

Routability Driven Modification Method of Monotonic Via Assignment for 2-layer Ball Grid Array Packages

Yoichi Tomioka

Atsushi Takahashi

Department of Communications and Integrated Systems, Tokyo Institute of Technology
2-12-1-S3-58 Ookayama, Meguro-ku, Tokyo, 152-8550 Japan
e-mail: {yoichi,atsushi}@lab.ss.titech.ac.jp

Abstract— Ball Grid Array packages in which I/O pins are arranged in a grid array pattern realize a number of connections between chips and a printed circuit board, but it takes much time in manual routing. We propose a fast routing method for 2-layer Ball Grid Array packages to support designers. Our method distributes wires evenly on top layer and increases completion ratio of nets by improving via assignment iteratively.

I. INTRODUCTION

In current VLSI circuits, there can be hundreds of required I/O pins. Instead of Dual In-line Package (DIP) or Quad Flat Package (QFP), in which the number of available I/O pins is small, Ball Grid Array (BGA) packages are used to realize a number of connections between VLSI chips and printed circuit boards (PCBs).

Though there exist many approaches for routing and parts of them are included in several tools, most tools are for routing on PCBs or VLSI chips. Most approaches proposed so far can not immediately apply to BGA packages routing which contains special requirements and constraints on BGA packages since it is hard to obtain a routing pattern as good as manual one. The structure of BGA packages is symmetric, and given netlists has some properties. In current design, designers generate routing patterns for BGA packages by using such properties. So, we introduce know-how in manual routing into our methods to obtain satisfactory routing pattern efficiently.

The first approach for BGA package routing was proposed in [1] and was improved in [2]. In these approaches, it is assumed that there is a single routing layer, and that a netlist is not given. These approaches generate a netlist and a global route for each net. Each net connects a finger and a ball. The objective is to balance the congestion over the routing area and shorten the wire length of each net. Since a netlist is usually given in package routing design, these approaches are primarily used for package architecture design or flip-chip bonding design. For a given netlist in two-layer BGA model, as shown in Fig. 1, global routing on layer 1 may be possible by using these algorithms if a candidate of via positions is considered as a ball. The feasibility of the global routes on layer 2, however, is not guaranteed.

Algorithms for multi-layer Pin Grid Array (PGA) and BGA routing have been proposed in [3] and [4], respectively. These algorithms first assign each net to a layer and then generate routes in each layer. However, neither the feasibility of the routes from the finger of each net to the assigned layer nor the routes from the assigned layer to the ball of the net are guaranteed. These routes require vias that are large compared to the wire width, and these algorithms omit the via assignment planning, which is the most difficult part of package routing.

A via assignment and global routing method for single-chip two-layer BGA packages that considers total wire

length and wire congestion have been proposed as the first stage of package substrate routing in [5], and this method has been improved in [6]. In these papers, the concepts of monotonic global routing and monotonic via assignment focusing mainly on layer 1 are introduced. In the method, a via assignment is iteratively improved to minimize the maximum wire congestion on layer 1 while the total wire length on layer 2 is kept to be small enough.

Though the method achieves small total wire length and congestion, several enhancements are required in order to use the method in actual package routing design. In the method, since the via of a net is placed near the ball of the net, the wire of each net on layer 2 is short and the routing on layer 2 seems not to be difficult. However, there is no guarantee that 100% routing on layer 2 is possible. Moreover, in package substrate, various kinds of obstacles exist. For example, mold gates from which resin is poured into the package are placed on layer 1. In the region at which a mold gate placed, routing on layer 1 is not allowed but routing on layer 2 is allowed. Mold gates make it difficult to generate 100% routing since the via of a net may be placed away from its ball if the ball is under a mold gate. Even if the evaluation of a via assignment by cost function is better, it can not be adopted if 100% routing is impossible.

In this paper, we propose a via assignment and global routing method which is an enhancement of the method proposed in [6]. In our proposed method, the maximum wire congestion on layer 1 and the wire length of a net on layer 1 and layer 2 are minimized. Moreover, a global routing on layer 2 is generated in the final stage to guarantee the 100% routing if a feasible via assignment is obtained. Our method consists of two phases. In the first phase, a via assignment is iteratively improved to minimize the maximum wire congestion on layer 1 while the total wire length on layer 2 is kept to be small enough. Though this phase is based on the method proposed in [6], the computational complexity to obtain the maximum gain is improved from $O(N^2)$ to $O(N)$, where N is the number of grid nodes. In the second phase, a via assignment is iteratively improved to improve the routability on layer 2 while the maximum wire congestion on layer 1 is maintained. In this phase, global routing on layer 2 is generated. New modification to improve the routability on layer 2 while maintaining the maximum wire congestion on layer 1 is introduced. In our experiments, our method obtains a via assignment which distributes wires evenly faster than the method proposed in [6], and the routability on layer 2 is drastically improved.

