Simulated Annealing Approach to Crosstalk Minimization in Gridded Channel Routing

Kyoung-Son Jhang, Soonhoi Ha, Chu Shik Jhon
Dept. of Computer Engineering
Seoul National University
Seoul, 151-742, KOREA

August 16, 1995

Abstract

The inter-wire spacing in a VLSI chip becomes closer as the VLSI fabrication technology rapidly evolves. Accordingly, it becomes important to minimize crosstalk caused by the coupling capacitance between adjacent wires in the layout design for the fast and safe VLSI circuits. We present a simulated annealing approach based on segment rearrangement to crosstalk minimization in an initially gridded channel routing. The proposed technique is compared with previous track-oriented techniques, especially a track permutation technique whose performance is bounded by an exhaustive track permutation algorithm. Experiments showed that the presented technique is more effective than the track permutation technique.

Keywords - VLSI, coupling capacitance, crosstalk, channel routing, track permutation, segment rearrangement.

INTRODUCTION

The inter-wire spacing in a VLSI chip becomes closer as the VLSI fabrication technology rapidly evolves. It is known that the coupling capacitance between wires becomes comparable to or larger than the ground capacitance in MOS technologies as the wire-to-wire

spacing(S) becomes smaller than 1μm[1, 2] as shown in Figure 1. Figure 1[1] indicates the relative magnitudes of ground capacitance and coupling capacitance assuming that the thickness of field oxide(X) is equal to the height of the wire(G)[1]. Moreover, coupling capacitance, the main source of crosstalk, can cause inadvertent logic transition. Therefore, it is important to minimize the crosstalk for the fast and safe VLSI circuits. For example, there have been proposed some techniques considering crosstalk in multi-chip module design. Cho *et al.*[3] proposed a layer assingment technique to minimize interference which includes crosstalk. Chen *et al.*[4] restricted the minimum distance between horizontal wires to satisfy the crosstalk constraints in the routing of horizontal inter-chip channels.

FIGURE 1

In VLSI layout design, there are two feasible approaches to consider crosstalk in channel routing. One is to use an unified objective function to minimize the crosstalk of nets as well as the number of tracks. However, routing problem with crosstalk constraints is more difficult to solve than conventional routing problem only to minimize the number of tracks because the objective function depends on relative positions as well as absolute positions of the wire segments. We are not aware of any proposed router using this approach. The other approach is to split the routing procedure into two phases, and to use different objective functions in each phase. In the first phase, the channel router minimizes the number of tracks. In the second phase, the objective is to minimize the crosstalk. Since this approach may utilize the existing channel routers[5, 6] in order to generate a routing in the first phase, it has lower time complexity than the former approach. In this paper, we focus on the second phase channel router.

Recently, several layout techniques acceptable in the second phase have been introduced[7, 8, 9]. Chaudhary *et al.*[7] presented a linear programming approach to increase/decrease the spacing between wire segments in order to optimize the interconnection delay and to reduce the crosstalk. Crosstalk minimization with this approach is limited because it cannot change ordering between wire segments. In addition, this technique may

[1] C_x, C_f, $C_{p\text{-}p}$, C_g, and C_{total} indicate unit length coupling capacitance, fringing capacitance, parallel-plate capacitance, ground capacitance, and interconnection capacitance, respectively. C_g is $C_f + C_{p\text{-}p}$, and C_{total} is $C_x + C_g$.

increase routing area[7]. Gao *et al.*[9] proposed a crosstalk minimization technique based on track permutation with the initial gridded channel routing. They formulated an approximated mixed ILP(integer linear programming) problem for crosstalk minimization. But, the number of possible permutations is limited by the ordering requirements among tracks imposed by the vertical constraints, which will be discussed later.

As a second phase router, we propose a crosstalk minimization technique based on the rearrangements of segments constituting tracks. This problem has $O(w^{|H|})$ search space when there are no vertical constraints that restrict the rearrangements(w is the number of tracks, and $|H|$ the number of wire segments). Gao *et al.*[9] solved a smaller and approximated version of our problem, i.e. crosstalk minimization based on track permutation by ILP solver whose complexity is not polynomial. No optimal algorithm with polynomial time complexity is known, even though NP-completeness of our problem has not been proved so far. Thus, we employ a simulated annealing technique to solve our problem. Simulated annealing technique is a well-known probabilistic hill climbing technique[10], which is guaranteed to find an optimal solution with probability one if the search process is executed infinitely. Kirkpatrick *et al.* applied first the annealing idea to the optimization problems such as placement, routing and traveling salesman problem. Afterwards, many researchers have applied the simulated annealing technique to such optimization problems[11, 12, 13, 14]. In this paper, we make use of the simulated annealing technique based on the rearrangement of horizontal wire segments. The basic rearrangements are 1) moving a horizontal wire segment(shortly segment) to another track and 2) swapping two sets of segments. The proposed technique establishes a cost function to be iteratively improved through rearrangements. However, it performs not only the rearrangements improving the cost but also those degrading the cost with non-zero probability in order to escape from the local optimal solutions.

