# A Novel Framework for Multilevel Full-Chip Gridless Routing

Area 1.2: Routing

## Abstract

Due to its great flexibility, gridless routing is desirable for nanometer circuit designs that use variable wire widths and spacings. Nevertheless, it is much more difficult than grid-based routing because of its larger solution space. In this paper, we present a novel $\Lambda$-shaped multilevel framework (LMF for short) that can better handle global circuit effects than the traditional $V$-shaped one. Unlike the traditional "V-shaped" multilevel framework, LMF works in the $\Lambda$-shaped manner: top-down uncoarsening followed by bottom-up coarsening. Based on the novel framework, we develop a multilevel full-chip *gridless* router (LMGR for short) for large-scale circuit designs. The top-down uncoarsening stage of LMGR starts from the coarsest regions and then processes down to finest ones level by level; at each level, it performs global pattern routing and detailed routing for local nets and then estimate the routing resource for the next level. Then, the bottom-up coarsening stage performs global maze routing and detailed routing to reroute failed connections and refine the solution level by level from the finest level to the coarsest one. We employs a dynamic congestion map to guide the global routing at all stage and propose a new cost function for congestion control. Experimental results show that our $\Lambda$-shaped multilevel gridless router achieves the best routability among all published gridless routers based on a set of commonly used MCNC benchmarks. Besides, LMGR can obtain significantly less wirelength and smaller critical path delay than the previous works. In particular, the $\Lambda$-shaped multilevel framework is general and thus can readily apply to other problems.

## 1 Introduction

Research in VLSI routing has received much attention in the literature. Routing is typically a very complex combinatorial problem. In order to make it manageable, the routing problem is usually solved using the two-stage approach of global routing followed by detailed routing. Global routing first partitions the routing area into tiles and decides tile-to-tile paths for all nets while detailed routing assigns actual tracks and vias for nets. Many routing algorithms adopt a flat framework of finding paths for all nets. Those algorithms can be classified into sequential and concurrent approaches. Early sequential routing algorithms include maze-searching approaches [21, 28] and line-searching approaches [16], which route net-by-net. Most concurrent algorithms apply network-flow or linear-assignment formulation [1, 27] to route a set of nets at one time.

The major problem of the flat frameworks lies in their scalability for handling larger designs. As technology advances, technology nodes are getting smaller and circuit sizes are getting larger. To cope with the increasing complexity, researchers proposed to use hierarchical approaches to handle the problem: Marek-Sadowska proposed a hierarchical global router based on linear assignment [26]; Heisterman and Lengauer presented a hierarchical integer linear programming approach for global routing [15]; Wang and Kuh proposed a hierarchical ($\alpha$, $\beta$)* algorithm for timing-driven multilayer MCM/IC routing [29]; Chang, Zhu, and Wong applied linear assignment to develop a hierarchical, concurrent global and detailed router for FPGA's [4].

The two-level, hierarchical routing framework, however, is still limited in handling the dramatically growing complexity in current and future IC designs which may contain hundreds of millions of gates in a single chip. As pointed out in [10], for a 0.07 $\mu m$ process technology, a 2.5 $\times$ 2.5 $cm^2$ chip may contain over 360,000 horizontal and vertical routing tracks. To handle such high design complexity, the two-level, hierarchical approach becomes insufficient. Therefore, it is desired to employ more levels of routing for larger IC designs.

### 1.1 Multilevel Framework

The multilevel framework has attracted much attention in the literature recently. It employs a two-stage technique: coarsening followed by uncoarsening. The coarsening stage iteratively groups a set of circuit components (e.g., circuit nodes, cells, modules, routing tiles, etc) based on a predefined cost metric until the number of components being considered is smaller than a threshold. Then, the uncoarsening stage iteratively ungroups a set of previously clustered circuit components and refines the solution by using a combinatorial optimization technique (e.g., simulated annealing, local refinement, etc). The multilevel framework has been successfully applied to VLSI physical design. For example, the famous multilevel partitioners, *ML* [2], *hMETIS* [19], and *HPM* [9], the multilevel placer, *mPL* [3], and the multilevel floorplanner/placer, *MB*-tree* [22], all show the promise of the multilevel framework for large-scale circuit partitioning, placement, and floorplanning.

A framework similar to multilevel routing was presented in [14, 23, 25]. Lin, Hsu, and Tsai in [23] and Hayashi and Tsukiyama in [14] presented hybrid hierarchical *global* routers for multi-layer VLSI's [14], in which both bottom-up (coarsening) and top-down (uncoarsening) techniques were used for global routing. Marek-Sadowska [25] proposed a global router based on the outer-most loop approach. The approach is similar to the coarsening stage of multilevel routing. Cong, Fang, and Zhang proposed a pioneering routability-driven multilevel global-routing approach for large-scale, full-chip routing [10]. Cong et al. later proposed an enhanced multilevel routing system, named MARS, which incorporates resource reservation, a graph-based Steiner tree heuristic, and a history-based multi-iteration scheme to improve the quality of the multilevel routing algorithm in [11, 12]. Their final results of the multilevel global routing are tile-to-tile paths for all nets. The results are then fed into a non-multilevel gridless detailed router, called DUNE [8], to find the exact connection for each net. (Therefore, MARS is in fact a multilevel *global* router, but not a *detailed* router.) Lin and Chang also proposed a multilevel approach for full-chip, *grid-based* routing, which considers both routability and performance [5, 24]. This framework integrates grid-based global routing, detailed routing, and resource estimation together at each level, leading to more accurate routing resource estimation during coarsening and thus facilitating the solution refinement during uncoarsening. Their experimental results show the best routability among the previous works for grid-based routing. Recently, Ho et al. in [17] and [18] presented another multilevel framework for full-chip, grid-based routing considering crosstalk and antenna effects, respectively. The framework incorporates an intermediate stage of layer/track assignment between the coarsening stage and the uncoarsening stage. The coarsening stage performs only global routing while global routing and detailed routing are integrated together at the uncoarsening stage.

## 1.2 Gridless Routing

Most of the previous routing algorithms are grid-based, assuming uniform wire/via sizes. However, the grid-based approach is not effective to handle modern routing problems with nanometer electrical effects, such as optical proximity correction (OPC) and phase-shift mask (PSM). To cope with these nanometer electrical effects, we need to consider designs of variable wire/via widths and spacings, for which gridless routers are desirable due to their great flexibility. The gridless routing, however, is much more difficult than the grid-based routing because the solution space of gridless routing is significantly larger than that of grid-based routing. Cong et al. in [8] proposed a three-level routing scheme with a wire-planning phase between the global routing and the detailed routing. However, for large-scale designs, even with the three-level routing system, the problem size at each level may still be very large. Therefore, as the designs grow, more levels of routing are needed [12]. Recently, Chen and Chang proposed an OPC-aware multilevel gridless router [6], which integrates gridless global and detailed routing at each level. Their router can handle non-uniform wire widths and reduce OPC pattern feature requirements.

