論文 / 著書情報
Article / Book Information

Title	Routability Driven Via Assignment Method for 2-Layer Ball Grid Array Packages
Authors	Yoichi Tomioka, Atsushi Takahashi
Citation	IEICE Trans. Fundamentals, Vol. E92-A, No. 6, pp. 1433-1441
Pub. date	2009, 6
URL	http://search.ieice.org/
Copyright	(c) 2009 Institute of Electronics, Information and Communication Engineers

PAPER

Routability Driven Via Assignment Method for 2-Layer Ball Grid Array Packages*

Yoichi TOMIOKA†a), *Nonmember and* **Atsushi TAKAHASHI**†b), *Member*

SUMMARY 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 that iteratively modifies via assignment. In experiments, in most cases, via assignment and global routing on both of layers in which all nets are realized and the violation of wire congestion on layer 1 is small are speedily obtained.

key words: *ball grid array, monotonic, package routing, via assignment*

1. Introduction

In current VLSI circuits, Ball Grid Array (BGA) packages are used to realize a number of connections between VLSI chips and printed circuit boards (PCBs). BGA packages has some routing layers, and wires on different layers are connected by vias. Since the quality of routing of each layer strongly depends on the placement of vias and the assignment of vias to nets, the via planning that determines the placement and assignment of vias is important to get a satisfactory routing. Manual routing including the via planning needs much time, and automation of BGA package routing is strongly required.

There are several works related to BGA package routing. Routing methods for problems in which terminals are placed in single-row and double row have been proposed in [2] and [3], respectively. Routing methods for flip chips using network flow algorithm and integer linear programming have been proposed in [4] and [5], respectively. The first routing method to minimize the maximum wire congestion in single-layer BGA package has been proposed in [6], and improved in [7]. These methods are for planar routing, and via planning is not included. While, a routing method for multi-layer BGA packages that uses a single-row routing method has been proposed in [8]. However, the method does not consider via planning.

A via assignment and global routing method for single-chip two-layer BGA packages has been proposed in [9], and this method has been improved in [10]. To the best of our knowledge, this is the only work considering via planning for BGA. The method proposed in [10] achieves a small total wire length and congestion on layer 1 while keeping the distances between vias and balls small instead of generating global routing on layer 2. However, there are several problems in order to guarantee 100% routing in actual package routing design.

First, global routing on layer 2 is not generated in the method. Even if the evaluation of a via assignment is better, it can not be adopted if 100% routing on layer 2 is impossible. Second, the method does not handle various kinds of obstacles in the routing region. For example, mold gates are placed on layer 1. In the region at which a mold gate is placed, routing on layer 1 is not allowed but routing on layer 2 is allowed. Therefore, all vias must be out of mold gate to guarantee 100% routing of layer 1. In addition, mold gates make it difficult to generate 100% routing on layer 2 since the via of a net may be placed away from its ball if the ball is under a mold gate.

In this paper, we propose a via assignment and global routing method under the existence of mold gates to realize 100% routing on both layers. Our method is an enhancement of the method proposed in [10], and consists of two phases. The first phase and the second phase are mainly for improving the routability of layer 1 and layer 2, respectively. The first phase is iterative via assignment modification based on the method proposed in [10]. However, the computational complexity of each iteration is improved, and vias on mold gates in an initial via assignment expected to be moved out of the mold gates through the iteration. The second phase is iterative modification to improve the completion ratio of nets on layer 2 and to reduce the amount of violations on layer 1. In this phase, a global routing on layer 2 is generated to guarantee 100% routing on layer 2.

In experiments, our method is applied to six data, and compared to a method extended to handle mold gates from the method in [10]. The first phase of our proposed method obtains the via assignments obtained by the extended method ten times faster, and vias are placed in the routing region in the obtained via assignments. In the second phase, the completion ratio of nets on layer 2 is drastically improved, and the total violation of layer 1 is decreased by 44.7% on average from the first phase. In most cases, our method obtains a via assignment and a routing pattern within several minutes, where all nets are realized and the violation of wire congestion on layer 1 is small.

Manuscript received December 15, 2008.

†The authors are with the Department of Communications and Integrated Systems, Tokyo Institute of Technology, Tokyo, 152-8550 Japan.

*The preliminary vesion was presented at [1].

a) E-mail: yoichi@lab.ss.titech.ac.jp

b) E-mail: atsushi@lab.ss.titech.ac.jp

DOI: 10.1587/transfun.E92.A.1433

2. Preliminary

2.1 Problem Definition

In this paper, we consider a basic model of BGA package that contains a single chip which is smaller than the package size and a package substrate with two routing layers as shown in Fig. 1. On the package substrate, bonding fingers, solder balls, and mold gates are placed.

A bonding finger, which is referred to as a finger, is placed on the perimeter of a rectangle enclosing the chip on layer 1, and is connected to the chip by a bonding wire. A solder ball, which is referred to as a ball, is an I/O terminal of the package, and is placed in a grid array pattern on layer 2. A mold gate which is used to pour resin into the package is placed in a corner of the substrate on layer 1.

In package routing design, a connection requirement is given as a net which consists of fingers and balls. In this paper, we assume that a net consists of one finger and one ball. A net is realized by using wires on each layer and vias which connect wires on different layers. However, there are several special constraints. In this paper, two types of constraints are considered. The first one is related to the mold gates. In the region at which a mold gate is placed, a wire on layer 1 is not allowed, but a wire on layer 2 is allowed. The second one is related to the electric plating to protect the wire. In order to enable electric plating, a net is requested to extend its wire to the substrate boundary on either layer.

