Approximation algorithms for the wafer to wafer integration problem

Rodolphe Giroudeau, Vincent Boudet, Marin Bougeret, Trivikram Dokka, Frits Spieksma

Approximation Algorithms for Wafer to Wafer Integration Problem*

Trivikram Dokka[1], Marin Bougeret[2], Vincent Boudet[2], and Frits C.R. Spieksma[1]

[1] ORSTAT, K.U.Leuven, Naamsestraat 69, B-3000 Leuven, Belgium.
trivikram.dokka;frits.spieksma@econ.kuleuven.be.
[2] LIRMM Montpellier, France

Abstract. Motivated by the yield optimization problem in semi-conductor manufacturing, we model the wafer to wafer integration problem as a multi-dimensional assignment problem and study it from an approximation point of view. We give approximation algorithms achieving an approximation factor of $\frac{3}{2}$ and $\frac{4}{3}$ for WWI-3. We show that a special case of yield optimization problem can be solved in polynomial time.

Keywords: wafer-to-wafer integration; approximation; computational complexity; efficient algorithm.

1 Introduction

Consider the following problem. Given are m sets V_i, $i = 1, ..., m$. Each set contains n p-dimensional vectors; each entry of each vector is a nonnegative integer. We define the *cost* of vector $u = (u_1, u_2, \ldots, u_p)$ as follows: $c(u) = \sum_{i=1}^{p} u_i$. Given a pair of vectors u, v, we can construct the vector $u \vee v$ by defining the operation $\vee$ as follows:

$$u \vee v = (\max(u_1, v_1), \max(u_2, v_2), \ldots, \max(u_p, v_p)).$$

Notice that $(u \vee v) \vee w = u \vee (v \vee w)$.

Consider now an m-tuple, ie, a set of m vectors $u^1, u^2, \ldots, u^m \in V_1 \times V_2 \times \ldots \times V_m$. The cost of an m-tuple equals $c(u^1 \vee u^2 \vee \ldots \vee u^m)$. Our problem, that we denote by WWI (see Section 1.1), is to find n disjoint m-tuples such that each vector is used exactly once, while total cost is minimum. In the figure below an instance with $m = 3, n = p = 2$ is depicted; notice that this instance has the property that each vector is a 0-1 vector; the value of an optimal solution to this instance equals 2.

We were motivated to look at this optimization problem by an application in the semi-conductor industry, that we now proceed to describe.

* This research was supported by OT Grant OT/07/015.

V_1	V_2	V_3
00	00	10
01	10	01

Fig. 1. WWI instance; $m = 3, n = p = 2$

1.1 The application

Our understanding of the semi-conductor industry, and in particular the wafer-to-wafer production process is primarily based upon ... In the semi-conductor industry, TSV based 3D-SIC is an emerging technology that provides large benefits: a smaller footprint, a higher interconnect density between stacked dies, higher performance, and lower power consumption due to shorter wires when compared to planar IC's. One of the key steps in the production of 3D-SIC's is stacking. There are three different ways of stacking: (1) wafer to wafer (2) die to wafer (3) die to die (see [5]). Of these three approaches, wafer to wafer stacking offers the highest manufacturing throughput coupled with other advantages. However, wafer to wafer stacking approach suffers from a drawback that it may have a low yield. The main motivation of this paper is to study this yield optimization problem in the wafer to wafer integration process.

The yield optimization problem in the semi-conductor industry can be informally described as follows: there are m lots of wafers called wafer lots, with each wafer lot consisting of n wafers. A wafer consists of a string of bad dies and good dies; in our context this translates to a '0' in case of a good die, and a '1' in case of a bad die (such a string corresponds to a vector in the description of WWI). The objective is to form n stacks (a stack corresponds to an m-tuple) by integrating one wafer from each lot (a set V_i) while maximizing the yield i.e., maximizing the total number of good dies in the resulting stacks (or equivalently, minimizing the total number of bad dies in the resulting stacks). Integrating two wafers can be seen as superimposing the two corresponding strings; in this operation the position in the merged string is only 'good' when the two corresponding entries are good, otherwise it is 'bad'. Due to this reason we call the above problem wafer to wafer integration (WWI) problem. We refer to it as WWI-m, where is m is the number of wafer-lots.

