MGR: Multi-Level Global Router

Yue Xu and Chris Chu
Department of Electrical and Computer Engineering
Iowa State University, Ames, Iowa 50011-3060, USA
{yuexu,cnchu}@iastate.edu

Abstract—**Global routing faces an increasing problem size and urgent demand on improvement in solution quality. Despite of the recent developments of global routers, there exist only two types of choices: slow 3D routers with good solution quality or efficient 2D routers with relatively poor solution quality. We propose a multi-level 3D global router called MGR to fill the gap. MGR resorts to an efficient multi-level framework to reroute nets in the congested region on the 3D grid graph. Routing on the coarsened grid graph speeds up the global router while 3D routing introduces less vias. The powerful multi-level rerouting framework wraps three innovative routing techniques together: adaptive resource reservation in coarsening process, a new 3-terminal maze routing algorithm and network flow based solution propagation in uncoarsening process. As a result, MGR can achieve the solution quality close to 3D routers with comparable runtime of 2D routers.**

I. Introduction

Due to the increasing size of modern circuits, together with complex demands from performance and manufacturability, every major physical design automation problem faces an exploding size and intricate tradeoff to balance. Global routing, one of the most traditional physical design problems, is no exception. The sheer problem size requires an efficient global routing solution. Meanwhile, increasingly unbalanced routing demands and resources requires global routers to find new methods to resolve routability issues.

In addition, design for manufacturability adds new factors to optimize in global routing. The number of vias has become a standard parameter to minimize in global routing. The reasons why via is so important in global routing come twofold. Via has a higher probability to cause open connection due to its manufacturing process, which lowers circuit yields. To make it worse, via has large variations in its resistance value, which causes performance degradation. Although double via insertion and post routing optimization eases the severity of such issues, the number of vias in global routing remains as one of the determining factors in solution quality.

There are generally two kinds of global routers depending on how they take via into consideration. 3D routers directly work on the metal layers in the layout while 2D routers append traditional single-layer global routers with layer assignment techniques. FGR [1] and GRIP [2] belong to the first category. While FGR sequentially uses 3D maze routing to reroute nets for an existing routing solution, GRIP employs integer linear programming (ILP) to select routing solutions. Despite of their leading results of wirelength and the number of vias, they run too slowly to be practical. The majority of academic global routers belong to the second category. NTHU-R 2.0 [3], NTUgr [4], FastRoute 4.0 [5], NCTU-R [6] and BoxRoute 2.0 [7] all use a two-stage framework to solve the global routing problem. In the first stage, the first four 2D global routers adopt sequential rip-up and reroute strategy while BoxRoute uses ILP to search the best 2D solutions. In the second stage, they all use layer assignment technique to extend the 2D solutions into final 3D solutions.

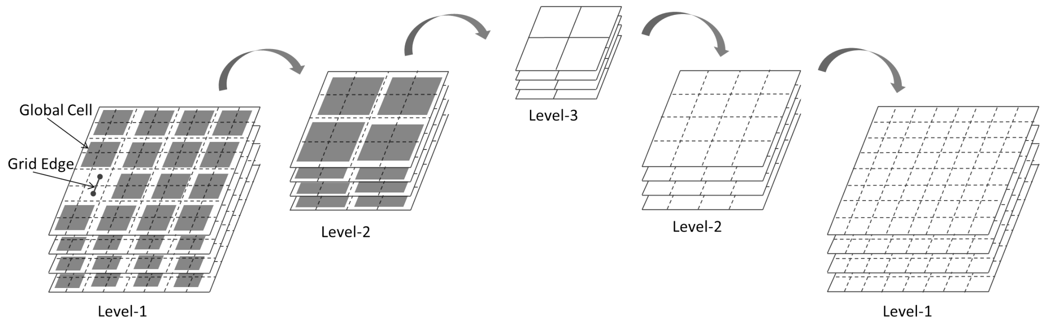
It is natural to wonder whether we can make 2D routers to take a leap to generate solutions as good as 3D routers or whether it is possible to dramatically cut down 3D routers' runtime to the level of 2D routers. The recent developments of 2D routers tend to disprove first attempt. Efforts to improve 2D maze routing by Liu et al. [6] or layer assignment by Lee et al. [8] only bring in marginal improvements. It seems that directly constructing routing solutions on multiple metal layers is indispensable for high quality global routers. To get the combination of short runtime and short wirelength, we propose a multi-level 3D global routing framework MGR to achieve high solution quality and keep runtime in check. MGR exploits 3D maze routing to explore a much larger solution space but formulates it on the coarsened grid graph to get small problem size.

