GLADE: A Modern Global Router Considering Layer Directives

Yen-Jung Chang, Tsung-Hsien Lee, and Ting-Chi Wang
Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan
jalamorm@gmail.com, tsunghsienlee@gmail.com, tcwang@cs.nthu.edu.tw

***Abstract*—Global routing is a very crucial stage in a design cycle, because it physically plans the routes of nets on a chip. In order to boost the research and development of global routing techniques, ISPD held contests and released benchmarks in 2007 and 2008, respectively. However, the contests may lead researchers away from facing other real problems in practice. In this paper we study a new global routing problem that not only considers traditional routing objectives such as overflow and wirelength but also focuses on honoring layer directives that are usually specified for timing-critical nets to alleviate performance degrading. Based on novel extensions of an academic router, we present a new global router called GLADE for the addressed problem. The experimental results show that GLADE can effectively generate a high-quality solution, which balances the metrics under consideration, for each test case from the set of recently released ICCAD 2009 benchmarks.**

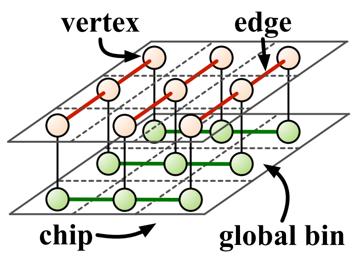

Fig. 1. A multi-layer design with preferred routing directions.

I. Introduction

High performance is one of the ultimate goals that are typically pursued for modern VLSI design. Among all stages in a design cycle, routing is a very important one because it influences the reliability, power consumption, and timing of a chip profoundly. Due to the problem size, routing is usually divided into global routing and detailed routing. A good global routing result can guide a detailed router to obtain a high-quality design. The metrics to measure a global router usually include the total overflow and wirelength of the routing result it generates.

In order to boost the research and development of global routing techniques, International Symposium on Physical Design (ISPD) held contests and released two sets of benchmarks in 2007 [1] and 2008 [2], respectively. Both sets of benchmarks provide multi-layer designs, and therefore all academic global routers that were developed in 2007 or afterward, e.g., [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], are able to generate three-dimensional (3D) routing solutions. Unfortunately, the ISPD contests may lead researchers away from facing other real problems in practice. In industrial designs, the goals of global routing are not only to minimize the total overflow and wirelength, but also to consider the factors that may degrade the quality of a design, e.g., the detours of timing-critical nets that may impair the performance of a chip. Consequently, the problem specifications of the ISPD contests have been questioned. For instance, the via count within a global bin should not be left unbound [13][14], and intra-bin congestion should be considered [15].

In this paper, we study a new global routing problem that not only considers traditional routing objectives such as overflow and wirelength but also focuses on honoring layer directives. Layer directives are often specified for timing-critical nets to meet the performance target in a modern physical design flow. For example, recent wire synthesis algorithms [16][17] use layer ranges for timing closure. A set of benchmarks that incorporate layer directives are modified from ISPD 2008 benchmarks and released in ICCAD last year [18]. Based on novel extensions of NTHU-Route 2.0 [11], we present a new global router called GLADE (standing for Global router for LAyer DirEctives) for the addressed problem. Further, the experimental results show that GLADE can effectively generate a high-quality solution, which balances the metrics under consideration, for each ICCAD 2009 benchmark.

The rest of the paper is organized as follows. We describe the background of global routing, the ICCAD 2009 benchmarks and the objectives of GLADE in Section II. Next, we give a brief review on NTHU-Route 2.0 in Section III. Then, we describe the details of GLADE in Sections IV and V. Finally, the experimental results are presented in Section VI and we conclude the paper in Section VII.

II. Preliminaries

A. Global Routing

For global routing, a multi-layer design can be modeled by a 3D grid graph G which is composed of a set V of vertices and a set E of edges. As depicted in Fig. 1, each global bin and each boundary between two bins correspond to a vertex and an edge, respectively. There is also a set of nets, where each net is composed of a set of pins and each pin corresponds to a vertex. The global routing problem for a net is to find a tree that connects all pins of the net by using edges and vias. Moreover, it is preferable to route each net with only one direction on a layer in order to prevent wires from crossing. Taking Fig. 1 for example, the design separates horizontal and vertical wires on different metal layers.

