A TWO-PROBE ROUTING MODEL AND AN ALGORITHM FOR MCM SUBSTRATES TESTING

Keisuke Murakami
National Institute of Informatics
Chiyoda-ku, Tokyo 101-8430, Japan
murakami@nii.ac.jp

Shunji Umetani and Hiroshi Morita
Department of Information Physics and Sciences
Osaka University
Suita, Osaka 565-0871, Japan
{umetani, morita}@ist.osaka-u.ac.jp

Abstract

This paper presents a novel model and an algorithm for the probe routing problem of MCM substrates testing. The probe routing problem is too difficult to be solved because of dense assembly of MCM substrates and large time of testing requires. Therefore, it is essential to reduce the testing time and our objective of this paper is to minimize the completion time of MCM substrates testing. We propose a constrained shortest path problem (CSP) model for MCM substrates testing problem. Furthermore, we propose solution methods using labeling approaches. In computation experiments, we compare the results of our model and algorithm with an existing method.

Keywords: MCM substrate, CSP, labeling approach, set covering problem.

1 INTRODUCTION

Multi-chip module (MCM) substrate is designed to pack two or more semiconductor chips. A lot of different chip dies are connected through a wired interconnect on a MCM substrate. The end of wired interconnect is called "pin" and a set of pins which are to be electrically connected is called "net" (Fig.1). MCM is a technology that provides the dense assembly capability and the increase in the packaging of chips at relatively low cost. Because of high density assembly, MCM has some advantages in terms of power consumption, package volume and so on (Yao et al. 1994). On the other hand, high density assembly causes faults on MCM substrates (Chu et al. 1994) (Ma and Lombardi 2008). These faults are classified into "open fault" and "short fault". An open fault is an electrical disconnection between two points (pins) which are to be connected. A short fault is an electrical connection between two nets which are not to be connected. Thus, testing of MCM substrates is essential. This testing checks whether faults exist or not, by touching two pins simultaneously. Then the number of probes is more than two. However, on the dense assembly such as MCM, this testing requires too much time. Therefore an efficient method for testing should be established.

Several previous works propose the two-phase method (Kim et al. 1999) (Kahng et al. 1996). The first phase determines the set of pairs of pins simultaneously touched and the second phase designs the route of probes. The second phase corresponds to solving TSP. The solution of this two-phase method depends heavily on the first phase. When we do not select an appropriate set of pairs on the first phase, the solution is also much worse even if we obtain an optimal solution to TSP on the second phase. Kahng et al. propose, on the first phase, the method for selecting a minimum pair set (the pair set size is minimum). This method might work well for some instances. However, we think that the minimum pair set is not always appropriate. For example, if there is a net which have most pins in one side (Fig.1), we should design the probes' route that one probe touches most pins of one side and the other probe stays a pin of the other side. Then the minimum pair sets is not useful at all. Therefore, we propose a model which can determine a set of pairs of pins and design the routes of probes at the same time. We model the MCM substrate testing problem as a constrained shortest path problem (CSP). Our model can ob-

Fig. 1 A net with seven pins configuration

Copyright © 2011 by the Japan Society of Mechanical Engineers
tain an optimal solution to non-large scale problem even if it is impossible for two-phase method. We solve this CSP by labeling approaches, which are improvement of the dynamic programming method and follow the line of Pareto-optima approaches. This CSP is also regarded as a kind of set covering problem. Thus we apply the greedy heuristic for the set covering problem to subproblems of CSP for labeling. In real world, there exist a MCM of varied sizes, up to about 10,000 pins. In this paper, we assume that the size of MCM is about 1,000 pins. Also, we assume that the number of probes is two.

This paper is organized as follows. Section 2 presents MCM substrates testing problem description. In Section 3, we model the MCM substrates problem as a CSP. In Section 4, we propose the exact and approximate solution methods, respectively. Both solution methods are based on labeling approaches. Furthermore, we improve CSP model in order to reflect the more real-world problems in Section 5. In Section 6, we compare the result of our methods with Kahng et al’s minimum pair set approach. Section 7 concludes this paper.

