設 計 自 動 化 71－3
（1994. 6. 24）

A New Performace-Driven Global Routing Algorithm for Cell-Based Layout

Takashi Fujii†, Tianxiong Xue‡ and Ernest S. Kuh‡

† ULSI Systems Development Laboratories
NEC Corporation
Kawasaki, 211, Japan

‡ Department of Electrical Engineering and Computer Sciences
University of California, Berkeley
Berkeley, CA 94720, U.S.A.

The global routing problem based on critical path-based timing analysis is formulated as a multi-commodity multi-terminal flow problem. The initial routing trees are generated by two net-based performance-driven Steiner tree algorithms. Later, some nets are rerouted to release over-congested channels. As long as the total accumulated delay on the critical paths affected by the nets still satisfy the constraints on the paths, the rerouted trees are accepted. The experimental results based on ISCAS benchmarks show that our method can obtain the same chip size as those which ignore timing completely, while achieving comparable path delay with the net-based method.

セルベースレイアウトのための性能ドリブン概略配線手法

藤井 隆志† Tianxiong Xue‡ Ernest S.Kuh‡

† 日本電気株式会社 ULSI システム開発研究所
‡ カリフォルニア大学バークレイ校

セルベースＶＬＳＩ（ゲートアレイ方式及びスタンダードセル方式）用の性能（タイミング）ドリブン概略配線手法を提案する．本手法では，クリティカルパスに与えられる遅延時間制約を各ネットごとに分割するのではなく，パス全体への制約として扱うことにより面積最適化を図りつつ遅延時間性能を保証することを可能としている．２つのタイミングドリブンスタイナー木アルゴリズム及び多種多シンクフロー問題に対するアルゴリズムを利用して実現したプログラムを ISCAS ベンチマークデータに適用した結果，遅延時間制約を全く考慮しない場合と同じ面積最小化を実現し且つネット単位の制約の場合と同等のタイミング性能を満足する結果を得た．

1 Introduction

Interconnection delay has become a dominant factor in circuit performance of a VLSI chip. Interconnection capacitance and resistance cannot be ignored for delay calculation since they are comparable to gate capacitance and output driver resistance under submicron technology [6, 18].

Since the global routing step in VLSI layout design determines a route for each signal net, these routes affect the interconnection delay. It is one of the most important problems to be considered for the purpose of maximizing the overall chip performance. In most previous timing-driven approaches, the main objective of global routing is to minimize the total length of each net on critical paths [8, 13]. Recently, several approaches have been proposed to minimize both the total length of a net and the longest path from the source pin to any sink pin, because shortest total wire length alone may not yield minimum delay under submicron technology [4, 5, 14]. In [2, 12], routing tree construction methods for minimizing the delay between the source and sinks in a net were described. These approaches were based on the assumption that the timing constraints on critical paths can be satisfied by obtaining the minimum delay Steiner tree for each net. It is likely that this requirement forced upon each net is more strict than actually necessary, because the actual timing requirement is specified for the paths not for each individual net.

This paper proposes a new performance-driven global routing method for gate array and standard cell layout based on critical path timing analysis. A distributed RC model [1, 9, 15, 16] is adopted for interconnection delay. An upper bound of the interconnection delay of each segment (partial path from the source to a sink in a net) on a critical path is derived from the estimation based on the delay model.

The performance-driven global routing problem is formulated as a multi-commodity multi-terminal flow problem [3, 11]. As the input, a set of initial possible routes for each net is created by the efficient performance-driven Steiner tree algorithms [12] which consider the timing requirement in the construction process. With multi-commodity approach, all initial routes are treated simultaneously. Then, if any channel is over congested, some nets have to be rerouted to release the heavily congested channels. The rerouted tree of the net should be checked with respect to the delay bound of critical paths for timing requirement. As long as the total wiring delay on a critical path satisfies the constraint, the route is acceptable. So even for some net where the delay between the source pin and the sink pin exceeds the constraint for that sink pin, the accumulated total delay on the critical paths affected by that net may still satisfy the constraint on the paths, and the route of the net should not be rejected. This is a major difference between our proposed path-based method and previous net-based methods. Unnecessary rejection of the nets' routes may result in increase of channel density, and thus chip area.