## 1.3 Our Contributions

In this paper, we present a novel $\Lambda$-shaped multilevel framework (LMF for short) that can better handle global circuit effects than the traditional $V$-shaped multilevel framework (VMF for short). VMF applies a two-stage technique of bottom-up coarsening followed by top-down uncoarsening. As an example adopted in [6] and shown in Figure 1(a), the coarsening stage of VMF uses a detailed router to route a local connection according to the tile-to-tile path found by a global router. Its uncoarsening stage considers the failed connections during the coarsening stage, and rip-up and re-route are performed to refine the routing solution. Experimental results showed that VMF obtains solutions of very high completion rates. Since VMF works in a bottom-up manner by processing smaller (finer) regions first, shorter local nets are routed earlier than longer global nets, and it is very likely that the shorter local nets might become routing blockages that hurt the later routing of global nets. Therefore, it is obvious that VMF is limited in handling global circuit effects such as critical path optimization, which is much more important for modern nanometer circuit designs.

Unlike VMF that adopts the V-shaped framework, LMF works in the $\Lambda$-shaped manner: top-down uncoarsening followed by bottom-up coarsening. (See Figure 1(b) for an illustration of LMF.) Based on LMF, we develop a $\Lambda$-shaped multilevel full-chip *gridless* router (LMGR for short) for large-scale circuit designs. The top-down uncoarsening stage of LMGR starts from the coarsest regions and then processes down to finest ones level by level; at each level, it performs global pattern routing and detailed routing for local nets and then estimate the routing resource for the next level. Then, the bottom-up coarsening stage performs global maze routing and detailed routing to reroute failed connections and refine the solution level by level from the finest level to the coarsest one.

LMF outperforms VMF in the optimization of global interconnect effects (such as wirelength, timing, and crosstalk optimization), since LMF considers the global configuration first and then processes down to local ones level by level, and thus the global effects can be handled at earlier stages. In particular, LMF is general and thus can readily apply to other problems.

In addition to the aforementioned characteristics, our LMF-based LMGR has the following distinguished features:

- The previous works [5, 17, 18, 24] are gird-based multilevel router, which cannot handle designs of variable wire/via widths and spacings. Thus, they cannot effectively handle modern routing problems with nanometer electrical effects such as OPC.

- The previous works [10, 11, 12, 15, 23] are mainly for global routing while LMGR integrates global and detailed routing.

- LMF considers the global longer nets first at the earlier uncoarsening stage, leading to better control on critical path delay and global interconnect effects.

- The previous works [5, 6, 24] perform *greedy global routing*, which determines the global path of the current net without considering the routing resource of succeeding nets. In contrast, LMGR employs a congestion map to guide the global routing at all stage. Initially, the map keeps the preliminary estimation of routing congestion based on the pin distribution. After routing a net, the map is updated dynamically based on the real route, previously routed nets, and estimated unrouted nets. As routing proceeds, we keep more and more accurate congestion information in the map. Therefore, we have better congestion control throughout the whole routing process.

- We use a new cost function based on *both* the total path congestion and the maximum channel congestion for global routing. The cost function obtains better solutions than those consider only total path congestion or the maximum channel congestion.

- LMGR has higher flexibility and keeps more global views, and thus more routing objectives (such as crosstalk and OPC) can be more easily considered in LMGR since exact track and wiring information at each level after detailed routing is known.

Table 1 compares the existing multilevel routing frameworks among [5, 24], [6], [10, 11, 12], [17, 18], and LMF.

Experimental results show that our LMGR achieves the best routability among all published gridless routers [6, 12] based on a set of commonly used MCNC benchmarks with non-uniform and uniform wire widths. Besides, LMGR can obtain significantly less wirelength and smaller critical path delay than the previous works [5, 24].

The rest of this paper is organized as follows. Section 2 presents the global, detailed, and $\Lambda$-shaped multilevel routing models. Section 3 presents our $\Lambda$-shaped multilevel routing framework . Experimental results are reported in Section 4. Finally, we give concluding remarks in Section 5.

| Work | Category of routing | Framework | Characteristics |
|---|---|---|---|
| Ours | • Gridless global and detailed routing | • Use Λ-shaped multilevel framework.<br>• Before uncoarsening: channel density initialization.<br>• Uncoarsening: GR+DR+RE.<br>• Coarsening: global and detailed maze refinement. | • Perform global and detailed routing at each level.<br>• Handle longer nets first and thus the wirelength and the critical path are reduced. |
| Chang et al. in [5, 24] | • Grid-based global and detailed routing | • Use V-shaped multilevel framework.<br>• Coarsening: GR+DR+RE.<br>• Uncoarsening: global and detailed maze refinement. | • Perform global and detailed routing at each level.<br>• Lack initial global routing. |
| Chen et al. in [6] | • Gridless global and detailed routing | • Use V-shaped multilevel framework.<br>• Coarsening: GR+DR+RE.<br>• Uncoarsening: global and detailed maze refinement. | • Perform global and detailed routing at each level.<br>• Lack initial global routing. |
| Cong et al. in [10, 11, 12] | • Gridless global routing + flat gridless detailed routing | • Use V-shaped multilevel framework.<br>• Coarsening: RE.<br>• Intermediate stage: multicommodity flow.<br>• Uncoarsening: global maze refinement. | • Perform global and detailed routing separately. |
| Ho et al. in [17, 18] | • Grid-based global and detailed routing | • Use V-shaped multilevel framework.<br>• Coarsening: GR+RE.<br>• Intermediate stage: track/layer assignment.<br>• Uncoarsening: global and detailed maze refinement. | • Perform global and detailed routing separately. |

Table 1: Multilevel framework comparisons among [5, 24], [6], [10, 11, 12], [17, 18], and LMGR. GR, DR, and RE denote global routing, detailed routing, and resource estimation, respectively.

## 2 Preliminaries

### 2.1 Modeling of Global Routing

Routing in modern IC's is a very complex process, and we can hardly obtain solutions directly. Our global routing algorithm is based on a graph search technique guided by the congestion associated with routing regions and topologies. The router assigns higher costs to route nets through congested areas to balance the net distribution among routing regions.

Before we can apply the graph search technique to multilevel routing, we first need to model the routing resource as a graph such that the graph topology can represent the chip structure. Figure 2 illustrates the graph modeling. For the modeling, we first

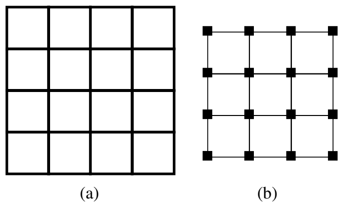

Figure 2: Modeling of global routing: (a) Partitioned layout; (b) Routing graph.

partition a chip into an array of rectangular subregions. These subregions are called *global routing cells (GRCs)*. A node in the graph represents a *GRC* in the chip, and an edge denotes the boundary between two adjacent *GRCs*. Each edge is assigned a capacity according to the width/height of a *GRC*. The graph is used to represent the routing area and is called a *multilevel routing graph*, denotes by $G_k$, where $k$ is the level ID. A global router finds *GRC*-to-*GRC* paths for all nets on $G_0$ to guide the detailed router. The goal of global routing is to route as many nets as possible while meeting the capacity constraint of each edge and any other constraint, if specified. Note that, because of the gridless nature of our routing problem, the cost of routing a net is associated with the wire width and spacing.