While the unit of rearrangement in the previous track permutation technique[9] is a track, the unit of rearrangement in our technique is a horizontal wire segment constituting tracks. Even when the ordering of tracks is fixed, there are usually more chances of segment based rearrangements further reducing the crosstalk. With experiments, we observed that the proposed technique is more effective than the exhaustive track permutation technique.

PROBLEM FORMULATION

In a gridded channel routing problem, the routing is performed only over the horizontal grid lines(or tracks) and the vertical grid lines(or columns) as shown in Figure 2. Two layers are available for routing: one for the vertical routing, and the other for the horizontal routing. The connection between wires in different layers is made through via(shown by x in Figure 2) whose position is restricted to the grid points where the vertical and the horizontal grid lines intersect. Tracks are numbered from bottom to top, and columns are numbered from left to right as shown in Figure 2. Let the number of tracks be w. The gridded channel routing problem is to connect all the pins of the same net number on the top and the bottom of the channel. The objective of the problem is usually to minimize the number of tracks w used.

First of all, we define the following notations.

- $[L_h, R_h]$: the interval of the horizontal segment h, where L_h is the left end column of the horizontal segment h, and R_h is the right end column of the horizontal segment h.
- $overlap(h_i, h_j)$: the number of columns in the overlapped interval between two horizontal segments h_i, h_j.

FIGURE 2

We may introduce jogs by splitting a segment h with an interval $[L_h, R_h]$ into a set of segments with intervals, $[L_h, L_h + 1]$, $[L_h + 1, L_h + 2]$, ..., $[R_h - 1, R_h]$ during the rearrangement process. However, since this results in too much increase in the number of segments and the running time, we avoid jogs to simplify the problem. In other words, we assume that there are no changes in the intervals of horizontal segments. On the other hand, the upper end track and the lower end track of each vertical wire segment may change according to the track positions of the connected horizontal wire segments. Then, a solution state(or routing) during rearrangement process can be completely defined by the positions of horizontal wire segments. A state can be defined by a function T mapping each horizontal segment to one of w tracks as follows.

$T : H \longrightarrow \{1, 2, ..., w\}$,

where H is the set of the horizontal segments.

But the state defined as above does not always indicate a valid routing. To be a valid state, each pair of horizontal segments h_i, h_j on the same track should not be overlapped, i.e. $T(h_i) = T(h_j) \longrightarrow overlap(h_i, h_j) = 0$. Also, the horizontal segments should satisfy the ordering relations, called vertical constraints, defined by the initial routing so that there should be no overlaps between vertical segments on a column. In Figure 2, the segment h_1 should always be placed above the segment h_3 lest the vertical segments on column 5 should be overlapped. We denote by $< h_i,\ h_j >$ the vertical constraint that the segment h_i should be placed above the segment h_j. Vertical constraint graph, abbreviated by VCG, is constructed from an initial routing. It is an acyclic directed graph where nodes represent horizontal segments, and edges represent vertical constraints. An example of VCG for Figure 2 is shown in Figure 3. Node g is said a predecessor of node h if there is a directed path from node g to node h. Node h is called a successor of node g. In Figure 3, h_7 is a predecessor of h_{11}, and h_{11} is a successor of h_7.

FIGURE 3

The crosstalk of a net depends on the coupling capacitance of wires on the net with wires on the other nets. Since the coupling capacitance between two non-adjacent wires is shielded by the other wires in-between, the coupling capacitance can be assumed to exist only between two visible wires. When two wires are visible from each other as shown in Figure 1(a), the coupling capacitance C is proportional to the length L of visible segment of wires, and inversely proportional to the distance S between two wires: $C = k * L/S^{1.34}$[15][2]

[2]Original formula for unit length coupling capacitance[15] is as follows.

$$C_X = \left[0.03\frac{W}{X} + 0.83\frac{G}{X} - 0.07\left(\frac{G}{X}\right)^{0.222}\right]\left(\frac{S}{X}\right)^{-1.34}$$

Relative error of this formula is less than 10% for $0.3 < W/X < 10$, $0.3 < G/X < 10$, $0.5 < S/X < 10$. We omitted process-dependent terms, since we can make the terms dependent on the process parameters(W, G, and X) into constant k for a given process.

The proposed technique is programmed considering only the coupling capacitances between wire segments in adjacent tracks or columns for simplicity although it can be easily extended to consider the coupling capacitances between visible non-adjacent wires. As the wire density of a channel grows, this approximation would approach to our formulation.

The crosstalk of a net depends also on the signal transition rate on the net. We may take into account the transition rate on a net as a weighting factor for its crosstalk computation. For example, if net n_1 has larger worst-case signal transition rate than net n_2, net n_1 is assigned bigger weighting factor than net n_2. If two nets n_i, n_j are adjacent, and their weights are ω_i and ω_j, the crosstalk between two nets would be scaled by a factor $MAX(\omega_i, \omega_j)$, since we consider the worst case. However, in this paper the crosstalk of a net is assumed to be normalized by signal transition rates, for simplicity, to make the crosstalk depend solely on the coupling capacitance.