The package substrate is divided into four sectors and the nets are divided accordingly. In the following, we focus on the bottom sector as shown in Fig. 2. Nets are labeled according to the order of fingers on the perimeter from the left to the right as $1, 2, 3, \ldots, N$ where N is the number of nets.

A grid array pattern of balls is modeled by a ball grid array. The interval of balls in the ball grid array almost decides the package size since a ball is large compared to a wire and a via. In order to get a smaller package, the interval is set to be small as long as the connection requirements are satisfied. In our model, the interval is set so that one wire can go through between adjacent balls on layer 2. It is the minimum except trivial cases. While, the number of possible wires going through the interval is larger than one if there is no obstacle.

Let the unit length of our coordinate system be the interval of balls in the ball grid array. According to the ball grid array, a sector is divided into unit squares called cells, and the set of cells is denoted by C. A cell corresponds to the area surrounded by four adjacent balls. Let c and c' be any distinct cells in C, and let (x_c, y_c) and (x'_c, y'_c) be the positions of the center points of cells c and c', respectively. c is said to be adjacent to c' if $|x_c - x'_c| + |y_c - y'_c| = 1$.

The candidate positions of vias are restricted since the size of a via is relatively large even though it is smaller than the size of a ball. In our model, the number of vias to be placed in a cell is at most one, and the via is placed at the center of a cell.

Vias are not allowed under a mold gate since the wire on layer 1 is not allowed. A cell in which a via can not be placed is called a mold cell. In a feasible solution, a via is not placed in a mold cell. However, in our method, a via is assigned to a mold cell in an intermediate solution to get a better feasible final solution.

Since the routing on layer 2 is tight, we restrict the structure of each net so that it has just one via, and that its plating lead is routed on layer 1. Let b_i and v_i be the ball and the via of net i, respectively. Let (x_i^b, y_i^b) and (x_i^v, y_i^v) be the positions of b_i and v_i, respectively. Note that (x_i^b, y_i^b) is an input and (x_i^v, y_i^v) is an output.

A via assignment is an assignment of vias into cells including mold cells. In our method, a dummy via is introduced so that the number of vias including dummy is equal to the number of cells, that is, $|C|$. Let $\mathcal{V}$ be the set of vias of nets, $\mathcal{E}$ be the set of dummy vias. Note that $|\mathcal{V}| = N$ and $|\mathcal{V}| + |\mathcal{E}| = |C|$, and that $|C|$ can be regarded as $O(N)$ since the number of cells is close to the number of nets in BGA package routing. In the following, a via assignment is represented by bijection $\Phi : \mathcal{V} \cup \mathcal{E} \rightarrow C$.

A solution is represented by a via assignment Φ and a routing pattern of layer 2. A routing pattern of layer 1 is obtained from the via assignment by restricting the via assignment and a routing pattern to be monotonic. A monotonic via assignment and the corresponding monotonic routing pattern are explained in Sect. 2.2. A routing pattern of layer 2 is represented by a routing graph which is explained in Sect. 5.2.

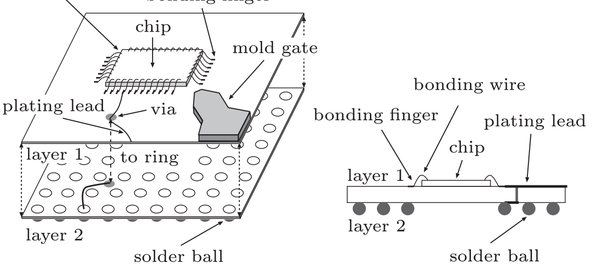


Fig. 1 A model of 2-layer BGA package.

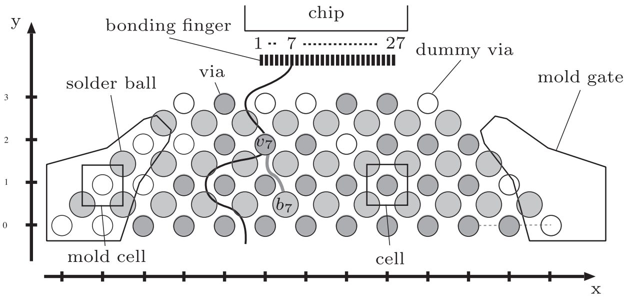


Fig. 2 Bottom sector.

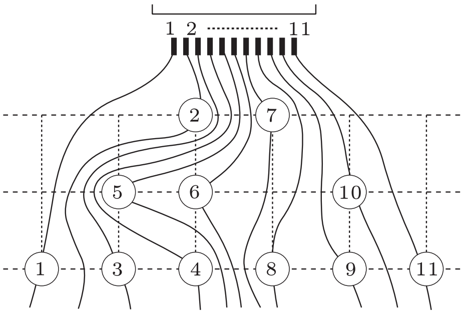


Fig. 3 A monotonic routing on layer 1 corresponding to a monotonic via assignment.

2.2 Monotonic Via Assignment

If the route of each net on layer 1 from its finger to the package boundary intersects any horizontal line at most 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 $x^v_i < x^v_j$ is satisfied for any pair of nets i and j ($i < j$) such that $y^v_i = y^v_j$. A via assignment is said to be *monotonic* if a monotonic routing of layer 1 is possible [9], [10].