As the process technology advances, multiple routing layers are possible. The number of layers in a modern chip can be more than six [13]. Wires in each layer run either horizontally or vertically. We refer to the layer as a horizontal (H) or a vertical (V) routing layer.

### 2.2 Modeling of Detailed Routing

In the detailed routing stage, seeking a design-rule-correct path between two given points in the routing region is our major concern. At first, for each obstacle (a pre-routed wire or a pin), we extend a range which is the sum of the obstacle (wire/via) spacing and a half of the width of the routing wire. As shown in Figure 3(a), the extended range is the sum of $D_S$ and $W_j/2$, where $D_S$ and $W_j$ are the design rules of wire/via spacing and the width of the routing wire, respectively. According to the boundaries of all extended regions and the center of the obstacle, we get three $x$-coordinates and three $y$-coordinates for each extended region. We store all $x$-coordinates and $y$-coordinates of all extended regions and all centers of all obstacles into two sorted array, $CG_x$ and $CG_y$, separately. Based on $CG_x$ and $CG_y$, we can construct a *connection graph* (*CG*). A node in a *CG* denotes that it is an intersection of one $x$-coordinate in $CG_x$ and one $y$-coordinate in $CG_y$. A node in a *CG* also denotes that it is a feasible point for routing the wire. As shown in Figure 3(a), the black circles in the intersection are the feasible points for routing the wire. Therefore, to seek for a design-rule-correct path from the source, $P_S$, to the target, $P_T$, we just need to search all feasible points and find if a successful path exists. As shown in Figure 3(b), a design-rule-correct path from $P_S$ to $P_T$ is found through the five feasible points shown on the $P_S$-$P_T$ path.

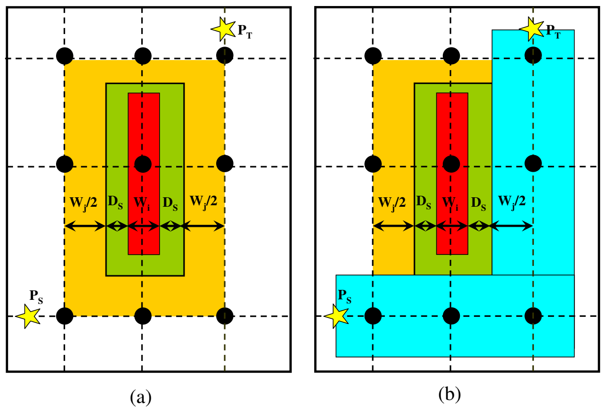

Figure 3: (a) We build the *CG* and locate the feasible points by extending a range which is the sum of the obstacle (wire/via) spacing and a half of the width of the routing wire. $D_S$, $W_i$, and $W_j$ are the design rule of wire/via spacing, the width of the pre-routed wire, and the width of the routing wire, respectively; (b) A design-rule-correct path from the source, $P_S$, to the target, $P_T$, is found through the five feasible points shown on the $P_S$-$P_T$ path.

### 2.3 Modeling of Λ-shaped Multilevel Routing

As illustrated in Figure 1(b), $G_0$ corresponds to the routing graph of the level 0 of the multilevel uncoarsening stage. Be-

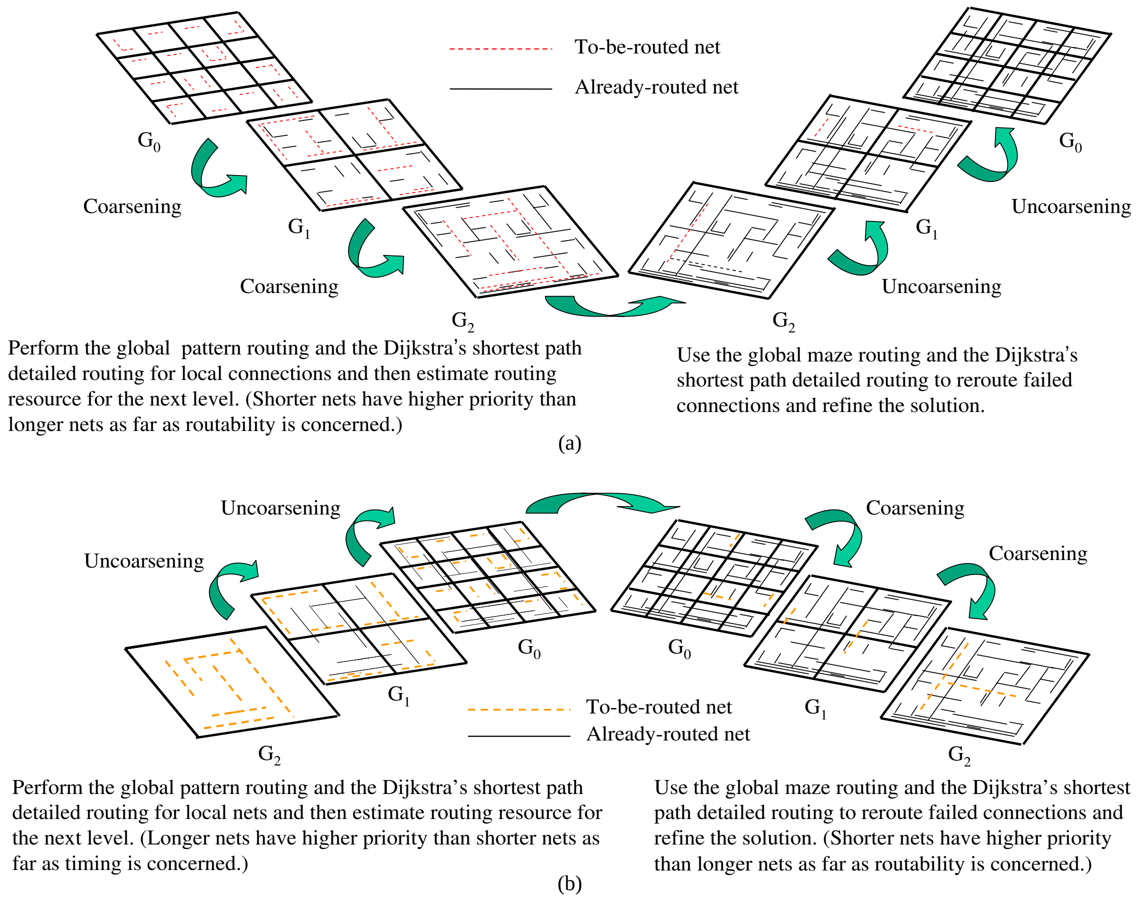
To-be-routed net
Already-routed net
G0
G1
G2
Coarsening
Coarsening
Uncoarsening
Uncoarsening
Perform the global pattern routing and the Dijkstra's shortest path detailed routing for local connections and then estimate routing resource for the next level. (Shorter nets have higher priority than longer nets as far as routability is concerned.)
Use the global maze routing and the Dijkstra's shortest path detailed routing to reroute failed connections and refine the solution.
(a)
Uncoarsening
Uncoarsening
Coarsening
Coarsening
To-be-routed net
Already-routed net
Perform the global pattern routing and the Dijkstra's shortest path detailed routing for local nets and then estimate routing resource for the next level. (Longer nets have higher priority than shorter nets as far as timing is concerned.)
Use the global maze routing and the Dijkstra's shortest path detailed routing to reroute failed connections and refine the solution. (Shorter nets have higher priority than longer nets as far as routability is concerned.)
(b)