II. PRELIMINARY

A. Problem definition

In this paper, we consider a basic model of BGA package as shown in Fig. 1. Our BGA package model has two routing layers and single chip which is smaller than

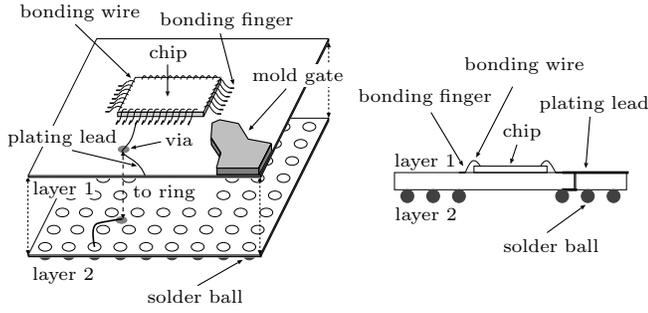


Fig. 1. A model of 2-layer BGA package.

package size. A bonding finger, which we will refer to as a finger, is connected to the chip by a bonding wire. Bonding fingers are placed on the perimeter of a rectangle enclosing the chip on layer 1. A solder ball, which we will refer to as a ball, is an I/O pin of the package, and is connected to the PCB. Solder balls are placed in a grid array pattern on layer 2. There are connection requirements between bounding fingers and solder balls. The connection requirement is called a net, and is realized by wires on each layer and vias which connect wires on different layer. The number of vias to be placed in the area surrounded by four adjacent balls is at most 1.

Mold gates are in some corners of top layer to pour resin into the package. In the region at which a mold gate placed, routing on layer 1 is not allowed but routing on layer 2 is allowed.

Ring structure which is used for electric plating surrounds the package. Each net should be connected to the ring in order to enable electric plating to protect its wires. The extra connection to the ring of a net is called a plating lead. The ring is cut when the package is used. A plating lead is redundant for operation, but is normally used to reduce the fabrication cost and to improve the reliability.

The routing area of a package is usually divided into sectors. Our approach is applied to each sector. In the following, we focus on the bottom sector as shown in Fig. 2.

In this paper, we assume that a net consists of a finger and a ball. Nets are labeled according to the order of fingers on the perimeter from the left to the right as n_1, n_2, n_3, \dots

Since the radius of a ball is large compared to the interval of the balls, the number of possible routes between adjacent balls on layer 2 is at most one. Therefore, routes on layer 2 should be short and most of plating leads should be routed on layer 1. For this reason, we restrict the route of each net so that it has only one via, the wire on layer 1 connects the finger of the net and the ring through the via of the net, and the wire on layer 2 connects the ball and the via of the net.

The set of candidate locations of vias which include the locations within a mold gate is represented by the via grid array \mathcal{N} . The interval of via grid array \mathcal{N} is the same as that of the balls, and is unit length as shown in Fig. 2. An element in \mathcal{N} is called a grid node. We assume the number of possible routes on layer 1 between two vias placed in adjacent grid nodes under a design rule is at most h where h is the number of rows of balls.

The ball and the via of net n_i are denoted by b_i and v_i , respectively. The positions of b_i and v_i are denoted by (x_i^b, y_i^b) and (x_i^v, y_i^v) , respectively.

Let \mathcal{V} be the set of vias of the nets. A via assignment to \mathcal{N} is represented by bijection $\Phi: \mathcal{V} \cup \mathcal{E} \rightarrow \mathcal{N}$, where \mathcal{E} is the set of dummy vias.

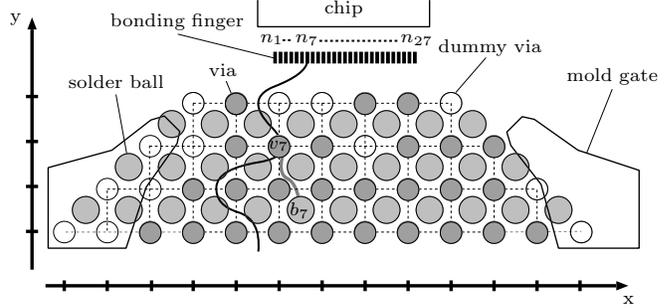


Fig. 2. Bottom sector.