Multi-level framework is a quite mature concept in physical design, commonly used to speed up partitioning and placement tools. Multi-level framework has also been used in gridless routing by Cong et. al. in MARS [9] and Chen et. al. [10]. Gridless routing includes the task of both global routing and detailed routing. It is now rarely used since the routing problem becomes too complex to handle all at once. Detailed routing nowadays is typically treated as a stand-alone problem to handle myriads of design rules.

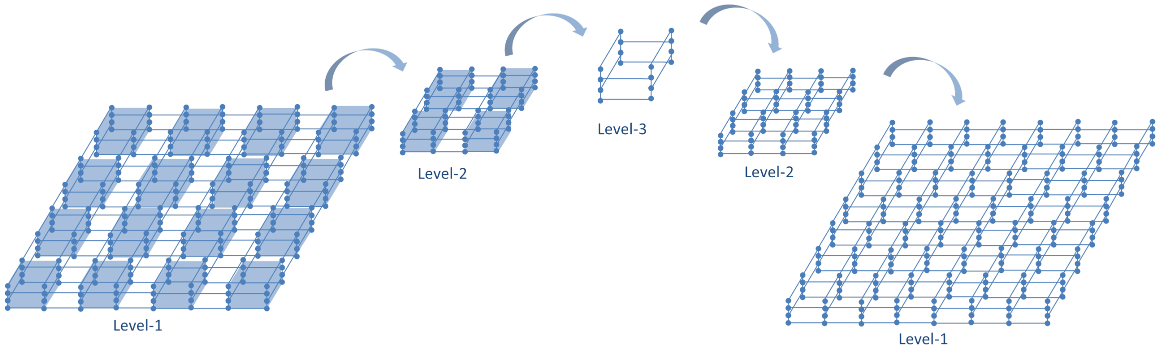
We propose the first academic multi-level global router and name it MGR. MGR uses pattern routing and layer assignment to initiate a 3D solution. Then it uses a coarsening-uncoarsening multi-level framework to reroute nets in the congested region. Rerouting on coarsened grid graph greatly speeds up MGR while 3D routing provides better solution quality by exploiting the entire 3D solution space. The two factors balance out and lead MGR to improve the performance and runtime of global routers. The key contributions of MGR include:

- A multi-level 3D global rerouting framework to efficiently generate high quality solutions. Unlike previous multi-level gridless routers, the framework starts with initial 3D solution which is refined by only one round

[1]This work was partially supported by IBM faculty award.



(a) Cell Structure Coarsening and Uncoarsening



(b) Grid Graph Coarsening and Uncoarsening

Fig. 1. Cell Structure and Grid Graph Views of the Multi-Level Framework

of coarsening and uncoarsening process.

- An adaptive resource reservation technique in coarsening process, which provides accurate routing resource calculation for rerouting on coarsened grid structure to guide routing in higher levels.
- A new 3-terminal maze routing algorithm to optimally connect 3 separate terminals for a net. It provides better routing solutions for multi-pin nets comparing to traditional 2-terminal maze routing based technique.
- A network flow based solution propagation technique in uncoarsening stage, which introduces certain degree of concurrency to achieve better solution refinement in wirelength, congestion and the number of vias together, comparing to pure sequential method.

The rest of this paper is organized as follows. Section 2 introduces the global routing grid graph model and the flow of MGR. Section 3 details the key techniques used in MGR. The experimental results are provided in Section 4 and the paper concludes with Section 5.

II. Preliminary

A. Global Routing Grid Graph

In global routing, the die is partitioned into 3D global cells. Every cell is modeled as a node. On each metal layer, the boundary between two neighboring cells is given certain capacity to accommodate several cross-cell nets while intra-cell nets are ignored in global routing. Grid edges are created to model the capacity and usage on cell boundaries. The capacity c_e of a grid edge is defined as the maximum number of nets that can use the grid edge. The usage u_e is the actual number of nets that use the grid edge. Overflow o_e is defined as $max(u_e - c_e, 0)$. All the grid edges together form the grid graph, as shown in the rightmost figure in Fig. 1(b).