Figure 1: (a) The V-shaped multilevel framework flow; (b) The $\Lambda$-shaped multilevel framework flow.

fore the uncoarsening stage is performed, we need to determine the number of levels and build $GRCs$ for each level. For each level $i$, we merge four $GRC_i$ of $G_i$ into a larger $GRC_{i+1}$. The process continues until the number of $GRCs$ at level $k$ is equal to one. After determining the number of levels, we start with the uncoarsening stage from the $k$-th level. At each level $i$, our global router just finds routing paths for the *local nets* (a local net at level $i$ denotes that all pins of the net can be included entirely by a $GRC_i$ and cannot be included totally by a $GRC_{i-1}$), and then the detailed router is used to determine the exact wiring. After the global and detailed routing are performed, we expand each $GRC_i$ to four finer $GRC_{i-1}$ and at the same time perform resource estimation. The uncoarsening stage continues until the 0-th level is arrived. After finishing the uncoarsening stage, the coarsening stage tries to refine the routing solution starting from the level 0. During the coarsening stage, the unroutable connections during the uncoarsening stage are considered, and point-to-path maze routing and rip-up and re-route are performed to refine the routing solution. Then we proceed to the next level (i.e., level 1 here) of the uncoarsening stage by merging four adjacent $GRC_0$ into a larger $GRC_1$. The process continues until we go back to level $k$ when the final routing solution is obtained.

## 3 $\Lambda$-shaped Multilevel Routing Framework

LMGR tends to route wider nets first since a wider net consumes more routing resource. Besides, LMGR tends to route longer nets first at the uncoarsening stage. It is obvious that the local nets at the higher level (say, level $k$) are usually longer than those at a lower level (say, level 0). Usually, a longer net has larger path delay. Thus, this observation implicitly suggests that a longer net has a higher priority than a shorter net as far as timing is concerned. Thought this net ordering scheme may not be the optimal solution for some routing problems (for example, when routability is considered, routing shorter nets first often leads to a better completion rate), it is still a better alternative to the optimization of global interconnect effects.

### 3.1 Channel Density Initialization and Update

If global routing, detailed routing, and resource estimation are performed separately, the re-routing process conducted at the global routing stage may be in vain since it does not know if the re-routing is useful for the detailed routing. Also, the detailed router may fail to find a path because of the low flexibility induced from the separated global routing. Therefore, making the three tasks interact with each other can significantly improve routing quality [5, 24]. However, the concept can only guide the latter nets passing through the area with lower congestion and cannot avoid a wrong decision made by *greedy global routing* which determines the global path of an early routed net without considering the routing resource of succeeding nets. Therefore, we initialize the routing congestion information based on the pin distribution and the global-path prediction of all nets, and then keep a congestion map that is updated dynamically based on both the already routed nets and the estimated unrouted nets. As routing proceeds, we keep more and more accurate congestion information in the map. Therefore, we have better congestion control throughout the whole routing process.

For a 2-pin connection $c$, we use L- and Z-shaped routes to determine the number of possible global routes $n_c$. We evenly distribute the wire density of the connection $c$, $w_c$, among all possible global routes. Therefore, the wire density of each possible global route is $w_c/n_c$. For each possible global route, we add the wire density of the possible global route to the channel density in the routing graph. For example, as shown in Figure 4(a), the connection $c$ (from the source S to the target T) has 2 L-shaped routes (S→a→b→c→g→T and S→d→h→i→j→T) and 3 Z-shaped routes (S→a→e→i→j→T, S→a→b→f→j→T, and S→d→e→f→g→T), implying that the wire density of each possible global route is $w_c$/5. Therefore, according to the number of possible global routes, we add the values which are shown in Figure 4(b) to the channel density in the routing graph between $S$ and $T$. After all 2-pin connections finish the process, we get an initial channel density. Note that the aforementioned approach is a natural way to estimate routing congestion, commonly used for interconnect-driven floorplanning.

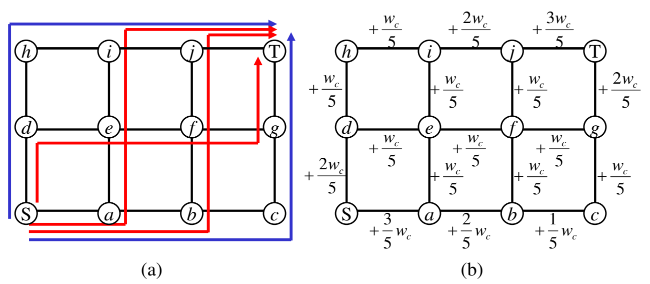

Figure 4: (a) Possible global routes: 2 L-shaped routes and 3 Z-shaped routes; (b) The values should be added to the channel density in the routing graph between $S$ and $T$.

At first, the channel density is totally estimated by the approach. After a connection has been routed successfully, the estimated cost induced by the connection will be removed from the channel density, and the wire density of the real path will be updated to the channel density (congestion map) dynamically. Therefore, our congestion control is based on congestion information induced by both the already routed nets and the estimated unrouted nets. As routing proceeds, we have more and more accurate congestion information for routing succeeding nets.

## 3.2 Cost Function for Global Routing

Let the multilevel routing graph be $G_0 = (V_0, E_0)$. Let $R_e = \{\, e \in E_0 \mid e$ is the edge chosen for routing$\}$. We apply the cost function $\alpha : E_0 \rightarrow \Re$ to guide the global routing:

$$\alpha(R_e) = \max_{e \in R_e} c_e + \frac{1}{|R_e|} \sum_{e \in R_e} c_e, \qquad (1)$$

where $c_e$ is the congestion of edge $e$ and is defined by

$$c_e = \frac{d_e}{p_e},$$

where $p_e$ and $d_e$ are the capacity and channel density associated with $e$, respectively. We measure the routing congestion based on the *channel density* defined by the sum of wire spacing and wire width for gridless routing. (Note that the definition is different from the case in grid-based routing, for which channel density is defined as the maximum number of parallel nets passing through a routing channel.)

There are two advantages by using this cost function for global routing. First, this cost function can avoid that we select a path which has lower total path congestion with a higher channel congestion. Second, this cost function can prevent us from choosing a worse global path with the higher overall path congestion when two global paths have the same maximum channel congestion. For example, as shown in Figure 5, the channel congestion between the source (S) and the target (T) is shown on it corresponding edge. If we determine the global path by the total path congestion, we will select the path $P_1$ (S→a→b→e→T), which has the minimal total path congestion, 1.5. However, this path has a very high channel congestion, 0.9. If we determine the global path by the maximum channel congestion, paths $P_2$ (S→c→d→e→T) and $P_3$ (S→c→f→g→T) have the same cost, 0.7. Selecting a path between $P_2$ and $P_3$ arbitrarily without considering other information seems not a good idea, since $P_3$ is a better solution with lower total path congestion. Therefore, since our cost function considers both the total path congestion and the maximum channel congestion, we will select the better solution $P_3$ with the minimum cost, 1.1.