2 PROBLEM DESCRIPTION

At the probe routing problem, the testings for open fault and short fault are almost not different, because both testings check faults by touching two pins. Even if testing for open fault selects two pins of a net and testing for short fault selects two pins from different nets, respectively, this does not make much of a difference at the probe routing problem. Hence, even though we deal with only one kind of fault, we can extend to two kinds of fault easily. Thus in this paper, we deal with only open fault.

MCM substrate is aggregation of nets and a net is aggregation of several pins which are connected through a wired interconnect each other. Two probes simultaneously touch two pins, respectively and then the wired interconnect between the two pins are checked. In other words, the interconnection wire between a via and a pin (or another via) is defined as a "segment wire" (Fig.1) and then, the probes check the connection of all segment wires between two pins. The probes are allowed to touch a pin many times even if the pin has already been touched.

During MCM substrate testing, the probes check the whole segment wire on MCM substrate. Our objective is to minimize the completion time. We assume that the probes’ travel time is equal to the Euclidean distance. The testing start and termination points (pins) of each probe are fixed and the pair of start and termination pins is same. The probes are allowed to travel from a net to another one before the testing of a net is completed. Two probes are not allowed to touch a pin at the same time. Furthermore, some pairs of pins are not able to be touched at the same time because the probes conflict with each other.

In more real-world problem, it is likely that one probe is allowed to go through a certain path, but the other probe is not because of the physical reason and so on. If this situation does not happen at all, we do not have to distinguish between two probes. However, if this situation happens, we must distinguish between two probes. When two probes are distinguished, the probe routing problem is more complex. In Section 3, we explain a CSP model where two probes are not distinguished. In Section 5, we present an improved CSP model where two probes are distinguished.

3 PROBE ROUTING MODEL

In this section, we model the probe routing problem as a constrained shortest path problem (CSP). We generate all available pairs of pins and then we eliminate unavailable pairs, which are not be touched at the same time. An available pair of pins is regarded as a node in CSP. Let N be the set of the nodes. Each node includes the set of segment wires checked by that node (touching pair of pins). Let S be the set of segment wires. The nodes connect to each other through an arc. The cost of the arc corresponds to the distance (time) of travel between the two nodes. The cost $c_{i,j}$ is computed as follows.

Let $p_i^1$ and $p_i^2$ be the two pins of node $i$, respectively and let $d(k,h)$ be the distance of travel from pin $k$ to pin $h$. The combinations of paths of two probes are only two ways (Fig 2). Way 1 is that one probe travels from $p_i^1$ to $p_j^1$, and the other probe travels from $p_i^2$ to $p_j^2$. Way 2 is that one probe travels from $p_i^1$ to $p_j^1$, and the other probe travels from $p_i^1$ to $p_j^2$. One way containing the maximum distance in the above four paths is not valid but the other way is valid, because the probes do absolutely not go through the path of the maximum distance. For example when the distance $d(p_i^2, p_j^1)$ is the maximum distance, Way I is valid. The two paths of the valid way are compared and then the larger distance of the two becomes the cost of travel from node $i$ to $j$. We describe a series of this by the following.

The maximum distance between node $i$ and $j$ is

$$d(p_i^1, p_j^1) = \max \{d(p_i^1, p_j^1), d(p_i^2, p_j^2), d(p_i^1, p_j^2), d(p_i^2, p_j^1)\}$$

(1)

where $p_i^1$ and $p_j^1$ are the two pins of the arc of the maximum distance. The travel cost from node $i$ to $j$ is

$$c_{i,j} = \max \{d(p_i^1, \{p_i^1\} \\{p_j^1\}), d(\{p_i^1\} \{p_i^2\} \{p_j^1\}, p_j^1)\}$$

(2)

Equation (2) implies when the maximum distance is eliminated, the candidates for the cost of the arc are $d(p_i^1, \{p_i^1\} \{p_j^1\})$ and $d(\{p_i^1\} \{p_i^2\} \{p_j^1\}, p_j^1)$. Then the cost $c_{i,j}$ is the larger distance of the two.