For multi-level global routing, we need to create multi-level grid graphs. The original cell structure and grid graph with finest structure is denoted as level-1. The level increases as global cells in cell structure become larger and grid graph becomes coarser. We derive the level-2 cell structure from level-1 cell structure in the following manner. 4 neighboring cells are merged into one, as shown in Fig. 1(a) by grey areas. The top level is denoted as level-t. Level-i grid graph models the crossing on boundaries of level-i cell structure. For level-(i+1) grid graph, the four corresponding level-i grid nodes grouped in shadow are merged into a node as shown in Fig. 1(b). One level-(i+1) grid edge corresponds to two level-i grid edges. The hierarchy is built up until each layer in the top level grid graph becomes smaller than or equal to the size of 8-by-8. In the hierarchy, nets that exist on level-i grid graph but disappear in level-(i+1) grid graph are denoted as level-i nets. Level-i nets do not belong to level-(i-1) nets because they still exist on level-i. Grid graphs at all levels have the same number of metal layers as level-1 grid graph and coarsening on different layers is vertically synchronized to maintain regular 3D cell and grid graph structures.

In the abstraction process, complex design rules are simplified away. So long as overflow for all grid edges stays 0 and all nets are connected, global routing is considered to have a valid solution.

B. Flow of MGR

The flow of MGR is illustrated in Fig. 2. MGR starts with 3D global routing initialization. MGR needs the initialization stage to provide guidance on how wire-length focused

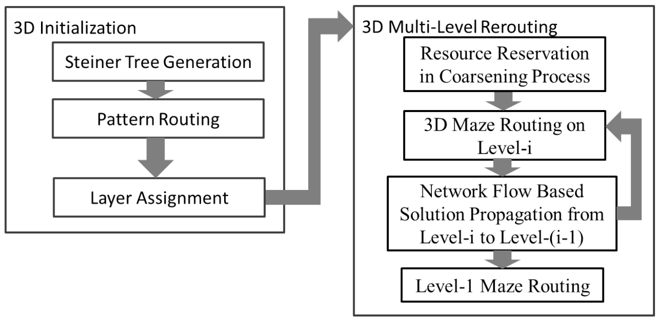

3D Initialization
Steiner Tree Generation
Pattern Routing
Layer Assignment
3D Multi-Level Rerouting
Resource Reservation in Coarsening Process
3D Maze Routing on Level-i
Network Flow Based Solution Propagation from Level-i to Level-(i-1)
Level-1 Maze Routing

Fig. 2. MGR Flow

routing would generate congestion so it can detour high-level nets from the congested region. Our initialization consists of Steiner tree generation [11], a pattern routing step and dynamic programming based greedy layer assignment step just like previous 2D sequential routers [3] [5] without maze routing. Such process is very fast and can provide accurate enough congestion information for the following 3D multi-level rerouting stage.

In the rerouting stage, we use 3D routing to explore the entire solution space and adopt multi-level framework to speed up such exploration. MGR first coarsens the grid graph to build hierarchy. The goal of coarsening is to generate higher level grid graphs to reduce the problem size and calculate accurate routing resource estimation based on the initial solution on those grid graphs. We use adaptive resource reservation to adjust the capacities for high level grid edges, so that high level nets would not over utilize routing resources in congested region, which would reduce the need to rip-up and reroute high level routing solutions in low level grid graphs and thus effectively control run time and improve solution quality. After the coarsening stage, we achieve all levels of grid graphs and their routing resource estimations. MGR then uses 3D maze routing, including the newly proposed 3-terminal maze routing, to eliminate congestion on the level-t grid graph. Then MGR uses network flow based solution propagation to project routing of level-t nets onto level-(t-1) grid graph. For a level-t net, its routing solution can only use grid edges on level-(t-1) grid graph its level-t solution represents. In general, before propagating level-i solutions to level-(i-1), we use 3D maze routing to minimize congestion on level-i. Our newly proposed 3-terminal maze routing is adopted to enhance MGR's congestion reduction capability. This rerouting and propagation iteration inner loops as shown in the right part of Fig. 2 works from level-t to level-2. At the end of this loop, we achieve a rerouted level-1 solution. If the overflow remains after multi-level rerouting, MGR uses 3D maze routing on level-1 grid graph to eliminate it.