The global routing algorithm is applied to three ISCAS benchmark circuits: C2, C7, and s13207. For each data, the following three timing modes are tested and compared:

(i) no timing analysis,
(ii) net-based timing analysis, and
(iii) critical path-based timing analysis.

The experimental results show that the critical path-based method can obtain the very close maximum or total channel density as the "no timing analysis" mode, while achieving comparable path delay with the net-based method.

2 Timing Model

Let u be the index of the source pin and v be the index of the sink pin in a net n. In order to calculate the wire delay $del(n, v)$ at sink pin v, a distributed RC delay model from [1, 9, 15, 16] is referenced. The circuit is modeled by a voltage source with the on-resistance of the transistor - Rtr, distributed RC lines of resistance R_j, capacitance C_j, and loading capacitors C_{Lj} (see, Fig.1). Take sink pin t_2 in Fig.1 as an example, the delay time at sink pin t_2, $del(s, t_2)$, is approximated by the following equation:

$$\begin{aligned} del(s,t_2) = {} & \beta * Rtr * (C_{11} + C_{12} + C_{21} + C_{L1} + C_{22} \\ & + C_3 + C_{L2} + C_4 + C_{L3}) + \beta * R_1 * (C_{12} \\ & + C_{21} + C_{L1} + C_{22} + C_3 + C_{L2} + C_4 \\ & + C_{L3}) + \beta * R_3 * (C_{22} + C_3 + C_{L2} + C_4 \\ & + C_{L3}) + \beta * R_4 * (C_3 + C_{L2}) \end{aligned}$$

where β equals 2.21 which yields the 90 percent of the signal's final delay time [16].

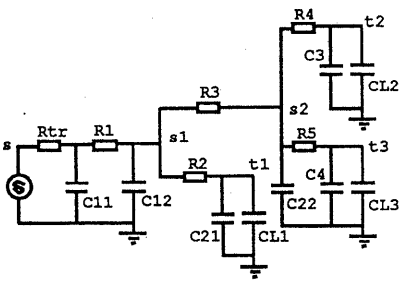

Figure 1. Multi-terminal net interconnection delay model.

In the case where tree-structured interconnections are used for connecting a multi-terminal net, the delay time at sink pin v in net n, $del(n,v)$, has an upper bound $Del(n,v)$ given by the following equation:

$$del(n,v) \leq Del(n,v) = \beta*(Rtr + r_1*L(u,v)) \\ *(c_1*W + C_{in}) \qquad (1)$$

where r_1 and c_1 are wire resistance and capacitance per unit length, respectively. W is the total wire length of the net, $L(u,v)$ is the path length from the source pin u to the sink pin v, and C_{in} is the total loading capacitance of all sink pins of the net. Equation (1) is used in following sections for sink delay estimation.

3 Timing Analysis

In this paper, cell delay is ignored in the estimation of signal propagation delay of a critical path, since it is constant and not affected by any global routes. Critical path information is obtained from placement output.

3.1 Net-Based Approach

One of the easy ways to obtain a good performance design is to route each net such that the delay between the source pin and each sink pin is less than the specified timing constraint on the sink.

Let $\tau(n,v)$ be the delay bound for sink pin v in net n. The routing tree for the net should satisfy:

$$del(n,v) \leq \tau(n,v), \quad \forall n. \qquad (2)$$

It is clear that the required performance is guaranteed if the route of every net satisfies the constraint; however, it may be too strict to require all nets to be routed with the "minimum" delay, because this would cause possible channel congestion.

3.2 Path-Based Approach

Let p be a critical path, which is represented as

$$p: \ v_0 \rightarrow v_1^{in} \simeq v_1^{out} \rightarrow v_2^{in} \simeq v_2^{out} \rightarrow \cdots \\ \rightarrow v_{m-1}^{in} \simeq v_{m-1}^{out} \rightarrow v_m,$$

where v_i^{in} and v_i^{out} are pins of a cell ($v^{in} \simeq v^{out}$ means that pins v^{in} and v^{out} belong to the same cell), and $u \rightarrow v$ represents the segment from source pin u to sink pin v. If a segment $u \rightarrow v$ $(u,v \in n)$ is on a critical path p, $u \rightarrow v$ is said to be included in p, and denoted by $(u \rightarrow v) \in p$. The delay of segment $u \rightarrow v$ is denoted by $del(u \rightarrow v)$.