The constrained shortest path problem is formulated as follows. Let $a$ and $b$ be the source and target node, respectively. Let $x_{i,j}$ be the 0-1 decision variable. If the probe travels from node $i$ to node $j$, $x_{i,j} = 1$; otherwise, $x_{i,j} = 0$. Let $\alpha_{i,j,s}$ be the 0-1 constant. If node $i$ or $j$ check segment wire $s$, $\alpha_{i,j,s} = 1$; otherwise, $\alpha_{i,j,s} = 0$. Let $y_i$ be the integer
The cost between two nodes

**Fig. 2** The cost between two nodes

variable at node $i$.

\[
\min \sum_{i, j \in N} c_{i, j} x_{i, j} \quad (3)
\]

s.t.

\[
\sum_{j \in N} x_{i, j} s_t \geq 1, \quad \forall s \in S \quad (4)
\]

\[
\sum_{j \in N_{\{a, b\}}} x_{i, j} - \sum_{j \in N_{\{a, b\}}} x_{k, j} = 0, \quad \forall k \in N \quad (5)
\]

\[
\sum_{j \in N_{\{a\}}} x_{i, j} = 1 \quad (6)
\]

\[
\sum_{j \in N_{\{b\}}} x_{i, j} = 1 \quad (7)
\]

\[
y_i - y_j + |N| x_{i, j} \leq |N| - 1, \quad \forall i \in N_{\{b\}}, \forall j \in N_{\{a\}}, i \neq j \quad (8)
\]

The objective function (3) is minimization of the sum of the cost. Equation (4) ensures that each segment wire is checked at least once. Equations (5)-(7) are the standard shortest path constraints: Equation (5) is the degree constraint for each node except for source node and target node and equations (6), (7) ensure that one arc leaves the source node and one arc enters the target node. Equation (8) is subtour breaking constraint.

### 4 ALGORITHM

In this section, we explain the labeling approach (Stroetmann 1997). This approach uses a set of labels for each node. A label on a node represents a path from source node to that node. When a new label on a node is generated, we evaluate whether the new label is dominated by the others on that node or not. If the new label is not dominated at all, it is added to that node; otherwise, it is removed.

We propose an exact algorithm and an approximation algorithm using the labeling approach.

#### 4.1 Exact Algorithm for CSP

A label $l$ consists of the set of segment wires $T_l$ and the cost $e_l$ of the corresponding path. The set of segment wires $T_l$ have already been checked. Let $(T_l, e_l)$ be the label $l$.

Then the dominance relation between two labels is defined as:

**Definition 4.1** $(T_m, e_m)$ dominates $(T_l, e_l)$ if and only if $T_m$ contains $T_l$ and $e_m$ is less than or equal to $e_l$

\[(T_l, e_l) \leq (T_m, e_m) \iff T_l \subseteq T_m, e_l \geq e_m\]

Also, the set of labels on node $i$ is represented by

\[L(i) = \{(T_l, e_l) \mid l \in R(i)\}\]

where $R(i)$ is the set of label’s number on node $i$. If a new label on node $i$ is not dominated by $L(i)$ at all, the label is added to $L(i)$. This process is continued until any $L(i)$ ($\forall i \in N$) are not updated. Let $S_j \subseteq S$ be the set of segment wire checked by node $j$ and then the exact algorithm is described as follows.

**Exact Algorithm**

1. do
2. for each $i \in N$ do
3. for each $j \in N_{\{i\}}$ do
4. for each $(T_m, e_m) \in L(i)$ do
5. if $(T_m \cup S_j, e_m + c_{i, j}) \notin (T_l, e_l)$ for each $(T_l, e_l) \in L(j)$
6. then $L(j) \leftarrow L(j) \cup (T_m \cup S_j, e_m + c_{i, j})$
7. end for
8. end for
9. until $L$ is not updated

The target node includes the solution (label) of the exact algorithm and the solution is expressed by

\[\arg \min_{(T_l, e_l) \in L(b)} \{e_l \mid T_l = S\}\]

where the minimum cost (min $e_l$) corresponds the objective function (3) and $T_l = S$ is equal to Equation (4). The other constraints (5) - (8) do not require to be confirmed, because the labeling approach takes account of these constraints.