The coupling capacitance of a net n in a routing, denoted $CC(n)$, is the sum of $C_V(n)$ and $C_H(n)$, where $C_V(n)(C_H(n))$ is the sum of coupling capacitances between vertical (horizontal) segments of net n and the other nets. The crosstalk slack of a net n, $slack(n)$, is defined as $UB_n - CC(n)$, where UB_n is the crosstalk upper bound (or crosstalk constraint) of net n. Each net usually has a different crosstalk upper bound. We define the *minslack* of a state as the smallest slack in the routing, and the crosstalk critical nets as the nets which have the smallest slack. The net with negative slack violates the crosstalk constraint. Therefore, in order to satisfy the crosstalk constraints, we have to find a routing with non-negative *minslack*. Also, in order to minimize the crosstalk, we should find a routing with the maximum *minslack*. Since the *minslack* indicates the safeness of the corresponding circuits, we can improve the safeness of the circuits by maximizing the *minslack*. In Figure 2, assuming that the crosstalk upper bound for each net is 20, the nets 5, 7, and 8 violates the crosstalk constraints, since the slacks of the nets 5, 7, and 8 are -1, -3, and -14, respectively.

TRACK PERMUTATION TECHNIQUES TO MINIMIZE THE CROSSTALK

Here, we review Gao *et al.*'s track permutation technique which is based on approximated mixed ILP formulation. Then, our exhaustive track permutation technique is de-

scribed, which is devised for comparison with the segment rearrangement technique which is presented in the next section.

Gao *et al.* employ a few types of variables to represent vertical constraints among tracks and the cost function. The initial routing is given by w tracks, t_1, t_2, ..., t_w. T' denotes the function which maps each track t_i to one of w tracks in the the final routing. The variable T_{ij} is defined to be 1 if a track t_i in the initial routing is mapped to j-th track position in the final routing, otherwise T_{ij} is defined to be 0. The final track position, $T'(t_i)$ of track t_i in the initial routing is represented by $\sum_{k=1}^{w} kT_{ik}$. So, the vertical constraint $< t_i, t_j >$ can be expressed as $\sum_{k=1}^{w} kT_{ik} < \sum_{k=1}^{w} kT_{jk}$. Another type of variable is P_{ij} variable, which represents adjacency between two tracks in the final routing. P_{ij} becomes 0 when track t_i and t_j are placed in the adjacent track positions in the final routing, otherwise P_{ij} is 0. The crosstalk of a segment h of a net in the final routing is represented by P_{ij} variable as $\sum_{allj<i} P_{ji}S_j + \sum_{allj>i} P_{ij}S_j$, where S_j denotes $\sum_{s \in t_j} overlap(h, s)$, the sum of *overlaps* between the segment h and all the segments on track t_j in the initial routing. However, the crosstalk between some vertical segments cannot be represented exactly with this formulation. For example, when a dogleg segment and other vertical segment are adjacent as shown in Figure 4(a), the crosstalk between them is estimated with a fixed value, the half of the length of the dogleg segment in the initial routing. In addition, the crosstalk between two vertical segments connected to the same side(top or bottom) as shown in Figure 4(b) is estimated with a fixed value which is also computed based on initial routing. Note that the latter case occurs very frequently in the actual routing examples. Since the cost function of Gao *et al.*'s is not exact, their approach cannot be guaranteed to find an optimal solution even to the problem of crosstalk minimization based on track permutation.

FIGURE 4

On the other hand, our exhaustive track permutation technique is based on the exact formulation, and explores the whole valid solution space which is reduced by the vertical constraint. Therefore, the formance of the exhaustive track permutation algorithm defines the upper bound of all track permutation techniques including Gao *et al.*'s. Our idea of

track permutation is driven by the systematic exchange of two tracks in the given routing. The technique is illustrated by displaying the permutation process for the character string(or routing with four tracks) "abcd" in the Figure 5, where each character in the string corresponds to a track in the routing. In the first level, three more permutations are generated from "abcd" by exchanging the character in the first position with the characters in the other positions. The two numbers in the parenthesis mean the positions where the characters in the left string are exchanged to get the right string. In the second level, we generate two more permutations for each permutation generated in the first level based on the similar exchange operation. But, the first character in each string is fixed, and the second character in the string is exchanged with the characters in the remaining positions. If this procedure continues to the $(w-1)$-th level, all the possible permutations of the string "abcd" are generated.

FIGURE 5

In Figure 5, if <a, b> is the vertical constraint defined by the initial routing, the string "bacd" violates the vertical constraint, and the permutations generated in $j(>1)$-th level from "bacd" also violates the constraint, since 'b' is fixed in the first position. Therefore, in order to reduce the explored solution space, we do not try to generate more permutations from the $(i+1)$-th level for each permutation which violates the vertical constraints in i-th level.

The worst case time complexity of the algorithm is $O(w! * |H|)$, where $|H|$ means the number of horizontal segments, and the time taken to compute the cost of each generated routing. However, since w is rather small(10-30) for the practical size examples, and the vertical constraints usually reduces the explored solution space, the proposed technique can be applied to the real examples.

SIMULATED ANNEALING APPROACH TO MINIMIZE THE CROSSTALK

In this section, we propose a simulated annealing approach to crosstalk minimization based on segment rearrangements from the initial routing.