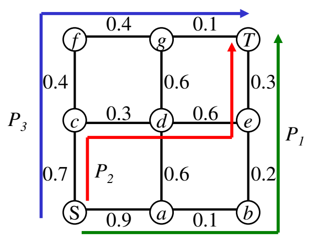

Figure 5: The global paths determined by different cost functions. $P_1$: total path congestion; $P_2$: maximum channel congestion; $P_3$: maximum channel congestion + average total path congestion.

## 3.3 Λ-shaped Multilevel Gridless Routing

In the following, we present our framework for LMGR and summarize it in Figure 6.

```
Algorithm:  Λ-shaped-Multilevel-Gridless-Routing($G$, $N$)
    Input:  $G$ - partitioned layout;
            $N$ - netlist of multi-terminal nets;
   Output:  routing solutions for $N$ on $G$
begin
1   Partition layout;
2   For each net $n \in N$
3       Construct an MST;
4       Decompose the MST into 2-pin connections;
5       For each 2-pin connection
6           Initialize channel density;
7   // Uncoarsening Stage
8   For each level at the uncoarsening stage
9       Choose a local net $n$;
10          For each connection $c \in n$;
11              Perform global pattern routing;
12              Perform detailed routing;
13              Update channel density;
14  // Coarsening Stage
15  For each level at the coarsening stage
16      Choose a failed connection at the uncoarsening stage
17          Perform global maze routing;
18          Perform detailed routing;
19          Update channel density;
20  Analyze timing for all nets;
21  return the routing layout;
end
```

Figure 6: Algorithm for Λ-shaped multilevel gridless routing.

Given a netlist, we first run a minimum spanning tree (MST) algorithm to construct the topology for each net, and then decompose each net into 2-pin connections, with each connection corresponding to an edge of the MST. According to those 2-pin connections, we use the heuristic in Section 3.1 to initialize the channel density in the routing graph by predicting the global paths of all nets in advance.

LMGR starts from uncoarsening the coarsenest tile of level $k$. At each level, tiles are processed one by one, and only local nets are routed. At each level, the two-stage routing approach of global routing followed by detailed routing is applied. The global routing is based on the approach used in the Pattern Router [20] and first routes local nets on the tiles of level k. Let the multilevel routing graph of level $i$ be $G_i = (V_i, E_i)$. Let $R_e = \{ e \in E_i \mid e$ is the edge chosen for routing$\}$. We apply the cost function in Section 3.2 to guide the routing.

After the global routing is completed, LMGR performs detailed routing with the guidance of the global-routing results and finds a real path in the chip. Our detailed router is based on the Dijkstra's shortest path algorithm and supports the *local refinement*. If detailed routing of a connection fails, it will be reconsidered (refined) at the coarsening stage. After a connection has been routed successfully, the estimated cost induced by the connection which calculated by the approach in Section 3.1 will be removed from the channel density, and the wire density of the real path will be updated to the channel density (congestion map) dynamically. This is called *resource estimation*. There are at least two advantages by using this approach. First, routing resource estimation is more accurate than that performing global routing alone since we can precisely evaluate the routing region. Second, we can obtain a good initial solution for the following refinement very effectively since pattern routing enjoys very low time complexity and uses fewer routing resources due to its simple L- and Z-shaped routing patterns.

The coarsening stage starts to refine each local failed connection, left from the uncoarsening stage. The global router is now changed to the maze router with the same cost function in the uncoarsening stage. Coarsening continues until the first level $k$ is reached and the final solution is found. Note that the global maze routing here serves as an elaborate rip-up and re-route processor, in contrast to the simple L- and Z-shaped routing during uncoarsening. (For rip-up and re-route in LMGR, we mean the maze routing at the coarsening stage. It is only applied to global routing for better efficiency and quality trade-off.) This two-stage approach of global and local refinement of detailed routing gives our overall refinement scheme.

## 4 Experimental Results

We have implemented LMGR in the C++ language on a 1 GHz SUN Blade-2000 workstation with 8 GB memory. We compared our results with the grid-based routers presented in [5, 17, 24] and the gridless routers presented in [6, 12] based on the 11 benchmark circuits provided by the authors. (Note that since the results of [12] is better than those of [10, 11], we just compare our results with [12].) The design rules for wire/via widths and wire/via spacings for detailed routing are the same as those used in [12].

Table 2 lists the set of benchmark circuits. In the table, "Circuit" gives the names of the circuits, "Size" gives the layout dimensions, "#Layers" denotes the number of routing layers used, and "#Nets" gives the number of two-pin connections after net decomposition. For delay computation, we use the Elmore delay model. All the parameters are the same as those used in [5, 17, 24]. A via is modeled as the Π-model circuit, with its resistance and capacitance being twice of those of a wire segment. As pointed out in [5, 24], Mcc1, Mcc2, Struct, Prim1, and Prim2 do not have the information of net sources. Therefore, we cannot calculate the path delay for those benchmark circuits. In the following experiments, we represent the critical path delay of these 5 benchmark circuits by the notation, –.

| Circuit | Size ($\mu m$) | #Layers | #Nets | #Pins |
|---|---|---|---|---|
| Mcc1 | 45000×39000 | 4 | 1693 | 3101 |
| Mcc2 | 161482×161482 | 4 | 7541 | 25024 |
| Struct | 4903×4904 | 3 | 3551 | 5471 |
| Prim1 | 7522×4988 | 3 | 2037 | 2941 |
| Prim2 | 10438×6488 | 3 | 8197 | 11226 |
| S5378 | 435×239 | 3 | 3124 | 4818 |
| S9234 | 404×225 | 3 | 2774 | 4260 |
| S13207 | 660×365 | 3 | 6995 | 10776 |
| S15850 | 705×389 | 3 | 8321 | 12793 |
| S38417 | 1144×619 | 3 | 21035 | 32344 |
| S38584 | 1295×672 | 3 | 28177 | 42931 |

Table 2: The benchmark circuits.

### 4.1 Multilevel Grid-based Routing with Uniform Nets

Table 3 lists the wirelength, the critical path delay, the numbers of failed nets, and the running time obtained by the V-shaped multilevel grid-based router with the routability mode in [5, 24], the V-shaped multilevel grid-based router in [17], and LMGR. (We just list the results for the benchmark circuits S5378∼S38584 obtained by [17], since they did not experiment on the other 5 benchmark circuits in their works.) In the table, "WL ($\mu m$)" represents the wirelength in $\mu m$, "#F. Nets" denotes the number of failed nets, "$D_{max}$ (psec)" represents the critical path delay in pico-second, and "Time (sec)" represents the running times in second.

Compared with [5, 24] with the routability mode, the experimental results show that LMGR achieved a 5.66X runtime speedup, reduced the respective maximum and average wirelength by about 18% and 10%, reduced the respective maximum and average critical path delay by about 17061% and 4734%. Compared with [17], the experimental results show that LMGR achieved a 1.46X runtime speedup, reduced the respective maximum and average wirelength by about 69% and 38%, reduced the respective maximum and average critical path delay by about 154% and 18%. Besides, LMGR obtained significantly better routing solutions than [17]. Note that the wirelength and the delay caused by failed nets in [17] should be calculated for fair comparison with LMGR. Because [17] did not complete routing for all benchmark circuits and did not calculate the cost of failed nets in their wirelength and delay, the router underestimates the wirelength and the critical path delay. (A net with multi-sink is routed as a tree. If an edge in this tree is not routed, the tree will be broken into two sutrees. Therefore, the router does not include the failed edge in the wirelength computation, and neither does the downstream capacitance/resistance of the failed edge for delay computation.) If the wirelength and the delay caused by the failed nets are also considered, the real result of the router will be even worse in its wirelength and critical path delay.