Notice that the yield optimization problem described here is a special case of WWI, since instances of the yield optimization problem have 0-1 vectors (instead of vectors with arbitrary integral entries). However, since the approximation results that we derived are valid for this more general setting, we opted to focus on the case of arbitrary vectors.

Dimensions of typical instances occurring in the semi-conductor industry have $m = 10, n = 50, p = 1000$ [5][8].

1.2 Goal and Related work

Our main intention in this paper is to formulate the WWI-m as a combinatorial optimization problem and study it from an approximation point of view. Usually, the yield optimization problem is formulated as a maximization problem, however, we feel that studying the minimization problem is especially relevant from approximation point of view. Indeed, owing to the fact that in the yield optimization instances, the number of bad dies in each wafer is typically much less than the number of good dies, it make sense to be able to approximate the (smaller) minimization optimum instead of the (larger) maximization optimum.

There is increasing attention for the yield optimization in the literature. One example is the contribution [5]. In [5] the problem is formulated as an multi-index assignment problem; further, computational performance with straightforward heuristics is reported. Some recent work on this problem is also reported in [7] [8]. As we will show, WWI can be seen as a multi-index assignment problem where the costs have a certain structure. Research on this type of problems is reported in [2]. Similar type of Multi-index assignment problems with decomposable costs are studied in [3],[1]. An survey on multi-dimensional assignment problems can be found in [6].

1.3 Our results

Our results can be summarized as follows:

- We present an IP-formulation that is an alternative to the traditional formulation given in [5]. This alternative formulation contains fewer variables, and may be more suited from a computational perspective (see Section 2).
- We prove that the yield optimization problem is NP-hard (see Section 3).
- We give two simple approximation algorithms for WWI-3, one with a $\frac{3}{2}$ performance guarantee, and one with a $\frac{4}{3}$ performance guarantee (see Section 4.1). We also show that natural extensions of these algorithms to the case of arbitrary m fail to provide a constant-factor guarantee (see Section 4.2).
- We show that, in case of a fixed m and a fixed p, the yield optimization problem is solvable in polynomial time (see Section 5).

2 Problem Formulation

In subsection 2.1 we give a straightforward formulation of the yield optimization problem in wafer to wafer integration as a m-dimensional axial assignment problem, see also [5]. Section 2.2 presents an alternative IP-formulation that may be more suited from a computational perspective.

2.1 IP formulation

We set $K = V_1 \times V_2 \times \ldots \times V_m$, ie, K corresponds to the set of m-tuples. Next, for each $a \in K$, there is a binary variable x_a indicating whether m-tuple a is selected ($x_a = 1$) or not ($x_a = 0$). The formulation is now as follows (see also [5])

$$\min \qquad \textstyle\sum_{a \in K} w(a) \cdot x_a \tag{1}$$

$$\textstyle\sum_{a:\ u \in a} x_a = 1 \ \text{ for each } \ u \in \cup_{i=1}^{m} V_i, \tag{2}$$

$$x_a \in \{0,1\} \ \text{ for each } a \in K. \tag{3}$$

Observe that constraints (2) ensure that each vector u is in an m-tuple.

2.2 Alternative IP formulation

In this section we give an IP formulation that is different from the classical formulation and contains fewer variables.

In this formulation, we model the problem by treating V_1 as the *hub*. Each node in $\cup_{i=2}^{m} V_i$ is assigned to a node in V_1; this decision is modelled by a binary variable as follows. There is a variable $z_{u,v}$, where $u \in V_1$ and $v \in \cup_{i=2}^{m} V_i$, such that:

$$\begin{aligned} z_{u,v} &= 1 \text{ if vectors } u \text{ and } v \text{ are contained in the same } m\text{-tuple}, \\ &= 0 \text{ otherwise.} \end{aligned}$$

In addition, we introduce variables $y_{u,\ell}$ as follows.