In a routing graph, the capacity c_e of an edge $e \in E$ represents the number of available routing tracks e contains, and the demand d_e represents the amount of nets that pass through e. For an edge e, if its demand d_e exceeds its capacity c_e, its overflow is defined as $(d_e - c_e)$; otherwise, its overflow is zero. A typical objective for global routing is to minimize the total overflow among all edges. Meanwhile, the total wirelength of all routes should be as short as possible. In multi-layer designs, wirelength calculation also involves vias.

B. ICCAD 2009 Benchmarks

ICCAD 2009 benchmarks [18] provide some representative global routing instances that are faced in industry. Firstly, lower metal layers have more routing tracks than higher metal layers, which shows the fact that wires become thicker and wider on higher metal layers. Secondly, they provide information for considering timing issues of a design. One of them is the layer directives for timing critical nets, which specify the target layer ranges. Each layer range is given by $[sl : el]$, where sl stands for the start layer, el stands for the end layer, and $sl < el$. Furthermore, sl is always an odd layer higher than layer 1, while el is always the highest layer and is an even layer. For a net i with layer range $[sl_i : el_i]$, we define its LD type as $(\lfloor sl_i/2 \rfloor + 1)$. For example, for a 8-layer design with preferred directions, sl could be the 3rd, 5th, or 7th layer, and el is the 8th layer. For a net with layer range $[5 : 8]$, its LD type is 3. Additionally, for a net without a layer directive, its LD type is defined as 1.

In the rest of this paper, a net with a layer directive (i.e., its LD type is larger than 1) is called a LD net. If a routed LD net passes through an edge on an non-preferred layer (a layer located outside the layer range of the net), then this edge induces one unit of LD violation for this net. For example, assuming that in a 6-layer design there is a net whose LD type is 2 (layer range from layer 3 to layer 6), then at least one unit of LD violation occurs if the net passes through layer 1 or layer 2.

C. Objectives

The goal of our global router, GLADE, is to minimize the total LD violation as well as the total overflow (TOF) of a design. Since it is difficult to tell which metric is the better one to measure the quality of a routing solution, GLADE would find a good balance between them. Besides the total LD violation and TOF, the second objective of GLADE is to shorten the total wirelength (TWL) of a design.

The routing method of GLADE is separated into two parts: 2D global routing followed by layer assignment, and each part has a different subgoal. During 2D global routing, the subgoal is to minimize the total LDOF and TOF, and then TWL. LDOF is a metric used by the 2D global routing method to estimate the total LD violation and additional overflow of the 3D routing solution produced after layer assignment. The details of LDOF will be given in Section IV-B. During layer assignment, the subgoal is to generate a 3D routing result which has the identical TOF as the 2D result and has the total LD violation and via count as small as possible.

III. Review on NTHU-Route 2.0

Since our global router GLADE extends NTHU-Route 2.0 [11] to take layer directives into account, we give a brief review on NTHU-Route 2.0 in this section. There are four stages in NTHU-Route 2.0: initial stage, main stage, refinement stage, and layer assignment.

In the initial stage, NTHU-Route 2.0 projects a multi-layer design onto a plane and generates a wirelength-driven 2D solution by FLUTE [19], a probabilistic routing method, an edge shifting technique [20], and L-shaped pattern routing.

In the main stage, NTHU-Route 2.0 improves the initial solution by iteratively ripping up and rerouting every overflowed net to reduce the TOF. Every ripped-up two-pin net is rerouted by monotonic routing [21] or multi-source multi-sink maze routing. In this stage, the cost of each edge e is defined as follows:

$$cost_e = B_e \times GC + H_e \times P_e + V_e \times GC \quad (1)$$

Here B_e is the wirelength of edge e, GC decreases its value moderately from 1 to 0 as the iteration count increases, H_e is the cost for historical overflow of edge e, P_e is the congestion cost of e and is defined as follows:

$$P_e = \left(\frac{d_e + 1}{c_e} \times f\right)^{k_1} \quad (2)$$

where d_e and c_e are the demand and capacity of edge e, k_1 is a user defined parameter, and f is a penalty amplifier which dramatically raises the value of P_e when e is nearly overflowed; V_e is the expected via cost when using edge e causes a bend.