#### 4.2 Approximation Algorithm for CSP

Indeed the exact algorithm in subsection 4.1 obtains an optimal solution, but this algorithm requires a lot of running time and memory. If the worst should happen, we can never obtain even one feasible solution. Thus, we cannot use the exact algorithm for large scale problems. And so, we propose an approximation algorithm, which can solve large scale problems.

In the exact algorithm, the large number of labels causes time and memory requirement. Therefore, we reduce the number of labels, but the framework of the exact algorithm remains. In the approximation algorithm, we evaluate two labels only by upper bound of cost and then the upper bound is added to the label. This upper bound provides two advantages. One advantage is that the number of labels on each node is always one only. Instead, we must add the route $P_i$
to the label $l$, and the route implies the path from the source node to that node. Since the previous label does not always remain, if $P_l$ does not exist, we cannot know the path from the source node to that node. However the number of labels is not so large that a large memory is not required. The other advantage is that at least one feasible solution is obtained early.

Let $u_l$ be the upper bound of solution on a node, and let $(T_l, e_l, u_l, P_l)$ be the label $l$. Then the dominance relation between two labels is defined as:

**Definition 4.2** $(T_{m}, e_{m}, u_{m}, P_{m})$ dominates $(T_l, e_l, u_l, P_l)$ if and only if $u_m$ is less than or equal to $u_l$

$$(T_l, e_l, u_l, P_l) \leq (T_{m}, e_{m}, u_{m}, P_{m}) \iff u_l \geq u_m$$

Also, the set of labels on node $i$ is represented by

$$L'(i) = \{(T_l, e_l, u_l, P_l) \mid l \in R'(i)\}$$

where $R'(i)$ is the set of label's number on node $i$.

Here we explain how the upper bound is computed. Our CSP model is also characterized as a set covering problem. For the set covering problem, the greedy heuristic has been proposed (Chvatal 1979). However the greedy heuristic does not take care of the shortest path constraints $(5) \cdot (8)$. So we improve the greedy heuristic. The improved greedy algorithm finds a next node to be visited one-by-one. The next node is found as follows.

Let $S_l'$ be the set of segment wires unchecked by the label $l$. In particular, let $S_i' \subseteq S_l'$ be the set of unchecked segment wires of node $i$ in $S_l'$. Then, $S_i'$ is given by

$$S_i' = S_l' \setminus T_i, \quad \forall i \in N$$

When the label $l$ exists on node $i$, we determine the next node by the ratio $|S_i' / c_{i,j}|$ (for $j \in N$). That is, the next node is represented by the following equation.

$$\arg \max_{j \in N} \{|S_i'| / c_{i,j}\}$$

When the next node $j$ is found by Equation (13), the set of unchecked segment wires $S'_j$ (for $k \in N$) are updated by

$$S_k' = S'_j \setminus S_k, \quad \forall k \in N$$

A series of these procedures is continued until all segment wires are checked. In addition, we use 2-opt method in order to improve the solution and let $e_l'$ be the solution (cost) of 2-opt method. The input of the 2-opt method is a route $P_l$, obtained previously by greedy heuristic. Note that, in the 2-opt method, we consider only some nodes which are contained in the route $P_l$, that is, the 2-opt method is exactly the same one that we often use to solve TSP. The improved greedy and 2-opt heuristic is described as follows:

**Improved Greedy+2-opt Heuristic (i, $e_l$, $S_l'$)**

1. $j \leftarrow i$
2. $z \leftarrow 0$
3. while $S'_h \neq \emptyset$ (for $h \in N$) do
4. $P(z) \leftarrow j$
5. $j \leftarrow \arg \max_{k \in N} \{|S'_h| / (c_{j,k})\}$
6. for each $k \in N$ do
7. $S'_k \leftarrow S'_k \setminus S'_j$
8. end for
9. $z \leftarrow z + 1$
10. end while
11. $e_l' \leftarrow 2$-opt method ($P_l$)
12. return $e_l + e_l'$