III. Multi-Level Global Rerouting Framework

A. Coarsening Process

The coarsening process generates smaller sized grid graphs for 3D maze routing. But it should not sacrifice routing quality with inaccurate estimations of routing resources on the coarsened grid graphs. MGR uses pattern routing and simple layer assignment to provide an initial solution so that the coarsening stage can calculate routing capability for higher level grid graphs based on the actual usage of lowest level grid graph. It differs from previous multi-level gridless routers in the way that previous works rely on inner cell routing blockage model to estimate boundary capacity.

In the coarsening process, we need to make sure that we do not render high level grid graphs with excessive capacity so that high level nets lack incentive to avoid congested region. Such situation will naturally happen if we simply add up the grid edge capacities of the two lower level grid edges corresponding to each higher level grid edge. A higher level grid edge represents larger layout boundary. The possibility of overflow on that boundary tends to diminish as the level goes higher, due to the fact that the demand will average out over a long boundary. The larger area a cell includes, the less likely demands on its boundary exceed the capacity. Just like ideal global routing would detour global nets from locally congested global cells, we want MGR to guide level-(i+1) nets to avoid congested edges on level-i. The most effective way to achieve this is through adaptively capacity adjustment for higher level edges. Without adjustment, the capacity of a level-(i+1) grid edge would be the sum of capacities of the two level-i grid edges it represents. This method has a major drawback. It ignores the congestion inside level-(i+1) cells and leads to much less detour than desirable amount. This stems from the nature of routing problem: If the boundary has ample routing resources while the cell is congested with intra-cell nets, routing on that boundary will aggravate intra-cell congestion. Thus, MGR adjusts the capacity of high level grid edges according to existing inner cell routing solution in an adaptive manner to guarantee that high level routing will not take up critical lower level routing resources for lower level nets.

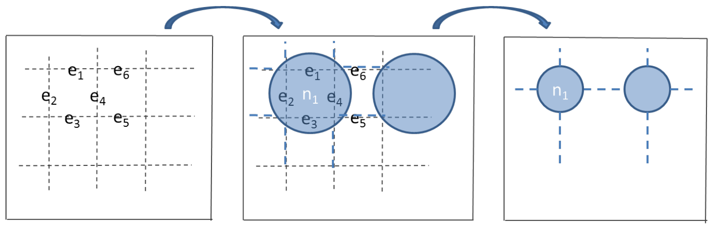


Fig. 3. Coarsening Process in Grid Graph (graph would be changed)

Consider the coarsening step from level-i to level-(i+1), Fig. 3 shows a part of the grid graph that will go through the coarsening step. First, MGR contracts e_1, e_2, e_3, and e_4 into a level-i+1 grid graph node. For the node, MGR assigns a node adjustment value av_n, which is $o_{e1} + o_{e2} + o_{e3} + o_{e4}$, the sum of overflow of the level-i grid edges it includes. The node adjustment value represents the need to discourage higher level nets to use the node in order to resolve congestion on level-i. Nevertheless, routing on a grid graph with both edge and node constraints overly complicates the problem. So in the second phase, we convert the node adjustment value to edge capacity adjustment on the same level by distributing the node adjustment value to the capacity of grid edges connecting to the node. For the level-(i+1) edges that connect to n_1, their capacities will each be reduced by a quarter of av_n. If the node is at the corner of grid graph, the capacities of the two edges connecting to the node will be reduced by half of the node adjustment value. In this way, we propagate level-i overflow information into level-(i+1) grid graph.

B. 3D Maze Rerouting

During the uncoarsening process, MGR needs to reroute some nets to eliminate congestion. Recently, Min Et. Al. expanded maze routing to multi-source and multi-sink maze routing to handle multi-pin nets to further reduce congestion [12]. Instead of rerouting two subtrees from the two fixed endpoints of the ripped-up edge, they use any points on the subtrees as the reconnecting point to adaptively adjust net topology. However, the effectiveness of such adjustment is limited because it only optimally configures the one segment that connects the two unconnected subtrees while the subtrees themselves remain suboptimal. For multi-pin nets, which count for more than 40% in the benchmarks used in ISPD 2008 global routing contest [13], relying on 2-terminal maze routing to restructure net topology may lead global router to run many iterations before it can get a fairly good solution. Optimally connecting multi-pin nets will greatly improved solution quality. Our first effort goes to routing 3-pin nets optimally and it can be extended to optimize multi-pin nets iteratively.

