SCHOOL OF OPERATIONS RESEARCH
AND INDUSTRIAL ENGINEERING
COLLEGE OF ENGINEERING
CORNELL UNIVERSITY
ITHACA, NY 14853-7501

TECHNICAL REPORT NO. 935

October 1990

CHIP ASSIGNMENT ALGORITHMS FOR
DYNAMIC WAFER DESIGN IN
SEMICONDUCTOR MANUFACTURING

by

D.C. Heath, P.L. Jackson, K. Levesque
M. Fret, S. Tlakula, C. Akkan

This research was supported in part by grants from the IBM Corporation and by the National Science Foundation grant no. DDM-8819542.
CHIP ASSIGNMENT ALGORITHMS FOR DYNAMIC WAFER DESIGN IN SEMICONDUCTOR MANUFACTURING

David C. Heath
Peter L. Jackson
Kathleen Levesque
Michelle Fret
Stanley Tiakula
Can Akkan
Cornell University

ABSTRACT

For a given number of jobs to be released into a semiconductor fabrication line, we consider the problem of assigning chip part numbers to the different wafers and to segments within the wafers in order to maximize the expected number of complete boards that can be assembled from the yield of good chips that the jobs will produce. We develop a general yield model for the problem and investigate two ways to implement it. We also propose and empirically compare two heuristic algorithms for solving the assignment problem.

1. INTRODUCTION

Manufacturers of semiconductor logic chips for low volume, high-end mainframe computers face a unique set of logistical problems. Although each process step in semiconductor fabrication can be accomplished relatively quickly and with high yield, the sheer number of process steps required for complex chip designs results in very long production lead times and highly variable overall yields.

There are many operational controls that affect the ability of the semiconductor manufacturer to meet the demand for chips. Among these are the quantity of different chip part numbers to release into the fabrication line during a given planning period. The second is the assignment of chip part numbers to different jobs, to different wafers within jobs, and to different chip sites on the wafers. As noted by Avram and Wein [1989], recent technological developments permit a number of different types of chips to be produced on the same wafer. Electron-beam technology permits the assignment of chip types to wafers at the time the job is released to production. The sequencing of job releases is another control and the expediting of jobs through the semiconductor line is yet another. In this paper, we focus on the chip release problem: how many chips of each type to release and how to assign these chips to jobs, wafers, and chip sites. The planning cycle and the number of jobs to be released is assumed to be given.

Demand for chips is a derived demand based on a bill of materials. The major levels in this bill of materials are illustrated in Figure 1. Chips are assembled into modules. At this level, some chips are unique to a particular module, while other chips are common to several modules. The role of common vs. unique chips and the need for stock of common chips is the subject of Glynn and Mostashari [1987]. At the next level of assembly are boards. From these two levels one can derive the number of chips of each type required to assemble a particular board. Chip types are associated with particular boards, so that there are no chips that are common to more than one board. The final level of assembly is the mainframe computer but the configuration of boards used in the computer is subject to customer choice so that it is not possible to specify a fixed bill of materials for a computer.

We focus on the board level in this bill of materials structure for several reasons. The first is that it is the highest level in the structure for which a fixed bill of materials (chips) can be specified. The second is that it leads to a natural decomposition of the chip release problem. Modules cannot be considered independently of one another because of the common vs. unique chip problem. However, we can focus release targets on particular boards and ignore others because their interactions are minimal. Furthermore, for practical reasons, jobs are designed in practice to consist of chips for one and only one board type.

Avram and Wein [1989] consider the fixed design of a wafer to maximize the average production rate of sets of wafers. This problem is closely related to the problem considered here except that our focus is on the dynamic design of wafers which is made possible by E-beam technology. The problem considered in Singh, Abraham, and Akella [1990] is appropriate for the dynamic wafer design problem. They have posed the problem of allocating a fixed number of chip sites to chip types to maximize the probability of satisfying a given demand for sets of wafers. The yield model developed in this paper reduces in application to the yield model proposed by Singh, Abraham, and Akella. Our objective, however, is different: we propose to maximize the expected number of sets (boards) that can be assembled from a given allocation of a fixed number of chip sites to chip types. We are indebted to the two papers mentioned and to unpublished papers by Gallego and Yao [1990] for fundamental modelling of the yield process.