Track Permutation vs. Segment Rearrangement

The track permutation technique[9] is not efficient at maximizing the *minslack* in cases where the possible number of permutations is small due to the vertical constraints. In Figure 2, the ordering between track 1 and 2 is fixed, and the ordering among track 3, 4, and 5 is also fixed, so only track 2 and 3 can be exchanged. If we set UB_n for each net n to 20, the slacks become -2, -6, and -13 for the nets 5, 7, and 8 respectively after the track permutation (2:3). But, if we apply the segment rearrangement technique, we can get the routing with the *minslack* 0, which satisfies the crosstalk constraints as shown in Figure 6.

FIGURE 6

The unit of rearrangement in the proposed approach is a segment. The basic rearrangements are 1) the movement of a segment to another track, called "move", and 2) the swapping of two sets of segments in two different tracks, called "swap". A swap rearrangement in a routing T is defined by a 4-tuple, (t_1, p_1, t_2, p_2), where $p_1 \subseteq \{h|T(h) = t_1\}, p_2 \subseteq \{g|T(g) = t_2\}$. If either p_1 or p_2 is an empty set, swap rearrangement becomes move rearrangement. Therefore, the 4-tuple represents both move and swap. In the new state T' obtained by applying the swap to the state T, p_1 is placed on track t_2, and p_2 on track t_1. We call a swap valid if the new state is a valid routing. A valid swap should satisfy the following two constraints. First, a valid swap should not cause horizontal segments to overlap in a new state. Or, the following conditions should be satisfied.

$$overlap(g_i, h_j) = 0 \text{ for } \forall g_i \in \{g|T(g) = t_1\} - p_1 \text{ and } \forall h_j \in p_2$$

$$overlap(g_j, h_i) = 0 \text{ for } \forall g_j \in \{g|T(g) = t_2\} - p_2 \text{ and } \forall h_i \in p_1$$

Second, a valid swap should not make vertical segments overlap in a new state. In other words, when $t_1 < t_2$, the predecessors of p_1 should be above t_2, and the successors of p_2 should be below t_1. This constraint can be formulated as follows.

$T(g) > t_2$ for $\forall g \in PRED(p_1)$,
$T(h) < t_1$ for $\forall h \in SUCC(p_2)$,
where $PRED(p_1)$ means the set of predecessors of p_1,
and $SUCC(p_2)$ means the set of successors of p_2.

A valid swap is called profitable swap when its application leads to the state with increased *minslack* or to the state with the same *minslack* but with the smaller number of critical nets.

Simulated Annealing Algorithm based on Swap

Annealing process is used to grow a crystal from a melted material, i.e. determines the low energy state of the material by first melting the substance, then lowering the temperature slowly, and spending a long time at temperatures in the vicinity of the freezing point[10]. In the proposed simulated annealing technique, particles of a material in the annealing process correspond to horizontal segments of a routing. Our simulated annealing algorithm employs valid swaps as basic move operations on the given routing, and its objective is to get a routing with the maximum *minslack*. So, the cost function is *minslack* defined in the problem formulation section. The algorithm to control the application of valid swaps is illustrated in Figure 7.

The algorithm generates a random valid swap and applies it if it is profitable(step 5). Even though it is not profitable(or *minslack* is decreased), the swap is applied (step 6) if a generated random number is smaller than the acceptance probability. The magnitude of acceptance probability, $e^{\delta/t}$ is proportional to the current temperature t, and inversely proportional to the cost difference $|\delta|$(or *minslack* decrement). Therefore, at high temperature, the unprofitable swaps are more likely to be allowed. However, at low temperature, such swaps are not usually allowed except swaps with smallest cost difference, i.e. the search process tends to become greedy. The temperature is decremented stepwisely (step 7) whenever the equilibrium condition is satisfied at each temperature. The process continues until the frozen or termination condition is satisfied(step 2).

The performance of a simulated annealing algorithm usually depends on the move set design and the annealing schedule. In our approach, the move set is the set of valid swaps(step 4) applicable to the current routing. Annealing schedule consists of four components; initial hot temperature(step 1), temperature decrement(step 7), equilibrium condition(step 3), and frozen or termination(step 2) condition.

We employ a simple schedule similar to that of [11] as follows. The initial hot temperature is determined by the formula, $t = |\text{initial } minslack| * |H|$. The current temperature

is decremented by the formula, $t = t * \alpha$. Equilibrium is considered established if the number of the accepted states are more than $\beta * |H|$. The constants α and β are determined, based on experiments so they results in good performance and reasonable amount of running time. We set α and β to 0.90 and 5, respectively. We terminate the annealing process if the finally accepted states at the end of inner loop are the same three times in succession.

FIGURE 7

EXPERIMENTAL RESULTS

The proposed technique, named SA(Simulated Annealing), was implemented in C-language on a Sun-Sparc2 workstation. An exhaustive track permutation algorithm, named EXTP, was also implemented for the performance comparison with SA. The performance obtained by Gao *et al.*'s approach[9] is bounded by the performance of EXTP.