$$y_{u,\ell} = \text{the value in the } \ell\text{-th position of the m-tuple containing vector } u \in V_1.$$

$$\min \sum_{u \in V_1} \sum_{\ell=1}^{p} y_{u,\ell} \tag{4}$$

$$\sum_{u \in V_1} z_{u,v} = 1 \tag{5}$$

$$\sum_{v \in V_i} z_{u,v} = 1 \tag{6}$$

$$y_{u,\ell} \geq \max(u_\ell, v_\ell) \cdot z_{u,v} \tag{7}$$

$$z_{u,v} \in \{0,1\} \tag{8}$$

(5) is for each $v \in \cup_{i=2}^{m} V_i$; (6) is for each $u \in V_1$, for each $i = 2, \ldots, m,$; (7) is for each $u \in V_1$, for each $v \in \cup_{i=2}^{m} V_i$, $1 \leq \ell \leq p$; (8) is for each $u \in V_1$, for each $v \in \cup_{i=2}^{m} V_i$.

Observe that this alternative formulation has very few variables ($O(mn^2 + np)$) when compared to the number of variables in classical assignment formulation ($O(n^m)$). Even for reasonably small instances it will be difficult to solve the resulting problem with IP solvers using the classical formulation, whereas we might be able to solve them using (4)-(8).

3 Complexity of WWI

In this section we describe a reduction from MAX-3DM to WWI. Recall that for a given pairwise disjoint sets X,Y,Z, and a set of ordered triples $T \subseteq X \times Y \times Z$, a *matching* in T is a subset of $M \subseteq T$ in which no two ordered triples in M agree in any coordinate. The goal of the MAXIMUM 3-DMENSIONAL MATCHING problem (shortly, MAX-3DM) is to find a matching in T of maximum cardinality.

Kann [4] showed that the 3-bounded MAX-3DM is Max SNP-complete (hence also APX-complete).

Reduction. Consider an arbitrary instance I of MAX-3DM with three sets $X = \{x_1, ..., x_q\}$, $Y = \{y_1, ..., y_q\}$, and $Z = \{z_1, ..., z_q\}$, and a subset $T \subseteq X \times Y \times Z$. Let the number of triples be denoted by $|T|$. Further, let the number of triples in which element x_i occurs, be denoted by $\#occ(x_i)$, $i = 1, \ldots, q$.

Starting from the instance I of MAX-3DM, we now build a corresponding instance I' of WWI-3 by specifying V_i $(i = 1, 2, 3)$, as follows:

- for each element in $x_i \in X$ there is a vector $v_{1i} \in V_1$
- for each element in $y_j \in Y$ there is a vector $v_{2j} \in V_2$
- for each element in $z_k \in Z$ there is a vector $v_{3k} \in V_3$
- each vector has length $|T|$ i.e., $p = |T|$
- for each triple $e = (x_i, y_j, z_k) \in T$, there is a position in each vector corresponding to that triple. The three vectors v_{1i}, v_{2j}, and v_{3k} corresponding to triple (x_i, y_j, z_k), have a '0' in that position, all other vectors have a '1' in that position.

This completes the description of WWI-3 instance.

It is easy to see that a solution to an instance of MAX-3DM with value k corresponds to a solution to the corresponding instance of WWI-3 with value $pq - k$. Hence NP-hardness of WWI-3, even for yield maximization, follows.

4 Approximation algorithms for WWI-m

In this section we first prove that a straightforward algorithm (called heuristic H) for WWI-3 is a $\frac{3}{2}$ approximation algorithm. We the show how a simple modification of this heuristic allows us to improve the worst-case ratio to $\frac{4}{3}$. We then show that natural extension of heuristic H to WWI-m can perform arbitrarily bad.

4.1 The Case $m = 3$

Theorem 1. *Heuristic H is a $\frac{3}{2}$-approximation algorithm for WWI-3. This bound is tight.*

Algorithm 1 Heuristic H

1. Solve an assignment problem between V_1 and V_2, based on costs $c(u \vee v)$, $u \in V_1, v \in V_2$. Call the resulting matching M.
2. Solve an assignment problem between M and V_3 based on costs $c((u \vee v) \vee w)$, $u \vee v \in M$, $w \in V_3$.