There is a growing literature on release and dispatch rules in semiconductor fabrication (see, for example, Glasssey and Resende [1988]) that is relevant to the context of this paper but it is not directly relevant to the problem considered here. A description of the semiconductor fabrication process can be found in Maly [1987].
Figure 2 illustrates the layout of a wafer. Each square represents a chip site. With E-beam technology, it is possible to assign a different chip part number to each site. In practice, however, only a small number of different part numbers are assigned to a single wafer. For allocation purposes, chip sites are logically grouped into segments. Each site within a segment must be assigned the same chip part number, so the number of segments determines the number of different chip types that can be assigned to a wafer. Each segment has roughly the same number of chip sites, but their locations on the wafer are not necessarily contiguous. Figure 2 shows a wafer with three segments each composed of a random scattering of chip sites. The segmentation design problem is not considered here.

A fixed number of wafers compose a job. In a given planning cycle, there are a fixed number of jobs of a given board type that will be released. The problem posed in this paper is to design each wafer in each job (i.e., assign chip part types to all possible segments) to maximize the expected number of boards that can be produced.

The paper is organized in the following sequence: section 2 presents a model for computing the number of boards that can be assembled from chips about to be released, chips released but unfinished, and finished chips. Section 3 considers two ways to implement the model and perform the calculations: Monte Carlo simulation and Clark's Method. Section 4 describes two chip assignment algorithms and the results of an empirical test comparing the algorithms. Section 5 concludes the paper.

2. JOB YIELD MODEL

General Model

Suppose that \( K \) types of chips are required to build a board. Let the numbers of each chip type required to make one board be denoted by \( n_1, n_2, ..., n_K \). There are three sources of chips to produce boards: finished goods of completed chips (FG), work-in-process (WIP), and work about to be released. We assume that jobs are processed in the order in which they are released (FIFO: first-in, first-out). In particular, no work released after now will finish before work released now. We desire to compute the distribution of the number of boards that can be produced up to and including the time at which the work being released now will finish.

The Chip Supply Vector and the Board Supply Variable

Denote the number of chips of each type that will result from the current WIP plus the inventory of completed chips by the random variables \( M_1, M_2, ..., M_K \). Denote the number of chips of each type that will result from the jobs we are about to release by \( N_1, N_2, ..., N_K \). Under our assumption that jobs are processed in FIFO order, when the currently released work finishes we shall have \( N_k + M_k \) chips of type \( k \) available. The maximum number of boards that can be produced using these chips is then:

\[
\min \left\{ \frac{N_1 + M_1}{n_1}, \frac{N_2 + M_2}{n_2}, ..., \frac{N_K + M_K}{n_K} \right\}
\]

This will be a difficult expression to evaluate in later analysis. In its place, we shall use the random variable \( L \) defined by:

\[
L = \min \left\{ \frac{N_1 + M_1}{n_1}, \frac{N_2 + M_2}{n_2}, ..., \frac{N_K + M_K}{n_K} \right\}
\]

Clearly, we have:

\[
L - 1 \leq \min \left\{ \frac{N_1 + M_1}{n_1}, \frac{N_2 + M_2}{n_2}, ..., \frac{N_K + M_K}{n_K} \right\} \leq L
\]

We will refer to \( L \) as the board supply variable since it approximates the number of boards that will be possible to build given the current FG, WIP, and work release.

Using vector notation, we define \( M = (M_1, ..., M_K) \), \( N = (N_1, ..., N_K) \), \( n = (n_1, ..., n_K) \), and \( Q = (M+N)/n \), where the operations are component by component. The vector \( Q \) is the vector of chips that will be available to produce boards, normalized by the bill of materials requirements, \( n \). We call \( Q \) the chip supply vector.