### 4.2 Multilevel Gridless Routing with Uniform Nets

Table 4 lists the wirelength, the critical path delay, the numbers of failed nets, and the running time obtained by the V-shaped multilevel gridless routing [6], the multilevel routing (multilevel global routing + flat gridless detailed routing) [12], and LMGR.

Compared with [6], the experimental results show that LMGR achieved a 2.22X runtime speedup, reduced the respective maximum and average wirelength by about 4% and 2%, reduced the respective maximum and average critical path delay by about 90% and 21%. Compared with [12], the experimental results show that LMGR achieved a 1.82X runtime speedup. (For a fair and reasonable comparison, we normalize the running time of [12] by the factor 440/1000.) Since [12] did not report their wirelength and the critical path delay in their paper, we cannot compare those results in LMGR with those in [12].

| Circuit | (A) Results of [5, 24] | | | | (B) Results of [17] | | | | (C) Our Results | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | WL ($\mu m$) | $D_{max}$ (psec) | #F. Nets | Time (sec) | WL ($\mu m$) | $D_{max}$ (psec) | #F. Nets | Time (sec) | WL ($\mu m$) | $D_{max}$ (psec) | #F. Nets | Time (sec) |
| Mcc1 | 2.9e7 | – | 0 | 108.3 | (*) | – | (*) | (*) | 2.7e7 | – | 0 | 63.1 |
| Mcc2 | 4.1e8 | – | 0 | 3961.7 | (*) | – | (*) | (*) | 4.0e8 | – | 0 | 1353.8 |
| Struct | 8.9e5 | – | 0 | 80.7 | (*) | – | (*) | (*) | 8.4e5 | – | 0 | 4.0 |
| Prim1 | 1.0e6 | – | 0 | 89.0 | (*) | – | (*) | (*) | 1.0e6 | – | 0 | 5.0 |
| Prim2 | 4.3e6 | – | 0 | 420.0 | (*) | – | (*) | (*) | 4.1e6 | – | 0 | 30.1 |
| S5378 | 8.5e4 | 89 | 0 | 6.0 | 8.4e4$^+$ | 28$^+$ | 5 | 10.6 | 7.4e4 | 11 | 0 | 7.6 |
| S9234 | 6.4e4 | 254 | 0 | 4.1 | 6.0e4$^+$ | 24$^+$ | 2 | 8.1 | 5.4e4 | 17 | 0 | 4.8 |
| S13207 | 2.0e5 | 465 | 0 | 20.8 | 2.3e5$^+$ | 52$^+$ | 10 | 22.6 | 1.8e5 | 33 | 0 | 20.0 |
| S15850 | 2.5e5 | 2670 | 0 | 31.1 | 2.9e5$^+$ | 68$^+$ | 19 | 62.6 | 2.2e5 | 84 | 0 | 24.5 |
| S38417 | 5.5e5 | 8541 | 0 | 70.3 | 8.0e5$^+$ | 106$^+$ | 33 | 71.3 | 4.7e5 | 174 | 0 | 91.1 |
| S38584 | 7.6e5 | 176090 | 0 | 171.9 | 1.1e6$^+$ | 132$^+$ | 54 | 255.6 | 6.6e5 | 1026 | 0 | 209.0 |
| Comp. | 1.10 | 48.34 | 0 | 5.66 | 1.38$^+$ | 1.18$^+$ | 123 | 1.46 | 1 | 1 | 0 | 1 |

Table 3: Comparison among (A) he V-shaped multilevel grid-based routing [5, 24], (B) he V-shaped multilevel grid-based routing [17], and (C) LMGR. Note: All works were run on a 1 GHz Sun Blade-2000 workstation with 8 GB memory. (+: Because (B) underestimated the wirelength and the path delay by ignoring the cost induced by failed nets, the actual results are worst than the numbers listed in the table.) (–: Because those benchmark circuits did not have the information of net sources, we cannot calculate the path delay for them.) (*: Since [17] did not experiment on those 5 benchmark circuits, we leave the corresponding fields blank.)

| Circuit | (A) Results of [6] | | | | (B) Results of [12] | | | | (C) Our Results | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | WL ($\mu m$) | $D_{max}$ (psec) | #F. Nets | Time (sec) | WL ($\mu m$) | $D_{max}$ (psec) | #F. Nets | Time (sec) | WL ($\mu m$) | $D_{max}$ (psec) | #F. Nets | Time (sec) |
| Mcc1 | 2.8e7 | – | 0 | 179.3 | (*) | – | 0 | 105.1 | 2.7e7 | – | 0 | 63.1 |
| Mcc2 | 4.1e8 | – | 0 | 5509.3 | (*) | – | 0 | 1916.9 | 4.0e8 | – | 0 | 1353.8 |
| Struct | 8.5e5 | – | 0 | 5.7 | (*) | – | 0 | 31.6 | 8.4e5 | – | 0 | 4.0 |
| Prim1 | 1.0e6 | – | 0 | 5.0 | (*) | – | 0 | 33.5 | 1.0e6 | – | 0 | 5.0 |
| Prim2 | 4.2e6 | – | 0 | 42.8 | (*) | – | 0 | 162.7 | 4.1e6 | – | 0 | 30.1 |
| S5378 | 7.6e4 | 21 | 0 | 16.4 | (*) | (*) | 0 | 30.0 | 7.4e4 | 11 | 0 | 7.6 |
| S9234 | 5.5e4 | 18 | 0 | 9.5 | (*) | (*) | 0 | 22.8 | 5.4e4 | 17 | 0 | 4.8 |
| S13207 | 1.8e5 | 37 | 0 | 48.8 | (*) | (*) | 0 | 85.2 | 1.8e5 | 33 | 0 | 20.0 |
| S15850 | 2.2e5 | 87 | 0 | 83.9 | (*) | (*) | 0 | 107.1 | 2.2e5 | 84 | 0 | 24.5 |
| S38417 | 4.8e5 | 183 | 0 | 168.5 | (*) | (*) | 0 | 250.9 | 4.7e5 | 174 | 0 | 91.1 |
| S38584 | 6.7e5 | 1086 | 0 | 369.9 | (*) | (*) | 0 | 466.1 | 6.6e5 | 1026 | 0 | 209.0 |
| Comp. | 1.02 | 1.21 | 0 | 2.22 | | | 0 | 1.82$^\#$ | 1 | 1 | 0 | 1 |

Table 4: Comparison among (A) the V-shaped multilevel gridless routing [6], (B) the V-shaped multilevel gridless global routing + flat gridless detailed routing [12], and (C) LMGR. Note: (A) and (C) were run on a 1 GHz Sun Blade-2000 with 8 GB memory; (B) was run on a 440 MHz Sun Ultra-5 with 384 MB memory. (–: Because those benchmark circuits did not have the information of net sources, we cannot calculate the path delay for them.) (*: Since [12] did not report their wirelength and the critical path delay in their paper, we leave the corresponding fields blank.) (#: For fair comparison, we normalize the running times of [12] by the factor 440/1000.)