Proof. We first introduce some notation. Let OPT denote the value of an optimal solution, and let $cost(H)$ refer to the value of the solution found by H. Let $c(V_i)$ equal total cost of the vectors in V_i, ie, $c(V_i) = \sum_{u \in V_i} c(u)$, for $i = 1, 2, 3$. Let c_{12}^{OPT} denote the value of a partial optimal solution restricted to $V_1 \times V_2$, ie, when we remove from the optimal solution the vectors from V_3; the total weight that remains equals c_{12}^{OPT}. Recall that M refers to matching found by H in the first step, and let c_{12}^{H} be the value of the partial solution obtained after Step 1 of the heuristic H.

Let us call x (y) the amount with which the value of a partial optimal (heuristic) solution increases when vectors from V_3 are matched optimally to the optimal (heuristic) pairs from $V_1 \times V_2$, ie, $x = OPT - c_{12}^{OPT}$.

The following inequality is valid:

$$\begin{aligned} c(V_3) &\le OPT \\ c(V_3) - x &\le c_{12}^{OPT}. \end{aligned}$$

Consider a set U consisting of n p-dimensional vectors with total cost $c(U) = \sum_{u \in U} c(u)$. In addition, consider a set V, also consisting of n p-dimensional vectors. Let us now assign the vectors from V to the vectors of U using as a cost $c(u \vee v)$ for each $(u, v) \in U \times V$. Let the value of the resulting optimal solution be denoted by $c(U \times V)$. We say that an amount equal to $c(V) - (c(U \times V) - c(U))$ from V is *covered* by U (or equivalently, we say that U is able to cover an amount of $c(V) + c(U) - c(U \times V)$ from V).

Consider now the partial heuristic solution found after Step 1, ie, consider M.

Lemma 1. *There exists a feasible assignment of the vectors in V_3 to the pairs from M such that at least the amount $\frac{1}{2}(c(V_3) - x)$ from V_3 is covered by M.*

Argument: To argue that the lemma is true, consider the partial optimal solution restricted to $V_1 \times V_2$. Apparently, these are able to cover from V_3 an amount equal to $c(V_3) - x$ when assigning the strings from V_3 to these pairs (since $OPT = c_{12}^{OPT} + x$). However, each vector in $V_1 \times V_2$ consists of p numbers, each one arising from either V_1 or V_2. Thus, we can partition the amount covered $c(V_3) - x$ into two parts: one part covered by numbers from V_1, one part covered by numbers from V_2. It follows that if one considers the following two assignments: one where you assign the vectors from V_3 to the vectors from V_1 as in the optimal solution, and one where you assign the vectors from V_3 to the vectors from V_2 as in the optimal solution, at least one of these solutions will cover $\frac{1}{2}(c(V_3) - x)$. This proves the lemma.

We can now derive

$$\begin{aligned}
\text{cost}(H) &= c_{12}^H + y \\
&\leq c_{12}^H + c(V_3) - (\frac{1}{2}c(V_3) + \frac{1}{2}x) \\
&= c_{12}^H + \frac{1}{2}c(V_3) + \frac{1}{2}x \\
&\leq c_{12}^{OPT} + \frac{1}{2}c(V_3) + \frac{1}{2}x \\
&\leq c_{12}^{OPT} + \frac{1}{2}[c_{12}^{OPT} + x] + \frac{1}{2}x \\
&\leq \frac{3}{2}c_{12}^{OPT} + \frac{3}{2}x = \frac{3}{2}OPT.
\end{aligned}$$

The first inequality follows from Lemma 1, the second inequality follows from the fact that the heuristic, in Step 1, computes an optimum assignment between sets V_1 and V_2 whose costs cannot exceed c_{12}^{OPT}, and the final inequality follows from the definition of x. Tightness follows from the instance depicted in Figure 1: observe that, for this instance, OPT = 2, whereas heuristic H might find a solution with value 3. □

A minor modification of heuristic H (denoted by H_{heavy}) allows us to improve the worst-case ratio without actually increasing the computational effort. Indeed, let us slightly modify H by ensuring that in Step 1 the *heaviest* set V_i is present, ie, we ensure that the set V_i for which $c(V_i)$ is maximal, is assigned to some $V_j, j \neq i$ in the first step.