The refinement stage focuses on finding overflow-free paths for all overflowed two-pin nets. It simply rips up and reroutes overflowed nets if they pass through any overflowed edge. The edge cost used in this stage is defined as follows:

$$cost'_e = \begin{cases} 0 & \text{if } e \text{ has been passed through by the same net} \\ 0 & \text{if } e \text{ has free routing tracks} \\ 1 & \text{otherwise} \end{cases}$$

In the layer assignment stage, NTHU-Route 2.0 applies the layer assignment algorithm, COLA [22], to map the 2D routing solution from the projected plane to its original multiple layers. The algorithm can be separated into two parts. First of all, it calculates a score, based on the pin count and wirelength of a net, for each net, and determines a net order according to the scores. It then follows the net order and applies a single-net layer assignment method, that is based a dynamic programming technique, to find the result with minimum via count for each net.

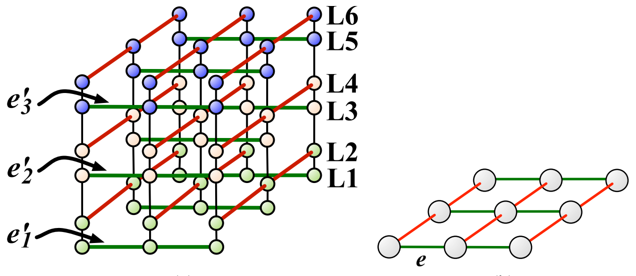

Fig. 2. (a) A 6-layer design with preferred routing directions. (b) The projected 2D routing graph.

IV. GLADE: 2D Global Routing Considering Layer Directives

In this section, we describe the 2D global routing method adopted by GLADE. We explain some necessary concepts in the first two subsections, followed by the details of the 2D global routing method.

A. Pseudo Layer Assignment

During 2D global routing, NTHU-Route 2.0 projects a multi-layer design onto a single plane (2D routing graph) and tries to find a routing tree for each net on the plane. Fig. 2(a) depicts a 3D routing graph for a 6-layer design with preferred routing directions and Fig. 2(b) illustrates the projected 2D plane. For distinguishing edges in the two routing graphs, the edges in Fig. 2(a) are called 3D edges and those in Fig. 2(b) are called 2D edges. Besides, the capacity of each 2D edge e in Fig. 2(b) is the sum of the capacities of the corresponding 3D edges e'_1, e'_2, and e'_3 in Fig. 2(a).

Since the 2D plane used by NTHU-Route 2.0 dose not have layer information, NTHU-Route 2.0 is unable to consider layer directives or predict the 3D routing solution generated by layer assignment when performing 2D global routing. To cope with this problem, GLADE

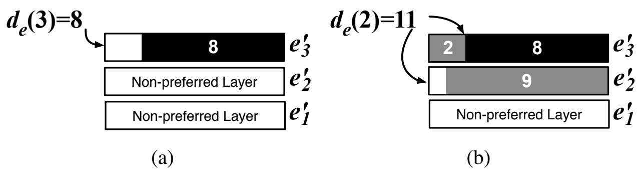

Fig. 3. (a) vc_e(3)=10 and vd_e(3)=8. (b) vc_e(2)=20 and vd_e(2)=19.

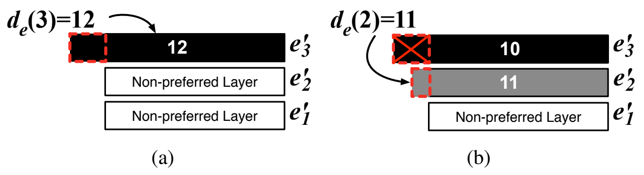

Fig. 4. (a) vc_e(3)=10 and vd_e(3)=12. (b) vc_e(2)=20 and vd_e(2)=21.

performs pseudo layer assignment during 2D global routing. Pseudo layer assignment dose not actually assign nets to 3D edges. It adopts the notions of virtual capacity and virtual demand to achieve the goal instead. In other words, when GLADE tries to find routes for LD nets on the plane, it uses the information on the 3D routing graph to retrieve different values of capacity and demand, which depends on the LD types of the nets, to calculate route costs.