Each test example consists of the specification on the vertical and the horizontal segments, the upper bound of the crosstalk for each net, and an initial routing solution(track positions for each horizontal segments). The test examples used in experiments have the characteristics shown in Table I. The test example **ex1** has the the same initial routing solution as shown in Figure 2, with the same crosstalk upper bound for each net which is 20. The test examples **small2**, **yk3c**, **d1**, **deutsch** were used in [9]. These test examples have the same initial routings to those used in [9], but **small2**, **yk3c**, **d1** have different crosstalk constraints from those used in [9][3]. The test examples **3a** and **3c** were used in YACR[6], and their crosstalk upper bounds were set randomly. The other examples were made randomly. The initial routing solutions of the test examples were either provided by the author of the paper[9] or generated by a channel router[16]. The example **deutsch′** is different from the example **deutsch** only in terms of the crosstalk upper bounds of nets. The data in the parenthesis are cited from [9] to compare with those of EXTP and SA.

[3]Therefore, Our results are different from those presented in Gao *et al.*'s paper[9]. We asked Gao to provide the same crosstalk constraints for those examples, but we are still waiting his reply.

TABLE I

The running times of Gao *et al.*'s approach for **yk3c** and **d1** are over 6 minutes according to Gao as shown in Table II. Considering that RS/6000 is about 10 times faster than Sparc2,[4] the running times will be over 1 hour in Sparc2 time. The time complexity of the algorithm solving ILP problem is not polynomial with respect to the number of integer variables. The linear integer programming formulation of track permutation problem requires $O(w^2)$ integer variables. Thus, its time complexity is roughly comparable to EXTP at least for the test examples we used in this paper.

We compared two techniques, SA and EXTP to see the performance difference between track permutation and segment rearrangement. The comparison results are shown in Table II. Table II displays the *minslacks* of initial routings, the maximum *minslacks* achieved by two techniques, and the times spent by two techniques in Sun-Sparc2 seconds for each test example. Since SA is not a deterministic algorithm, we ran SA 10 times for each test example. So, we display the best *minslack* and the average *minslack*. In addition, the execution time of SA indicates the average termination time. We display the time spent by EXTP to indirectly show how many track permutations each test example has. For all examples, SA satisfies the crosstalk constraints and has no smaller *minslack* than EXTP. Performance(*minslack*) difference between EXTP and SA is not significant for the test examples that have only a few segments on each track such as **yk3c**, **3a**, **3c**. But the performance of EXTP degrades significantly for the test examples that have a few track permutations such as **ex1**, **rand1**, and **rand3**, or for the test examples which have relatively greater number of segments on a track such as **deutsch′** and **d1**. The total numbers of track permutations of test examples **small2**, **deutsch**, **rand1**, **rand2**, and **rand3** are 6, 55, 2, 164, and 3, respectively.

TABLE II

To examine the effect of upper bounds on the solution state, we modified the crosstalk upperbounds of a few nets of the **deutsch** example to make the **deustch′** example. It

[4]This comparison is based on linpack computation

changes the order of segment rearrangements, which affects the running time of the algorithm. We noticed the significant distinction on the performance between SA and EXTP in this example.

TABLE III

We determined the annealing parameters, α and β, based on experiments. Table III shows the average $minslack$ and the average running time of our algorithm according to changes of the annealing parameters. The bigger parameter β is, the longer time the algorithm takes to reach the equilibrium state. The closer to 1 parameter α is, the smaller the temperature decrement is. Unexpectedly, however, slowing down the temperature decrement does not always lead to better performance in spite of the larger execution time. It seems that the performance is more sensitive to parameter β than parameter α. Thus, we select the combination, $\alpha/\beta = 0.90/5$, since it leads to reasonable amount of running time and good performance.

SUMMARY AND FUTURE WORKS

This paper proposes a crosstalk minimization technique based on segment rearrangement for gridded channel routing. We make use of the simulated annealing approach, a well-known probabilistic hill climbing technique. With experiments, we observed that the presented technique is more effective than the track permutation technique: especially, for the test examples that have only a few chances of track permutation or for the test examples that have relatively large number of segments on a track.

However, the time complexity of simulated annealing technique is rather high. In order to reduce the running time, we may need to find a more suitable annealing schedule. We developed another heuristic[17], which repeatedly rearranges horizontal segments around the crosstalk critical nets to rapidly improve $minslack$. However, it may sacrifice performance($minslack$).

As future works, We may need to extend our proposed approach to more general routing models such as overlap routing model, three or more layer routing model, and PCB routing model.

References

[1] H.B. Bakoglu, "Circuits, interconnections, and packaging for VLSI," *Addison wesley*, Chapter 4.2, 1990.

[2] E. Barke, "Line-to-ground capacitance calculation for VLSI: a comparison, " *IEEE Trans. CAD*, vol. 7, no. 2. pp.295-298, Feb. 1988.

[3] J.D. Cho, S. Raje, M. Sarrafzadeh, M. Sriram, S.M. Kang, "Crosstalk-minimum layer assignment," *IEEE 1993 Custom Integrated Circuits Conference*, pp.29.7.1-29.7.4, May 1993.

[4] H.H. Chen, C.K. Wong, "Wiring and crosstalk avoidance in multi-chip module design," *IEEE 1992 Custom Integrated Circuits Conference*, pp.28.6.1-28.6.4, May 1992.