Algorithm 2 Heuristic H_{heavy}

0. Let $j = \arg\max_{i=1,2,3} c(V_i)$.
1. Solve an assignment problem between V_j and some $V_i, i \neq j$, based on costs $c(u \vee v)$, $u \in V_j, v \in V_i$. Call the resulting matching M.
2. Solve an assignment problem between M and the remaining set $V_k, k \neq j, k \neq i$ based on costs $c((u \vee v) \vee w)$, $u \vee v \in M$, $w \in V_k$.

Theorem 2. *Heuristic H_{heavy} is a $\frac{4}{3}$-approximation algorithm for WWI-3. This bound is tight.*

Proof. Let us assume, without loss of generality, that set V_1 is the heaviest set. Thus, we have $c(V_1) \geq c(V_2)$ as well as $c(V_1) \geq c(V_3)$. Even more, let us assume (again wlog) that in Step 1 of H_{heavy} sets V_1 and V_2 are assigned to each other. We distinguish three cases.

Case 1: $0 \leq c(V_1) \leq \frac{1}{3}$OPT.
This case is trivial since any feasible solution is in fact optimal: $\text{cost}(H_{heavy}) \leq c(V_1) + c(V_2) + c(V_3) \leq 3 \cdot \frac{1}{3}\text{OPT} = \text{OPT}$.

Case 2: $\frac{1}{3}\text{OPT} < c(V_1) \leq \frac{2}{3}\text{OPT}$.
This case is similar to the analysis in Theorem 1. We derive:

$$\begin{aligned}\text{cost}(H_{heavy}) &= c_{12}^{H_{heavy}} + y \\ &\leq c_{12}^{H_{heavy}} + c(V_3) - (\frac{1}{2}c(V_3) + \frac{1}{2}x) \\ &= c_{12}^{H_{heavy}} + \frac{1}{2}c(V_3) + \frac{1}{2}x \\ &\leq \text{OPT} + \frac{1}{2}c(V_3) \leq \frac{4}{3}OPT.\end{aligned}$$

The last inequality follows from the assumption in this particular case, and the fact that $c(V_3) \leq c(V_1)$.

Case 2: $\frac{2}{3}\text{OPT} < c(V_1) \leq \text{OPT}$.
We denote by Q the weight from V_3 that is covered by V_1 when we solve an assignment problem between V_1 and V_3. The following is true:

$$c(V_1) + c(V_3) - Q \leq \text{OPT}. \tag{9}$$

We now derive:

$$\begin{aligned}\text{cost}(H_{heavy}) = c_{12}^{H_{heavy}} + y &\leq c_{12}^{H_{heavy}} + c(V_3) - Q \\ &\leq c_{12}^{OPT} + c(V_3) - Q \\ &\leq c_{12}^{OPT} + \text{OPT} - c(V_1) \\ &\leq \text{OPT} + \frac{1}{3}\text{OPT} = \frac{4}{3}OPT.\end{aligned}$$

The first inequality follows from Step 2 of H_{heavy}, the second from Step 1 of H_{heavy}, the third inequality follows from (9), and the last inequality follows from the assumption in this particular case.

Tightness follows from the instance depicted in Figure 2: observe that, for this instance, OPT $= 6$, whereas heuristic H_{heavy} might find a solution with value 8.

□

An obvious improvement to heuristic H and H_{heavy} would consist of a heuristic that runs H for all possible pairs in the first step, add the remaining set in the last step, and then choosing the best of the three feasible solutions found. Interestingly, this heuristic (which involves solving 6 assignment problems) does not have a lower worst case ratio than H_{heavy} (which only solves two assignment problems). This also follows from the example depicted in Figure 2.

Notice that heuristic H, in contrast to H_{heavy} can be seen as an online algorithm for a natural, online variant of WWI-3. Indeed, consider the setting where the sets V_1, V_2, and V_3 arrive sequentially over time, and that, before the arrival of a next set, the just arrived set V_i must be assigned to the partial tuples. Results given above imply directly:

V_1	V_2	V_3
100000	000010	001000
010000	000001	000100
001000	100000	000010
000100	010000	000001
000000	000000	000000
000000	000000	000000

Fig. 2. H_{heavy} bad example; OPT = 6, SH = 8

Corollary 1. *Heuristic H is a $\frac{3}{2}$ competitive algorithm.*

Clearly, in this framework, H_{heavy} is not an online algorithm.