Given a monotonic via assignment, monotonic routing on layer 1 is uniquely determined. For example, the via assignment shown in Fig. 3 is monotonic. Three vias v_5, v_6 and v_{10} are assigned on the middle row in Fig. 3. As shown in Fig. 3, the routes of nets 1, 2, 3, and 4 in monotonic routing need to pass to the left of v_5, and the routes of nets 7, 8, and 9 need to pass between v_6 and v_{10}.

2.3 Indices for Evaluation of a Via Assignment

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

Let v_i and v_j be any distinct vias in $\mathcal{V}$. Via v_j is said to be to the left of via v_i in Φ if $\Phi(v_j)$ is to the left of $\Phi(v_i)$ on the same row, and cells between $\Phi(v_i)$ and $\Phi(v_j)$ have no vias except v_i and v_j. Similarly, vias above, below, and to the right of v_i are defined. Moreover, via v_i is said to be adjacent to via v_j if v_j is above, below, to the left of, or to the right of v_i.

If v is the leftmost via on a row, then the left interval of v is the interval between v and the left boundary of the routing region of the row. Otherwise, the left interval of v is the interval between v and the via to the left of v. Similarly, the right, upper, and lower intervals of v are defined. The set of left, right, upper and lower intervals of all vias is denoted by $\mathcal{I}$.

The number of wires on layer 1 which goes through the upper interval of via v is denoted by $cut_a(v)$. Details are explained in [10]. It is used to estimate the length of routing on layer 1. The Manhattan distance between via v_i and ball b_i is denoted by $d(v_i)$. It is used to estimate the length of routing on layer 2.

The wire congestion of an interval I in $\mathcal{I}$ is the number of wires going through the interval over the number of possible wires going through the interval, and is denoted by $density(I)$. In our method, some via may be assigned to mold cells in an intermediate solution. The definition of the wire congestion of an interval related to a via assigned to a mold cell is modified to estimate the wire congestion in latter solutions in earlier stage. In the definition of the wire congestion of the left interval of via v, if v is the leftmost via on a row and assigned to a mold cell, then the number of possible wire is defined as if v is assigned to the leftmost cell on the row in the routing region. If v is not the leftmost via on a row and the via to the left of v is assigned to a mold cell, then the number of possible wire obtained by ignoring the existence of the mold cell is used. The wire congestion of the right interval of v is similarly defined.

The difference between the wire congestion of the left and right intervals of v is denoted by $F(v)$. That is, $F(v) = |density_\ell(v) - density_r(v)|$.

The indices defined above are also used in [10], and their calculations are discussed in [10], while the indices defined below are mainly used to improve the completion ratio of nets on layer 2 which are not used in [10].

Whether via v is assigned to a mold cell or not is denoted by $mold(v)$. That is, if v is assigned to a mold cell, $mold(v) = 1$. Otherwise, $mold(v) = 0$.

The amount of violation of an interval I in $\mathcal{I}$ is denoted by $vio(I)$. That is, $vio(I) = \max\{density(I) - 1, 0\}$. The total violation is denoted by Δ. That is, Δ is the sum of violations of intervals in $\mathcal{I}$.

In order to evaluate the wire length and the completion ratio of nets on layer 2, a routing graph is defined. A routing pattern on layer 2 is generated on the graph by using the rip-up and reroute technique which is explained in Sect. 5. The number of unconnected nets and the total wire length in the routing pattern on layer 2 obtained from the graph are denoted by U and L, respectively.

2.4 Modifications

There are many ways to modify a via assignment. Since every via assignment is bijection, a modification of a via assignment is a permutation and is represented as the product of rotations. Therefore, rotations are used as basic modification operations in our methods.

A rotation is defined by using some distinct vias. A rotation defined by n distinct vias is called n-rotation ($n \geq 2$). Let $R = (u_0, u_1, \ldots, u_{n-1})$ be a n-rotation where $u_0, u_1, \ldots, u_{n-1}$ are n distinct vias ($n \geq 2$). In the via assignment Φ′ obtained from a via assignment Φ by applying rotation R, via u_i is assigned to the cell to which via u_{i+1} is assigned in Φ (modulo n). In Fig. 4, 4-rotation (v_i, v_j, v_k, v_l) is shown.

The range of a rotation is defined as the range of the net numbers of vias in the rotation. That is, the range of rotation R is $p - q + 1$ where $p = \max\{i \mid v_i \in R\}$ and $q = \min\{i \mid v_i \in R\}$. When R is applied to Φ under the

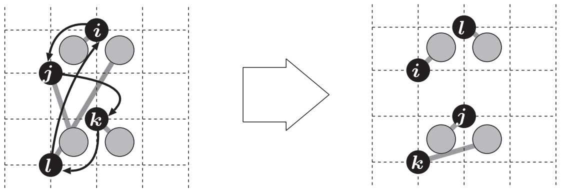
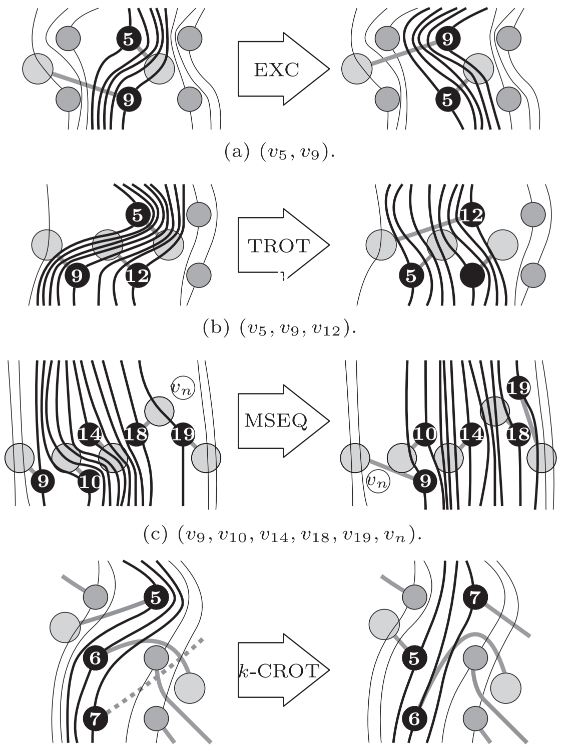