On a second thought, rather than starting the analysis for rerouting 3-pin nets and later extending it to connecting subtrees, we draw lessons from [12] and propose an optimal method to connect nets ripped-up into 3 terminals directly. Here a terminal could be a pin or a subtree. Generating the minimum cost solution to connect nets ripped-up into three terminals has two parts. While searching for a point to be the optimal Steiner point is the first obstacle, fiiguring out the paths between the Steiner point and each terminal stands as the second catch. One easy way to accomplish the two parts is through three independent wavefront propagations from the 3 terminals respectively. We can add the cost for the 3 propagation together and set the point with least sum to be the Steiner node. The optimal solution can be derived from back tracking the wavefront from the Steiner point to the terminals. The only problem is the efficiency of such technique. The search region has to be large enough to contain the optimal Steiner point, which means that the wavefront propagation might propagate to a large area. On the contrary, traditional 2-terminal maze routing can stop when the two wavefronts meet each other. Fortunately, we proved Theorem 1 to greatly limit the propagation area and thus improve the efficiency of the 3-terminal maze routing technique.

Theorem 1. *The wavefront propagation from one terminal used in 3 terminal maze routing can stop whenever it reaches any of the two other terminals.*

Proof: As shown in Fig. 4, we have three terminals: t_A, t_B and t_C. Without loss of generality, we assume that we have an optimal Steiner point P, $md(t_B, P) > min(md(t_B, t_A), md(t_B, t_C))$. Here, "$md$" is the minimum path distance between two terminals. The inequality indicates that point P is out of range of the wavefront propagation started from terminal B. It is obvious that $md(t_A, P) + md(t_C, P) \geq$

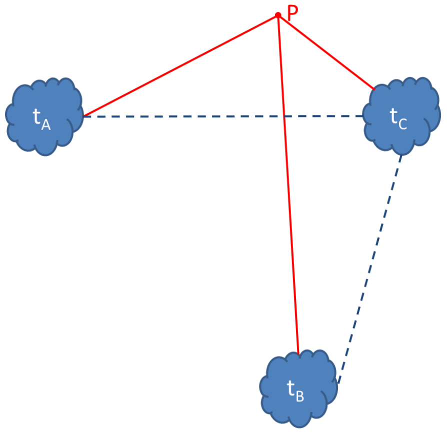


Fig. 4. 3 Terminal Maze Routing

$md(t_A, t_C)$ due to triangular inequality. If we add the two inequalities together, we can get $md(t_A, P) + md(t_C, P) + md(t_B, P) > md(t_A, t_C) + min(md(t_B, t_A), md(t_B, t_C))$. The right part of the inequalities represents a solution that can connect the three terminals together with lower cost than the shortest paths through point P. Thus, we prove Theorem 1 by contradiction.

■

In the proof of Theorem 1, we do not need to specify whether the terminal is a point or a sub-net. Thus, we can apply the multi-source multi-sink maze routing proposed in FastRoute 2.0 [12] to get a fast 3 terminal multi-source multi-sink maze routing. Besides, since minimum distance could be adjusted by any non-decreasing cost function, the algorithm works well for obstructions so long as routing over them gives large cost.

So the 3-terminal maze routing algorithm for nets with 3 pins or more runs as follows. Instead of ripping-up a 2-pin edge, MGR rip-ups three 2-pin edges that connect to a shared Steiner node, generating 3 separate terminals waiting to be connected. Then MGR starts the maze wavefront propagation for each terminal until it meets any one of the other two terminals. After the propagations, we find the least cost Steiner node by checking the region visited by all three wavefront propagation procedure. With the least cost Steiner node at hand, the algorithm backtracks and constructs the paths to connect the 3 terminals together. The runtime bottleneck for 3-terminal maze routing still is the wavefront propagation, which results in a complexity of $O(n \cdot log(n))$, the same as traditional 2-terminal maze routing.