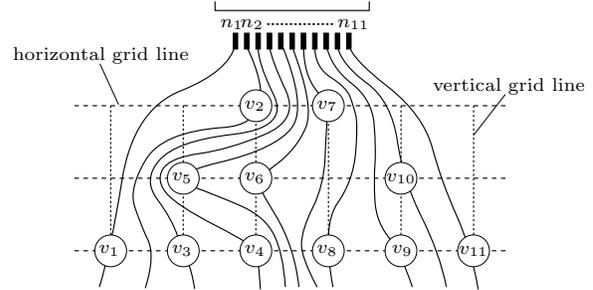


Fig. 3. A monotonic routing corresponding to a monotonic via assignment.

The routing problem for a two-layer BGA package is defined as follows:

| A routing problem for 2-layer BGA | |
|-----------------------------------|--|
| Input: | Fingers, balls, and netlist (Connection Requirements between fingers and balls) |
| Output: | A via assignment Φ , corresponding routing on layer 1, and routing on layer 2 |
| Objective: | Minimize the total wire length and the maximum wire congestion |
| Constraint: | All nets are realized, and vias are placed out of mold gates. |

B. Monotonic via assignment

If the route of each net on layer 1 from its finger to the outer ring intersects every horizontal grid line only once, then the route is said to be monotonic. Otherwise, it is said to be non-monotonic. It is clear that a monotonic routing is possible for via assignment Φ if and only if $x_i^v < x_j^v$ is satisfied for any pair of nets n_i and n_j ($i < j$) such that $y_i^v = y_j^v$. A via assignment is said to be monotonic if a monotonic routing of layer 1 is possible [5, 6].

Given a monotonic via assignment, monotonic routing on layer 1 is uniquely determined. The via assignment shown in Fig. 3 is monotonic, and its routing is unique. For example, three vias v_5 , v_6 and v_{10} are assigned on middle row in Fig. 3. The route of nets n_1, n_2, n_3 , and n_4 in monotonic routing need to pass to the left of v_5 as shown in Fig. 3.

C. Indices for Evaluation of a via assignment

In this section, indices of a via assignment which are used in the evaluation of the via assignment is explained briefly.

The number of wires on layer 1 between via v and the via above v is denoted by $cut_a(v)$. If no via exists above v , $cut_a(v)$ is zero. Details are explained in [6]. The Man-

3A-3

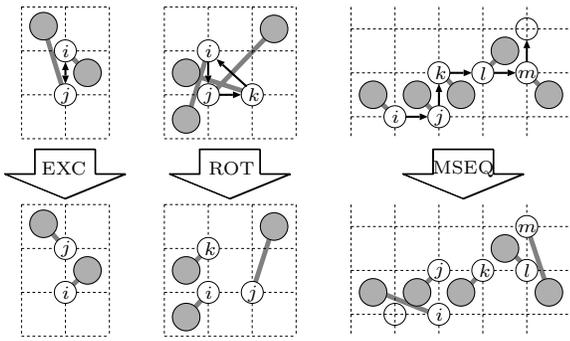


Fig. 4. Examples of three modifications.

hattan distance between via v and the ball of the net is denoted by $d(v)$.

The wire congestion of layer 1 between via v and the via to the left of v is denoted by $density_l(v)$. That is, $density_l(v)$ is the number of wires of layer 1 between them over the distance between them. If no via exists to the left of v and v is in the routing region, then $density_l(v)$ is the wire congestion of layer 1 between v and the left boundary of routing region. If no via exists to the left of v and v is within the mold gates, then $density_l(v)$ is the number of wires of layer 1 which pass to the left of v over the distance between v and the left boundary of the sector. Similarly, $density_r(v)$ is defined.

The balance of wire congestion of via v is denoted by $F(v)$. That is, $F(v) = |density_l(v) - density_r(v)|$.

The indices defined above are also used in [6], and their calculations are discussed in [6]. While, the indices defined below are mainly used to improve the completion ratio of nets which are not used in [6].

The illegality of via v is denoted by $obs(v)$. That is, if v is on a mold gate, $obs(v) = 1$. Otherwise, $obs(v) = 0$.

The violation of wire congestion of layer 1 between via v and the left of v is denoted by $viol_l(v)$. That is, $viol_l(v) = \max\{density_l(v) - C_{MAX}, 0\}$ where C_{MAX} be the allowable wire congestion of layer 1. The total violation is denoted by Δ . That is, Δ is the sum of violations on a whole via grid array.

The number of unconnected nets is denoted by U . The total wire length on layer 2 of connected nets is denoted by L . In order to evaluate U and L , the routing graph for layer 2 is defined and a rip-up and reroute method is used which are explained in Section V.

D. Modifications

There are many ways to modify a via assignment. In this paper, four simple modifications are used.

