Int J Adv Manuf Technol (2006) 27: 1228–1234
DOI 10.1007/s00170-004-2316-z

ORIGINAL ARTICLE

Muh-Cherng Wu · C.S. Chien · K.S. Lu

Yield improvement planning for the recycle processes of test wafers

Received: 23 April 2004 / Accepted: 23 June 2004 / Published online: 16 March 2005

Abstract Test wafers are used to measure the process quality in semiconductor manufacturing. Test wafers are reusable by recycle cleaning and can be downgraded to the downstream processes. Most previous studies on test wafers aimed to reduce the use of test wafers by making appropriate operational decisions. Yet, the effective improvement of yield in the recycle process of test wafers is seldom explored. This paper formulates a decision problem and proposes two solution methods for selecting the yield improvement alternatives in the test wafer recycle processes. The decision problem is to determine the yield improvement target for each recycle process in order to minimize the use of test wafers, under a given budget for yield improvement. The two solution methods involve a genetic algorithm and a marginal allocation algorithm. The two methods yield very close solutions, but the marginal allocation method is better because it requires less computation time.

Keywords Control wafers · Monitor wafers · Test wafers · Yield improvement

1 Introduction

In semiconductor manufacturing, *test wafers* are indispensable materials, used in ensuring the production quality. Test wafers, also called *control wafers* or *monitor wafers*, are used to monitor the quality of tools and processes. To control a tool/process, test wafers may be run before or concurrently with product wafers. Output parameters are then taken from test wafers to make adjustments on the tool/process, if necessary.

A semiconductor fab keeps many types of test wafers with different specifications. Test wafers of a particular specification are stored in a dedicated buffer, which supplies to one or many tools. A test wafer, after being used in a tool, is sent to a *cleaning recycle* process for possible reuse. The recycled test wafers if meeting the original specification are kept in the present buffer. Those becoming lower in grade are downgraded to some other buffers. Test wafers in a buffer can be repeatedly recycled up to a limited number.

The process flow of using test wafers typically involves the following five steps: *preprocessing*, *in-use*, *cleaning recycle*, *downgrade*, and *grinding reclaim*. A test wafer for measuring the quality of an etching process is used to explain these steps. The preprocessing step is to deposit a film on the wafer. The in-use step measures the thickness of the film before and after the etching process to monitor the process quality. The cleaning recycle step, as mentioned above, is to remove the film and clean the test wafer for reuse. The downgrade step is to deliver the test wafer to lower-grade buffers. The downgrade relationship among the test buffers is a directed graph (Fig. 1). The grinding reclaim step is to grind off some 20–30 μm silicon materials from the test wafer for reuse; a reclaimed test wafer is functionally like a brand-new one.

Much literature on test wafers has been published. Wong and Hood [13] studied the impact on cycle time and throughput caused by increasing the number of process monitoring, which consequently increases the demand of test wafers. Wu [14] examined the dispatching policy of test wafers and product wafers in the preprocessing stage. Popovich et al. [9] developed an automated ordering process to maximize the reuse of test wafers. Chu [4] investigated the policy for setting safety stock level in each test wafer buffer. Watanabe et al. [11] proposed a procedure to increase the use ratio of reclaimed test wafers. Another addresses the downgrade decision problem; that is, how many test wafers should be delivered to each of its descendant buffers from a particular buffer. Some studies [1, 3, 6] developed the downgrade decision methods by considering the instantaneous work in process (WIP) level and demand of test wafers. Lu [8] analyzed the cost structure of test wafers and solved the problem by considering the long-term demand and supply of each buffer. In summary, most previous studies focused on the improvement of operation policies for test wafers, under a given set of system parameters. Yet, very few examine the improvement of these

M.-C. Wu (✉) · C.S. Chien · K.S. Lu
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