[5] T.-T. Ho, S.S. Iyengar, S.-Q. Zheng, "A general greedy channel routing algorithm," *IEEE Trans. CAD*, vol.10, pp.204-211, Feb. 1991.

[6] A. Sangiovanni-Vincentelli, . Santomauro, "A new gridless channel router: Yet Another Channel Router(YACR-II)," *Proc. ICCAD'*84, pp.72-75, 1984.

[7] K. Chaudhary, A. Onozawa, E.S. Kuh, "A spacing algorithm for performance enhancement and cross-talk reduction," *Proc. ICCAD '93*, pp.697-702, Nov. 1993.

[8] A. Onozawa, "Layout compaction with attractive and repulsive constraints," *Proc. 27th Design Automation Conference*, pp.369-376, June 1990.

[9] T. Gao, C.L. Liu, "Minimum crosstalk channel routing," *Proc. ICCAD '93*, pp.692-696, Nov. 1993.

[10] S. Kirkpatrick, C.D. Gelatt, .P. Vecchi, "Optimization by Simulated Annealing," *Science*, vol.220, no.4598, pp.671-680, May 1983.

[11] C. Sechen, "VLSI Placement and global routing using simulated annealing," *Kluwer Academic Publishers*, 1988.

[12] H.W. Leong, D.F. Wong, C.L. Liu, "A simulated-annealing channel router," *IEEE ICCAD '85*, pp.226-228, 1985.

[13] H. Flisher, *et al.*, "Simulated annealing as a tool for logic optimization in a CAD environment," *Proc. ICCAD '85*, pp.203-205, 1985.

[14] R.H.J.M. Otten, L.P.P.P van Ginneken, "Floorplan design using simulated annealing," *Proc. ICCAD '84*, pp.96-98, 1984.

[15] T. Sakurai, K. Tamaru, "Simple formulas for two- and three- dimensional capacitance," *IEEE Trans. Electron Devices*, Vol.ED-30, No.2, pp.183-185, Feb. 1983.

[16] K.-S. Jhang, C.S. Jhon, "A channel router with the coupling capacitance reduction," *1994 Joint Technical Conference on Circuits/Systems, Computers and Communications*, pp.370-375, July 1994.

[17] K.-S. Jhang, S. Ha, C.S. Jhon, "A Segment rearrangement approach to channel routing under the crosstalk constraints," *1994 IEEE Asia-Pacific Conference on Circuits and Systems*, pp. 536-541, Dec. 1994.

Biographies

KYOUNG-SON JHANG received his B.S.(1986), M.S.(1988), and Ph.D.(1995) degrees in Computer Engineering from Seoul National University, Seoul, KOREA. Currently, he has worked as a special research staff in Research Institute of Advanced Computer Technology at Seoul National University since he finished his Ph.D. degree. His research interests include VLSI design automation and optimization algorithms.

SOONHOI HA received his Bachelors (1985) and Masters (1987) in Electronics Engineering from Seoul National University, and Ph.D.(1992) degrees in Electrical Engineering and Computer Science from University of California, Berkeley. He is currently a faculty member of Department of Computer Engineering at Seoul National University. His research interests include hardware-software codesign, design methodology for signal processing, parallel computing, and microprocessor architecture.

CHU SHIK JHON received his B.S.(1974) degree in Applied Mathematics from Seoul National University, his M.S.(1976) degree in Computer Science from Korea Advanced Institute of Science and Technology, and his Ph.D.(1982) degree in Computer Science from University of Utah. He worked as a faculty member of University of Iowa, and is currently a professor of Department of Computer Engineering at Seoul National University. His research interests include VLSI CAD and parallel computer architecture.

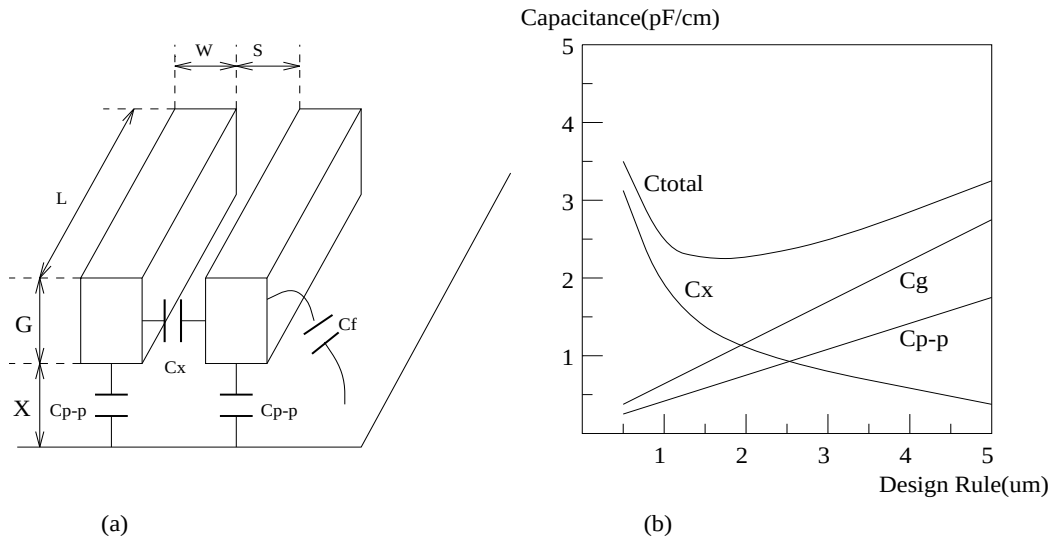

FIGURE 1 Interconnection capacitance and its trend, where X = G = 1μm, and W = S is used as "Design Rule"[1].