(EXC) Two adjacent vias on a vertical grid line are exchanged.

(ROT) Three vias on a unit square on the via grid array are rotated.

(MSEQ) Vias are moved to their adjacent grid nodes on a via grid array one by one until reaching a grid node without a via in which the direction of every horizontal movement of vias is either left or right and that of every vertical movement is either above or below.

(CEXC) Any two vias are exchanged.

EXC, ROT, and MSEQ have been proposed in [5] to improve the wire congestion of layer 1 while keeping the

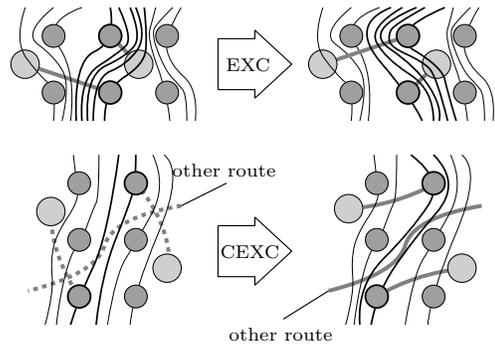


Fig. 5. The change of routes by each exchange.

wire length of layer 2 as small as possible. Examples of them are shown in Fig. 4. While, a CEXC is introduced here to improve the routability of layer 2 while keeping the global structure of routing pattern of layer 1. See Fig. 5. EXC may drastically change routes of layer 1 while keeping the distance between the via and the ball, whereas CEXC may improve the routability of layer 2 without changing routes of layer 1 drastically if the number of wires of layer 1 between vias is small.

III. OUTLINE OF OUR METHOD

In our proposed method, an initial monotonic via assignment is generated by the method proposed in [5]. Then the initial via assignment is iteratively improved. Our method consists of two phases. In the first phase, a via assignment is iteratively improved under the monotonic condition to minimize the maximum wire congestion on layer 1 while the total wire length on layer 2 is kept to be small enough. In the second phase, a via assignment is iteratively improved under the monotonic condition to improve the routability on layer 2 while the maximum wire congestion on layer 1 is maintained.

The first phase is based on the method proposed in [6]. In this phase, three types of modification EXC, ROT, and MSEQ are used. In each iteration, a modification with the maximum gain on EXCs, ROTs, and MSEQs is applied to the current via assignment to improve the total wire length and the wire congestion. Though the initial via assignment has vias placed on a obstacle, all vias are moved to routing region in this iterative modification. In [6], it takes $O(|\mathcal{N}|^2)$ time to find a MSEQ with the maximum gain. However, we show that it can be obtained in $O(|\mathcal{N}|)$. In the first phase of our proposed method, each iteration takes only $O(|\mathcal{N}|)$ time.

In the second phase, the via assignment generated by the first phase is iteratively modified by CEXC which is proposed here so that the maximum wire congestion on layer 1 is maintained and the routability on layer 2 is improved. In order to evaluate the routability on layer 2, the routing graph corresponding to a routing problem on layer 2 is introduced, and routes are generated on it. In each iteration, an allowable CEXC with the maximum gain is applied. In order to find an allowable CEXC with the maximum gain, the routing corresponding to each CEXC needs to be generated. However it is not so time consuming since the routes are changed incrementally and the most of routes are not changed even if CEXC is applied.

IV. THE FIRST PHASE

A. Cost of a via assignment

The routing cost for monotonic via assignment used in [6] is extended to move vias out of mold gate since the initial via assignment may have vias on a mold gate.

The routing cost for monotonic via assignment Φ , which is denoted by $COST_1(\Phi)$, is defined as follows:

$$COST_1(\Phi) = \sum_{v \in \mathcal{V}} (\alpha_1 cut_a(v) + \beta_1 d(v) + \gamma_1 F(v) + \delta_1 obs(v))$$

where α_1 , β_1 , γ_1 , and δ_1 are coefficients. Note that δ_1 is set to much larger than the others in order to obtain a via assignment where vias are out of mold gates.

B. Improvement of the maximum gain computation

In the first phase of our method, a modification with the maximum gain under the monotonic condition is selected and applied to the current via assignment. The number of patterns on EXCs and ROTs is $O(|\mathcal{N}|)$, which is small enough to enumerate all the patterns, while the number of patterns on MSEQs is exponential in the terms of the number of grid nodes. In order to find a MSEQ with the maximum gain in polynomial time, the cost graphs are used.