Generally a VLSI designer specifies the timing requirement as the delay bound between signal input to a chip and its output response from the chip. Thus, it is necessary to check the delay of a critical path between I/O pads. Let P denote the set of all critical paths, and $\mathcal{T}(p)$ be the timing constraint for critical path $p: Q_I \rightarrow \cdots \rightarrow Q_O$ $(p \in P)$ between I/O pads Q_I and Q_O. The path delay should satisfy:

$$\sum_{(u \rightarrow v) \in p} del(u \rightarrow v) \leq \mathcal{T}(p), \quad \forall p \in P. \qquad (3)$$

In our global routing algorithm, the timing constraint $\mathcal{T}(p)$ for each critical path p is given by the placement program RITUAL [17]. $\mathcal{T}(p)$ is computed by the delay of each net on p from the output of register to the input of gate based on the clock cycle time.

According to the above restriction, nets may not be routed with its minimum delay. When the tree of a net on a critical path is rerouted to release the congested area, the timing constraints specified on each net may be violated, but the rerouted tree may still be accepted as long as the constraint on the critical path is satisfied.

In our global routing, the path-based timing analysis method is adopted. Our global routing algorithm generates an initial routing tree for each net according to the timing constraint for each net on a critical path (*i.e.*, the net-based timing analysis method). Then, routes of some nets are modified in order to decrease the channel congestion based on the path-based timing analysis method. It is clear that the satisfaction of timing constraint on the path (Inequality (3)) is the necessary condition of satisfaction of timing constraint of the nets on that path (Inequality (2)), while the later is the sufficient condition of the former. In other words, the timing constraint is loosened so that more nets can be rerouted in the global routing process to release the congested area, which results in

smaller maximum channel density compared with the net-based timing analysis while the timing constraints on the critical paths are still satisfied.

4 Problem Formulation

4.1 Definitions

Let $G = (V, E)$ denote the global routing graph shown by the dotted lines in Fig.2. The chip is dissected into blocks of regions which are represented by V, a set E of edges represents the adjacencies among those blocks, which can be further classified as $E = E_H \cup E_V$, where E_H represents the routing channels, E_V represents the connection of vertical adjacent blocks. Each edge $e \in E$ is assigned a capacity $C(e)$. Each row of the dissected region covers a channel. E_{H_r} denotes the set of horizontal edges in channel r. Ch denotes the set of channels (excluding the top and bottom channels) on the chip. The channel density for channel $r \in Ch$ is denoted by A_r. For gate array layout, each edge $e \in E_{H_r}$ has the same fixed channel capacity C_{ch}. Also, a distance function $d(e)$ is assigned to each $e \in E$ to represent the cost of edge e.

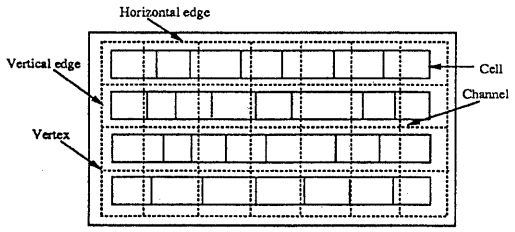

Figure 2. Dissection of chip area for graph generation.

Let N denote the set of nets to be routed on G under timing constraint. Each net $n \in N$ has a demand $B(n)$ which is set to one in general. The routing of net n can be formulated as embedding a routing tree t into G. Since $G(V, E)$ is a finite graph, so t is simply a subset of E. The distance of t is defined as: $D(t) = \sum_{e \in t} d(e)$, which can be interpreted as the cost function for routing t.

Let T_n be a set of all possible routing trees for net n which satisfy the timing constraint. Obviously, $|T_n|$ is finite. Furthermore, since only the best tree according to the distance function is generated during each routing iteration, $|T_n|$ is expected to be relatively small. $f(n, t)$ is denoted as the flow of routing tree $t \in T_n$ for net n. $\delta(n, t, e)$ is a binary function defined to indicate whether tree t for net n passes edge e. The flow on edge e is defined as $F(e) = \sum_{n \in N} \sum_{t \in T_n} f(n, t) * \delta(n, t, e)$.