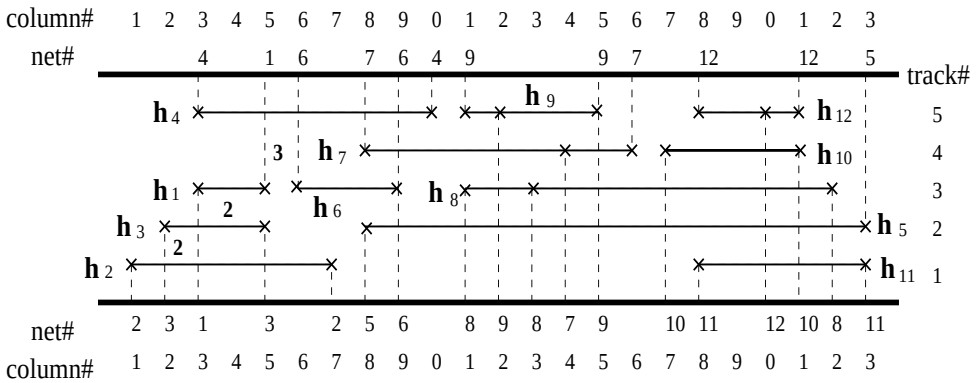

FIGURE 2 An example of gridded channel routing solution.

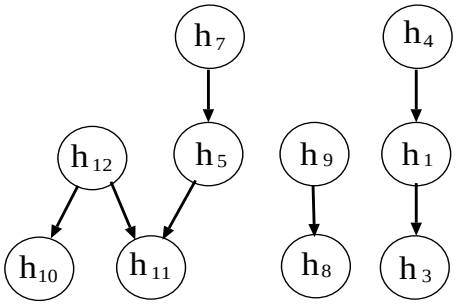

FIGURE 3 Vertical constraint graph for Figure 2.

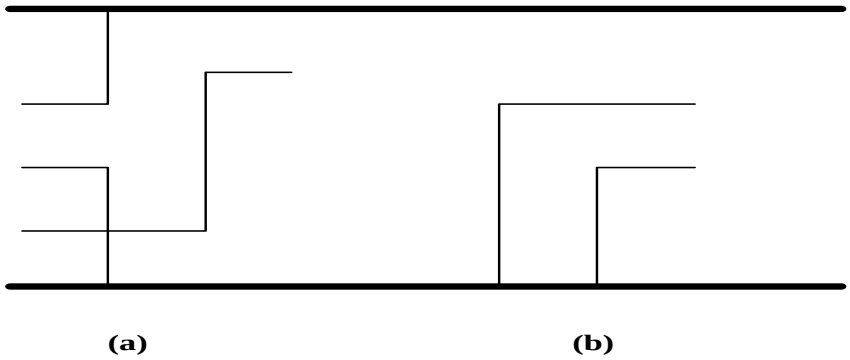

FIGURE 4 Cases where the crosstalk between vertical segments cannot be represented in linear programming style.

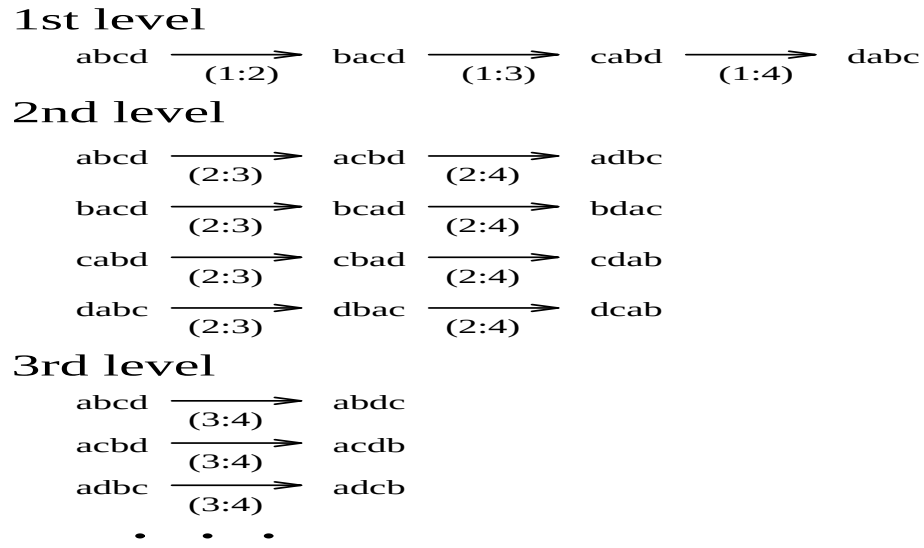

FIGURE 5 Exhaustive generation of permutations for string "abcd".