4.2 The Case of arbitrary m

A natural extension of heuristic H to the case of arbitrary m is as follows. We iteratively assign set V_i to the existing partial tuples from $V_1 \times V_2 \times \ldots \times V_{i-1}$. Let us call the resulting heuristic H_{seq}. The performance of H_{seq} can be arbitrarily bad as can be seen from the description of the following instances. To understand these instances, it can be helpful to see each vector as a circle with p positions; in such a circle, the 1s, as well as the 0s, will appear consecutively. Let $v_{i,j}$ denote the j-th vector from V_i. Formally, the instances are described as follows:

Choose m such that there exists a value of p with $m = p(p-1)+1$ (thus, in these instances, the length of a vector increases with m), and set $n = p$.

- for each $k \in \{1, \ldots, p-1\}$, there are 1s in position $i-(k-1)p$ to position $i-(k-1)p+k-1$ (modulo p) in vector $v_{i,1}$, for each $i \in \{(k-1)p+1, kp\}$.
- There is a 1 in each position of the vector $v_{(p(p-1)+1,1}$.
- Each other vector is an all-zero vector.

Notice that the cost of an optimal solution equals p, whereas H_{seq} may find a solution with cost $m = p(p-1)+1$. Therefore, the worst-case ratio of H_{seq} is at least $O(\sqrt{m})$. An instance with $p = 3$ is depicted in Figure 3.

Another natural heuristic to consider is the so-called Multiple Hub-Heuristic (see [1]), which can be informally described as in Algorithm (3):

The performance of the multiple hub heuristic MH can be arbitrarily bad. Indeed, consider the following instance. The length of each vector equals 2, ie, $p = 2$, and consider some even value for the number of sets m. let $n = \frac{m}{2}+1$.

V_1	V_2	V_3	V_4	V_5	V_6	V_7
100	000	000	110	000	000	111
000	010	000	000	011	000	000
000	000	001	000	000	101	000

Fig. 3. H_{seq}: a bad example; OPT = 3, MH = 7

Algorithm 3 Multi-Hub-Heuristic MH

for h = 1 to m **do**
  for i = 1 to m **do**
    1. Solve an assignment problem between V_h and V_i, $i \neq h$, based on costs $c(u \vee v)$, $u \in V_h, v \in V_i$. Call the resulting matching M_{hi}.
  end for
  Combine all M_{hi}, to construct M_h.
end for
Output the min-cost solution of all M_h.

The first vector of each of the sets V_i, $i = 1, 2, \ldots, \frac{m}{2}$ is specified as follows: For $i = 1, 2, \ldots, n$, put $v_{i,1} = (1\ 0)$; for $i = \frac{m}{2} + 1, \ldots, m$ put $v_{i1} = (0\ 1)$. All other vectors in the instance are equal to $(0\ 0)$. It can be seen that $OPT = 2$ whereas $\text{cost}(MH) = \frac{m}{2} + 1$.

Notice that this performance is in contrast with the performance of the multiple hub-heuristic for other variants of decomposable minimum cost m-dimensional assignment problems, see [1].

5 Fixed p

In this section we consider the yield optimization problem, i.e., we consider instances that feature 0-1 vectors only. We will argue that instances of the yield optimization problem with a fixed p can be solved in polynomial time (for each fixed m).

Consider a solution of the yield optimization problem. It consists of n 0-1 vectors. Thus, we can classify these n 0-1 vectors as belonging to at most 2^p different types (each type corresponding to a distinct 0-1 vector of length p). We use the symbol t to index these types.

We say that a vector from type t is *compatible* with a vector from type s if the vector of type t has a '1' in each of the positions where the vector of type s has a '1'. We write type t is *compatible* with a vector from type s as $t \succ s$. Further, given an instance of the yield optimization problem, we let k_s^i denote the number of 0-1 vectors of type s in set V_i, $s = 1, \ldots, 2^p$, $i = 1, \ldots, m$.