A cost graph is a directed acyclic graph (DAG), and has some sources and sinks. All sources in the graph correspond to the start vias of MSEQs, and all sinks in the graph correspond to the end dummy vias of MSEQs. Every directed path from source to sink corresponds to a MSEQ, and the length of the path corresponds to the gain on the MSEQ. The type of an MSEQ is either above-left, above-right, below-left, or below-right since the directions are restricted. A MSEQ with the maximum gain is obtained by generating cost graphs for each type and searching a longest path on the graphs.

In [6], the cost graph for every MSEQ beginning with a via is constructed and a longest path in each cost graph is obtained. A longest path of DAG can be obtained in $O(n+m)$, where n and m are the numbers of vertices and edges, respectively. Since the numbers of vertices and edges in a cost graph for each via are $O(|\mathcal{N}|)$, a longest path is obtained in $O(|\mathcal{N}|)$ for each cost graph. The maximum gain on MSEQs can be obtained in $O(|\mathcal{N}|^2)$ since the number of cost graphs is $O(|\mathcal{N}|)$.

In the following, we show that the cost graphs beginning with different vias can be combined. Since just four cost graphs are constructed where the numbers of vertices and edges are $O(|\mathcal{N}|)$, an MSEQ with the maximum gain can be obtained in $O(|\mathcal{N}|)$.

An MSEQ M is represented by a sequence of vias, where the last via is dummy.

Let $g(M)$ be the gain of an MSEQ M that is defined by $COST(\Phi) - COST(\Phi')$, where Φ' is the via assignment obtained from Φ by applying M . For any MSEQ M , $g(M)$ can be represented as the sum of local gains $g_M(v)$, where v is a via contained in M . $g_M(v)$ can be calculated if the subsequence of M which consists of four vias around v is known, as described in [5]. Namely, even if two MSEQs M_1 and M_2 begin with different vias, $g_{M_1}(v)$ and $g_{M_2}(v)$ are the same if the subsequences of M_1 and M_2 around v are the same.

Each vertex of a cost graph is labeled by the sequence of three vias. In a cost graph, a subsequence (v_{pp}, v_p, v, v_n) of M corresponds to vertex (v_{pp}, v_p, v) , vertex (v_p, v, v_n) , and the edge between them with weight $g_M(v)$, where v_n is the next via of v in M , v_p is the previous via of v in M , and v_{pp} is the previous via of v_p in M . Note that only v_n is allowed to be dummy.

The algorithm of the above-right type cost graph construction is shown in Fig. 6. In a cost graph, the number of vertices in which v is the last element of label is at most

```

ConstructCostGraph(A via assignment  $\Phi$ )
 $\mathbf{X} \leftarrow \mathcal{V} \cup \mathcal{E}$ 
 $\mathbf{C} \leftarrow$  the set of vias
    s.t. no via in  $\mathbf{X}$  exists in above-right of them
while  $\mathbf{C} \neq \emptyset$  do
  select  $v$  from  $\mathbf{C}$ 
  Let  $v_n$  be the via above or to the right of  $v$ 
  Let  $v_p$  be the via lower or to the left of  $v$ 
  Let  $v_{pp}$  be the via lower or to the left of  $v_p$ 
  if  $v$  is dummy then
    generate feasible vertices  $(\emptyset, v_p, v)$  and  $(v_{pp}, v_p, v)$ 
  else
    if  $(\emptyset, v, v_n)$  exists then
      generate  $(\emptyset, \emptyset, v)$ 
      generate an edge from  $(\emptyset, \emptyset, v)$  to  $(\emptyset, v, v_n)$ 
    if  $(v_p, v, v_n)$  exists then
      generate feasible vertices  $(\emptyset, v_p, v)$  and  $(v_{pp}, v_p, v)$ 
      generate edges from these vertices to  $(v_p, v, v_n)$ 
   $\mathbf{X} \leftarrow \mathbf{X} \setminus \{v\}$ 
   $\mathbf{C} \leftarrow$  the set of vias
    s.t. no via in  $\mathbf{X}$  exists in above-right of them
done
  
```

Fig. 6. Algorithm of graph construction for above-right direction.

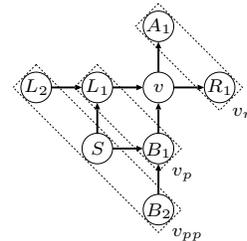


Fig. 7. Vias used to calculate $g_M(v_i)$.

seven. Note that a vertex is not generated if a via assignment becomes non-monotonic when the modification corresponding to the vertex is applied. The number of edges incident from a vertex is at most two. Therefore, the numbers of vertices and edges of a cost graph are $O(|\mathcal{N}|)$. For example, in the via assignment shown in Fig. 7, labels in which v is the last element are (L_2, L_1, v) , (S, L_1, v) , (S, B_1, v) , (B_2, B_1, v) , (\emptyset, L_1, v) , (\emptyset, B_1, v) , and $(\emptyset, \emptyset, v)$. The edges incident from (L_2, L_1, v) are (L_1, v, A_1) and (L_1, v, R_1) . Therefore, a modification with the maximum gain on MSEQs can be obtained in $O(|\mathcal{N}|)$.