Distribution of the Chip Supply Vector

We make two simplifying assumptions at this stage. The first is that the conditions of the central limit theorem hold so that \( Q \) is nearly normally distributed. With this assumption, it is sufficient to determine its mean vector and variance-covariance matrix in order to determine the distribution of \( Q \) (and hence of \( L \)). The second assumption is that \( N \) and \( M \) are independent. Hence, the means, variances, and covariances of the total numbers of chips available can be computed by adding the means and variance-covariance matrices of the two random vectors. The current stock
of completed chips (FG) is nonrandom, so its mean and variance are known. The methodology of this section will permit the computation of the means, variances, and covariances of both \( N \) and the WIP, so the results of this section will permit the computation of the mean vector and variance-covariance matrix of \( Q \).

We will describe the model for computing the mean vector and variance-covariance matrix of \( N \), the chips that will result from the current release. The same computations, with different names, can be used to compute the mean vector and variance-covariance matrix of the chips resulting from the current WIP.

Let \( N_1, N_2, \ldots, N_K \) denote the number of chips of each type being released into the line in the current period. These releases are decision variables. In addition to specifying the number of chips to release, it is necessary to specify the assignment of these chips to particular chip sites. A chip site consists of the triple \((j, w, d)\) in which \( j \) is the index of the job containing the chip, \( w \) is the index of the wafer on which the chip resides, and \( d \) is the location (segment) of the chip on that wafer. Let \( s_{ij} \) denote the site of the \( j \)th chip of type \( i \), \( j = 1, \ldots, N_i \), \( i = 1, \ldots, K \).

Let \( X_{ij} \) be the random variable which takes on the value 1 if the \( j \)th chip of type \( i \) is good, and 0 if it is bad. We assume that the probability that a chip of type \( i \) is good is a function both of the chip type \( i \) and the chip site \( s_{ij} \). Because we need to model variances and covariances, we also assume that the probability that two different chips, say, chips \( i_1, j_1 \) and \( i_2, j_2 \), are both good is some function of the chip types, \( i_1 \) and \( i_2 \), and of the chip sites, \( s_{i_1,j_1} \) and \( s_{i_2,j_2} \).

More precisely, we assume the existence of functions \( p(\cdot, \cdot) \) and \( p(\cdot, \cdot, \cdot) \) such that

\[
E(X_{ij}) = p(i, s_{ij})
\]

and

\[
E(X_{i1,j1}X_{i2,j2}) = p(i_1,i_2,s_{i1,j1},s_{i2,j2}).
\]

for all \( i, j, i_1, i_2, j_1, \) and \( j_2 \).

Given the above, we can compute the mean of the \( N \) vector:

\[
E(N) = \sum_{j=1}^{N_i} E(X_{i,j}) = \sum_{j=1}^{N_i} p(i, s_{ij}).
\]  

Similarly, we can compute the second moments, getting:

\[
E(N^2) = E(N) + 2 \sum_{j_1=1}^{N_i} \sum_{j_2=1}^{N_i} p(i_1, i_2, s_{i1,j1}, s_{i2,j2}).
\]

and

\[
E(N_{i1}N_{i2}) = \sum_{j_1=1}^{N_{i1}} \sum_{j_2=1}^{N_{i2}} p(i_1, i_2, s_{i1,j1}, s_{i2,j2}).
\]  

Equations (1), (2), and (3) above are the fundamental results permitting computation of the means, variances, and covariances of the numbers of chips of each type which will result from the jobs about to be released.

**Simplified Model**

In this section, we suggest some simple forms for the functions \( p(\cdot, \cdot) \) and \( p(\cdot, \cdot, \cdot) \) and work out the corresponding form of the variance-covariance matrix for \( N \). For simplicity, we shall be assuming that the yield of chips located on sites in different jobs are independent.