1) Virtual Capacity: For each 2D edge e, a virtual capacity $vc_e(t)$ is defined for each LD type t ($t > 1$), and is calculated by the sum of the capacities of the corresponding 3D edges located on the preferred layers with respect to the LD type t. For example, in Figure 2 if the capacities of the 3D edges e'_1, e'_2, and e'_3 are denoted by $c_{e'_1}$, $c_{e'_2}$, and $c_{e'_3}$, then we have $vc_e(3) = c_{e'_3}$ and $vc_e(2) = c_{e'_3} + c_{e'_2}$. It is clear to see that $vc_e(t)$ specifies the capacity that e can provide for LD nets of type t.

2) Virtual Demand: For each 2D edge e, a virtual demand $vd_e(t)$ can be also defined for each LD type t ($t > 1$). When calculating $vd_e(t)$, GLADE presumes that nets whose LD types are larger than t have a higher priority to be processed during layer assignment, because they have smaller layer ranges than LD nets of type t. Therefore some certain amount of the virtual capacity $vc_e(t)$ may be consumed by those nets. Therefore the virtual demand $vd_e(t)$ will take into account the demands caused by LD nets whose types are larger than t. The $vd_e(t)$ is calculated by $d_e(t) + min(vd_e(t+1), vc_e(t+1))$, where $d_e(t)$ is the amount of LD nets of type t that pass through e. We use Fig. 3 and 4 to help illustrate how to calculate $vd_e(t)$. For both figures, the 3D edges, e'_1, e'_2, and e'_3 each have a capacity of 10 and are projected to the 2D edge e; the amounts of each type of LD nets passing through e are shown in the top-left corners; the gray and black parts indicate the capacities that are virtually consumed by LD nets of types 2 and 3, respectively. Under the assumption that LD nets of type 3 will be assigned first during layer assignment; the hollow parts indicate the remaining capacities. For the example shown in Fig. 3, we have $vc_e(3) = c_{e'_3} = 10$ and $vc_e(2) = c_{e'_3} + c_{e'_2} = 10 + 10 = 20$, $vd_e(3) = d_e(3) = 8$, and $vd_e(2) = d_e(2) + min(vd_e(3), vc_e(3)) = 11 + min(8, 10) = 11 + 8 = 19$. In addition, the capacity of each 3D edge e'_t is assumed to be preserved for LD nets of type t, and therefore the virtual overflow caused by LD nets of type larger than t does not carry over to e'_t. This is why we put the min term in the formula for calculating virtual demand. For the example in Fig. 4, the LD nets of type 3 in Fig. 4(a) are virtually assigned to e'_3 first, which causes two routing tracks short on e'_3. Since the 2 units of overflow do not carry over to layers lower than e'_3, we cross them out in Fig. 4(b). After the LD nets of type 2 are virtually assigned to e'_2, we get the virtual demands for LD type 2, which is 21 ($vd_e(2) = d_e(2) + c_{e'_3} = 11 + 10$).

B. LDOF

During 2D global routing, GLADE needs a guidance which can estimate the possible total LD violation and additional TOF that a 3D solution, when obtained from the current 2D solution by layer assignment, might have. In fact, a LD net may cause LD violation

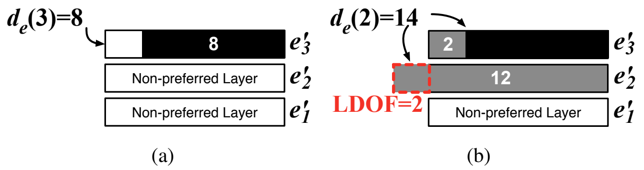

Fig. 5. (a) GLADE assumes that LD nets of type 3 will be assigned first during layer assignment. (b) Then LD nets of type 2 will be assigned afterward.

and/or overflow after layer assignment. Therefore for each 2D edge e, we define a *LDOF* for each LD type t ($t > 1$) , and it is calculated by $max(vd_e(t) - vc_e(t), 0)$. The LDOF of each 2D edge is the sum of the LDOFs among all LD types, and the total LDOF of a 2D global routing solution is the sum of the LDOFs among all 2D edges.