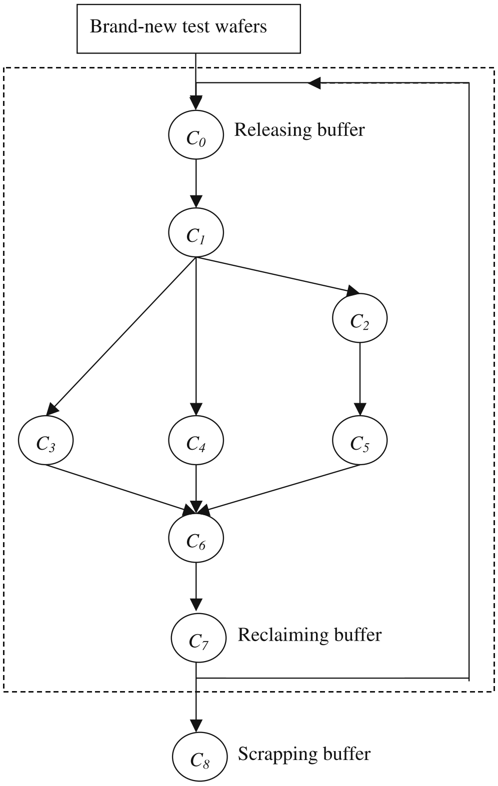

Fig. 1. Downgrade relationships among test buffers

system parameters, such as yield rates of recycle processes, for reducing the use of test wafers.

This paper studies how to establish an effective yield improvement plan for cleaning recycle processes. The decision problem is to determine the target yield rate of each buffer so that the use of brand-new test wafers can be minimized under a given budget for yield improvement. We adopt the downgrade decision model developed by Lu [8] to determine the usage of brand-new wafers for a particular set of recycling yield rates. Changing the set of yield rates will change the usage of brand-new test wafers. Two solution methods are developed to find a set of yield rates in order to minimize the usage of brand-new test wafers. These two solution methods involve a genetic algorithm (GA) and a marginal allocation algorithm.

The remainder of this paper is organized as follows. Sect. 2 reviews the downgrade decision model developed by Lu [8]. Sect. 3 describes the problem of planning the yield improvement of the cleaning recycle. Sect. 4 presents the two solution methods as well as the experiment results. Experiment results are presented in Sect. 5 and concluding remarks in Sect. 6.

2 Downgrade decision model

2.1 Cost analysis

The cost of test wafers in a fab involves three major items: (1) the cost of machine idleness due to lack of test wafers, (2) the usage cost of test wafers, and (2) the storage cost of test wafer (WIP) in shop floor. By interviewing several 8 inch fab sites in industry, Lu [8] estimates that the storage cost of test wafers is about 2.4% of the usage cost; and is at most 5% of the machine idleness cost.

From the cost analysis, the safety stock level of test wafers can be assumed to be high enough to always fulfill the time varying demand. Based on such an assumption, Lu [8] modeled the downgrade decision as a static decision problem. That is, the input and output average daily flow rates of each test wafer buffer should be balanced.

2.2 Downgrade decision problem

In a typical fab, the downgrade relationship among test wafer buffers is a directed graph. Referring to Fig. 1, the directed graph involves four types of buffers. *Working buffers* ($c_1 - c_6$) directly supply test wafers to tools. The *releasing buffer* (c_0) releases brand-new or reclaimed test wafers to working buffers. The *reclaiming buffer* (c_7) reclaims test wafers, and sends them to either the releasing buffer or the scrapping buffer. The *scrapping buffer* (c_8) scraps the test wafers that cannot be reclaimed further.

A working buffer stores m categories of test wafers, where m denotes the maximum number of cleaning recycles. Category $i(1 \leq i \leq m)$ represents test wafers that have received cleaning recycle i times. A test wafer in category i, after receiving one more cleaning recycle, becomes one in category $i+1$. Any test wafer in a particular working buffer, whatever category it belongs to, is regarded as the same in specification. Each cleaning recycle in a certain buffer has a distinct yield rate. Figure 2 shows various categories of test wafers in a working buffer.

The downgrade decision problem is to determine the daily flow rate of test wafers to be downgraded among buffers in order to minimize the usage of brand-new test wafers.