**Model Development**

We begin by supposing that \( p(i, s) = p(i)q(s) \). It is easy to see that we then have:

\[
E(N_i) = \sum_{j=1}^{N_i} p(i, s_{ij}) = p(i) \sum_{j=1}^{N_i} q(s_{ij}).
\]

To model the correlation in yield from chip to chip, we first compute:

\[
\begin{align*}
\text{cov}(X_{i1,j1}, X_{i2,j2}) &= E(X_{i1,j1}X_{i2,j2}) - E(X_{i1,j1})E(X_{i2,j2}) \\
&= p(i_1, i_2, s_{i1,j1}, s_{i2,j2}) \\
&\quad - p(i_1) p(i_2) q(s_{i1,j1}) q(s_{i2,j2}).
\end{align*}
\]

To simplify this expression, we now suppose that the first ratio inside the parentheses does not depend on \( i_1 \) or \( i_2 \). This means that there is another function (which we also call \( q \)) such that:

\[
\text{cov}(X_{i1,j1}, X_{i2,j2}) = p(i_1) p(i_2) \left[ \frac{q(s_{i1,j1}, s_{i2,j2})}{p(i_1) p(i_2)} - q(s_{i1,j1}) q(s_{i2,j2}) \right].
\]

Now, what sort of function do we expect \( q(\cdot, \cdot) \) to be? Clearly, we need define it only when its two arguments are different. Recalling that \( s = (j, w, d) \), we shall suppose, again for simplicity, that

\[
q(s_1, s_2) =
\begin{cases}
q(s_1)q(s_2) + \alpha(j_1, j_2) & \text{if } j_1 \neq j_2 \\
q(s_1)q(s_2) + \beta(j_1, j_2) + \gamma(w_1, w_2) & \text{if } j_1 = j_2 \text{ but } w_1 \neq w_2 \\
q(s_1)q(s_2) + \beta(j_1, j_2) + \gamma(w_1, w_2) & \text{if } j_1 = j_2 \text{ and } w_1 = w_2
\end{cases}
\]

where \( \alpha, \beta, \) and \( \gamma \) are non-negative and satisfy the following consistency condition:

\[
\alpha(j_1, j_2) + \beta(w_1, w_2) + \gamma(1, 1) + \gamma(1, 1) < q(s_1)(1 - q(s_2))
\]

and
\[ \alpha(j_1,j_2) + \beta(s_1,j_2) + \gamma(1,1,2) \leq q(s_2)(1-q(s_1)) \]

for all \( s_1 \) and \( s_2 \).

It seems plausible to suppose that, for the sorts of applications we have in mind, \( \gamma \) can be thought of as a constant.

The reason is that we plan to assign chip types to sites in groups (called segments) so that each group of sites is nearly uniformly distributed across the wafer. Consequently, each segment is reasonable "close" to each other segment.

Another reasonable assumption is that \( \alpha \) is zero. The reason for this is that we shall confine our analysis to jobs for just one type of board. When jobs are released, only some will be of the type we are analyzing. Furthermore, we assume that some effort is made when deciding a release sequence to separate jobs of the same type by jobs of other types. Thus, jobs of the type we are analyzing will usually not be "next" to one another even at release. The greater the spacing between jobs of the same type in the release sequence, the more reasonable is the assumption that all the yields from different jobs are independent.

It is reasonable that the function \( \beta \) will have larger values for wafers near one another in a job, and smaller values for wafers which are further apart. This is because some of these steps operate on the wafers in the job sequentially. If each process step in the line operated on all wafers in the job simultaneously and identically, then \( \beta \) would be a constant.

**Model Results**

For a simple application of the above model, we shall compute the means, variances, and covariances assuming \( \alpha = 0 \), \( \beta = 0 \), and \( \gamma = \delta \). These assumptions are equivalent to the common modelling assumptions (eg., Singh, Abraham, and Akella) that the yield of a chip site is independent of every other chip site, except when the whole wafer is flawed or when the whole job is lost. To see this, suppose the probability that a job is good is \( \delta \) and the probability that a wafer is good, given that the job is good, is \( \varepsilon \). In this case, \( q(s) = \delta \varepsilon \) for every chip site \( s \). It is also easy to verify that

\[
q(s_1,s_2) = \begin{cases} 
\delta^2 \varepsilon^2 & \text{if } j_1 \neq j_2 \\
\delta \varepsilon & \text{if } j_1 = j_2 \text{ and } w_1 \neq w_2 \\
\delta \varepsilon (1-\varepsilon) & \text{if } j_1 = j_2 \text{ and } w_1 = w_2 
\end{cases}
\]

Observe that this corresponds to \( \alpha = 0, \beta = \delta \varepsilon^2 (1-\delta), \) and \( \gamma = \delta \varepsilon (1-\varepsilon) \).