Fig. 5 illustrates an example to calculate LDOF, where each 3D edge e'_t is assumed to have the capacity of 10, and there are 8 and 14 LD nets of types 3 and 2 passing through the 2D edge e, respectively. Firstly, GLADE presumes the 8 LD nets of type 3 will be assign to e'_3 first during layer assignment (see Fig. 5(a)), because they have a smaller layer range than the LD nets of type 2. We get $vc_e(3) = 10, vd_e(3) = 8$, and the LODF of type 3 is $max(vd_e(3) - vc_e(3), 0) = max(8 - 10, 0) = 0$. After then, LD nets of type 2 will be virtually assigned to e'_3 and e'_2 because the two remaining routing tracks of e'_3 can still be used by them (see Fig. 5(b)). We get $vc_e(2) = 10 + 10 = 20$, $vd_e(2) = d_e(2) + vd_e(3) = 14 + 8 = 22$, and the LODF of type 2 is $max(vd_e(3) - vc_e(3), 0) = max(22 - 20, 0) = 2$. As a result, the LDOF of the 2D edge e is 2. Non-zero LDOF implies that additional overflow (if layer directives must be honored) or LD violation (if overflow must be minimized) will occur after layer assignment, regardless of the layers to which the two excess LD nets of type 2 are assigned.

C. The Flow of 2D Global Routing Considering Layer Directives

Now we are ready to explain our 2D global routing method which is based on the first three stages of NTHU-Route 2.0 with proper modifications to handle LD nets. In the initial stage, our method follows NTHU-Route 2.0 and does not perform any LDOF optimization, because the initial stage plays an important role in controlling total overflow and wirelength. Besides, the total LDOF can be effectively eliminated in the main stage and refinement stage.

During the main stage, our method reduces the total LDOF and overflow of the solution obtained in the initial stage iteratively. Firstly, it replaces c_e and d_e by the virtual capacity $vc_e(t)$ and the virtual demand $vd_e(t)$ when routing a LD net of type t. In other words, our method uses $vc_e(t)$ and $vd_e(t)$ in Eq. (2) when calculating the cost of edge e. Also, it uses $vd_e(t)$ and $vc_e(t)$ to determine if edge e is overflowed. Finally, our method calculates the total LDOF and the total overflow at the end of each iteration in order to check if the solution gets converged.

In the refinement stage, our method again replaces c_e and d_e by $vc_e(t)$ and $vd_e(t)$ when routing a LD net of type t. Meanwhile, it determines the cost of each edge e by the following rules in order to completely focus on finding a route which does not cause additional LDOF or overflow:

$$cost''_e = \begin{cases} 0 & \text{if } e \text{ has been passed through by the same net} \\ 0 & \text{if } d_e \leq c_e \text{ (for a non-LD net)} \\ 0 & \text{if } vd_e(t) \leq vc_e(t) \text{ (for a LD net)} \\ 1 & \text{otherwise} \end{cases}$$

V. GLADE: Layer Assignment Considering Layer Directives

The objective in the layer assignment stage of GLADE is to generate a 3D routing solution which respects the TOF from a given 2D routing solution and also minimizes the amount of LD violations. To achieve that, we apply a new net ordering method, that is modified from the layer assignment algorithm COLA [22] of NTHU-Route 2.0, by taking the layer ranges of LD nets into consideration. Following the net order, an amended single-net layer assignment method is performed for each net in order to achieve the goal of this stage. In the following subsections, we will describe the layer assignment algorithm adopted by GLADE in detail.

A. Net Ordering for Considering Layer Directives

COLA [22] guarantees to generate a 3D layer assignment result with identical TOF from a 2D routing solution regardless of the net order if it holds the prevention condition for TOF. The characteristic of COLA provides GLADE the flexibilities to design a net order which can deal with layer directives and also satisfy the objective of preserving the TOF of the given 2D routing solution. Since nets with higher LD types have narrower layer ranges, we should give them a higher priority to choose their preferred layers. Therefore, our net ordering method first sorts all nets by their LD types in an non-increasing order and then uses the original score function of COLA as the tie-breaker.

B. Single-net Layer Assignment for Considering Layer Directives

To hold the prevention condition for TOF, the single-net layer assignment method adopted in COLA applies dynamic programming to find an optimal 3D layer assignment result on via count for each net. For considering layer directives, GLADE extends the method to find the optimal result on LD violation first and via count second. The method adopted by GLADE is similar to the optimal method in COLA: GLADE assigns each possible layer on each edge in a top-down manner and tries all combinations of subtrees to find an optimal layer assignment. It chooses the layers for a LD net by the following rule: If the net is assigned to its preferred layers, the cost of this assignment is the one used in COLA, i.e., via count; otherwise, an additional penalty cost which is much greater than via count will be added into the cost in order to discourage the assignment.