2.3 Notations

Let the downgrade path between the *reclaiming buffer* and the *releasing buffer* be called the *feedback path*. By eliminating the feedback path, the directed graph becomes one without a loop, which can be denoted by $G = (V, E)$ where $V = \{c_0, c_1, \ldots, c_r, c_{r+1}\}$ is a finite set of buffers and E is a set of arcs. An arc represents an ordered pair of two buffers. A *path* from c_i to c_j exists if one can traverse from c_i to c_j through passing k arcs ($k \geq 1$). If there is a path from c_i to c_j, then c_i is said to be an *ancestor* of c_j, and c_j is said to be a *descendant* of c_i. Additionally including the feedback path, the overall downgrade relationships can be denoted by $S = (G, f)$, where f is the arc $c_r \rightarrow c_0$. Referring to $S = (G, f)$, the following notations are used to formulate the downgrade decision problem.

Designations and sets.

c_i Designation of test buffer i; $0 \leqq i \leqq r+1$, c_0 is the releasing buffer, c_r is the reclaiming buffer, c_{r+1} is the scrapping buffer, and c_i $(1 \leqq i \leqq r-1)$ is a working buffer.

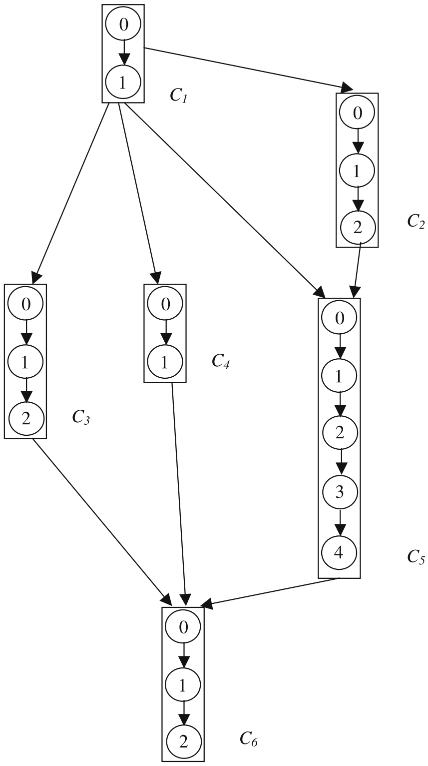

Fig. 2. A working buffer stores several categories of test wafers

$P(i)$ The set of ancestor buffers of c_i in diagraph G, excluding c_0, i.e. $c_0 \notin P(i)$.

$S(i)$ The set of descendant buffers for c_i in diagraph G.

Parameters.

D_i Average daily demand of test wafers in c_i, $1 \leqq i \leqq r-1$.

$m(i)$ Maximum number of cleaning recycle in c_i, $1 \leqq i \leqq r-1$.

$r_i^{[k]}$ The yield of kth cleaning recycle in c_i, $1 \leqq k \leqq m(i)$

n Maximum number of grinding reclaim in c_r

$h^{[k]}$ The yield of kth grinding reclaim in c_r, $1 \leqq k \leqq n$.

Variables.

O_{ij} Daily quantity of test wafers downgraded from c_i to c_j in diagraph G.

N_i Daily quantity of brand-new test wafers downgraded to c_i from c_0, $1 \leqq i \leqq r-1$.

N Daily quantity of brand-new test wafers downgraded from c_0; $N = \sum_{i=1}^{r-1} N_i$.

Y_i Daily quantity of reclaimed test wafers downgraded to c_i from c_0.

$Z^{[k]}$ Daily quantity of test wafers, with k times of reclaim, sent to c_0 from c_r.

Z Daily quantity of reclaimed test wafers sent to c_0 from c_r, $Z = \sum_{k=1}^{n} Z^{[k]}$.

$X_i^{[k]}$ Daily quantity of test wafers in c_i with kth cleaning recycle.