### 4.3 Multilevel Gridless Routing with Non-uniform Nets

We also performed experiments on the benchmark circuits of non-uniform wire widths. We modify the original circuits of uniform wire sizes to generate a set of circuits of non-uniform wire sizes by using the following rules, which was proposed by [12]. The longest 10% nets are widened to twice the original width, while the next 10% are widened to 150% the original width. However, because the benchmark circuits S5378∼S38584 are standard-cell designs, widening any pin violates the design rules for via spacing. Therefore, it is unreasonable and incorrect to test these six benchmark circuits.

As shown in Table 5, LMGR still achieved 100% routing completion for all benchmark circuits while [6] and [12] completed routing for only 4 circuits. **Note that LMGR is the first router to complete the routing for this set of benchmarks of non-uniform wire sizes.** Figures 7 and 8 show the full-chip and partial routing solutions for ”Mcc1” with non-uniform nets obtained from LMGR, respectively. The bounding box in Figure 7 is the boundary of this benchmark circuit. We can see in Figure 8 that the three left-most vertical lines have different widths.

### 4.4 Timing-Driven Multilevel Routing with Uniform Nets

Finally, we compared our results with timing-driven routers. [5, 24] is the first and the only one timing-driven multilevel grid-based router. Because there is no timing-driven multilevel gridless router, we selected the first multilevel gridless router [6] which can calculate the path delay. First, we constructed a shortest path tree for a net by connecting all sinks directly to their net source to obtain the timing constraints. We then assigned the timing bound of each sink as the multiplication of the constant $k$ and the shortest path delay of the net.

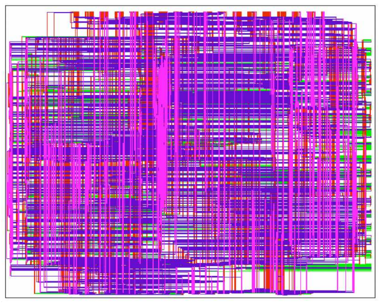
Figure 7: The full-chip routing solution for ”Mcc1” with non-uniform nets obtained from LMGR. The bounding box is the boundary of this benchmark circuit.

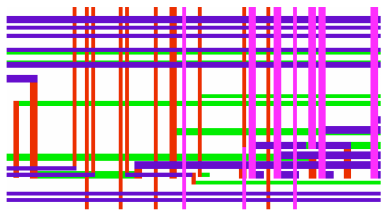
Figure 8: A partial routing solution for ”Mcc1” with non-uniform nets obtained from LMGR. We can see that the three left-most vertical lines have different widths.

Table 6 lists the wirelength, the critical path delay, the number of failed nets, the number of nets which violates timing constraints (#V. Nets in this table), and the running time obtained by the V-shaped multilevel grid-based router with the timing mode [5, 24], the multilevel gridless routing [6], and LMGR when $k = 2.5$.

Compared with [5, 24] with the timing mode, the experimental results show that our router achieved a 9.76X runtime speedup, reduced the respective maximum and average wire-

| Circuit | (A) Results of [6] | | | (B) Results of [12] | | | (C) Our Results | | |
|---|---|---|---|---|---|---|---|---|---|
| | WL ($\mu m$) | #F. Nets | Time (sec) | WL ($\mu m$) | #F. Nets (#Total Sub-nets) | Time (sec) | WL ($\mu m$) | #F. Nets | Time (sec) |
| Mcc1 | 2.8e7 | 0 | 199.6 | (*) | 0 | 148.1 | 2.7e7 | 0 | 65.4 |
| Mcc2 | 4.1e8 | 383 | 36581.5 | (*) | 27(99715) | 3388.8 | 4.1e8 | 0 | 23383.3 |
| Struct | 8.5e5 | 0 | 15.3 | (*) | 0 | 36.3 | 8.4e5 | 0 | 10.3 |
| Prim1 | 1.0e6 | 0 | 19.2 | (*) | 0 | 47.4 | 1.0e6 | 0 | 12.2 |
| Prim2 | 4.2e6 | 0 | 150.8 | (*) | 0 | 296.7 | 4.1e6 | 0 | 80.0 |
| Comp. | 1.02 | 238 | 1.91 | | 27 | 1.19 | 1 | 0 | 1 |

Table 5: Comparison among (A) the V-shaped multilevel gridless routing [6], (B) the V-shaped multilevel gridless global routing + flat gridless detailed routing [12], and (C) LMGR. Note: (A) and (C) were run on a 1 GHz Sun Blade-2000 with 8 GB memory; (B) was run on a 440 MHz Sun Ultra-5 with 384 MB memory. (Note that because the benchmark circuits S5378∼S38584 violate the design rules of via spacing, we did not list these cases in this table.) (*: Since [12] did not report their wirelength in their paper, we leave the corresponding fields blank.) (#: For fair comparison, we normalize the running time of [12] by the factor 440/1000.)

| Circuit | (A) Results of [5, 24] | | | | | (B) Results of [6] | | | | | (C) Our Results | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | WL ($\mu m$) | $D_{max}$ (psec) | #F. Nets | #V. Nets | Time (sec) | WL ($\mu m$) | $D_{max}$ (psec) | #F. Nets | #V. Nets | Time (sec) | WL ($\mu m$) | $D_{max}$ (psec) | #F. Nets | #V. Nets | Time (sec) |
| S5378 | $120766^+$ | $14^+$ | 19 | 0 | 38.1 | 75602 | 21 | 0 | 3 | 16.4 | 73818 | 11 | 0 | 0 | 7.6 |
| S9234 | $92453^+$ | $12^+$ | 25 | 0 | 27.2 | 55319 | 18 | 0 | 2 | 9.5 | 54199 | 17 | 0 | 0 | 4.8 |
| S13207 | $2.9e5^+$ | $27^+$ | 42 | 0 | 113.2 | 1.8e5 | 37 | 0 | 2 | 48.8 | 1.8e5 | 33 | 0 | 0 | 20.0 |
| S15850 | $3.5e5^+$ | $35^+$ | 47 | 0 | 549.9 | 2.2e5 | 87 | 0 | 3 | 83.9 | 2.2e5 | 84 | 0 | 0 | 24.5 |
| S38417 | $9.6e5^+$ | $54^+$ | 121 | 0 | 962.2 | 4.8e5 | 183 | 0 | 3 | 168.5 | 4.7e5 | 174 | 0 | 0 | 91.1 |
| S38584 | $1.2e6^+$ | $133^+$ | 173 | 0 | 1933.0 | 6.7e5 | 1086 | 0 | 10 | 369.9 | 6.6e5 | 1026 | 0 | 0 | 209.0 |
| Comp. | $1.76^+$ | $0.62^+$ | 427 | 0 | 9.76 | 1.02 | 1.21 | 0 | 23 | 2.3 | 1 | 1 | 0 | 0 | 1 |