4.2 Linear Programming Formulation for Gate Array

In gate array layout, C_{ch} is fixed, so the goal in global routing is to search for a flow solution which minimizes the maximum channel density M. If in the final solution, $M > C_{ch}$, then the routing is not feasible. As chip height is defined as the product of maximum channel density and number of rows on the chip, so minimizing M also means minimizing chip height. The global routing problem is formulated as the linear programming problem in the following:

$$\begin{aligned}
&\text{minimize:} && M \\
&\text{subject to:} \\
&\sum_{t \in T_n} f(n,t) \geq B(n), && \forall n \in N \\
&F(e) \leq M, && \forall e \in E_{H_r}, \forall r \in Ch \\
&F(e) \leq C(e), && \forall e \in E_v \\
&\sum_{(u \to v) \in p} Del(u \to v) \leq \mathcal{T}(p), && \forall p \in P \\
&f(n,t) \geq 0, && \forall t \in T_n, \forall n \in N \\
&F(e) \geq 0, && \forall e \in E.
\end{aligned}$$

The first constraint specifies that the total fractional flow of all possible routing trees for net n satisfies the demand for that net. The second constraint defines M as the maximum channel density. The third one demonstrates that the total flow on each vertical edge in E_v must be restricted by its capacity. The fourth one specifies the timing constraint for each critical path, which is enforced by the fact that the routing tree t is picked from the set T_n which only contains those trees satisfying the timing requirement.

4.3 Linear Programming Formulation for Standard Cell

In standard cell layout, C_{ch} is adjustable, so the goal in global routing is to search for a flow solution which minimizes the total channel density $\sum_r A_r$. Here chip height is defined as the summation of channel densities of all rows on the chip, so minimizing $\sum_r A_r$ also means minimizing chip height. The linear programming problem is formulated as:

minimize: $\sum_r A_r$

subject to:

$$\sum_{t \in T_n} f(n,t) \geq B(n), \quad \forall n \in N$$

$$F(e) \leq A_r, \quad \forall e \in E_{H_r}, \forall r \in Ch$$

$$F(e) \leq C(e), \quad \forall e \in E_v$$

$$\sum_{(u \to v) \in p} Del(u \to v) \leq \mathcal{T}(p), \quad \forall p \in P$$

$$f(n,t) \geq 0, \quad \forall t \in T_n, \forall n \in N$$

$$F(e) \geq 0, \quad \forall e \in E.$$

The second constraint defines A_r as the channel density. Other constraints have the same meaning as in the gate array case.

5 Global Routing

In this section, two major issues are discussed. The first is the generation of Steiner trees which satisfy the timing constraint. Various performance-driven Steiner algorithms are used for generating the initial trees for global routing and the rerouted trees for releasing the congested area. The second is a multi-commodity multi-terminal flow global routing algorithm. The global routing algorithm discussed here adopts the methodology in [3], but with different Steiner tree approaches and enforced timing analysis during the routing process to guarantee the satisfaction of specified timing constraints.

5.1 Performance-Driven Steiner Tree Algorithms

(1) Initial Performance-Driven Steiner Tree Construction

Two performance-driven Steiner tree algorithms (IDW and CFD) [12] are used to construct the initial routing trees for the global routing process which satisfy the specified timing constraints. (For details, please refer to [12].)

A) Iterative Dreyfus-Wagner based algorithm (IDW)

The first approach is an iterative Dreyfus-Wagner based Steiner tree algorithm (IDW). Its essence calls for two recursions, which derives the Steiner tree of a given set of nodes from the Steiner trees of its subsets. It is an extension of original Dreyfus-Wagner algorithm [7], but instead of minimizing the total wire length of the net, it adopts a more general objective, *i.e.*, to minimize the delay upper bound at the sink - $Del(n,v)$ in Eq. (1) which incorporates both the total wire length and the distance between the source and the specified sink (minimizing $Del(n,v)$ can guarantee that the real delay at sink - $del(n,v)$ will be restricted below a limit). The nonlinear effect of the timing model discussed in Section 2 is approximated by using the methodology of gradient descent approach.

The algorithm starts with an initial solution. During each iteration, the coefficients of the objective function are updated after the locally optimal solution is found, then the new objective is to be minimized. This process continues until the result converges or the number of iterations goes beyond a given limit.