The "Improved Greedy+2-opt Heuristic" computes the upper bound of the label $l$ on the node $i$. The input $S_l'$ is equal to Equation (12). Line 5 corresponds to Equation (13) and lines 6-8 correspond to Equation (14). We use "Improved Greedy+2-opt Heuristic" as a subroutine of labeling approach. Then if we want to obtain better solutions, the number of labels is allowed to increase. Instead of that, the more time and memory are required. Thus we should take into consideration the trade-off between solution and time (memory) when the number of labels is set. Here, we present our approximation algorithm. Let $u_{\ell}'$ be the upper bound of label $m$. Let $(P_m)'$ be the route, generated by adding the next node $j$ at the tail of $(P_m)$. Let $L^*$ be the best label in all labels that have been found. We further set the upper threshold of the number of label on each node, denoted by $\ell$. We describe our approximation algorithm as follows.

**Approximation Algorithm**

1. $L^* \leftarrow (0, 0, \infty, 0)$
2. do
3. for each $i \in N$ do
4. for each $j \in N' \setminus \{i\}$ do
5. for each $(T_{m}, e_{m}, u_{m}, P_{m}) \in L'(i)$ do
6. $S''_i = S_i \setminus \{T_{m}, u_{m} \mid k \in N\}$
7. $u_{m}' \leftarrow$ Improved Greedy+2-opt Heuristic $(j, e_{m}, S''_i)$
8. for each $(T_{i}, e_{i}, u_{i}, P_{i}) \in L'(j)$ do
9. if $(T_{i}, e_{i}, u_{i}, P_{i}) \leq (T_{m} \cup S_{j}, e_{m} + c_{i,j}, u_{m}', (P_{m})'$
   then $L'(j) \leftarrow L'(j) \cup \{T_{i}, e_{i}, u_{i}, (P_{i})'\}$
10. if $|L'(j)| > \ell$ then $L'(j) \leftarrow \min L'(j)$
11. if $L^* \leq (T_{m} \cup S_{j}, e_{m} + c_{i,j}, u_{m}', (P_{m})'$
   then $L^* \leftarrow (T_{m} \cup S_{j}, e_{m} + c_{i,j}, u_{m}', (P_{m})'$
12. go to next
13. end for
14. next
15. end for
16. end for
17. end for
18. until $L$ is not updated

$L^*$ is the solution, when the algorithm terminates. On line 9, if the new label dominates at least one label on that node, the new label is added to the set of label on that node. Then if the number of labels on that node is more than the upper threshold, the minimum label on that node is removed.
on line 10. The minimum label contains the largest upper bound in that node (Definition 4.2). Besides if the new label dominates all labels on all nodes, the new label becomes the provisional solution, on line 11.

5 THE IMPROVED MODEL

In Section 3, we present the model where two probes are not distinguished. In this section, we explain the improved model, where two probes are distinguished. When two probes are distinguished, we must assign one probe to one pin of the nodes. In order to do so, we divide one node into two nodes. Here we name two probes "Probe A" and "Probe B", and there are two pins $p_i^1$ and $p_i^2$ on node $i$. Then, one of the two divided nodes represents the case where Probe A touches $p_i^1$ and Probe B touches $p_i^2$. The other node represents the case where Probe A touches $p_i^2$ and Probe B touches $p_i^1$. These nodes allows us to describe more real-world situations. For example, if Probe A can touch $p_i^1$, but cannot touch $p_i^2$ because of the physical reason, one of the two divided nodes is not generated. In addition, if Probe A can travel from $p_i^1$ to $p_j^1$, but Probe B cannot travel from $p_i^2$ to $p_j^1$, this arc is cut (Fig. 3). In Fig. 3, nodes $i-1$ and $i-2$ represent the two divided nodes of node $i$.