Traditionally, sequential routers run as little maze routing as possible to save runtime. One advantage for MGR is that it has a relatively small problem on higher level grid graphs so it can afford running more maze routing. On higher levels, MGR picks up extra duty to balance out usage profile between large global cells to help eliminate lower level local congestions. To achieve that, MGR rips-up and reroutes every net using 3D maze routing for the top 2 levels to get better solutions. Running these extra maze routing improves high level solutions and consume little runtime due to small grid graphs and limited number of high level nets. On lower levels, MGR only use 3D maze routing to reroute congested nets, just like the behavior of traditional 2D sequential routers.

C. Routing Propagation to Lower Level

Once MGR reaches the top level and balances routing demands according to routing resources, it needs to propagate routing decisions made in the higher level to lower levels, in other words, from coarser grid graph to finer grid graphs.

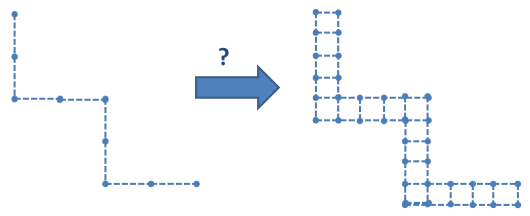


Fig. 5. High Level Routing Propagation into Lower Level Grid Graph

The propagation for a single net is a relatively easy problem. As shown in Fig. 5, the higher level net just need to choose among the edges that is abstracted away during the corresponding coarsening process. Such propagation can easily be realized by dynamic programming. However, sequential net by net propagation ignores the impacts of one net onto its spatially correlated nets. The ideal solution with best results would be a simultaneous net propagation for all the high level nets. However, it has too big a problem size to finish in tight schedule, which is especially true for uncoarsening process for the few bottom levels. In the end, we come up with a balanced solution between solution quality and runtime concern. MGR propagates all the nets or part of the nets that use the same column or same row in the grid graph. We call each column or row a slice. In a grid graph, a slice is defined as the grid nodes on all metal layers of same row or column index together with all the grid edges among the nodes.

Without loss of generality, we pick a row on level-(i+1) for the following analysis. This row consists of 2 neighboring rows in level-i grid graph, together with all the vertical edges that connect them. This forms the backbone of one subproblem. In each subproblem, we use network flow algorithm to propagate routing solution on the row into the finer lower level grid structure. To save more via, we relax the problem by allowing each high level net to choose new routing layer instead of the metal layer decided in its high level solution. This relaxation helps MGR to correct any layer assignment mistakes made in the previous level. The uncoarsening problem is decomposed into network flow sub-problems on slices. The decomposition disentangles the complex uncoarsening problems but still concurrently considers the competitions among all the nets that use a single slice.

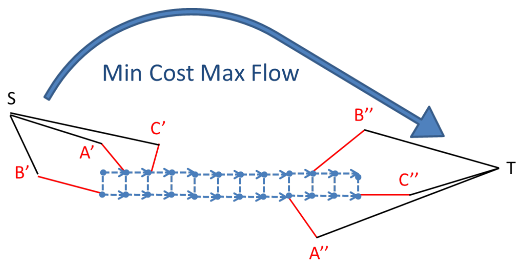


Fig. 6. Network Flow for High Level Net Propagation

The formal problem formulation is defined as follows. We slice the entire column or row from the grid graph. On level-i, we have a single line with multiple layers while at level-(i-1), we have 2 grid-edge lines together with a row of vertical grid edges that connects them. This slice of grids forms the backbone structure of the network flow problem. The grid graph edges are assigned a cost depending on the initial routing solutions and current routing solutions. The capacity and usage for the backbone is assigned in the following manner. Initially, the backbone has the capacity estimated during the coarsening process. Based on that, we add fixed usage that represents the propagated routing solution by level-i propagation subproblems finished earlier, if any. Furthermore, for each level-i nets not propagated yet but will use the vertical grid edges in the backbone, we add 0.5 usage on each of the two possible grid edges. For the network flow problem, we would compute the cost for each grid edge by a logistic function similar to [12] based on the capacity and usage calculated just above.