2.4 Model

Lu [8] formulates an LP model for the downgrading decision as follows. The model assumes that the input flow rate should equal the output flow rate for each buffer. Otherwise, the WIP level of each buffer may increase to infinity.

$$\text{Min} \sum_{i=1}^{r-1} N_i$$

s. t.

$$N + Z = \sum_{j=1}^{r-1} N_j + \sum_{j=1}^{r-1} Y_j \tag{1}$$

$$N_j + Y_j + \sum_{i \in P(j)} O_{ij} = X_j^{[0]} \quad 1 \le j \le r-1 \tag{2}$$

$$X_j^{[k]} = r_j^{[k]} \cdot X_j^{[k-1]} \quad 1 \le j \le r-1 \tag{3}$$

$$\sum_{k=0}^{m(j)} X_j^{[k]} = D_j \quad 1 \le j \le r-1 \tag{4}$$

$$\sum_{j \in s(i)} O_{ij} = X_0^{[0]} \quad 1 \le j \le r-1 \tag{5}$$

$$\sum_{j \in P(r)} O_{jr} = \sum_{k=1}^{n} Z^{[k]} + O_{r,r+1} \quad 1 \le j \le r-1 \tag{6}$$

$$Z^{[1]} = h^{[1]} \cdot N \tag{7}$$

$$Z^{[k]} = h^{[k]} \cdot Z^{[k-1]} \tag{8}$$

$$O_{r,r+1} = N \tag{9}$$

$$N_i \ge 0;\ \ Z_i \ge 0;\ \ Y_i \ge 0;\ \ O_{ij} \ge 0 \tag{10}$$

The objective function is to minimize the daily usage of brand-new test wafers. Constraint (1) denotes the flow balance relationship in buffer c_0. Constraints (2) indicate the inputs to a working buffer. Constraints (3) describe the yield relationship of a cleaning recycle in a working buffer. Constraints (4) denote that the demand in a working buffer c_i is supplied by several categories of test wafers. Constraints (5) indicate the output of a working buffer. The left-hand side describes where the output test wafers are downgraded. The right-hand side denotes the sources of the output.

Constraint (6) denotes the flow balance relationship of the reclaiming buffer c_r. The inputs are from all working buffers, represented in the left-hand side. The output involves two types of reclaimed test wafers, either within specification for reuse or out-of-specification for scrapping. Constraints (7) and (8) represent the yield relationships of grinding reclaim. Constraint (9) denotes that flow balance of the whole fab; that is, all the brand-new buffers finally have to go to the scrapping buffer c_{r+1}. Constraints (10) denote that all variables are non-negative.

3 Problem of planning yield targets

The decision problem of planning the recycle yield rate for each working buffer is discussed below. The cleaning recycle of test wafers is usually executed in a wet etch process. That is, putting test wafers in chemical solution for some time to clean the surface of test wafers. By changing recipes and process parameters, such as solution concentration or the time of bathing, the yield rate of recycle would change. Engineers need to experiment for the yield improvement. According to engineers' experiences, the higher is the target yield improvement, the more is the number of experiments and consequently the higher is the cost incurred.

To explain the yield planning problem, the following notations as well as those presented in Sect. 2 are referred.

Notations

$R = [R_1, R_2, \ldots, R_{r-1}]$: the *current* yield vector of the fab, where $R_i = [r_i^{[1]}, r_i^{[2]}, \ldots, r_i^{[m(i)]}]$: denotes the *current* yields at buffer c_i and $r_i^{[k]}$ is the yield of the kth recycle.

$\tilde{R} = [\tilde{R}_1, \tilde{R}_2, \ldots, \tilde{R}_{r-1}]$: a *new* yield vector of the fab, where $\tilde{R}_i = [\tilde{r}_i^{[1]}, \tilde{r}_i^{[2]}, \ldots, \tilde{r}_i^{[m(i)]}]$ denotes the *new* yields at buffer c_i and $\tilde{r}_i^{[k]}$ is the *new* yield of the kth recycle.

$X = [X_1, X_2, \ldots, X_{r-1}]$: a yield improvement plan, where $X_i = \tilde{R}_i - R_i = [x_i^{[1]}, x_i^{[2]}, \ldots, x_i^{[m(i)]}]$ denotes the yield improvement targets at buffer c_i, and $x_i^{[k]} = \tilde{r}_i^{[k]} - r_i^{[k]}$ denotes the yield improvement target of the kth recycle.

$C(x_i^{[k]})$: the cost incurred for the yield improvement in the kth recycle at buffer c_i.

$C(X) = \sum_{i=1}^{r-1} \sum_{l=1}^{m(i)} C(x_i^{[k]})$: the cost incurred for a yield improvement plan X.