B) Constructive force directed algorithm (CFD)

Due to the exponential nature of the IDW algorithm, it only works efficiently for nets with small number of pins. For large nets, a constructive force directed Steiner tree algorithm (CFD) is used. The algorithm first maps the original pins of the net onto the routing graph and denotes them as active nodes. A weighted medium point (WMP) is computed for each active node j in such a way that it tends to be closer to those nodes near j. Then two directions $w_{WMP}(j)$ and $w_S(j)$ are defined for each j in the routing tree which point to the WMP and the source S, respectively. A growing direction is determined for j by combining $w_S(j)$ and $w_{WMP}(j)$ with an appropriate balancing factor.

Initially, pins of the net are mapped onto the graph as active nodes. Each active node grows in the chosen direction one step at a time. The moving paths are recorded accordingly. Previous active nodes are deactivated and the newly arrived nodes become active, and then their WMP's are calculated and the new growing directions are decided. If two paths are touched in the growing process, the two active nodes are merged into one. This construction process stops when all growing segments are merged into one tree. Since both the total wire length and the distance between the source and each sink are considered in the construction process by setting a proper balancing factor, CFD algorithm results in minimizing the delay at the sinks.

(2) Performance-Driven Steiner Tree Improvement (STI)

During the global routing process, the congested portions are rerouted to reduce the routing density

under current tree configurations. The congestion is measured by distance function $d(e)$ which is discussed later. For each tree t for net n, the most expensive edge is identified by $d(e)$. Let S and T denote the two subtrees separated by e. The path connecting S and T is deleted and S and T are reconnected by applying a variant of Dijkstra's shortest path algorithm [10]. The newly rerouted tree is checked by the timing analysis methods described in previous section. If it results in violation of the specified timing constraint, then it is not included in the valid routing tree set T_n.

5.2 Performance-Driven Global Routing Algorithm

(1) Edge Distance Calculation

A) Gate Array

Initially, there is no flow on the edges, so $d(e)$ is set as the inverse proportion of the capacity $C(e)$, $d(e) = 1/C(e)$. After initialization, $d(e)$ is defined as the exponential function of the edge congestion, $d(e) = \exp(\alpha * F(e)/C(e))$, where α is a constant. α is set as the extent of penalty of the congested area, which can be tuned according to the precision of computing environment to avoid overflow.

B) Standard Cell

Initially, $d(e) = 1/C(e)$ for each edge $e \in E_V$, and $d(e) = 1/C$ for each edge $e \in E_H$, where C is the average of capacities $C(e)$'s of all vertical edges. After initialization, the $d(e)$ for each vertical edge e is defined as: $d(e) = \exp(\alpha * F(e)/C(e))/k$, where $k = K * \sum_{e \in E_V} \exp(\alpha * F(e)/C(e))$ and K is a constant coefficient. For each horizontal edge $e \in E_{H_r}$: $d(e) = \exp(\alpha * F(e)/A_r)/S_r$, where $S_r = \sum_{e \in E_{H_r}} \exp(\alpha * F(e)/A_r)$, which is used to normalize the distance function.

(2) Global Routing Algorithm GR

The essence of the algorithm can be summarized in four steps (S1-S4).

Algorithm GR

S1: Initialization and generation of initial performance-driven Steiner trees;
S2: while solution is not ϵ-optimal do
  S2.1: Compute edge distance and generate new routing Steiner trees under the timing analysis method;
  S2.2: Check if solution is ϵ-optimal;
  S2.3: Select and rerout net;
S3: Solution integerization;
S4: while not feasible do
  S4.1: Improve integerized solution.

Steps S1 and S2 work toward achieving an ϵ-optimal solution with fractional flows under the condition of timing constraint. Here, ϵ-optimal means that the error margin between the current flow and its upper bound is less than or equal to a given ϵ [3]. Step S1 achieves an initial feasible solution. The algorithms IDW and CFD are applied here to generate such initial feasible routing trees that the delay constraint for each net (Inequality (2)) is satisfied. It is clear that the generated routes satisfy the timing restriction for each critical path (Inequality (3)). Step S2.1 updates edge distance and recomputes the improved Steiner tree for each net under timing constraint. The improvement of a routing tree is performed by the STI algorithm. For a modified tree of net n, each critical path p including net n is checked to decide whether the path p satisfies the delay restriction $\mathcal{T}(p)$. If not, the modified tree is rejected. Step S2.2 determines whether the solution is ϵ-optimal. If it is, then it breaks from the loop and proceeds on to the integerization. Otherwise, Step S2.3 selects a net to reroute and the amount of flow to be rerouted.

