, the tool cost of Q_n without meeting any cycle time constraint, estimated by a heuristic method, should be relatively low. The heuristic method will be presented below.

The second procedure is to determine the optimum Q from the X-space. Notice that each Q matrix has a corresponding Q_n vector. By applying the marginal allocation procedure developed by Connor et al. [7], a Q_n can be used to yield an associated minimum cost toolset X_n that meets the cycle time constraint formulated in Eq. 3. Since there is a quantity M of Q matrices in the X-space, there is correspondingly quantity M of X_n toolsets. Among the M toolsets, the X_n that minimizes tool cost is the optimum toolset for fab F_n, and the corresponding Q is the optimum allocation of demand D.

This study develops a GA technique to determine the X-space, which stores the M good Q matrices. GA techniques were first proposed in the early 1980s [9]. Various applications have shown that GAs are powerful techniques for effectively and efficiently solving large scale space-search problems [10].

A GA is an iterative procedure that maintains a constant-sized population $P(t)$ of candidate solutions (called chromosomes). During each iteration step t, called a generation, new chromosomes are created by invoking genetic operators. Each existing and newly generated chromosome is evaluated to determine its fitness value, which denotes how good the solution is. Based on these evaluations, a set of chromosomes are screened out by a selection procedure to form the new population $P(t+1)$. The procedure is iteratively performed until the termination conditions are met.

The methods in our GA for creating chromosomes, genetic operators, the fitness function, the selection strategy for forming new population, and termination conditions are presented below. The method for creating the X-space is also described.

4.1 Representation and generation of chromosomes

A chromosome in our GA is the Q matrix defined above, where each element q_{ij} (a positive integer) is called a gene. Let N_p be the total number of chromosomes in the population $P(t)$. The initial population $P(0)$ is created by randomly generating N_p chromosomes. Notice that some of the randomly generated chromosomes may not be put in $P(t)$. In our GA, only the chromosome Q whose first $n-1$ row vectors Q_i $(i = 1, \ldots, n-1)$

meet the cycle time constraint formulated in Eq. 3 can be put in the population $P(t)$. Such a chromosome hereafter is called a valid chromosome; otherwise, it called an invalid chromosome.

In generating a chromosome Q, the value of each gene q_{ij} is randomly chosen from the interval $[LB(q_{ij}), UB(q_{ij})]$, where $LB(q_{ij})$ and $UB(q_{ij})$ denote the lower and upper bounds of q_{ij}, respectively. The lower bound $LB(q_{ij})$ is set to 0. The method for determining $UB(q_{ij})$ is presented below.

Let q_{ij}^u represent the maximum throughput of fab F_i when it produces only product j and satisfies the cycle time constraint formulated in Eq. 3. q_{ij}^u can be easily obtained by increasing q_{ij} stepwise and performing the queuing model in [7]; the search of q_{ij}^u can also be faster by applying the binary search method. To determine $UB(q_{ij})$ for a particular product j for each existing fab F_i $(i = 1, \ldots, n-1)$, the $n-1$ fabs are first randomly ordered. Let W_s $(s = 1, \ldots, n-1)$ represent the sequentially ordered fabs, where the function $\phi(s) \rightarrow i$ defines the mapping of the two subscripts. The procedure to determine $UB(q_{ij})$ $(i = 1, \ldots, n-1)$ of a chromosome, for product j $(j = 1, \ldots, k)$, is expressed below.

Step 1: Randomly generate a sequential order for the $n-1$ existing fabs, $\phi(s) \rightarrow i$
Step 2: Compute q_{ij}^u for each existing fab F_i, $(i = 1, \ldots, n-1)$
Step 3: $s = 1$, $H = 0$
While $s < n$, do
{
$h = \phi(s)$
$UB(q_{hj}) = \text{Min}\{q_{hj}^u,\ d_1 - H\}$
Randomly select a value v_{hj} for gene q_{hj} from the interval $[0, UB(q_{hj})]$
$q_{hj} = v_{hj}$
$H = H + v_{hj}$
$s = s + 1$
}
Step 4: Determine the value of q_{nj}
$q_{nj} = d_j - \sum_{i=1}^{n-1} q_{ij}$

The above procedure for determining $UB(q_{ij})$ aims to reduce the search space of genes and maintain the constraint $\sum_{i=1}^{n} q_{ij} = d_j$. Notice that $UB(q_{ij})$ creates different chromosomes because the random ordering procedure in step 1 is independently performed. This random ordering procedure aims to prevent our GA from being trapped in a local optimum solution.