B: the budget for improving the recycle yield of working buffers.

The LP model presented in Sect. 2 can be alternatively interpreted as $N = L(R)$. That is, given a yield vector R, the LP model L can compute the minimum daily usage of brand-new test wafers N. The problem of planning yield improvement targets can thus be formulated as follows.

$$\begin{aligned}
&\mathrm{Max}\, L(R) - L(\tilde{R}) \\
&X = \tilde{R} - R \\
&C(X) \le B
\end{aligned}$$

The decision variables are represented by a yield improvement plan X. The objective function is to maximize the saving in the usage of brand-new test wafers, under a given budge B.

4 Solution methods

Two solution methods have been developed for solving the yield planning problem. One is a genetic algorithm and the other is a marginal allocation algorithm.

4.1 Genetic algorithm

Genetic algorithm (GA) techniques, widely applied in various areas, are a random-search method for locating efficiently a near-optimal solution in the enormous space [2, 5, 7]. 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 some *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 chosen by a *selection procedure* to form the new population $P(t+1)$. The procedure is iteratively performed until the *termination conditions* are met.

The proposed GA for solving the yield planning problem is presented below.

A chromosome and initial population. A chromosome, a yield improvement plan, is denoted by $X = [X_1, X_2, \ldots, X_{r-1}]$ consisting of $r-1$ strings, where a string $X_i = [x_i^{[1]}, x_i^{[2]}, \ldots, x_i^{[m(i)]}]$ contains $m(i)$ positive numbers. Let $\mathrm{UB}(x_i^{[k]})$ represents the upper bound of $x_i^{[k]}$ and the interval $[0, \mathrm{UB}(x_i^{[k]})]$ be divided into n segments, where $n = \text{round-up}\left(\frac{\mathrm{UB}(x_i^{[k]})}{d}\right)$. Each of the first $n-1$ segments has a distance d, and the distance of the last segment is $\mathrm{UB}(x_i^{[k]}) - (n-1)d$. The value of $x_i^{[k]}$ is chosen from the set of the $n+1$ end points, denoted by $S(x_i^{[k]})$. 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.

B fitness function. The fitness function of X is defined as follows, where the first term denotes the objective function.

$$F(X) = [f(R) - f(\tilde{R})] - Y[C(X) - B],$$

where

$$\begin{aligned}
Y &= 0 \quad \text{if } C(X) \le B \\
&= M \quad \text{else, where } M \text{ is a very large positive number}
\end{aligned}$$

The second term is a penalty function [10], which leads to a small fitness value if the solution violates the budget constraint. A chromosome with a small fitness value is less likely to survive during the evolution of the population and tends to finally be excluded from the population. The penalty design is to keep "*good genes*" in the population. For a budget violation chromosome, particular segments of its genes may exactly match a part of the optimum solution. Possibly carrying good genes, violation chromosomes shall not be forcibly excluded from each population.

C crossover and mutation operators. The proposed GA defines two genetic operators, *crossover* and *mutation*, to create new chromosomes. The crossover operator is designed to create $N_p \times P_{\mathrm{cr}}$ new chromosomes in each generation, where P_{cr} is a predefined crossover probability. This operator is applied by first randomly choosing $N_p \times P_{\mathrm{cr}}$ chromosomes from $P(t)$ and randomly grouping them into $(N_p \times P_{\mathrm{cr}})/2$ pairs. For each pair of chromosomes, a position in a chromosome (called the *crossover point*) is randomly chosen, and the segments to the right of the crossover point exchanged.

The mutation operator is designed to create $N_p \times P_{mu}$ new chromosomes from $P(t)$, where P_{mu} is a predefined probability of mutation. This operator is applied by first randomly selecting $N_p \times P_{mu}$ chromosomes from $P(t)$. For each chosen chromosome, a gene $x_i^{[k]}$ is randomly selected and is subsequently replaced by a number randomly chosen from the set $S(x_i^{[k]})$.