V. THE SECOND PHASE

Though a via is placed near its ball in the output of the first phase, all nets can not be connected if the via assignment is bad. So, in the second phase, the via assignment generated by the first phase is iteratively modified by CEXC to improve the routability of layer 2 while the maximum wire congestion on layer 1 is maintained.

A. Evaluation of a via assignment

In the second phase, we use another cost defined here, since the target is different from the first phase. The routability improvement has priority since a near optimal via assignment is obtained by first phase.

The cost of a via assignment is defined as follows:

$$COST_2(\Phi) = \alpha_2 \Delta + \beta_2 L + \gamma_2 U$$

where α_2 , β_2 , and γ_2 are coefficients. Note that γ_2 is set

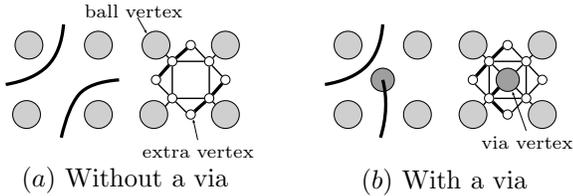


Fig. 8. Routing subgraphs on layer 2.

to much larger than the others in order to realize more nets.

The total violation Δ corresponds to the routability on layer 1, while the number of unconnected nets U corresponds to the routability on layer 2. The total wire length L on layer 2 is used to decrease U .

B. Routing graph on layer 2

The routing graph representing routing resource on layer 2 is constructed. The structure of it is changed depending on a via assignment. The routing graph has ball vertices, via vertices, and extra vertices. A ball vertex and a via vertex correspond to a ball and a via, respectively. The number of routes intersecting between two adjacent balls is at most one since ball radius is so big. A subgraph of a routing graph in Fig. 8(a) corresponds to a grid in which a ball exists in each corner and to which a via is not assigned. A subgraph of a routing graph in Fig. 8(b) corresponds to a grid to which a via is assigned.

A global routing on layer 2 is obtained by using a rip-up and reroute technique on the routing graph. In each iteration of rip-up and reroute method, a shortest path of each net is sequentially generated on the routing graph regarding the routes of the other nets as obstacles. If the route of a net can not be found, then a shortest path is generated in the graph without other routes, and the routes of the other nets which intersect the found shortest path are ripped up. Whenever a route is ripped up, the weight of each vertex on the ripped up route and on the found shortest path is increased to avoid iterations such as the routes of two nets are alternately generated and ripped up. The number of iteration of rip-up and reroute method is restricted. Therefore, there are several unconnected nets if 100% routing can not be found in prespecified times.

In order to find an allowable CEXC with the maximum gain, a global routing on layer 2 is generated for each routing graph which is obtained by applying an allowable CEXC. In the routing graph corresponding to an allowable CEXC, routes of two nets whose vias are exchanged have to be regenerated. However, most of routes apart from exchanged vias are not modified. Therefore, the execution time of a global routing on layer 2 in each iteration is not so large.

C. Local modification for routability

In the first phase, the wire congestion is minimized effectively by using EXCs, ROTs, and MSEQs. In the second phase, routes on layer 1 should not be changed drastically to maintain the quality. So, in the second phase, a local modification CEXC which has high probability on improving the routability on layer 2 though the effect on routes on layer 1 is small is used. In CEXC, two vias are exchanged, but pairs of vias to be exchanged are restricted to be allowable.

If CEXC satisfies the following three conditions, then it is said to be allowable. In each iteration, a CEXC with the maximum gain is selected among allowable CEXCs.

TABLE I
THE INITIAL COST.

| | #net | C | D | F | OBS |
|-------|------|-----|-----|--------|-----|
| data1 | 316 | 255 | 316 | 398.72 | 21 |
| data2 | 192 | 447 | 192 | 706.00 | 16 |
| data3 | 160 | 370 | 168 | 503.55 | 20 |
| data4 | 160 | 353 | 164 | 471.42 | 16 |
| data5 | 160 | 276 | 160 | 457.07 | 14 |

First, CEXC is restricted to satisfy the monotonic condition after it is applied.

Second, CEXC is restricted not to increase the maximum wire congestion on layer 1. Let c and c' be the maximum wire congestion around v_i and v_j before they are exchanged and after they are exchanged, respectively. If $c \leq C_{MAX}$, then CEXC is allowed only if $c' \leq C_{MAX}$. Otherwise, CEXC is allowed only if $c' < c$.