The penalty cost does help assign LD nets to their preferred layers. However, a layer has limited routing tracks. Once their preferred layers are out of routing tracks, an inappropriate layer assignment could induce additional LD violations. Fig. 6 illustrates an example of an inappropriate assignment. Fig. 6(a) shows a 3-layer design where L1 and L2 have one unit of routing resource left and L3 has no routing resource left. Assume that there are two LD nets: net a and net b. Since net a has a larger LD type, GLADE will assign it first. Fig. 6(b) shows that GLADE assigns net a to L2 and causes one unit of LD violation, because L3 is already full. Consequently, net b is forced to be assigned to L1 and causes an additional LD violation due to the inappropriate assignment of net a. However, the inappropriate assignment can be avoided by choosing the lowest available layers for those excess LD nets. Fig. 6(c) shows the solution which has only one unit of LD violation if GLADE first assigns net a to L1 (the lowest available layer) and then assigns net b to L2. Therefore, GLADE chooses the lowest available layer for the excess nets in order to generate the 3D routing results with smaller total LD violation. On the other hand, GLADE adopts the original single-net method in COLA for non-LD nets in order to generate the result with minimum via count.

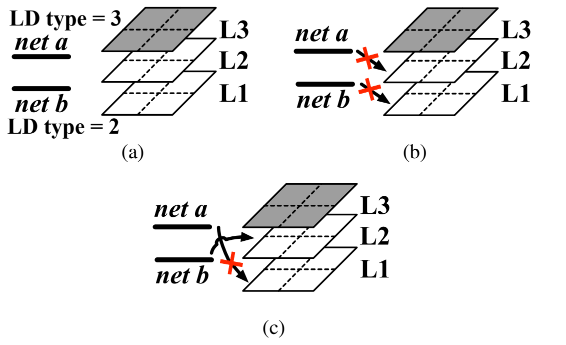

Fig. 6. (a) A 3-layer design with two LD nets and the gray layer has no free routing tracks. (b) A layer assignment result with two units of LD violations. (c) A layer assignment result with one unit of LD violation.

TABLE I
Detailed Information of ICCAD 2009 benchmarks.

Benchmark	#Nets	#LD nets	#Tiles	#Pins	#Layers
adaptec1	219794	970	324×324	942705	6
adaptec2	260159	3756	424×424	1063632	6
adaptec3	466295	2576	774×779	1874576	6
adaptec4	515304	3206	774×779	1911773	6
adaptec5	867441	1511	465×468	3492790	6
bigblue1	282974	1186	227×227	282974	6
bigblue2	576816	47	468×471	2121863	6
bigblue3	1122340	1157	555×557	3832388	8
bigblue4	2228903	2278	403×405	8899095	8
newblue1	331663	152	399×399	1237104	6
newblue2	463213	3380	557×463	1771849	6
newblue4	636195	1610	455×458	2498322	6
newblue5	1257555	487	637×640	4931147	6
newblue6	1286452	428	463×464	5305603	6
newblue7	2635625	4397	488×490	10103725	8

VI. Experimental Results

We implemented GLADE in ANSI C++ and used ICCAD 2009 benchmarks to compare GLADE with NTHU-Route 2.0 [11] in terms of total overflow (TOF), total LD violation due to layer directives (LDVio), total LDOF (LDOF), total wirelength (TWL), via count (VWL), and run time (CPU). The experiments were conducted on a machine with an Intel Core Duo 2.2Ghz CPU and 8GB memory. The detail information of ICCAD 2009 benchmarks is listed in TABLE I[1].

TABLE II shows that GLADE was able to generate zero LD violation for all benchmarks and zero TOF for the benchmarks, except newblue1, that NTHU-Route 2.0 could solve with overflow-free solutions. Although GLADE generated 2 units of TOF on newblue1, we believe the overflow was induced by the constrained solution

[1]Note that benchmark "newblue3" which exists in ISPD 2008 benchmarks is excluded in ICCAD 2009 benchmarks.

TABLE II
COMPARISON BETWEEN NTHU-ROUTE 2.0 AND GLADE ON ICCAD 2009 BENCHMARKS.