Fig. 4 4-rotation (v_i, v_j, v_k, v_l).



(a) (v_5, v_9).

(b) (v_5, v_9, v_{12}).

(c) $(v_9, v_{10}, v_{14}, v_{18}, v_{19}, v_n)$.

(d) (v_5, v_6, v_7).

Fig. 5 Four kinds of rotations.

monotonic condition, in layer 1, the routes of all nets except nets whose net numbers are between p and q are kept. The range of R is equal to the number of routes changed on layer 1 by applying R.

In our method, rotations which have higher potential to improve the current via assignment are focused on. In the following, rotations used in our method are explained. Let Φ be the current via assignment.

A 2-rotation $R = (u_0, u_1)$ is said to be exchange (EXC) for Φ if cells $\Phi(u_0)$ and $\Phi(u_1)$ are adjacent vertically.

A 3-rotation $R = (u_0, u_1, u_2)$ is said to be triangle rotation (TROT) for Φ if $\Phi(u_0)$, $\Phi(u_1)$ and $\Phi(u_2)$ are three cells among a 2×2 cells.

An n-rotation $R = (u_0, u_1, \dots, u_{n-1})$ $(n \geq 2)$ where $u_0, u_1, \dots, u_{n-2}$ are not dummy and u_{n-1} is dummy is said to be above-right monotonic sequence (MSEQ) for Φ if cells $\Phi(u_i)$ and $\Phi(u_{i+1})$ are adjacent to each other and $\Phi(u_{i+1})$ is above or right of $\Phi(u_i)$ $(0 \leq i \leq n-2)$. Above-left, below-left, and below-right MSEQs are similarly defined.

An n-rotation $R = (u_0, u_1, \dots, u_{n-1})$ $(n \geq 2)$ is said to be k-coast rotation (k-CROT) for Φ if the range of R is at most k.

Examples of EXC, TROT, MSEQ, and k-CROT $(k \geq 3)$ are shown in Fig. 5.

EXC, TROT, and MSEQ have been introduced in [9]. In the rotations in EXC, TROT, and MSEQ, the change of a via location is restricted to be small. If the distance between vias and balls are small enough in a via assignment, then it is expected that the routability on layer 2 is kept well after the rotations are iteratively applied to the via assignment. While, routes on layer 1 can be drastically changed.

In k-CROTs, the change of routes on layer 1 is small if k is small. If the total wire length on layer 1 is small enough in a via assignment, then it is expected that the total wire length is kept small after k-CROTs are iteratively applied to the via assignment. While the change of via location can be drastic and the routability on layer 2 is expected to be improved. In our experiment, 3-CROTs are used.

3. Outline of Our Method

In our proposed method, via assignment and routing on layer 1 is restricted to be monotonic. First, an initial monotonic via assignment is generated by the method proposed in [9]. Then the initial via assignment is iteratively improved.

Our method consists of two phases which have different objectives. In the first phase, a via assignment is iteratively improved under the monotonic condition to minimize the maximum wire congestion and the total wire length 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 and the total wire length on layer 1 is kept small enough.

The first phase is based on the method proposed in [10]. In this phase, three types of rotations MSEQ, EXC, and TROT are used. In each iteration, a rotation with the maximum gain among MSEQs, EXCs, and TROTs that satisfies the monotonic condition is applied to the current via assignment. Though the initial via assignment may have some vias assigned to mold cells, the via assignment is improved so that the vias are moved to routing region in this iterative modification. In [10], it takes $O(N^2)$ time to find an MSEQ with the maximum gain. However, we show that it can be obtained in $O(N)$, and each iteration takes only $O(N)$ time in the first phase.

In the second phase, the via assignment generated by the first phase is iteratively modified by k-CROTs. In each iteration, a rotation with the maximum gain in k-CROTs is applied. In order to evaluate the completion ratio of nets and the total wire length on layer 2, the routing graph corresponding to a routing problem on layer 2 is introduced, and routes are generated on it.

4. The First Phase

This phase is iterative via modification by MSEQs, EXCs, and TROTs. In each iteration, a rotation with the maximum gain among them that satisfies the monotonic condition is applied.

Our cost function of this phase and the maximum gain computation are explained in 4.1 and 4.2, respectively.

4.1 Cost of a Via Assignment

The routing cost for monotonic via assignment used in [10] is extended to move vias out of mold gate since some vias may be assigned to mold cells in the initial via assignment.