Under these assumptions, we want to work out \( \text{E}(N_i) \), \( \text{E}(N_i^2) \), and \( \text{E}(N_i N_j) \). The mean chip supply of type \( i \) is given by:

\[
\text{E}(N_i) = \sum_{j=1}^{N} p(1,\bar{s}_i,j) = p(i) \sum_{j=1}^{N} q(s_i,j).
\]

The second moment is given by:

\[
\text{E}(N_i^2) = \text{E} \left[ \sum_{j=1}^{N} X_i,j \right]^2
\]

The first term in the above line is just \( \text{E}(N_i^2) \) since \( X_{ij}^2 = X_{ij} \). The second term is equal to:

\[
2 \sum_{j_1=1}^{N} \sum_{j_2=j_1+1}^{N} \text{cov}(X_{1,j_1},X_{1,j_2}) + \text{E}(X_{1,j_1}) \text{E}(X_{1,j_2})
\]

where the selection in the first term depends on whether the chip sites are in different jobs, in the same job on different wafers, or on the same wafer, respectively. Thus, the sum is determined by how many wafers of type \( i \) end up in the same job, and on the same wafer. Hence, we define \( J(i) = \#\{j_1,j_2: j_1 \neq j_2 \text{ and } s_{i,j_1} \text{ and } s_{i,j_2} \text{ are in the same job} \} \text{ and } W(i) = \#\{j_1,j_2: j_1 \neq j_2 \text{ and } j_1 \text{ and } s_{i,j_2} \text{ are in the same wafer} \}. \) Putting all of the above together, we get:

\[
\text{E}(N_i^2) = \text{E}(N_i) + 2 \sum_{j_1=1}^{N} \sum_{j_2=j_1+1}^{N} \text{cov}(X_{1,j_1},X_{1,j_2}) + \text{E}(X_{1,j_1}) \text{E}(X_{1,j_2})
\]

Then \( \text{var}(N_i) = \text{E}(N_i^2) - (\text{E}(N_i))^2 \) is easy to compute. Similar computations show:

\[
\text{E}(N_i N_j) = \text{E}(N_i) \text{E}(N_j) + p(i) p(j) \left( J(i,j) \text{ or } W(i,j) \right)
\]

where \( J(i,j) = \#\{j_1,j_2: s_{i,j_1} \text{ and } s_{i,j_2} \text{ are in the same job} \} \text{ and } W(i,j) = \#\{j_1,j_2: s_{i,j_1} \text{ and } s_{i,j_2} \text{ are in the same wafer} \}. \) The covariance is easy to obtain:

\[
\text{cov}(N_i,N_j) = \text{E}(N_i N_j) - \text{E}(N_i) \text{E}(N_j).
\]
3. YIELD EVALUATION

Distribution of the Board Supply Variable

The previous section outlined a methodology for determining the mean vector and variance-covariance matrix of the chip supply vector $Q$. Given these data, it would be simple to obtain the distribution of the board supply variable $L$, if we could compute the cumulative distribution function for a normally distributed vector. Unfortunately, this seems to be quite difficult. Two approaches appear to be effective. The first is to simulate $Q$ and $L$. The second is to apply the method developed by Clark (1961).

Monte Carlo Simulation and the Critical Chip

To produce random values for $Q$, we must find a matrix $R$ such that $R'R$ is the variance-covariance matrix for $Q$. Once we have a "square root" $R$ of the variance-covariance matrix, we generate random values for $Q$ by computing $E(Q) + R'W$, where $W$ is a standard normal vector (with independent components each having mean 0 and variance 1). Standard techniques are available for generating random standard normal vectors, $W$. For each random value of $Q$, we compute the corresponding value of $L$. We also identify the chip type $k$ that determined $L$: $L = Q_k = (M_k + N_k)/2$. We refer to this chip type as the critical chip. Simulation can thus be used to compute an empirical distribution of $L$ and to estimate the probability that any given chip type will be critical.