4.2 Fitness function

The fitness function is defined to evaluate the quality of a chromosome. In our GA, each chromosome Q yields an nth row vector Q_n, the allocated demand of new fab F_n. The fitness function of Q is defined as the estimated tool cost of producing Q_n when a cycle time constraint is not considered. The study proposes the following heuristic method to estimate such a tool cost of Q_n.

Let t_{ij} denote the required processing time of tool group i in producing one unit of product j. To produce Q_n, the total required processing time of tool group i is $T_{ni} = \sum_{j=1}^{k} q_{nj} \cdot t_{ij}$. Let H_i denote the available machine hours per tool for tool group i. Then, $y_{ni} = T_{ni}/H_i$ $(i = 1, \ldots, m)$ denotes the lower bound or the least number of tools required for tool group i to produce Q_n. Notice that $y_{ni} \in R^+$. Let $Y_n = [y_{n1}, \ldots, y_{nm}]^T$. The fitness function of a chromosome Q is defined as $C \cdot Y_n$, the lower bound of tool cost for producing Q_n.

4.3 Crossover and mutation operators

The proposed GA defines two genetic operators known as crossover and mutation to create new chromosomes.

The crossover operator is designed to create $N_p \times P_{cr}$ new chromosomes in each generation, where P_{cr} is a predefined crossover probability. This operator is applied by first randomly choosing $(N_p \times P_{cr})/2$ pairs of chromosomes from $P(t)$. For each pair of chromosomes (matrices), a column position in the matrix (called the crossover line) is randomly chosen, and the segments to the right of the crossover line exchanged.

To facilitate presenting the crossover operation, let A_i represent a chromosome. Suppose a pair of chromosomes for crossover to be A_1 and A_2 shown below.

$$A_1 = \begin{bmatrix} q_{11} q_{12} q_{13} q_{14} \\ q_{21} q_{22} q_{23} q_{24} \end{bmatrix} \quad A_2 = \begin{bmatrix} q'_{11}\ q'_{12}\ q'_{13}\ q'_{14} \\ q'_{21}\ q'_{22}\ q'_{23}\ q'_{24} \end{bmatrix}$$

If the crossover line is located between columns 3 and 4, then two new chromosomes A_3 and A_4 can be generated, as shown below.

$$A_3 = \begin{bmatrix} q_{11} q_{12} q_{13} q'_{14} \\ q_{21} q_{22} q_{23} q'_{24} \end{bmatrix} \quad A_{\text{new}} = \begin{bmatrix} q'_{11}\ q'_{12}\ q'_{13}\ q_{14} \\ q'_{21}\ q'_{22}\ q'_{23}\ q_{24} \end{bmatrix}$$

The mutation operator is applied by first randomly selecting $N_p \times P_{mu}$ new chromosomes from $P(t)$, where P_{mu} is a predefined probability of mutation. For each selected chromosome Q, $k \times (n-1)$ new chromosomes are created by replacing the value of each gene q_{ij} $(1 \le i \le n-1$ and $1 \le j \le k)$ by an integer q'_{ij} randomly chosen from the interval $[LB(q_{ij}), UB(q_{ij})]$. The gene q_{nj} for new fab F_n is subsequently replaced by a value $q_{nj} + (q_{ij} - q'_{ij})$ in order to ensure that $\sum_{i=1}^{n} q_{ij} = d_j$ in these newly created chromosomes. The number of new chromosomes created by mutation is then $N_p \times P_{mu} \times k \times (n-1)$.

The new chromosomes created by crossover and mutation may not be valid; that is, they may not satisfy the cycle time constraint in Eq. 3. These invalid chromosomes are scrapped, and only valid chromosomes are kept.