The routing cost of the first phase for monotonic via assignment Φ is denoted by $COST_1(\Phi)$, and defined as follows:

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

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

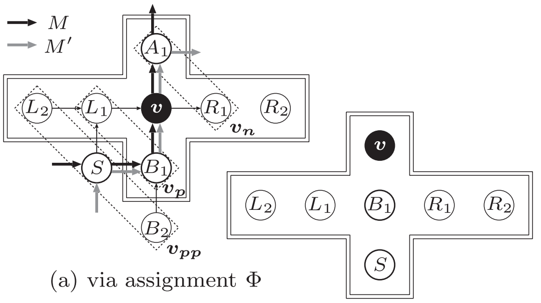
$cut_a(v)$, $F(v)$, $d(v)$, and $mold(v)$ are defined in Sect. 2.3. $cut_a(v)$ and $F(v)$ are the number of wires on layer 1 between via v and the via above v and the balance of wire congestion, respectively. They are used to improve the routing on layer 1. $d(v)$ is the Manhattan distance between via v and the ball of the net, and it is used to keep the routability on layer 2. $mold(v)$ is whether via v is assigned to a mold cell or not, and it is used to move vias out of mold gates.

4.2 Improvement of the Maximum Gain Computation

The number of patterns on EXCs and TROTs is $O(N)$, 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 cells. In order to find an MSEQ with the maximum gain in polynomial time, cost-graphs are used in [10].

The type of an MSEQ is either above-left, above-right, below-left, or below-right since the directions are restricted. If an MSEQ with the maximum gain of each type is obtained, then an MSEQ with the maximum gain among all MSEQs is obtained. In the following, we focus on above-right MSEQs.

An MSEQ is a n-rotation where the last via of the rotation is dummy, and it is represented by a sequence $(u_0, u_1, u_2, \ldots, u_{n-1})$. In [9], they show that all MSEQs beginning with a via can be represented by a directed acyclic graph with $O(N)$ vertices and $O(N)$ edges. The directed acyclic graph is called a cost-graph, and a cost-graph corresponding to all MSEQs beginning with via v is denoted by $G(v)$. $G(v)$ has one source and some sinks corresponding the start via v and the last vias in MSEQs, respectively. A path from the source to a sink corresponds to an MSEQ, and the length of the path corresponds to the gain of the MSEQ. Since a longest path of a directed graph H can be obtained in $O(|V(H)| + |E(H)|)$, it takes $O(N)$ to obtain an MSEQ with the maximum gain among MSEQs beginning with v. Moreover, it takes $O(N^2)$ to obtain an MSEQ with the maximum gain among all MSEQs since there are $O(N)$ cost-graphs. The detail is described in [9].



(a) via assignment Φ

(b) via assignment Φ'

Fig. 6 Vias used to calculate $g_M(v)$.

In this paper, we show that all MSEQs can be represented by one cost-graph with $O(N)$ vertices and $O(N)$ edges, and that an MSEQ with the maximum gain can be obtained in $O(N)$.

Let M be an MSEQ in a via assignment Φ. Let $g(M)$ be the gain of an MSEQ M that is defined by $COST_1(\Phi) - COST_1(\Phi')$, where Φ' is the via assignment obtained from Φ by applying M. Parts of via assignments Φ and Φ' are shown in Figs. 6(a) and (b), respectively. In [9], it is shown that 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 by via assignments in the indicated regions of Φ and Φ'. Therefore, if the subsequence of M which consists of four vias around v is known, then $g_M(v)$ can be calculated since via assignment in the indicated region of Φ' is obtained.

Even if two MSEQs M and M' begin with different vias, $g_M(v)$ and $g_{M'}(v)$ are same if the subsequences of M and M' around v are the same. Due to the fact, we can combine cost-graphs $G(v_1), G(v_2), \ldots, G(v_N)$ into one cost-graph which is denoted by G.

In the cost-graph G, there are some sources and sinks, and each path from a source to a sink corresponds to an MSEQ. Each vertex is labeled by the sequence of three vias. Let v_n, v_p, and v_{pp} be the next via of v in M, the previous via of v in M, and the previous via of v_p in M, respectively. 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 $g_M(v)$. Note that only v_n is allowed to be dummy, and that a vertex (v_{pp}, v_p, v) is not generated if via assignment obtained from the current via assignment by replacing v to a dummy via and by applying rotation (v_{pp}, v_p, v) is non-monotonic.

The number of vertices in which v is the last element of label is at most seven. The number of edges incident from a vertex is at most two. Therefore, the numbers of vertices

```
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 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 $(\emptyset, v, v_n)$
      endif
      if $(v_p, v, v_n)$ exists then
        generate vertices $(\emptyset, v_p, v)$ and $(v_{pp}, v_p, v)$
        generate edges from these vertices to $(v_p, v, v_n)$
      endif
    endif
    $\mathbf{X} \leftarrow \mathbf{X} \backslash \{v\}$
    $\mathbf{C} \leftarrow$ the set of vias
                s.t. no via in $\mathbf{X}$ exists in above-right of them
  done
```

Fig. 7 Algorithm of graph construction for above-right direction.

and edges of cost-graph G are $O(N)$. For example, in the via assignment shown in Fig. 6(a), 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). Moreover, G is a directed acyclic graph since all vias except the last dummy via in an above-right type MSEQ moved to the right or above. Therefore, the maximum gain among all MSEQs can be obtained in $O(N)$.

The algorithm of the above-right type cost-graph construction is shown in Fig. 7. In the algorithm, the cost-graph is constructed from sinks to sources.

Since a rotation with the maximum gain among MSEQs, EXCs, and TROTs is obtained in $O(N)$, each iteration takes only $O(N)$ in the first phase.