One method to find $R$ is given in Hoffman and Kunze (1961, pp. 244-245). Another is to decompose the original variance-covariance matrix as $VDV$, where $V$ is an orthogonal matrix (see Hoffman-Kunze, p. 152), and $D$ is a diagonal matrix whose entries are automatically non-negative. Then if $E$ is the diagonal matrix whose entries are the square roots of those of $D$, we get a suitable $R$ as $VEV$. This is essentially the method one obtains from Hoffman-Kunze's Theorem 27 (p. 278). This latter technique is the one implemented in the empirical study described below.

Clark's Method

An alternative method to approximate the distribution of $L$ is to apply the method of Clark (1961). The method of Clark is to iteratively compute the moments of the minimum of $K$ normal random variables. The method is exact for $K = 2$. Each iteration consists of computing the moments of the minimum of one of the random variables compared with the minimum of all the random variables considered on earlier iterations. The method is sensitive to the order in which the variables are considered and can lead to substantial errors in estimating the true moments. Nevertheless, the algorithm performed well in the environment of the empirical study described below and it is much faster than the simulation technique described above.

Empirical Validation of Clark's Method

The parameters for the empirical study were as follows: 40 chip types, 5 jobs, 18 wafers per job, 12 segments per wafer, 14 chip sites per segment, 85% probability that the site is good, 85% probability that the wafer is good, and 85% probability that the job is good. The yield of a chip type was set separately for each chip type in the range 0.85 to 0.99. The number of trials in the Monte Carlo simulation was set to 500.

Figure 3 shows the empirical distribution of the board supply variable for a particular chip assignment. The negative observations are a consequence of the normality assumption. Observe that the distribution is skewed to the right slightly. Since the Clark method assumes normality of distributions at each iteration, it cannot capture this aspect of the distribution. The simulated mean board supply is 70.45. The Clark method predicts a mean 65.71. Because of the skew, the Clark method tends to underestimate the mean. Nevertheless, the Clark method is an exceptionally practical alternative to simulation.

4. CHIP ASSIGNMENT ALGORITHMS

In this section, we describe two heuristic procedures to assign chips to sites with the purpose of maximizing the expected value of the board supply variable. The first method, called the Critical Chip Algorithm, takes advantage of the simulation method by using the identification of the critical chip to guide the selection of the next chip part type to assign. The second method, called Gradient Optimization, tries all possible chip part types at each step of the algorithm and chooses the one that results in the greatest marginal increase of the expected board supply. The Gradient Optimization method can be implemented using either simulation or Clark's method for yield evaluation. The relationships are summarized in Figure 4.

Both methods are computationally intensive. To reduce the computational burden, each algorithm begins with an initial partial assignment of chips to wafers. This initial partial assignment is referred to as the base allocation. The techniques used to generate a base allocation are described next, followed by descriptions of the Critical Chip and Gradient Optimization algorithms.

Base Allocation

We assume that there are a fixed number of jobs available to which to assign chips. This determines the total number of segments available. Each segment will be assigned one chip type. A portion of these segments, called the base allocation, is assigned by means of a simple heuristic.

The available segments can be thought of as a series of two-dimensional matrices, one matrix for each job. The rows of each matrix represent the individual wafers that make up the job. The columns represent the segments that make up the wafer. Therefore, each cell in the matrix represents a group, or segment, of chip sites. The entry in that cell is the assignment of a common chip type to each chip site in that particular segment. Figure 5 illustrates.

![Figure 3: Histogram of Board Supply Variable](image)

Figure 3: Histogram of Board Supply Variable

To test whether the order in which the chip types were considered in the Clark method had any effect, we reversed the order and ran the method again. There was no appreciable difference in the results.
Let \( c \) denote the number of chip sites in a segment. Given \( A \), the total number of available segments in the base allocation (which is a design parameter in the overall optimization), we determine an appropriate fraction of these segments to which to assign each chip type. Ignoring WIP and FG, the number of segments to allocate to chip type \( i \) is the greatest integer less than or equal to

\[
\left( \frac{n_i / Y_i}{\sum_k(n_k / Y_k)} \right) A,
\]

where \( Y_i = cp(i)i\sigma_e \), the expected yield of a segment of chip type \( i \).