	NTHU-Route 2.0					GLADE					
Benchmark	TOF	LDVio	TWL	VWL	CPU	TOF	LDVio	LDOF	TWL	VWL	CPU
adaptec1	0	129440	45.1	9.3	5.3	0	0	0	45.4	9.6	7.0
adaptec2	0	379086	43.1	10.1	1.3	0	0	0	43.9	10.9	1.4
adaptec3	0	450740	114.9	18.7	5.6	0	0	0	115.2	18.9	7.2
adaptec4	0	594576	105.9	16.3	1.7	0	0	0	106.5	17.0	1.8
adaptec5	0	143542	129.8	26.7	15.7	0	0	0	130.1	26.9	15.2
bigblue1	0	121392	47.8	10.3	7.0	0	0	0	48.3	10.7	8.7
bigblue2	0	19428	69.3	20.9	6.0	0	0	0	69.6	20.9	7.8
bigblue3	0	103810	105.7	27.4	3.7	0	0	0	105.9	27.6	3.8
bigblue4	162	152068	178.7	56.8	75.8	188	0	0	178.9	57.0	121.0
newblue1	0	12508	35.6	11.3	3.9	2	0	0	35.6	11.3	4.8
newblue2	0	98962	59.4	12.2	0.9	0	0	0	59.7	13.4	0.8
newblue4	138	100680	108.3	24.9	65.4	140	0	0	108.1	24.9	40.1
newblue5	0	21288	190.7	43.5	12.8	0	0	0	190.7	43.5	12.6
newblue6	0	58506	139.8	37.5	10.4	0	0	0	139.8	37.6	11.5
newblue7	62	257626	279.8	92.5	57.8	78	0	0	281.7	93.3	119.9
ratio	-	-	1.000	1.000	1.000	-	-	-	1.005	1.022	1.185

[a]The benchmark “newblue3” which exists in ISPD 2008 benchmarks is excluded in ICCAD 2009 benchmarks.

[b]Via is set to be 1 unit of wirelength in ICCAD 2009 benchmarks.

[c]Run time is given in minutes.

space due to layer directives. Meanwhile, the raises of TOF on bigblue4, newblue4, and newblue7 could be due to the same reason. Further, the zero LDOF and LD violation show that LDOF is a good guidance for considering layer directives during 2D global routing and also show the effectiveness of our layer assignment algorithm. As can be seen from the table, the increases of TWL were mainly dominated by the increases of via count, which shows the fact that LD nets are asked to use higher layers and hence require more vias. Finally, since GLADE has to improve the solution quality for the conventional metrics and honor the layer directives at the same time, its run time was raised by 18.5% on average. However, the run time is much faster than performing full 3D global routing.

VII. CONCLUSION

In this paper, we have presented GLADE, a global router for considering layer directives, that is based on extensions of NTHU-Route 2.0, one of the state-of-the-art global routers. The experimental results show that GLADE successfully eliminated the violations due to layer directives for all ICCAD 2009 benchmarks. Meanwhile, it can still rival with NTHU-Route 2.0 in total overflow, wirelength, and run time.

A future work for considering layer directives is to devise another 2D and 3D routing methods for general layer range rules which are different from the one in ICCAD 2009 benchmarks. In ICCAD 2009 benchmarks, all the end layers of layer ranges end at the highest layer of a design. However, the end layer of a general layer range rule may not be the highest layer in real industry designs. In other words, two layer ranges may *overlap* and cause this problem to become more challenging. Therefore, another routing algorithm is needed for considering the general problem in practice.

VIII. ACKNOWLEDGMENTS

This work was supported in part by National Science Council of Taiwan under Grants NSC-98-2220-E-007-011, NSC-98-2220-E-007-032, and NSC-99-2221-E-007-113-MY3.

REFERENCES

[1] ISPD 2007 Global Routing Contest. [Online]. Available: http://www.sigda.org/ispd2007/rcontest/

[2] ISPD 2008 Global Routing Contest. [Online]. Available: http://www.sigda.org/ispd2008/contests/ispd08rc.html

[3] M. M. Ozdal and M. D. F. Wong, “Archer: a history-driven global routing algorithm,” in *Proceedings of International Conference on Computer-Aided Design*, San Jose, CA, 2007, pp. 488–495.