5. The Second Phase

Though a via is placed near its ball in the via assignment obtained from the first phase, all nets can not be connected on layer 2 if the via assignment is bad. In the second phase, the via assignment generated by the first phase is iteratively modified by k-CROTs to improve the completion ratio of nets on layer 2 while the maximum wire congestion on layer 1 is maintained.

5.1 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 on layer 2 has priority since a near optimal via assignment for layer 1 is obtained by the first phase.

The routing cost of the second phase for monotonic via assignment Φ is denoted by $COST_2(\Phi)$, and 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 to much large than the others in order to realize more nets in layer 2.

Δ is the total violation of the wire congestion on layer 1 which is defined in Sect. 2.3, and is mainly used to improve the routing on layer 1. U and L are the total wire length on layer 2 and the completion ratio of nets on layer 2, respectively. U and L are mainly used to improve the routing on layer 2.

5.2 Routing Graph on Layer 2

A 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 and via vertices. A ball vertex and a via vertex correspond to a ball and a via, respectively. In our model, the number of routes intersecting between two adjacent balls is at most one. Two routes can pass through a cell as shown in Fig. 8(a) if a via is not assigned to the cell, while enough space for two routes does not exist around via as shown in Fig. 8(b) if a via exists in it. Extra vertices are introduced in the routing graph to generate routes to satisfy the constraints. A subgraph of a routing graph is shown in Fig. 8(c).

Each vertex has a weight. The weight of a path is the sum of the weight of vertices on the path. A shortest path of net i is a path that connects b_i and v_i with the minimum weight.

A global routing on layer 2 is obtained by using a rip-up and reroute technique on the routing graph. The weight of each vertex is initially set to one. In each iteration of a 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 length of a route on the routing graph is the number of edges on the route. In actual routing design, most of routes do not detour, and the detours are so small even if they do. Therefore, the length of a route of each net is restricted in order to improve the searching efficiency. In our implementation, a shortest path among paths whose length is at most the predefined upper bound is selected.

In addition, the number of iterations of rip-up and reroute is restricted. Therefore, there are cases that some unconnected nets still exist after rip-up and reroute is terminated.

In order to evaluate a k-CROT, a global routing on layer 2 is generated for each routing graph which is obtained

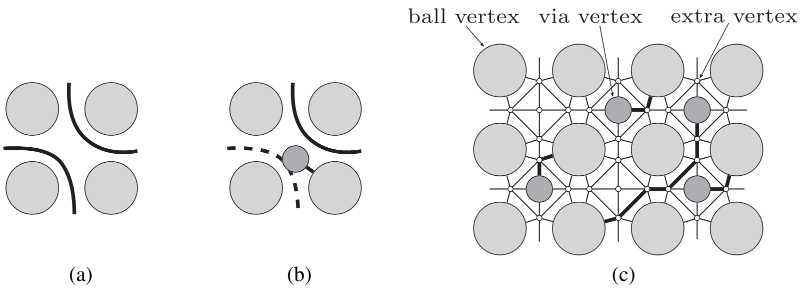


Fig. 8 Feasible routing patterns and routing sub graph.

by applying a k-CROT. In the routing graph corresponding to a k-CROT, routes of several nets whose vias are rotated need to be regenerated. However, most of routes apart from rotated vias are not modified. Therefore, the execution time to generate routing on layer 2 for each k-CROT is not so large.

5.3 Modifications for the Routability

In the first phase, the wire congestion is minimized effectively by using EXCs, TROTs, and MSEQs. In the second phase, routes on layer 1 should not be changed drastically to maintain the quality, and k-CROTs are used.

In order to evaluate a k-CROT with dummy vias, the net number of dummy via need to be determined. The net number of a dummy via is set to the net number of a net whose wire passes nearest the via. That is, the net number of dummy via v is

$$\left\lfloor \frac{l \cdot (x_r^v - x) + r \cdot (x - x_l^v)}{x_r^v - x_l^v} \right\rfloor .$$

where x is the x-coordinate of v, and v_l and v_r are the nearest non-dummy vias to the left and right of v, respectively.

In each iteration, a rotation with the maximum gain among k-CROTs which satisfies the following two constraints is applied.

First, a k-CROT is restricted not to increase the maximum wire congestion on layer 1. Let c and c' be the maximum wire congestion on layer 1 before and after applying a k-CROT, respectively. If $c \leq 1$, then k-CROT is allowed only if $c' \leq 1$. Otherwise, k-CROT is allowed only if $c' \leq c$.

Second, a k-CROT is restricted to be n-rotation ($n \leq k$). In other words, at most k vias are rotated at the same time.

The number of k-CROTs including specific via can be regarded as a constant number. Therefore, the number of k-CROTs is $O(N)$.

6. Experiments and Results

We implemented the proposed method in C++ language. The method was applied to six test cases which are artificially generated so that they are similar to actual routing problems. Especially, netlists from data1 to data5 are identical to those of [10], and mold gates whose sizes are proportional to the package size are added on the corners. The number of rows of balls at each sector is 4 in all data. The program ran on a personal computer with a 2.66 GHz Quad CPU and 4 GB of memory.

Table 1 The initial cost.