Once the number of segments per chip type is known, there are a variety of ways to assign each chip type to a particular segment in the set of jobs being released. We tested two simple methods: "spread by job" and "spread by wafer." In both methods, chips are assigned in order of part numbers, i.e., all chip segments of type 1 are assigned, then, all chip segments of type 2 are assigned, and so on. The "spread by job" technique means that the matrix is filled by moving through the jobs in sequence, or from left to right as in Figure 5a. The job index is incremented first, then the wafer index, and finally, the segment index. The "spread by wafer" technique increments the wafer index first, the job index second, and the segment index third. Figure 5b illustrates.

**Critical Chip Algorithm**

The Critical Chip Algorithm begins with a base allocation and then iteratively considers each segment that has not been assigned a part type and assigns a part type in an attempt to fine-tune the overall assignment. The order in which remaining segments are considered is similar to the spread by job technique: the job index is incremented most frequently. On each iteration, the yield of the current assignment is analyzed using the Monte Carlo simulation method. For each chip type, the simulation method returns an estimate of the probability that that chip type is critical. For each chip type, we compute its criticality ratio: the ratio of this probability to the bill of materials quantity of this chip, \( n_i \). We choose the chip type with the maximum criticality ratio and assign it to the next available segment. The reason for dividing by the bill of materials quantity is that we are using the criticality ratio to approximate the probability that adding a chip of type \( i \) will result in another board being produced. Even if type \( i \) is critical, an extra type \( i \) chip will effectively result in only \( 1/n_i \) additional boards.

**Gradient Optimization Method**

Observe that, under the Critical Chip Algorithm, the chip type to assign to the next available segment is chosen without regard to the particular segment to which it will be assigned. Adding a chip type to a wafer where several segments have already been assigned to the same chip type could significantly increase the variance of the yield of this chip type. This increased variance reduces the effectiveness of the assignment.

An alternative, and more thorough, chip type selection procedure is to consider, in turn, assigning each different chip type to the next available segment, compute the expected number of boards that would result from the new assignment, and select the chip type that maximizes this number. This approach, called Gradient Optimization, should dominate the Critical Chip Algorithm because the criticality ratio is a only a rough estimate of the marginal benefit that will result from assigning a chip type to the next available segment whereas the Gradient approach results in a much more accurate estimate.

Because the assignment is made myopically, there is no guarantee that even the Gradient Optimization Algorithm will find a globally optimal assignment.

The Gradient Optimization Algorithm can be implemented with either of the two yield evaluation techniques: Monte Carlo simulation or Clark's method.

---

**Figure 5a: Spread by Job**

<table>
<thead>
<tr>
<th>segments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st</td>
</tr>
<tr>
<td>3rd</td>
</tr>
<tr>
<td>5th</td>
</tr>
</tbody>
</table>

**Figure 5b: Spread by Wafer**

<table>
<thead>
<tr>
<th>segments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st</td>
</tr>
<tr>
<td>2nd</td>
</tr>
<tr>
<td>3rd</td>
</tr>
</tbody>
</table>

**Figure 5: Base Allocation Techniques**
Computational Results

The parameters of the test cases are described above in the discussion of Clark's method. To simplify the study, WIP and FG were assumed to be zero in order to focus on the impact of the chip assignment algorithms.

Spreading by Job vs. Spreading by Wafer

To study the impact of the base allocation technique, we set the proportion of segments allocated to the base portion to 90% (i.e. A = 0.9) and used the Critical Chip Algorithm to assign the remaining 10% of the total available segments. One run was made using the spread by job base allocation and another run was made using the spread by wafer technique. Figure 6 compares the effect of the two techniques in terms of the expected number of boards produced. The horizontal axis measures the number of iterations. Iteration 0 refers to the base allocation assignment. Clearly, the spread by job technique dominates the spread by wafer technique. This result can be explained by the fact that the spread by wafer technique results in a higher variance of the yield of each chip type by concentrating assignments of that chip in the same job. The loss of one job has a drastic impact on the number of boards that can be produced. Observe that the Critical Chip Algorithm cannot recover from the poor base allocation of the spread by wafer technique and the improvement at each step of the algorithm is not as steady as with the better base allocation.