D selection strategy. The chromosomes in population $P(t)$ and the newly created chromosomes are put in a *pool*, called S, where the number of chromosomes is $h = N_p \cdot (1 + P_{\mathrm{cr}} + P_{mu})$. N_p chromosomes are to be selected from S to the population $P(t+1)$, by the *rank-space* method [12] for preventing 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 $Z_1, Z_2, \ldots, Z_h$ be the sorted result. Such a ranking of Z_i, termed *quality-ranking*, is represented by $R_q(Z_i)$.

Step 2 Move the best quality-ranking chromosome from S to $P(t+1)$.

$$\begin{aligned}
&S = S - \{Z_1\};\\
&P(t+1) \leftarrow Z_1;\\
&Y_1 = Z_1; \quad /* \text{ rename the chromosome selected for } P(t+1) */\\
&N = 1; \quad /* \text{ count the chromosome number in } P(t+1) */
\end{aligned}$$

Step 3 For each chromosome Z_i in S, compute the *diversity index* $D(Z_i)$.

$$D(Z_i) = \sum_{k=1}^{N} \frac{1}{|Z_i - Y_k|}; \quad /* \; Y_k \text{ is a chromosome in } P(t+1) */$$

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

Step 5 Compute the sum of quality-ranking and diversity-ranking of Z_i in S.

$$T(Z_i) = R_q(Z_i) + R_d(Z_i)$$

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

Step 7 For each chromosome in S, compute the probability of putting Z_i in $P(t+1)$.

$$r = R_c(Z_i)$$

$Prob(Z_i) = p \cdot (1-p)^{r-1}$; /* 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 Z_m be the selected chromosome.

$$\begin{aligned}
&S = S - \{Z_m\}; \quad /* \text{ Move } Z_m \text{ from } S \text{ to } P(t+1) */\\
&P(t+1) \leftarrow Z_m;\\
&Y_m = Z_m; \quad /* \text{ rename the chromosome selected for } P(t+1) */\\
&N = N + 1; \quad /* \text{ update the chromosome number in } P(t+1) */
\end{aligned}$$

Step 9 Termination check

If $N < N_p$ then go to Step 3
Else Stop

E terminating conditions. Population $P(t)$ is iteratively updated until a particular chromosome keeps the best solution for over N_G generations or N_E generation has been created.

4.2 Marginal allocation algorithm

The proposed marginal allocation algorithm is an analytical method. The idea of this algorithm is to compute the cost and benefit caused by one unit yield improvement for each recycle at each buffer. Then, select the one, which is the most beneficial and meets the budget constraint, to update the yield parameters. The process is repeated until the cost incurred is over the budget. The procedure of this algorithm is presented below.

Step 0 Initialization and function definition

$$\begin{aligned}
&X = [0, 0, \ldots, 0]; \quad \text{i.e., } x_i^{[k]} = 0, \text{ for } 1 \le i \le r-1;\ 1 \le k \le m(i)\\
&E_i^k: \text{ a unit vector by replacing the value of } x_i^{[k]} \text{ in } X \text{ by } 1\\
&k = \text{Up_Arg}(\tilde{R}_i^k) \quad /* \text{ define function } \mathit{Up_Arg}() */\\
&i = \text{Low_Arg}(\tilde{R}_i^k) \quad /* \text{ define function } \mathit{Low_Arg}() */
\end{aligned}$$

Step 1 Compute the benefit of increasing yield by one unit of d.

$$\tilde{R}_i^k = R + (X + d \cdot E_i^k); \quad \text{for} 1 \le i \le r-1;\ 1 \le k \le m(i)$$

Step 2 Select the most beneficial alternative.

$$\begin{aligned}
&p = \text{Up_Arg}(\operatorname*{Max}_{i,k}(\tilde{R}_i^k)); \quad \text{for } 1 \le i \le r-1;\ 1 \le k \le m(i)\\
&q = \text{Low_Arg}(\operatorname*{Max}_{i,k}(\tilde{R}_i^k)); \quad \text{for } 1 \le i \le r-1;\ 1 \le k \le m(i)
\end{aligned}$$

Step 4 Check if the cost is within the budget.