DATA	#NET	COST1					COST2		
		C	F	D	MOLD	TOTAL	Δ	U	L
data0	400	957	213.7	424	51	2235.9	73.3	0	872
data1	316	193	93.4	316	24	882.6	7.0	0	632
data2	192	447	150.5	192	16	1241.0	69.4	0	384
data3	160	282	96.9	160	14	829.5	25.7	0	320
data4	160	368	109.4	167	16	972.6	45.5	0	343
data5	160	393	104.1	171	21	980.3	32.9	4	345

In our experiments, the number of used edges on routing graph is used as the total wire length of routing on layer 2. α_1 and β_1 are set to 1, and γ_1 and α_2 are set to 4. β_2 is set to $\frac{1}{3}$ and that corresponds to the fact that the distance between two vias assigned to adjacent cells is regarded as 1 since the distance on routing graph is 3. δ_1 and γ_2 are set to much larger than the others. The upper bound of the length of a route of net i on the routing graph is set to $3(d(v_i) + 2)$, and the iteration in each rip-up and reroute is restricted to be 20 times.

In the tables, C, D, F, and MOLD are $\sum cut_a(v)$, $\sum d(v)$, $\sum F(v)$, and $\sum mold(v)$, respectively, and TOTAL is $COST_1$ except MOLD. Δ is the total violations of the wire congestion on layer 1. L is the wire length and U is the number of unconnected nets for routing on layer 2.

The initial cost of each data is shown in Table 1.

The results of first phase are shown in Table 2. The last three columns show the execution time of the method proposed in [10], that of our proposed method, and the improvement, respectively. In ours, the via assignment which is identical to that of [10] is obtained about ten times faster than the method in [10].

Table 3 shows the results of the second phase by 3-CROTs. $N(k)$ is the number of applied rotations whose range is k. C_{MAX} and #VIO are the max congestion on layer 1 and the number of violated intervals in $\mathcal{I}$, respectively. WIRE LENG is the total wire length of global routes on both layers. In the case that the ball and the via of a net are not connected, the length of the route on layer 2 of the net is evaluated by the euclidean distance between the vias and the balls.

The output of the first phase for data5 is shown in Fig. 9, and the output of the second phase is shown in Fig. 10. Unconnected nets are represented by thick dotted

Table 2 The results of first phase.

DATA	WIRE	$COST_1$					$COST_2$			#MOD				TIME [sec.]	
	LENG	C	F	D	MOLD	TOTAL	Δ	L	U	EXC	ROT	MSEQ	ALL	[10]	OURS
data0	3528.9	311	115.5	576	0	1348.8	28.2	1218	19	33	7	86	126	72.1	3.7 (−94.9%)
data1	3209.1	137	51.6	400	0	743.3	4.6	869	0	8	5	64	77	39.3	2.1 (−94.7%)
data2	1722.6	112	60.6	253	0	607.2	17.8	550	1	22	4	34	60	6.5	0.7 (−89.2%)
data3	1298.1	79	37.2	215	0	442.7	4.5	436	3	15	0	36	51	4.2	0.5 (−88.1%)
data4	1337.3	62	33.0	251	0	444.9	5.1	539	4	18	2	55	75	6.7	0.8 (−88.1%)
data5	1362.7	73	40.1	262	0	495.5	2.7	607	6	12	5	48	65	5.6	0.7 (−87.5%)
Ave.															(−90.4%)

Table 3 The results of second phase.

DATA	WIRE	C_{MAX}	#VIO	$COST_1$					$COST_2$			#MOD				TIME
	LENG			C	F	D	MOLD	TOTAL	Δ	L	U	N(1)	N(2)	N(3)	ALL	[sec.]
data0	3482.9 (−1.3%)	2.5	47	333	131.9	569	0	1429.7	20.9 (−26.0%)	1186	1	3	26	29	58	213.1
data1	3198.5 (−0.3%)	1.6	5	181	71.3	372	0	838.0	2.2 (−51.1%)	759	0	0	24	13	37	32.2
data2	1693.6 (−1.7%)	2.1	28	121	70.0	247	0	648.0	11.0 (−38.0%)	506	0	0	16	8	24	10.7
data3	1286.2 (−0.9%)	1.8	10	81	43.8	208	0	464.1	3.3 (−27.8%)	422	0	0	2	5	7	5.9
data4	1316.9 (−1.5%)	1.8	6	63	37.7	251	0	464.9	1.4 (−71.8%)	534	0	1	10	7	18	12.3
data5	1317.4 (−3.3%)	1.3	6	105	54.9	244	0	568.4	1.3 (−53.7%)	523	0	1	15	13	29	29.8
Ave.	(−1.5%)								(−44.7%)							

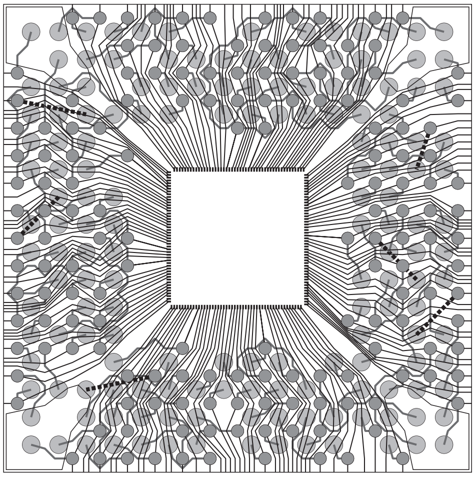


Fig. 9 The output routes of the first phase for data5.