Steps S3 and S4 control the multi-commodity flow integerization. Step S3 integerizes the fractional flow solution. For each net n, the current flow of each tree is regarded as a probability. A tree t is selected according to the probability, and its flow $f(n,t)$ is set to the demand $B(n)$. Based on the result of integerization, Step 4.1 identifies the overflow edges, updates the edge distances, and reroutes the nets which utilize the overflow edges. Step S4 improves the solution until either no further improvement is possible, or the solution becomes valid.

6 Experiment Results

The performance-driven global routing algorithm has been implemented in C language on DEC station 3100 under UNIX. Three ISCAS benchmark circuits, C2, C7, and s13207 have been tested. The three benchmark specifications are listed in Table 1. The global routing graph is similar to the one used in [3, 11]. The placement results and timing information are obtained

from the placement program RITUAL [17] developed at U.C. Berkeley. Those timing information includes the delay requirements at each sink pin of the nets and critical path specifications. The sink delay information is used for net-based timing analysis, while the critical path information is used for path-based timing analysis method. Three timing analysis modes are defined and tested for comparison:

Mode 0 : No timing analysis during the global routing process.

Mode 1 : Net-based timing analysis method.

Mode 2 : Critical path-based timing analysis method.

Table 2 shows the global routing results for gate array (GA). The maximum channel densities under three different timing modes are listed. Table 3 compares the changes in actual critical path delays under three different modes. The maximum, minimum and average percentage of increase of critical path delay under *Mode 0* and *Mode 2* compared with *Mode 1* are reported. The running CPU time for largest benchmark data s13207 under *Mode 2* (α=0.001 and the channel capacity=18) is less than 215 minutes.

Table 1. Benchmark Specification

Circuit	Cells	Nets	Critical Paths	Grid Description	
				Rows	Columns
C2	590	963	8	10	20
C7	2150	2465	2	19	38
s13207	4267	5757	15	25	50

Table 2. Maximum Channel Density

Circuit	Maximum Channel Density		
	Timing Mode 0	Timing Mode 1	Timing Mode 2
C2	5	7	5
C7	9	11	9
s13207	18	19	18

Table 3. Delay on Critical Paths for Gate Array

Circuit	Percentage of Increase of Delay on Critical Paths					
	Timing Mode 0			Timing Mode 2		
	min	max	ave	min	max	ave
C2	20.97	69.01	42.61	-1.31	0.00	-0.52
C7	20.61	77.16	48.89	0.84	1.06	0.95
s13207	30.77	60.18	45.23	-3.29	-0.55	-1.57

Table 4 shows the global routing results for standard cell (SC). The total channel densities under three different timing modes are listed. Table 5 compares the changes in actual critical path delays under three different modes.

Table 4. Total Channel Density

Circuit	Maximum Channel Density		
	Timing Mode 0	Timing Mode 1	Timing Mode 2
C2	40	44	40
C7	152	164	153
s13207	409	420	410

Table 5. Delay on Critical Paths for Standard Cell

Circuit	Percentage of Increase of Delay on Critical Paths					
	Timing Mode 0			Timing Mode 2		
	min	max	ave	min	max	ave
C2	13.48	56.42	38.56	0.21	2.47	1.05
C7	31.39	44.28	37.83	-2.63	0.32	-1.31
s13207	33.33	54.55	38.78	-3.50	1.91	-1.45

Results show that *Mode 0* achieves the smallest maximum channel density (GA) and the smallest total channel density (SC) with large critical path delay because no timing analysis is conducted. *Mode 1* restricts the delay on critical path but results in increment in maximum channel density (GA) and total channel density (SC) because timing analysis is conducted for each rerouted net, so some rerouted trees are rejected unnecessarily. *Mode 2* achieves the same maximum channel density (GA) and very close total channel density (SC) as *Mode 0* while still keeping the critical path delay comparable with *Mode 1*, because only the routing trees for nets on critical paths are analyzed. (Note that routing results of both *Mode 1* and *Mode 2* satisfy the timing constraint for all critical paths.) This demonstrates that the path-based timing analysis method is effective, and leads to smaller chip height than the previously used net-based approach.