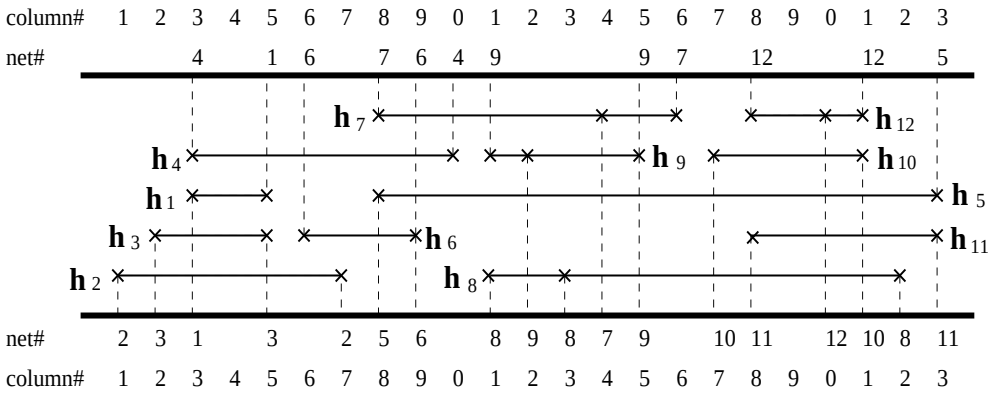

FIGURE 6 A routing solution satisfying the crosstalk constraints generated by the segment rearrangement technique.

```
1: t ⟵ initial hot temperature
   current routing ⟵ given initial routing
   current cost ⟵ minslack of current routing
2: repeat until (frozen condition satisfied) {
3:         repeat until (equilibrium is established) {
4:                 generate a random valid swap
                   new routing ⟵ apply the swap on current routing
                   δ = minslack of new routing - current cost
5:                 if (the swap is profitable) then
                       current routing ⟵ new routing
                       current cost ⟵ minslack
6:                 else
                       generate a random number
                       if (the random number < e^(δ/t)) then
                           current routing ⟵ new routing
                           current cost ⟵ minslack
   }
7: decrement the temperature, t
}
```

FIGURE 7 Simulated annealing algorithm for crosstalk minimization.

TABLE I
The characteristics of test examples.

test examples	# of nets	# of segments	# of columns	# of tracks
ex1	12	12	23	5
small2	10	12	20	5
yk3c	54	54	79	18
d1	65	77	155	19
deutsch	72	118	174	19
deutsch'	72	118	174	19
3a	30	30	45	16
3c	54	54	103	18
rand1	20	20	29	7
rand2	18	18	36	8
rand3	40	40	58	8

.

TABLE II
The comparison results on crosstalk minimization.

test examples	initial *minslack*	EXTP		SA		
		minslack	time	*minslack*1	*minslack*2	time
ex1	-14.0	-13.0	0.1	0.0	0.0	22.3
small2	-17.0(-17.0)	6.0(7.0)	0.0(6)	8.0	8.0	2.6
yk3c	-56.0(-17.0)	24.0(43.0)	1804.0(390)	32.0	30.6	534.3
d1	-98.0(-38.0)	12.0(57.0)	28254.0(360)	32.0	29.3	930.7
deutsch	2.0(2.0)	12.0(12.0)	3.2(24)	12.0	12.0	3058.0
deutsch'	-8.0	-5.0	3.2	29.0	26.5	3369.7
3a	-12.0	8.0	56243.6	11.0	11.0	155.8
3c	-29.0	37.0	4495.3	48.0	46.4	663.4
rand1	-12.0	-9.0	0.1	5.0	5.0	36.2
rand2	-11.0	0.0	0.4	4.0	4.0	49.9
rand3	-2.0	7.0	0.2	23.0	23.0	226.6

Data in () cited from [4]. Running times in () are in RS/6000 seconds.

.

TABLE III
The experiments to show performance dependency on the annealing parameters.

test examples	α/β=0.90/5		α/β=0.90/3		α/β=0.95/5		α/β=0.95/3	
	minslack	time	*minslack*	time	*minslack*	time	*minslack*	time
ex1	0.0	22.3	0.0	12.8	0.0	41.4	0.0	13.7
small2	8.0	2.6	7.7	1.7	7.9	2.5	7.8	1.8
yk3c	30.6	534.3	29.7	294.6	30.9	1065.0	30.4	620.3
d1	29.3	930.7	26.4	543.0	30.5	1862.8	25.7	1160.0
deutsch	12.0	3058.0	11.0	1925.0	12.0	6098.0	12.0	3642.3
deutsch'	26.5	3369.7	26.2	2032.0	27.0	7030.3	16.1	198.2
3a	11.0	155.8	10.8	89.3	11.0	300.7	11.0	185.0
3c	46.4	663.4	45.3	420.6	45.2	1192.3	45.2	723.4
rand1	5.0	36.2	4.8	14.2	4.5	31.7	3.6	21.0
rand2	4.0	49.9	4.0	31.9	4.0	106.9	4.0	62.5
rand3	23.0	226.6	22.4	119.8	22.4	412.0	23.0	265.7

.