If $C(X+d\cdot E_q^p) > B$

then return X; Stop

Else $X = X + d \cdot E_q^p$ /∗ update the yield improvement plan X ∗/

Go to Step 1

5 Experiment results

Experiments for justifying the two solution methods are executed by using a simplified fab involving six working buffers as shown in Fig. 1. The daily demand of test wafers and the current yield rate of each recycle at each working buffer are listed in Table 1. Assume $\mathrm{UB}(x_i^{[k]}) = 9\%$ for $1 \le i \le r-1$ and $1 \le k \le m(i)$. The cost function for improving the yield of the kth recycle at each buffer is the same, i.e., $C(x_i^{[k]}) = C(x_j^{[k]})$ for $i \ne j$, as shown in Table 2. The budget for yield improvement is $B = \$50,000$ and the unit yield increment is $d = 1\%$. The current daily usage rate of brand-new test wafers, computed by the LP model, is $L(R) = 1188.79$. Table 3 shows the cost and benefit of the solution obtained by applying the marginal allocation method, where $L(\tilde{R}) = 1036.76$. Notice that buffers c_1, c_2, and c_6 are not suggested to improve the yield at the present budget. This implies that their local improvements have very few or no impact on the global improvement. Buffer c_5 has the highest priority for yield improving. Such a yield target planning can effectively guide engineers to prioritize their jobs in order to maximize their contributions.

The GA is coded in C++ which calls the downgrade decision LP model implemented in CPLEX. The parameters of GA are set as follows: $P_{\mathrm{cr}} = 0.8$, $P_{mu} = 0.05$, $N_p = 100$, $N_G = 500$, and $N_E = 10,000$. Table 4 shows the results of 20 experiment runs. These 20 different yield improvement plans are quite close in the $C(X)$ and $L(R) - L(\tilde{R})$. The mean of $L(R) - L(\tilde{R})$ is 150.09 and the standard deviation is 1.526.

The results obtained by the marginal allocation method are slightly better than those obtained by GA. Also, the computation time per run for GA takes about 30 min. while that for the marginal allocation method takes only 10 s. A typical fab may include more than 100 working buffers, which implies that a chromosome may involve 400 genes. Accordingly, the computation time of the GA method may increase substantially. The marginal allocation method seems better in solving the addressed problem in the real world.

Table 1. Daily demand and current recycle yields at working buffer c_j

	c_1	c_2	c_3	c_4	c_5	c_6
D_j	3665	2538	2226	2336	6110	1448
$r_i^{[1]}$	90%	90%	90%	90%	90%	90%
$r_i^{[2]}$	80%	80%	80%	80%-	80%	80%
$r_i^{[3]}$	70%	70%	70%	70%	70%	70%
$r_i^{[4]}$	60%	60%	60%	60%	60%	60%

Table 2. Cost function of yield improvement for kth recycle at each buffer

Variable definition	Cost function
$y = x_i^{[1]}$	$C(y) = 100 + 450y + 30y^2$
$y = x_i^{[2]}$	$C(y) = 100 + 500y + 35y^2$
$y = x_i^{[3]}$	$C(y) = 100 + 550y + 40y^2$
$y = x_i^{[4]}$	$C(y) = 100 + 600y + 45y^2$

Table 3. Solution obtained by marginal allocation method

Chromosome	$C(X)$	$L(R) - L(\tilde{R})$
0000|0000|9800|9900|9991|0000	$49,985	152.03

Table 4. Solutions obtained by 20 runs of GA

Chromosome	$C(X)$	$L(R) - L(\tilde{R})$
0000|0000|9700|8810|9985|0000	$49,845	150.588
0000|0000|9900|9740|9980|0000	$49,925	150.924
0000|0000|9730|9800|9991|0000	$49,975	151.937
0000|0000|9901|8900|8991|0000	$49,905	147.39
0000|0000|9900|9910|9963|0000	$49,880	149.779
0000|0000|9620|7850|9982|0000	$49,765	149.506
0000|0000|9500|9761|9990|0000	$49,840	150.115
0000|0000|9530|9900|9920|0000	$49,965	151.739
0000|0000|9820|9500|9994|0000	$49,860	151.529
0000|0000|9900|9900|9891|0000	$49,985	149.927
0000|0000|9700|9900|9992|0000	$49, 695	151.832
0000|0000|7700|9600|9997|0000	$49,785	149.66
0000|0000|9700|9813|8991|0000	$49,900	147.046
0000|0000|9700|9900|9992|0000	$49,695	151.832
0000|0000|9900|8741|9980|0000	$49,710	149.566
0000|0000|9900|7800|9993|0000	$49,685	150.48
0000|0000|9760|9700|8990|0000	$49,975	147.513
0000|0000|9900|9910|9980|0000	$49,795	150.981
0000|0000|9700|9940|9980|0000	$49,925	151.046
0000|0000|9700|9680|9970|0000	$49,820	148.348