4.4 Selection strategy

The chromosomes in population $P(t)$ together with the valid chromosomes created by crossover and mutation are put in a pool. Let S represent the pool in which the number of chromosomes is h. N_p chromosomes are to be selected from S to form the population $P(t+1)$. The selection strategy used in this paper is based on the rank-space method [11], which was developed to prevent the genetic search from becoming trapped at a local

optimum solution. The procedure of the rank-space method is presented below.

Step 1: Sort in descending order the chromosomes in S according to their fitness values. Let $A_1, A_2, \ldots, A_h$ be the sorted result. Such a ranking of A_i, termed *quality-ranking*, is represented by $R_q(A_i)$

Step 2: Move the best quality-ranking chromosome from S to $P(t+1)$:
Let $S = S - \{A_1\}$;
$P(t) = P(t+1) + \{A_1\}$;
$B_1 = A_1$.
Rename the chromosome selected for $P(t+1)$:
$N = 1$.
The chromosome number in $P(t+1)$ is 1

Step 3: For each chromosome A_i in S, compute the diversity index $D(A_i)$:
$D(A_i) = \sum_{k=1}^{N} \frac{1}{|A_i - B_k|}$.
B_k is a chromosome in $P(t+1)$.
Let $A = [a_{ij}]$, $B = [b_{ij}]$ $1 \le i \le n$; $1 \le j \le k$.
$|A - B| = \left(\sum_{i=1}^{n} \sum_{j=1}^{k} (a_{ij} - b_{ij})^2\right)^{1/2}$

Step 4: Sort in ascending order the chromosomes in S according to $D(A_i)$. Such a ranking of A_i, termed diversity-ranking, is represented by $R_d(A_i)$

Step 5: Compute the sum of quality-ranking and diversity-ranking of A_i in S:
$T(A_i) = R_q(A_i) + R_d(A_i)$

Step 6: Sort in ascending order the chromosomes in S according to $T(A_i)$. Such a ranking of A_i, termed combined-ranking, is represented by $R_c(A_i)$.

Step 7: For each chromosome in S, compute the probability of putting A_i in $P(t+1)$:
Let $r = R_c(A_i)$;
and $\text{Prob}(A_i) = p \cdot (1-p)^{r-1}$.
p is a predefined probability, typically set to 0.667

Step 8: Generate a random number and determine which chromosome in S is selected. Let A_m be the selected chromosome. Move A_m from S to $P(t+1)$:
$S = S - \{A_m\}$;
$P(t) = P(t+1) + \{A_m\}$;
and $N = N + 1$.
Update the chromosome number in $P(t+1)$:
$B_N = A_m$.
Rename the chromosome selected for $P(t+1)$

Step 9: Termination check
If $N < N_p$, then go to step 3.
Else, stop

4.5 Terminating conditions

Population $P(t)$ is iteratively updated until the following termination condition is met: let N_G be a predefined large integer and $BS(t)$ be the best solution in each population $P(t)$. Our GA stops when a particular $BS(t_0)$ keeps the best solution for over N_G generations. That is, if $BS(t) = BS(t+1)$ for $t_0 \le t \le t_0 + N_G$, then the GA stops at $t_0 + N_G$; $BS(t_0 + N_G)$ is the final solution.

4.6 Updating the X-space

Notice that in the generation of each $P(t)$, the X-space needs to be continuously updated. The X-space stores M chromosomes, which, when evaluated by the fitness values, are the top M number of chromosomes among all valid chromosomes that have been visited by our GA. The X-space will be used to find the optimum solution (Q, X_n) for the investigated tool planning problem.

Let A_i $(i = 1, \ldots, M)$ represent the M chromosomes in the X-space, and R_{ni} represent the nth row vector of A_i; that is, the demand allocated to fab F_n. Given an R_{ni} vector, the marginal allocation algorithm [7] can be used to determine a toolset X_{ni} which minimizes tool cost and meets the cycle time constraint formulated in Eq. 3; that is, $CT(R_{ni}, X_{ni}) \le CT_0$. The optimum solution of the research problem is the chromosome A_s, the minimum one in tool cost in the X-space, as expressed below.

$$s = \text{Arg}\{\text{Min}_{1 \le i \le M}(C \cdot X_{ni})\}$$

5 Testing examples

The performance of the proposed solution methodology is evaluated by solving the following tool-planning problem in which the routing and tool cost data was provided by a real semiconductor company in industry: the semiconductor company of interest has two existing fabs *Fab_1* and *Fab_2*, and is planning a new fab *Fab_3*. The three fab sites are so far apart that each site cannot use the tools of other sites as backup. That is, a product is completely manufactured at one fab; it is impossible to manufacture some operations of the product at a fab and manufacture other operations at another fab.