After the costs are calculated, we assign net or part of net, which we denote as a path from here, to the backbone to model routing demands on the slice. If a path has a pin on the slice, we create an anchor at the pin location. Otherwise, if the path just “passes” the slice, we create the anchor depending on the direction in which the path turns. Take net A in Fig. 6 as an example, it comes from up, use the the slice and goes down. So we create anchor A' at the top row of the backbone and another anchor A” at the bottom. Every path that uses the slice has two anchors. If we scan through the slice from its minimum index to its maximum, whenever we find the first anchor belonging to a path, we connect it to the source node “s”. When we find the second anchor, we connect it to the sink node “t”. The scan adds edges to the backbone to model routing demands. Every edge connected to “s” or “t” has a capacity of 1 and a cost of 0. Besides, with the scan direction, we can assign the two rows of grid edges the same direction, which simplifies the network flow problem.

The optimal solution we want to find is a minimum cost maximum flow from “s” to “t” on the network we just created. We use ILP to solve this discrete flow problem. Since the problem size is relatively small, comparing to the entire routing problem, runtime is not a concern here. If the maximum flow equals to the number of paths that has demands on the slice, the uncoarsening process for this slice is successful. If not, then some high level nets cannot be routed. All the slices are sorted by how congested they are. The higher congestion one slice has, the earlier its network flow propagation would be solved, because less congested columns or rows could afford the reduced flexibility due to more fixed usage added onto their network flow backbone from earlier probagation subproblems.

We wait for the uncoarsening process for all slices to finish before we run reroute for all unconnected high level nets, and some level-(i-1) nets if they are involved in congestion too. In the reroute process, we use 3-terminal maze routing mentioned in previous section for nets with more than 2 pins and traditional maze routing for 2-pin nets. Then uncoarsening process will proceed to solve network flow based propagation problem from current level to the next lower level.

TABLE I
COMPARISON OF ROUTING RESULTS ON ROUTABLE BENCHMARKS

	MGR		NTHU-R 2.0 [3]		NTUgr [4]		FastRoute 4.0 [5]	
Name	Wirelength	cpu(s)	Wirelength	cpu(s)	Wirelength	cpu(s)	Wirelength	cpu(s)
adaptec1	5282K	304	5344K	611	5740K	291	5460K	279
adaptec2	5146K	72	5229K	102	5370K	71	5277K	59
adaptec3	12892K	334	13101K	549	13500K	284	13213K	240
adaptec4	11996K	98	12169K	130	12370K	78	12249K	41
adaptec5	15323K	550	15538K	1160	15990K	988	15866K	660
newblue1	4558K	312	4653K	312	4930K	63161	4686K	377
newblue2	7446K	55	7570K	61	7690K	39	7636K	18
newblue5	22800K	453	23158K	977	24490K	1324	23377K	777
newblue6	17486K	487	17689K	912	18660K	1376	18078K	884
bigblue1	5582K	349	5595K	690	6000K	1169	5775K	530
bigblue2	8892K	415	9059K	427	9120K	16044	9352K	792
bigblue3	12875K	200	13068K	253	13350K	258	13073K	158
Comparison	1	1	1.015	1.7	1.053	23.5	1.029	1.33

TABLE II
COMPARISON OF ROUTING RESULTS ON UNROUTABLE BENCHMARKS

	MGR			NTHU-R 2.0 [3]			NTUgr [4]			FastRoute 4.0 [5]		
Name	Overflow	WL	cpu(s)	Overflow	WL	cpu(s)	Overflow	WL	cpu(s)	Overflow	WL	cpu(s)
bigblue4	134	22573K	1475	162	23090K	6633	188	24280K	26692	150	25147K	3640
newblue3	31026	10722K	1384	31454	10653K	6168	31024	18830K	57120	31634	10752K	1181
newblue4	136	12854K	1083	138	13046K	4873	142	14380K	72246	140	13821K	2382
newblue7	56	34902K	7624	62	35522K	7252	310	37220K	93401	58	35974K	10209
Comparison	1	1	1	1.015	1.016	2.16	1.01	1.175	21.6	1.02	1.057	1.5

IV. Experimental Results

We implement MGR in C and conduct all the experiments on a Linux machine powered by a $2.6GHZ$ Intel Processor with $16GB$ memory. We use the benchmarks from ISPD 08 global routing contest and compare the results of our work with leading academic global routers: NTHU-R 2.0 [3], NTUgr [4] and FastRoute 4.0 [5].