Third, CEXC is restricted so that the number of changed routes on layer 1 is small. Even if the distance of v_i and v_j is large, the number of changed routes on layer 1 by exchanging v_i and v_j is small if $|i - j|$ is small. Therefore, the vias which can be exchanged to v_i are restricted as follows. If v_i is not a dummy via, then v_i can be exchanged to v_j only if $i - 2 \leq j \leq i + 2$. If v_i is dummy a via, then v_i can be exchanged to the via of a net only if the route of the net passes near v_i . Formally, such nets are defined as follows: Let v_l and v_r be vias to the left and right of dummy via v_i , respectively. A net whose route passes near v_i is n_j ($k - 2 \leq j \leq k + 2$), where

$$k = \frac{l \cdot (x_r^v - x_i^v) + r \cdot (x_i^v - x_l^v)}{x_r^v - x_l^v}.$$

VI. EXPERIMENTS AND RESULTS

We implemented the proposed method in C++ language and applied it to several test cases which has mold gates as shown in Fig. 2. The number of rows of balls is 4 in all data. The program ran on a personal computer with a 3.4GHz CPU and 1 GB of memory.

In our experiment, the number of used edges on routing graph is used as the total wire length of routing on layer 2. $\alpha_1, \beta_1, \gamma_1$, and α_2 are set to 1. δ_1 and γ_2 are set to much larger than the others. β_2 is set to $\frac{1}{4}$ to balance each term, and that corresponds to the fact that the distance between adjacent two grid nodes is regarded as 1 since the distance on routing graph is 4. In addition, the iteration in each rip-up and reroute is restricted to be 20 times.

In the tables, C, D, F, and OBS are $\sum cut_a(v)$, $\sum d(v)$, $\sum F(v)$, and $\sum obs(v)$, respectively, and TOTAL is the sum of them. Δ is the violations of the wire congestion. L is the wire length and U is the number of unconnected nets for routing on layer 2. OLD is the execution time of the method proposed in [6], and PROP is the execution time of our proposed method where the cost graph is improved.

The initial cost of each data is shown in Table I.

The outputs of the first phase for most inputs are improved drastically as shown in Table II.

Although the method proposed in [6] needs 45 second for data1, our proposed method obtains the identical output within 3 second by improving the computational complexity of each iteration.

Table III shows the result of the second phase. Though TOTAL of the second phase increases in comparison to that of the first phase, the routability is improved. In this experiment, all nets are realized in data2, data3, data4, and data5, and L gets lower than the output of the first phase in all data.

The total violations on layer 1 are reduced by the second phase. The result with violations can not be used

TABLE II
THE RESULT IN THE FIRST PHASE.

| | $COST_1$ | | | | $COST_2$ | | | #Modification | | | | Execution time [sec] | | |
|-------|----------|-----|--------|-----|----------|----------|-----|---------------|-----|-----|------|----------------------|-------|------|
| | C | D | F | OBS | TOTAL | Δ | L | U | EXC | ROT | MSEQ | ALL | OLD | PROP |
| data1 | 118 | 412 | 233.98 | 0 | 763.98 | 2.80 | 885 | 11 | 11 | 0 | 48 | 59 | 44.96 | 2.58 |
| data2 | 147 | 284 | 349.67 | 0 | 780.67 | 47.6 | 709 | 3 | 20 | 5 | 32 | 57 | 8.37 | 0.86 |
| data3 | 81 | 255 | 165.52 | 0 | 501.52 | 7.28 | 663 | 5 | 16 | 5 | 57 | 78 | 10.58 | 1.29 |
| data4 | 63 | 241 | 171.40 | 0 | 475.40 | 11.5 | 597 | 3 | 21 | 2 | 45 | 68 | 9.57 | 1.09 |
| data5 | 68 | 216 | 153.45 | 0 | 437.45 | 7.14 | 565 | 0 | 12 | 2 | 41 | 55 | 8.33 | 0.85 |

TABLE III
THE RESULT IN THE SECOND PHASE.