Table 6: Comparison among (A) The V-shaped multilevel grid-based routing with the timing mode [5, 24], (B) The V-shaped multilevel gridless routing [6], and (C) LMGR. Note: All works were run on a 1 GHz Sun Blade-2000 with 8 GB memory. (+: Because (A) underestimated the wirelength and the path delay by ignoring the cost of failed nets, the actual values shall able to even worst.)

length by about 103% and 76%. Since [5, 24] underestimated the wirelength and the path delay by ignoring the cost induced by failed nets, the actual values shall be even worse. Compared with [6], the experimental results show that our router achieved a 2.3X runtime speedup, reduced the respective maximum and average wirelength by about 4% and 2%, reduced the respective maximum and average critical path delay by about 90% and 21%. Although some path delays of [6] violated timing constraints, the router completed routing for all benchmark circuits. Therefore, we can compare the results of LMGR with those of [6]. Besides, LMGR obtained significantly better routing solutions than [5, 24] and [6] under the same timing constraints. ($k$ = 2.5)

## 5 Conclusion

In this paper, we have proposed a novel Λ-shaped framework for multilevel, full-chip gridless routing. The Λ-shaped multilevel framework adopts a two-stage technique, top-down uncoarsening followed by bottom-up coarsening. Experimental results have shown that our Λ-shaped multilevel gridless router can obtain 100% routing completion rates with less wirelength and smaller critical path delay than previous works. Besides, it can handle designs with non-uniform wire widths well and obtained better routing solutions than previous works. **In particular, our gridless router is the first to complete the routing for the set of commonly used benchmarks of non-uniform wire sizes listed in the preceding section.**

## References

[1] C. Albrecht, “Global routing by new approximation algorithms for multicommodity flow,” *IEEE Trans. CAD*, vol. 20, no. 5, pp. 622–632, May 2001.

[2] C. J. Alpert, J.-H. Huang, and A. B. Kahng, “Multilevel circuit partitioning,” *IEEE Trans. CAD*, vol. 17, no. 8, pp. 655–667, August 1998.

[3] T. Chan, J. Cong, T. Kong, and J. Shinnerl, “Multilevel optimization for large-scale circuit placement,” *Proc. ICCAD*, pp. 171–176, Nov. 2000.

[4] Y.-W. Chang, K. Zhu, and D.-F. Wong, “Timing-driven routing for symmetrical-array-based FPGAs,” *ACM Trans. Design Automation of Electronic Systems*, vol. 5, no. 3, pp. 433–450, July 2000.

[5] Y.-W. Chang and S.-P. Lin, “MR: A new framework for multilevel full-chip routing,” *IEEE Trans. CAD*, vol. 23, no. 5, pp. 793–800, May 2004.

[6] T.-C. Chen and Y.-W. Chang, “Multilevel Gridless Routing considering Optical Proximity Correction,” *Proc. ASP-DAC*, pp. 1160–1163, January 2005.

[7] J. Cong, A. B. Kahng, G. Robins, M. Sarrafzadeh, and C. K. Wong, “Provably good performance-driven global routing”, *IEEE Trans. CAD*, vol 11, no 6, pp. 739-752, Jun. 1992.

[8] J. Cong, J. Fang, and K. Khoo, “DUNE: A multi-layer gridless routing system with wire planning,” *Proc. ISPD*, pp. 12–18, April 2000.

[9] J. Cong, S. Lim, and C. Wu, “Performance driven multilevel and multiway partitioning with retiming,” *Proc. DAC*, pp. 274–279, June 2000.

[10] J. Cong, J. Fang, and Y. Zhang, “Multilevel approach to full-chip gridless routing,” *Proc. ICCAD*, pp. 396–403, Nov. 2001.

[11] J. Cong, M. Xie, and Y. Zhang, “An enhanced multilevel routing system,” *Proc. ICCAD*, pp. 51–58, Nov. 2002.

[12] J. Cong, J. Fang, M. Xie, and Y Zhang, “MARS–A Multilevel Full-Chip Gridless Routing System,” *IEEE Trans. CAD*, vol. 24, no. 3, pp. 382–394, March 2005.

[13] T. Deguchi, T. Koide and S. Wakabayashi, “Timing-driven hierarchical global routing with wire-sizing and buffer-insertion for VLSI with multi-routing-layer”, *Proc. ASP-DAC*, pp. 99-104, Jan. 2000.

[14] M. Hayashi and S. Tsukiyama, “A hybrid hierarchical global router for multi-layer VLSIs,” *IEICE Trans. Fundamentals*, vol. E78-A, no. 3, pp. 337–344, 1995.

[15] J. Heisterman and T. Lengauer, “The efficient solutions of integer programs for hierarchical global routing,” *IEEE Trans. CAD*, vol. 10, no. 6, pp. 748–753, June 1991.

[16] D. Hightower, “A solution to line routing problems on the continuous plane,” *Proc. Design Automation Workshop*, pp. 1–24, 1969.

[17] T.-Y. Ho, Y.-W. Chang, S.-J. Chen, and D. T. Lee, “A Fast Crosstalk- and Performance-Driven Multilevel Routing System,” *Proc. ICCAD*, pp.382–387, Nov. 2003.

[18] T.-Y. Ho, Y.-W. Chang, and S.-J. Chen, “Multilevel Routing with Antenna Avoidance,” *Proc. ISPD*, pp. 34–40, April 2004.

[19] G. Karypis, R. Aggarwal, V. Kumar, and S. shekhar, “Multilevel hypergraph partitioning: Application in VLSI domain,” *IEEE Trans. VLSI Systems*, vol. 7, pp. 69–79, March 1999.

[20] R. Kastner, E. Bozorgzadeh and M. Sarrafzadeh, “Predictable routing,” *Proc. ICCAD*, pp. 110–114, Nov. 2000.

[21] Lee, “An algorithm for path connection and its application,” *IRE Trans. Electronic Computer*, EC-10, 1961.

[22] H.-C. Lee, Y.-W. Chang, J.-M. Hsu, and H. Yang, “Multilevel floorplanning/placement for large-scale modules using B*-trees,” *Proc. DAC*, pp. 812–817, June 2003.

[23] Y. L. Lin, Y. C. Hsu, and F. S. Tsai, “Hybrid routing,” *IEEE Trans. CAD*, vol. 9, no. 2, pp. 151–157, Feb. 1990.

[24] S. P. Lin and Y. W. Chang, “A novel framework for multilevel routing considering routability and performance,” *Proc. ICCAD*, pp. 44–50, Nov. 2002.

[25] M. Marek-Sadowska, “Global Router for gate array,” *Proc. ICCD*, pp. 332–337, Oct. 1984.

[26] M. Marek-Sadowska, “Router planner for custom chip design,” *Proc. ICCAD*, Nov. 1986.

[27] G. Meixner and U. Lauther, “A new global router based on a flow model and linear assignment,” *Proc. ICCAD*, pp. 44–47, Nov. 1990.

[28] J. Soukup, “Fast maze router,” *Proc. DAC*, pp. 100–102, June 1978.

[29] D. Wang and E. Kuh, “A new timing-driven multilayer MCM/IC routing algorithm,” *Proc. Multi-chip Module Conference*, pp. 89–94, Feb. 1997.