The company produces four product families: U, V, W, and Z. A typical product in each product family is chosen to represent the family. Each routing of these four representative products includes 400–500 operations. The mean processing time of these four representative products is $PT_0 = 10$ days.

Fab_1 was optimally designed to produce the monthly demand mix $(U : V : W : Z) = (360, 340, 480, 0)$. That is, the total monthly output involves 360 lots U, 340 lots V, and 480 lots W. *Fab_2* was optimally designed for producing the product mix $(U : V : W : Z) = (600, 360, 240, 0)$. When *Fab_1* and *Fab_2* were planned, new product Z was not available; therefore, their demand mixes did not include product Z.

Due to a change in market, the demand of product W is diminishing. The future monthly demand for the company is forecasted to be $(U : V : W : Z) = (1200, 1200, 0, 1200)$. The amount of future demand is so high that the company needs to establish a new fab *Fab_3*. Without the demand of product W, the demand mix for the company is now significantly different from those of *Fab_1* and *Fab2*. To plan the toolset of *Fab_3*, the company needs to re-allocate the demand to each fab site.

The company requests that the mean cycle time of each fab should be less than a target CT_0. The decision problem is figuring out how to allocate the demand to each fab so that *Fab_1* and

Fab_2 can produce the allocated demand in the targeted mean cycle time and the tool cost of *Fab_3* is minimized.

The performance of the proposed method will be compared with that of a heuristic method used in industry. The heuristic method involves requesting each fab to produce a demand mix that is as close as possible to the originally designed demand mix. Taking products U and V as examples (Table 1), *Fab_1* and *Fab_2* produce exactly the same amount as the originally designed demand mix. Given allocated demands of product U and V, the maximum quantity of product Z that can be allocated to *Fab_1* can be determined by repeatedly performing the queuing model in [7]. That is, by trying various amounts of product Z allocated to *Fab_1*, the queuing model yields various CT_1. The maximum quantity of product Z that can be allocated is the amount that makes CT_1 closest to CT_0.

Table 1 compares the performance of the two methods for tool planning when $CT_0 = 30$ days. The saving in tool cost by applying the proposed method is about US $76 million, about a 5.2% difference.

The proposed method was coded in C^{++} language and performed on a Pentium 4 computer. In the GA, the crossover rate was set to $P_{cr} = 0.6$, the mutation rate to $P_{mu} = 0.1$, and the population size to $N_p = 100$. The search terminates when a particular toolset maintains the best solution for over 500 generations.

The proposed method involves two procedures. The first procedure is to find the X-space, which stores 500 good demand allocation portfolios. For each demand allocation portfolio in the X-space, the second procedure is to compute the toolset of *Fab_3* that minimizes tool cost and should meet the cycle time constraint. For a demand portfolio in the X-space, the tool cost computed by the first procedure is called the initial tool cost. The tool cost computed by the second procedure is called the final tool cost.

In the first procedure, the number of visited demand portfolios is about 2.4×10^5. The GA search proceeds for about 1,000 generations before it terminates. The computation time for the GA search to identify the X-space is about three hours. About three hours are required for the computation of toolsets for 100 demand portfolios in the X-space, and about 15 hours are required for 500 demand portfolios. The total computation time seems acceptable for a long-term decision-making problem such as tool planning.