The comparison are conducted in two parts: for routable benchmarks and for unroutable benchmarks, separated by whether academic global routers can generate congestion free solutions up to now, because routers may adopt very distinctive behavior when facing the final few violations to resolve. Table I shows the comparison for routable benchmarks. Comparing to the contest winners, MGR generates solutions with least wirelength using much less run time. The solutions generated by MGR has 1.5%, 5.3% and 2.9% less wirelength comparing to NTHU-R 2.0, NTUgr and FastRoute 4.0 respectively. In addition, the new routing framework runs $1.7X$, $1.4X$ and $19.8X$ faster than the three global routers.

Table II shows the comparison for unroutable benchmarks. Once more, MGR generate solutions with shortest wirelength while using least runtime. The improvement for unroutable benchmarks is more significant than the improvement for routable benchmarks. It is because MGR could resolve the violations during the multi-level coarsening and uncoarsening process in a faster manner and later resort to level-1 3D maze routing to minimize the number of violations.

V. Conclusions

In this work, we propose a multi-level global router called MGR. MGR consists of uncoarsening process and coarsening process. In the uncoarsening process, MGR adaptively adjusts the capacities for high level grid graphs to effectively direct high-level nets from congested region. In the coarsening process, we propose a novel three-terminal maze routing and network flow based solution propagation. These two new techniques together guarantee better routing solutions. This multi-level solution can greatly speeds up global routing and provides better solutions comparing to existing academic global routers.

References

[1] J. A.Roy and I. L. Markov, "High-performance routing at the nanometer scale," *Proc. IEEE/ACM Intl. Conf. on Compuer-Aided Design*, pp. 496-502, 2007.

[2] T. Wu, A. Davoodi and J. Linderoth, "GRIP: scalable 3D global routing using integer programming," *Proc. of Design Automation Conference*, pp. 320-325, 2009.

[3] Y.-J. Chang, Y.-T. Lee, and T.-C. Wang, "NTHU-Route 2.0: a fast and stable global router," *Proc. International Conference on Computer Aided Design*, pp. 338-343, 2008.

[4] Y. Chen, C. Hsu and Y. Chang"High-Performance Global Routing with Fast Overflow Reduction," *Proc. Asia and South Pacific Design Automation Conf.*, pp. 582-587, 2009.

[5] Y. Xu, Y. Zhang and C. Chu, "FastRoute 4.0: global router with efficient via minimization," *Proc. Asia and South Pacific Design Automation Conf.*, pp. 576-581, 2009.

[6] W.-H. Liu, W.-C. Kao, Y.-L. Li, and K.-Y. Chao, "Multi-Threaded Collusion-Aware Global Routing with Bounded-Length Maze Routing," *Proc. of Design Automation Conference*, pp. 200-205, 2010.

[7] M. Cho, K. Lu, K. Yuan, and D. Z. Pan, "BoxRoute 2.0: Architecture and Implementation of a Hybrid and Robust Global Router," *Proc. International Conference on Computer Aided Design*, pp. 503-508, 2007.

[8] T.-H. Lee and T.-C. Wang, "Robust Layer Assignment for Via Optimization in Multi-layer Global Routing," *Proc. International Symposium on Physical Design*, pp. 159-166, 2009.

[9] J. Cong, M. Xie and Y. Zhang, "MARS - A multilevel fullchip gridless routing system," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 3, pp. 382-394, 2005.

[10] T.-C. Chen, Y.-W. Chang, and S.-C. Lin, "A novel Framework for Multilevl Full-Chip Gridless Routing," *Proc. Asia and South Pacific Design Automation Conf.*, pp. 636 - 641, 2006.

[11] C. Chu and Y. C.Wong, "FLUTE: Fast Lookup Table-based Rectilinear Steiner Minimal Tree Algorithm for VLSI Design," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 1, pp. 7083, 2008.

[12] M. Pan and C. Chu, "FastRoute 2.0: A High-quality and Efficient Global Router," *Proc. Asia and South Pacific Design Automation Conf.*, pp. 250 - 255, 2007.

[13] http://www.ispd.cc/contests/ispd08rc.html.