[4] M. Cho, K. Lu, K. Yuan, and D. Z. Pan, “Boxrouter 2.0: architecture and implementation of a hybrid and robust global router,” in *Proceedings of International Conference on Computer-Aided Design*, San Jose, CA, 2007, pp. 503–508.

[5] Y. Zhang, Y. Xu, and C. Chu, “Fastroute 3.0: A fast and high quality global router based on virtual capacity,” in *Proceedings of International Conference on Computer-Aided Design*, San Jose, CA, 2008, pp. 344–349.

[6] Y. Xu, Y. Zhang, and C. Chu, “Fastroute 4.0: Global router with efficient via minimization,” in *Proceedings of Asia and South Pacific Design Automation Conference*, Yokohama, Japan, 2009, pp. 576–581.

[7] J. A. Roy and I. L. Markov, “High-performance routing at the nanometer scale,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 6, pp. 1066–1077, 2008.

[8] K.-R. Dai, W.-H. Liu, and Y.-L. Li, “Efficient simulated evolution based rerouting and congestion-relaxed layer assignment on 3-d global routing,” in *Proceedings of Asia and South Pacific Design Automation Conference*, Yokohama, Japan, 2009, pp. 570–575.

[9] H.-Y. Chen, C.-H. Hsu, and Y.-W. Chang, “High-performance global routing with fast overflow reduction,” in *Proceedings of Asia and South Pacific Design Automation Conference*, Yokohama, Japan, 2009, pp. 582–587.

[10] J.-R. Gao, P.-C. Wu, and T.-C. Wang, “A new global router for modern designs,” in *Proceedings of Asia and South Pacific Design Automation Conference*, Seoul, Korea, 2008, pp. 232–237.

[11] Y.-J. Chang, Y.-T. Lee, and T.-C. Wang, “NTHU-Route 2.0: A fast and stable global router,” in *Proceedings of International Conference on Computer-Aided Design*, CA, USA, 2008, pp. 338–343.

[12] T.-H. Wu, A. Davoodi, and J. T. Linderoth, “GRIP: scalable 3d global routing using integer programming,” in *Proceedings of Design Automation Conference*, San Francisco, CA, 2009, pp. 320–325.

[13] C.-H. Hsu, H.-Y. Chen, and Y.-W. Chang, “Multi-layer global routing considering via and wire capacities,” in *Proceedings of International Conference on Computer-Aided Design*, CA, USA, 2008, pp. 350–355.

[14] T.-H. Lee and T.-C. Wang, “Robust layer assignment for via optimization in multi-layer global routing,” in *Proceedings of International Symposium on Physical Design*, CA, USA, 2009, pp. 159–166.

[15] W. Swartz, “Issues in global routing,” in *Proceedings of International Symposium on Physical Design*, Portland, USA, 2008, pp. 142–147.

[16] C. J. Alpert, C. Chu, and P. G. Villarrubia, “The coming of age of physical synthesis,” in *Proceedings of International Conference on Computer-Aided Design*, San Jose, CA, 2007, pp. 246–249.

[17] Z. Li, C. J. Alpert, S. Hu, T. Muhmud, S. T. Quay, and P. G. Villarrubia, “Fast interconnect synthesis with layer assignment,” in *Proceedings of International Symposium on Physical Design*, Portland, OR, 2008, pp. 71–77.

[18] M. D. Moffitt, “Global routing revisited,” in *Proceedings of International Conference on Computer-Aided Design*, CA, USA, 2009, pp. 805–808.

[19] C. Chu and Y.-C. Wong, “FLUTE: Fast lookup table based rectilinear steiner minimal tree algorithm for VLSI design,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 1, pp. 70–83, 2008.

[20] M. Pan and C. Chu, “Fastroute: a step to integrate global routing into placement,” in *Proceedings of International Conference on Computer-Aided Design*, San Jose, CA, 2006, pp. 464–471.

[21] ——, “Fastroute 2.0: A high-quality and efficient global router,” in *Proceedings of Asia and South Pacific Design Automation Conference*, Yokohama, Japan, 2007, pp. 250–255.

[22] T.-H. Lee and T.-C. Wang, “Congestion-constrained layer assignment for via minimization in global routing,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 9, pp. 1643–1656, 2008.