We construct the following formulation that features variables x_t:

$$x_t = \text{number of 0-1 vectors of type } t \text{ in the final solution, } t = 1, \ldots, 2^p.$$

We also need "transportation" type variables; for each $i = 1, \ldots, m, s, t = 1, \ldots, 2^p$:

$$z^i_{s,t} = \text{number of 0-1 vectors of type } s \text{ from set } V_i \text{ assigned to class } t.$$

The formulation (with parameter c_t referring to the number of '1's in a vector from type t):

$$\min \sum_{t=1}^{2^p} c_t x_t \tag{10}$$

$$\sum_{s:\ t \succ s} z^i_{s,t} = x_t \qquad \text{for each } t = 1, \ldots, 2^p, i = 1, \ldots, m, \tag{11}$$

$$\sum_{t:\ t \succ s} z^i_{s,t} = k^i_s \qquad \text{for each } s = 1, \ldots, 2^p, i = 1, \ldots, m, \tag{12}$$

$$x_t, z^i_{s,t} \quad \text{integer} \qquad \text{for each } s, t = 1, \ldots, 2^p, i = 1, \ldots, m, \tag{13}$$

The objective funcion (10) minimizes the total cost. (11)-(12) resemble the familier transportation constraints. Constraints (12) enforce that each vector in V_i is assigned to some type t. Constriants (11) enforce that each vector of type t such that $x_t > 0$ is assigned to exactly one vector in V_i. Given a feasible solution to (10)-(13) one can construct a feasible solution to WWI-m as follows: (1) Create a set X of n vectors with x_t vectors of type t, (2) Solve an assignment problem between X and V_i, for each $i = 1, \ldots, m$, (3) Construct m-tuples of vectors by matching m vectors one from each V_i together in an m-tuple if they all are matched to same vector in X in (2).

Observe that this formulation involves $O(m2^{2p})$ variables, and $O(m2^p)$ constraints.

Lemma 2. *Formulation is correct.*

Proof. Consider a feasible solution to the yield maximization problem. This solution prescribes for each type of vector in each set V_i how many of these vectors are assigned to a vector of type t. This determines the $z^i_{s,t}$ values; clearly, these values will satisfy constraints (10)-(13), since our solution is valid. Vice versa, consider $z^i_{s,t}$ values that satisfy (10)-(13). This corresponds directly to a feasible solution. □

When we fix p and m the above formulation has a fixed number of variables and constraints. Thus we can use Lenstra's algorithm to solve this IP in polynomial time. This implies:

Corollary 2. *For each fixed p, and for each fixed m, the yield maximization problem can be solved in polynomial time.*

References

1. Bandelt, H., Crama, Y., Spieksma, F.: Approximation algorithms for multi-dimensional assignment problems with decomposable costs. Discrete Applied Mathematics 49, 25–50 (1994)
2. Burkard, R., Dell'Amico, M., Martello, S.: Assignment Problems. SIAM (2009)
3. Crama, Y., Spieksma, F.: Approximation algorithms for three-dimensional assignment problems with triangle inequalities. European Journal of Operational Research 60, 273–279 (1992)
4. Kann, V.: Maximum bounded 3-dimensional matching is max snp complete. Information Processing Letters 37, 27–35 (1991)
5. Reda, S., Smith, L., Smith, G.: Maximizing the functional yield of wafer-to-wafer integration. IEEE Transactions on VLSI Systems 17, 1357–1362 (2009)
6. Spieksma, F.C.R.: Multi-index assignment problems: complexity, approximation, applications. in L. Pitsoulis and P. Pardalos, eds., Nonlinear Assignment Problems, Algorithms and Applications (Kluwer Academic Publishers pp. 1–12 (2000)
7. Taouil, M., Hamdioui, S.: Layer redundancy based yield improvement for 3d wafer-to-wafer stacked memories. IEEE European Test Symposium pp. 45–50 (2011)
8. Verbree, J., Marinissen, E., Roussel, P., Velenis, D.: On the cost-effectiveness of matching repositories of pre-tested wafers for wafer-to-wafer 3d chip stacking. IEEE European Test Symposium pp. 36–41 (2010)