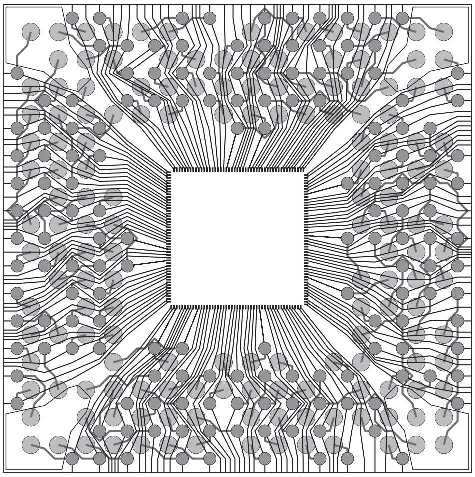


Fig. 10 The output routes of the second phase for data5.

lines in Fig. 9.

In the second phase, the completion ratio is drastically improved and the violation of layer 1 is decreased by 44.7% on average from the first phase while keeping the total wire length on both of layers. The result with violations of layer 1 can not be used as it is. However, even though the violations still exist after the second phase, 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.

On the other hand, our method does not realize all nets in data0 where the output of the first phase has many unconnected nets. In the case that 4-CROT is used in the second phase instead of 3-CROT, all nets are connected in all data and the violation of layer 1 is decreased by 59.3% on average from the first phase. However, the execution time of the second phase increases 4.4 times.

In larger packages, feasible solutions may be obtained quickly if the initial via assignment in which vias are placed out of a mold gate is created with the routability analysis. In addition, 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.

7. Conclusion

We showed that a modification with the maximum gain is obtained in $O(N)$, though it takes $O(N^2)$ times in [10]. Moreover, we introduced a routing graph for routing on layer 2, and a local modification k-CROT 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 about ten times faster than the method proposed in [10] in the first phase, and the routability on layer 2 is improved drastically in the second phase while keeping the maximum wire congestion and the total wire length of both layers. Our proposed method explores monotonic via assignments effectively, and a via assignment which guarantees 100% routing on layer 2 is obtained in most of test cases. To achieve better routability in large package and to reduce the violation of wire congestion on layer 1 by the realization of some plating leads on layer 2 are in our future work.

Acknowledgments

This research was partially supported by Grant-in-Aid for JSPS Fellows (19·9340).

References

[1] Y. Tomioka and A. Takahashi, "Routability driven modification method of monotonic via assignment for 2-layer ball grid array packages," Proc. Asia and South Pacific Design Automation Conference 2008 (ASP-DAC 2008), pp.238–243, Jan. 2008.

[2] E.S. Kuh, T. Kashiwabara, and T. Fujisawa, "On optimum single-row routing," IEEE Trans. Circuits Syst., vol.CAS-26, no.6, pp.361–368, 1979.

[3] S. Tsukiyama and E.S. Kuh, "Double-row planar routing and permutation layout," Networks, vol.12, no.3, pp.287–316, 1982.

[4] J.W. Fang, I.J. Lin, P.H. Yuh, Y.W. Chang, and J.H. Wang, "A routing algorithm for flip-chip design," ICCAD'05: Proc. 2005 IEEE/ACM International Conference on Computer-Aided Design, pp.753–758, IEEE Computer Society, Washington, DC, USA, 2005.

[5] J.W. Fang, C.H. Hsu, and Y.W. Chang, "An integer linear programming based routing algorithm for flip-chip design," DAC'07: Proc. 44th Annual Conference on Design Automation, pp.606–611, ACM, New York, NY, USA, 2007.

[6] M.F. Yu and W.W.M. Dai, "Single-layer fanout routing and routability analysis for ball grid arrays," Proc. International Conference Computer-Aided Design, pp.581–586, 1995.

[7] S. Shibata, K. Ukai, N. Togawa, M. Sato, and T. Ohtsuki, "A BGA package routing algorithm on sketch layout system," J. Japan Institute for Interconnecting and Packaging Electronic Circuits, vol.12, no.4, pp.241–246, 1997.

[8] S.S. Chen, J.J. Chen, C.C. Tsai, and S.J. Chen, "An even wiring approach to the ball grid array package routing," Proc. International Conference on Computer Design, pp.303–306, 1999.

[9] Y. Kubo and A. Takahashi, "A via assignment and global routing method for 2-layer ball grid array packages," IEICE Trans. Fundamentals, vol.E88-A, no.5, pp.1283–1289, May 2005.

[10] Y. Kubo and A. Takahashi, "Global routing by iterative improvements for 2-layer ball grid array packages," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol.25, no.4, pp.725–733, 2006.

Yoichi Tomioka received his B.E. and M.E. degrees from Tokyo Institute of Technology, Tokyo, Japan, in 2005 and 2006, respectively. He is currently a Ph.D. student of Department of Communications and Integrated Systems in Tokyo Institute of Technology. His research interests are in VLSI package design automation and combinational algorithms. He is a member of IPSJ.

Atsushi Takahashi received his B.E., M.E., and D.E. degrees in electrical and electronic engineering from Tokyo Institute of Technology, Tokyo, Japan, in 1989, 1991, and 1996, respectively. He had been with the Tokyo Institute of Technology as a research associate from 1991 to 1997 and has been an associate professor since 1997. He visited University of California, Los Angeles, U.S.A., as a visiting scholar from 2001 to 2002. He is currently with Department of Communications and Integrated Systems, Graduate School of Science and Engineering, Tokyo Institute of Technology. His research interests are in VLSI layout design and combinational algorithms. He is a member of IEEE and IPSJ.