The initial tool cost and the final tool cost for the 500 demand portfolios in the X-space are shown in Fig. 1, which shows

Table 1. Comparison of performances for $CT_0 = 30$ days

CT_0 = 30 days	Proposed method	Heuristic method
Fab_1: (U, V, W, Z)	(383, 393, 0, 98)	(360, 340, 0, 110)
Fab_2: (U, V, W, Z)	(606, 394, 0, 108)	(600, 360, 0, 115)
Fab_3: (U, V, W, Z)	(211, 413, 0, 994)	(240, 500, 0, 975)
Tool cost for *Fab_3*	US $1,461 million	US $1,537 million
Tool cost difference	0	US $76 million
Cost reduction by %	0%	5.2%
Final tool number	517	548

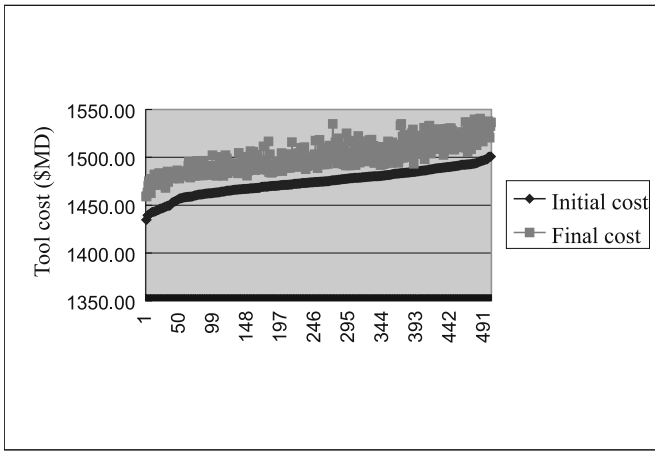

Fig. 1. Tool cost of 500 candidates

that the distribution of the final tool cost is quite consistent with that of the initial tool cost. This implies that we cannot find a better toolset by increasing the number of selected demand portfolios. In fact, this testing problem shows that the optimum demand portfolio appears in the first 100 demand portfolios. We can therefore reasonably claim that the proposed method can find the optimum solution.

6 Concluding remarks

This study presents a methodology for solving the tool planning problem for a semiconductor company that has several existing fabs. The demand mix for the whole company is known and must be allocated to the new and existing fabs before proceeding with the task of tool planning. The proposed methodology develops a GA technique to identify some hundreds of good demand allocation patterns, and put them in an X-space. A good demand allocation pattern is one in which each existing fab meets the cycle time requirement, and the new fab is relatively low in tool cost without meeting the cycle time requirement. The optimum demand allocation pattern is then identified from the X-space by exhaustively evaluating the tool cost of each member with the cycle time requirement.

Experiments show that the tool cost with cycle time constraint is highly correlated with that without cycle time constraint. We therefore can reasonably claim that the proposed method is valid in identifying the optimum solution. The saving in tool cost by the proposed method exceeds a heuristic method by over US $70 million.

References

1. Kao CE, Chou YC (1999) A tool portfolio planning methodology for semiconductor wafer fabs. IEEE Proceedings of the eighth international symposium on semiconductor manufacturing, pp 84–90

2. Grewal NS, Bruska AC, Wulf TM, Robinson JK (1998) Integrating targeted cycle-time reduction into the capital planning process. Proceeding of the 1998 winter simulation conference, pp 1005–1010
3. Mollaghsemi M, Evans GW (1994) Multicriteria design of manufacturing systems through simulation optimization. IEEE Trans Syst Man Cybern 24(9):1407–1411
4. Chen LH, Chen YH (1996) A design procedure for a robust job shop manufacturing system under a constraint using computer simulation experiments. Comput Ind Eng 30(1):1–12
5. Yoneda K, Wada I, Haruki K (1992) Job shop configuration with queueing networks and simulated annealing. Proceedings of the IEEE international conference on systems engineering, pp 407–410
6. Bretthauer KM (1996) Capacity planning in manufacturing and computer networks. Eur J Oper Res 91:386–394
7. Connors DP, Feigin GE, Yao D (1996) A queueing network model for semiconductor manufacturing. IEEE Trans Semicond Manuf 9(3): 412–427
8. Swaminathan JM (2000) Tool capacity planning for semiconductor fabrication facilities under demand uncertainty. Eur J Oper Res 120:545–558
9. Bethke AD (1981) Genetic algorithm as function optimizers. Dissertation, University of Michigan
10. Gen M, Cheng R (2000) Genetic algorithms and engineering optimization. Wiley, New York
11. Winston PH (1992) Artificial intelligence. Addison-Wesley, Boston