![Fig. 3 The improved model for CSP](image)

In this way, we generate all feasible nodes and arcs, and then the CSP of the improved model is constructed as Section 3. Note that the two divided nodes are not connected by arc each other. Our algorithms in Section 4 is also available to solve the CSP of the improved model. However, the scale of the CSP of the improved model is larger than that of CSP in Section 3, because the number of nodes of the CSP of the improved model is almost twice as large as that of the CSP in Section 3. When the scale of problem is larger, the solution of approximate algorithm is generally more different from an optimal solution.

If the special cases such as the above examples (Fig. 3) happen at many nodes and arcs, this improved model is effective. If not, this improved model vainly forces the solution to be larger. Thus, if the special cases do not happen so often, we should not employ the improved model. Instead of that, we solve the CSP by the following method. First we solve the CSP for the model of Section 3 even if the special cases happen. Secondly we modify the solution by assigning one probe to one pin of the nodes. Note that the route remains itself. That is, we solve an assignment problem of probes to pins, while we consider the special cases. Then the Dijkstra algorithm is available to solve this assignment problem and we can obtain the solution easily. We expect that this solution suffice when the special cases do not happen so often, though the optimality property is lost.

6 COMPUTATION EXPERIMENTS

We show the results of computational experiments of our method and the existing one. All algorithms are implemented in C++ using STL vector. All the tests are performed on a 2.8 GHz Core2 Duo with Windows operating system using 3 GB of RAM memory.

We compare our exact and approximate algorithms for CSP with the existing method (Kahng et al. 1996). We call the existing method "the minimum pair set method" (for short, MPS). Then we improve and implement MPS as follows. We first enumerate the minimum pair sets of each net. The minimum pair set represents the set minimizing the number of pairs of pins and checking all segment wires. Secondly, we find an optimal combination of the minimum pair sets of each net by branch-and-bound method, while we solve TSP by the nearest neighbor and 3-opt heuristics.

We prepare several patterns of net consisting of 10 pins, such as Fig. 4. Then we select some nets from the patterns of net and generate several instances, where the number of nets is 1, 2, 3, 20, 40, 80 and 100. Then, we name these instances "net1", "net2", ..., "net100", respectively. In these instances, we do not consider the special cases of Section 5. The data of these instances is listed in Table 1, where $|N|$ represents the number of node of CSP and $|S|$ represents the number of segment wires.

![Fig. 4 An example of a net](image)

First, we show the results of the three methods for small scale problems. In Table 2, total cost of each method is shown and the computation time of each method is shown in Table 3. Note that a computational experiment is performed up to 3 days.

In Tables 2 and 3, the exact algorithm found the optimal solutions to smaller instances, but the computation time is too large. The approximate algorithm found the optimal solutions to instance net1 and net2 and the computation time...
is small. The computation time of MPS is also small, but the total cost is not smaller than the others, especially in the instance net2.

Secondly, we show the results of our approximate algorithm and MPS for large scale problems in Table 4 and Table 5. Note that a computational experiment is performed up to 10800 seconds (3 hours). If the experiment is not over in 10800 seconds, the provisional solution is shown. In Table 4, the total cost of our approximate algorithm is smaller in all instances except for net100. The total cost of MPS is smaller only in net100. We think this is because the instance net100 happens to be convenient to MPS, that is, minimum pair set just works well for net100. In Table 5, both the approximate algorithm and MPS are not faster for larger scale instances.

7 CONCLUSION

The main contributions of this paper are twofold. First, for MCM substrates testing problem, we propose a novel CSP model. Second, we propose the approximate algorithm for CSP. Thanks to CSP model and our approximate algorithm, we can obtain an optimal solution to small scale problem, and we can avoid the risk that only much worse solutions or no solution are found though two-phase method has this risk, because our approximate algorithm is available for even large scale problems.

In the computational experiments, we show the solutions of our approximate algorithm is close to exact solutions for smaller scale problems. Next, for larger scale problems, we show the solutions of our approximate algorithm are superior to that of MPS for most problems.

Future work improves the time performance of our approximate algorithm in order to solve further large scale problems.

References