7 Conclusions

We have proposed a new performance-driven global routing algorithm which concentrates on critical paths composed of nets in order to satisfy delay restriction for a complete circuit. The experimental results show that our router with critical path delay constraints can obtain the same chip size as the one which ignores timing. Moreover, much smaller chip size is achieved compared with the net-based one while keeping the comparable critical path delay.

References

[1] H.B.Bakoglu, "*Circuits, Interconnections, and Packaging for VLSI*," Reading, MA: Addison-Wesley, 1990.

[2] K.D.Boese, J.Cong, A.B.Kahng, K.S.Leung and D.Zhou, "On high-speed VLSI interconnects: Analysis and design," *Proc. APCCAS-92*, pp.35-40, 1992.

[3] R.C.Carden IV and C.-K.Cheng, "A global router using an efficient approximate multicommodity multiterminal flow algorithm," *Proc. 28th DAC*, pp.316-321, 1991.

[4] J.P.Cohoon and L.J.Randall, "Critical net routing," *Proc. ICCD'91*, pp.174-177, 1991.

[5] J.Cong, A.B.Kahng, G.Robins, M.Sarrafzadeh and C.K.Wong, "Provably good performance-driven global routing," *IEEE Trans. CAD*, 11, 6, pp.739-752, 1992.

[6] W.E.Donath, R.J.Norman, B.K.Agrawal, S.E. Bello, S.Y.Han, J.M.Kurtzberg, P.Lowy and R.I. McMillan, "Timing driven placement using complete path delays," *Proc. 27th DAC*, pp.84-89, 1990.

[7] S.E.Dreyfus and R.A.Wagner, "The Steiner problem in graphs," *Networks*, 1, 3, pp.195-207, 1972.

[8] A.E.Dunlop, V.D.Agrawal, D.N.Deutsh, M.F. Jukl, P.Kozak and M.Wiesel, "Chip layout optimization using critical path weighting," *Proc. 21st DAC*, pp.133-136, 1984.

[9] W.C.Elmore, "The transient response of damped linear networks with particular regard to wide-band amplifiers," *J. Applied Physics*, 19, 1, pp.55-63, 1948.

[10] T.C.Hu. "*Combinatorial Algorithms*," Reading, MA: Addison-Wesley, 1982.

[11] J.Huang, X.-L.Hong, C.-K.Cheng and E.S.Kuh, "An efficient timing-driven global routing algorithm," *Proc. 30th DAC*, pp.596-600, 1993.

[12] X.Hong, T.Xue, E.S.Kuh, C.-K.Cheng and J. Huang, "Performance-driven Steiner tree algorithms for global routing," *Proc. 30th DAC*, pp.177-181, 1993.

[13] M.A.B.Jackson, E.S.Kuh and M.Marek-Sadowska, "Timing-driven routing for building block layout," *Proc. ISCAS*, pp.518-519, 1987.

[14] S.Prasitjutrakul and W.J.Kubitz, "A performance-driven global router for custom VLSI chip design," *IEEE Trans. CAD*, 11, 8, pp.1044-1051, 1992.

[15] J.Rubinstein, P.Penfield,Jr. and M.A.Horowitz, "Signal delay in RC tree networks," *IEEE Trans. CAD*, CAD-2, 3, pp.202-211, 1983.

[16] T.Sakurai, "Approximation of wiring delay in MOSFET LSI," *IEEE J. Solid-State Circuits*, SC-18, 4, pp.418-426, 1983.

[17] A.Srinivasan, K.Chaudhary and E.S.Kuh, "RITUAL: A performance-driven placement algorithm," *IEEE Trans. CAS-II: Analog and Digital Processing*, 39, 11, pp.825-840, 1992.

[18] S.Sutanthavibul and E.Shragowitz, "An adaptive timing-driven layout for high speed VLSI," *Proc. 27th DAC*, pp.90-95, 1990.