| | $COST_1$ | | | | $COST_2$ | | | #Modification CEXC | Execution time [sec] | |
|-------|----------|-----|--------|-----|----------------|----------|-----|-----------------------|----------------------|-------|
| | C | D | F | OBS | TOTAL (%) | Δ | L | | | U |
| data1 | 149 | 404 | 285.98 | 0 | 838.98 (109.8) | 2.80 | 893 | 5 | 20 | 61.32 |
| data2 | 173 | 275 | 385.24 | 0 | 833.24 (106.7) | 40.0 | 659 | 0 | 12 | 3.33 |
| data3 | 110 | 243 | 216.12 | 0 | 569.12 (113.4) | 5.27 | 602 | 0 | 21 | 4.31 |
| data4 | 77 | 235 | 241.20 | 0 | 553.20 (116.3) | 11.5 | 575 | 0 | 9 | 2.31 |
| data5 | 76 | 205 | 215.24 | 0 | 496.24 (113.4) | 2.78 | 458 | 0 | 11 | 0.83 |

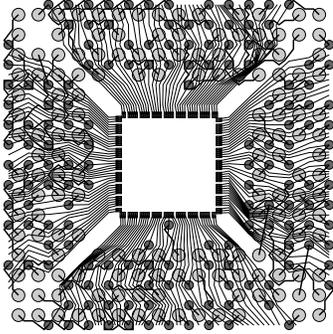


Fig. 9. The output routes of the first phase for data4.

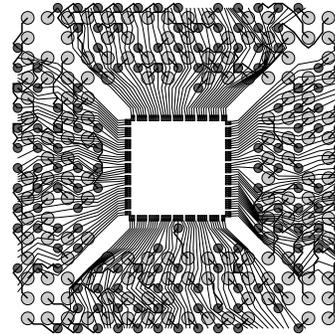


Fig. 10. The output routes of the second phase for data4.

as it is. However, even though the violations still exist, the result with few violations might be acceptable in design scene. A few violations would be eliminated easily by manual modifications and/or neglected by allowing to use a few narrow wire segments for non-critical parts and signals.

The output of the first phase for data4 is shown in Fig. 9, and the output of the second phase is shown in Fig. 10. Mold gates are not drawn in these figures. In the second phase, routability is improved without changing the structure of global routing of layer 1 drastically.

VII. CONCLUSION

We showed that a modification with the maximum gain is obtained in $O(|N|)$, though it takes $O(|N|^2)$ times in [6]. Moreover, we gave a routing graph for routing on layer 2, and a local modification to improve the routability on layer 2 while maintaining the maximum wire congestion.

In our experiments, our method obtains a via assignment which distributes wires evenly faster than the method proposed in [6], and the routability on layer 2 is improved drastically. Our proposed method explores monotonic via assignments effectively, and a via assignment which guarantees 100% routability on layer 2 is obtained in most of test cases.

On the other hand, our method does not realize all net in data1 where the output of the first phase has many unconnected nets. In addition, though most wires are distributed evenly, there exist places with high wire congestion near mold gates. This is caused since moving vias out of mold gates has priority over improving or maintaining the wire congestion and the distance between a via and a ball in the first phase. These bad effects will be relaxed if the initial via assignment in which vias are placed out of a mold gate is created with the routability analysis, and the wire congestion on layer 1 can be improved if a part of plating leads is realized on layer 2. But,

these are in our future works. Moreover, we will consider the method where parts of plating leads are realized on layer 2.

ACKNOWLEDGMENTS

This research was partially supported by Grant-in-Aid for Scientific Research (C) (18500034) and Grant-in-Aid for JSPS Fellows (19009340).

REFERENCES

- [1] M.-F. Yu and W. W.-M. Dai, "Single-Layer Fanout Routing and Routability Analysis for Ball Grid Arrays," in *Proceedings of International Conference Computer-Aided Design*, pp. 581–586, 1995.
- [2] S. Shibata, K. Ukai, N. Togawa, M. Sato, and T. Ohtsuki, "A BGA Package Routing Algorithm on Sketch Layout System," *The journal of Japan Institute for Interconnecting and Packaging Electronic Circuits*, vol. 12, no. 4, pp. 241–246, 1997. (In Japanese).
- [3] C.-C. Tsai, C.-M. Wang, and S.-J. Chen, "NEWS: A Net-Even-Wiring System for the Routing on a Multilayer PGA Package," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 17, no. 2, pp. 182–189, 1998.
- [4] S.-S. Chen, J.-J. Chen, C.-C. Tsai, and S.-J. Chen, "An Even Wiring Approach to the Ball Grid Array Package Routing," in *Proceedings of International Conference on Computer Design*, pp. 303–306, 1999.
- [5] Y. Kubo and A. Takahashi, "A Via Assignment and Global Routing Method for 2-Layer Ball Grid Array Packages," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E88-A, no. 5, pp. 1283–1289, 2005.
- [6] Y. Kubo and A. Takahashi, "Global Routing by Iterative Improvements for 2-Layer Ball Grid Array Packages," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 25, no. 4, pp. 725–733, 2006.