6 Concluding remarks

This research formulates and solves a problem for planning the yield targets of test wafer recycle processes. Test wafers are used for monitoring tool or process quality in semiconductor manufacturing. Test wafers after use are often recycled for possible reuse. A test wafer allows a limited number of recycles. The cost and benefit for improving the yield rate in each recycle at each buffer may be different. The addressed problem is to determine the allocation of yield improvement targets in order to maximize the benefit under a given budget.

An LP model is used to evaluate the benefit of a yield improvement plan, by computing the minimum daily usage of brand-new test wafers. Based on the benefit evaluation module, two methods for finding an optimum yield improvement plan are proposed. One is the genetic algorithm and the other is the marginal allocation algorithm. The solution obtained by the marginal allocation method is slightly better, which also takes much less time in computation. This fast computation feature becomes much more important when the problem size substantially increases.

The solution of the addressed problem can effectively guide engineers to prioritize the jobs of yield improving in order to maximize their contributions. Inappropriate priority settings may increase the yield of a particular buffer but have very little or no impact on saving brand-new test wafers.

Acknowledgement The authors would like to thank the Taiwan Semiconductor Manufacturing Corp. for characterizing this research problem.

References

1. Chen HC (2003) Control and dummy wafers management. Dissertation, National Chiao Tung University, Hsin-Chu, Taiwan
2. Chen JH, Fu LC, Lin MH, and Hunang AC (2001) Petri-net and GA-based approach to modeling, scheduling, and performance evaluation for wafer fabrication. IEEE Trans Robot and Automat 17(3):619–636
3. Chen HC, Lee CE (2000) Downgrading management for control and dummy wafers. J Chin Inst Ind Eng 17:437–449
4. Chu YF (1998) The inventory management model for control and dummy wafers. Dissertation, National Chiao Tung University, Hsin-Chu, Taiwan
5. Gen M, Cheng R (2000) Genetic algorithms and engineering optimization. Wiley, New York
6. Foster B, Meyersdorf D, Padillo JM, and Brenner R (1998) Simulation of test wafer consumption in a semiconductor facility. IEEE/SEMI Advanced Semiconductor Manufacturing Conference, pp 298–302
7. Holland JH (1975) Adaptation in neural and artificial systems. Univ Michigan Press, Ann Arbor, MI
8. Lu KS (2003) Decisions for downgrading control/dummy wafers in semiconductor fabs. Dissertation, National Chiao Tung University, Hsin-Chu, Taiwan
9. Popovich SB, Chilton SR, and Kilgore B (1997) Implementation of a test wafer inventory tracking system to increase efficiency in monitor wafer usage. IEEE/SEMI Advanced Semiconductor Manufacturing Conference, pp 440–443
10. Rietman EA and Frye RC (1996) A genetic algorithm for low variance control in semiconductor device manufacturing: some early results. IEEE Trans Semicond Manuf 9:223–229
11. Watanabe A, Kobayashi T, Egi T, and Yoshida T (1999) Continuous and independent monitor wafer reduction in DRAM fab. IEEE International Symposium on Semiconductor Manufacturing Conference, pp303–306
12. Winston PH (1992) Artificial Intelligence. Addison-Wesley, Boston, MA pp 520–527
13. Wong CY, Hood SJ (1994) Impact of process monitoring in semiconductor manufacturing. IEEE/CPMT International Electronics Manufacturing Technology Symposium, pp 221–225
14. Wu JE (1997) The construction of dispatching rule for control wafer in diffusion area. Dissertation, National Chiao Tung University, Hsin-Chu, Taiwan