The spread by job technique is used for base allocation in all the experiments described below.

Effect of the Base Allocation Proportion

The computational requirements of both the Critical Chip Algorithm and Gradient Optimization are greatly affected by the proportion of segments assigned using the base allocation techniques. On the other hand, too high a proportion of base allocation could prevent either algorithm from reaching its best solution.

Figure 7 illustrates the performance of Gradient Optimization (using simulation) under two different levels of the base allocation proportion (A = 0.75 and A = 0.9). Observe that starting from a base allocation of 75%, Gradient Optimization is able to find an assignment that clearly dominates the base allocation of 90% (58 boards, in expectation, using Gradient Optimization, as opposed to 50 boards using only the spread by job technique). However, the Gradient Optimization starting from the 90% base recovers quickly. By the stage at which 95% of the available segments have been assigned, the methods yield virtually identical results. The same phenomenon occurs with the Critical Chip Algorithm: the difference in outcomes between A = 0.75 and A = 0.90 is not statistically significant. We conclude that a 90% base allocation is adequate for the purpose of comparing the chip assignment algorithms.

Comparison of Chip Assignment Algorithms

Figure 8 shows the progress of the Critical Chip Algorithm (labelled critical) and the two variants of Gradient Optimization (labelled gradsim for the simulation version and gradclark for the version using Clark's Method). All three algorithms began from a 90% base allocation.

Note that the progress of the gradclark version is marked by the estimate of the expected boards produced using Clark's method. The other two algorithms, gradsim and critical, use simulation to estimate the same quantity so the graphs are not comparable in a strict sense. When the final assignment generated by gradclark is simulated, it results in a mean number of boards produced of 70.45. This is 0.9 boards higher than the mean number of boards produced using gradsim and the difference is statistically significant.

Both gradsim and gradclark statistically dominate critical in terms of the final assignment but the differences are fewer than 1.5 boards out of 70, in expectation. One must trade the value of an additional board against the cost of the additional computation time. The running time for gradsim on an IBM RT was on the order of 22 hours. The running time for gradclark was 105 cpu minutes. The time for critical was 70 cpu minutes.
5. CONCLUSIONS

In this paper, we have analyzed the problem of releasing jobs into a semiconductor fabrication line to maximize the expected number of boards that can be assembled from the resulting semiconductor chips after yield considerations. We developed a general model to represent the distribution of board supply as a function of FG, WIP, and job releases and we applied the model to the common modeling assumptions that yields of chips on the same good wafer are independent, wafers of good jobs are good or bad with a known probability, and jobs are good or bad with a known probability. We considered two techniques to evaluate the yield of a given job configuration (chip assignment): Monte Carlo simulation and Clark’s Method and found Clark’s Method to perform well. We considered several methods for assigning chip types to chip sites (job/wafer/segment assignments). The so-called Critical Chip Algorithm and the Gradient Optimization both resulted in substantial improvements over a naive base allocation algorithm. The Gradient Optimization technique dominated the performance of the Critical Chip Algorithm but at the expense of greater computational effort. Two methods of implementing Gradient Optimization were compared. The method that relied on Clark’s Method for yield evaluation required an order of magnitude less computation time than the method relying on simulation, and it actually resulted in an improved solution.

ACKNOWLEDGEMENTS

We are indebted to John Milne of IBM East Fishkill for suggesting this problem and to IBM and NSF for sponsoring this research. The yield and other data used in the empirical tests reported in this paper were not provided by IBM and are not intended to reflect their current manufacturing capability.

REFERENCES


Figure 1. Bill of Materials Structure
Figure 2. Wafer Segmentation
Figure 3: Histogram of Board Supply Variable
Figure 4. Algorithm/Model Relations
Figure 5a: Spread by Job

Figure 5b: Spread by Wafer

Figure 5: Base Allocation Techniques
Figure 6: Comparison of Base Allocation Techniques
Figure 7: Sensitivity to Proportion of Base Allocation