TritonRoute: An Initial Detailed Router for Advanced VLSI Technologies

Andrew B. Kahng[†‡], Lutong Wang[‡], Bangqi Xu[‡]

[†]CSE and [‡]ECE Departments, UC San Diego, La Jolla, CA 92093
{abk, luw002, bangqixu}@ucsd.edu

ABSTRACT

Detailed routing is a dead-or-alive critical element in design automation tooling for advanced node enablement. However, very few works address detailed routing in the recent open literature, particularly in the context of modern industrial designs and a complete, end-to-end flow. The ISPD-2018 Initial Detailed Routing Contest addressed this gap for modern industrial designs, using a reduced design rules set. In this work, we present TritonRoute, an initial detailed router for the ISPD-2018 contest. Given route guides from global routing, the initial detailed routing stage should generate a detailed routing solution honoring the route guides as much as possible, while minimizing wirelength, via count and various design rule violations. In our work, the key contribution is intra-layer parallel routing, where we partition each layer into parallel panels and route each panel using an Integer Linear Programming-based algorithm. We sequentially route layer by layer from the bottom to the top. We evaluate our router using the official ISPD-2018 benchmark suite and show that we reduce the contest metric by up to 74%, and on average 50%, compared to the first-place routing solution for each testcase.

ACM Reference Format:
Andrew B. Kahng[†‡], Lutong Wang[‡], Bangqi Xu[‡]. 2018. TritonRoute: An Initial Detailed Router for Advanced VLSI Technologies. In *IEEE/ACM INTERNATIONAL CONFERENCE ON COMPUTER-AIDED DESIGN (ICCAD '18), November 5–8, 2018, San Diego, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3240765.3240766

1 INTRODUCTION

Detailed routing is a dead-or-alive critical element of advanced node enablement. New technology nodes come with smaller feature sizes, while fundamental physical (lithographic patterning, CMP, reliability, variability, etc.) and circuit (crosstalk, delay, etc.) limitations remain. As a result, ever-more complex design rules must be comprehended and satisfied at the detailed routing stage, greatly challenging routability as well as the architecture and strategy of the detailed router itself.

ICCAD '18, November 5–8, 2018, San Diego, CA, USA

ACM ISBN 978-1-4503-5950-4/18/11…$15.00
https://doi.org/10.1145/3240765.3240766

Due to the high complexity and enormous solution space for the VLSI routing problem, the routing is typically split into global routing and detailed routing stages. In global routing, the routing region is divided into rectangular grid cells and represented using a coarse-grained 3D routing graph. Capacities and various constraints are assigned to the edges and vertices in this 3D routing graph so that overall routing topology and layer assignment can be optimized considering routability, timing, crosstalk, power, etc. The ensuing detailed routing stage attempts to realize the segments and vias according to the global routing solution, while minimizing design rule violations.

The detailed routing problem has been extensively studied for more than five decades. The fundamental algorithms (e.g., Lee's algorithm, unidirectional and bidirectional A* search, ripup-and-reroute paradigm, etc.) and problem formulations (e.g., channel routing and switchbox routing) have largely remained intact in commercial tools for several decades; see [1] for a thorough review. These algorithms and formulations are elaborated to meet real-world requirements (design-rule correctness, quality of result, scalability, and turnaround time) and widely deployed in today's commercial tools that support N7, N5 or even N3 nodes. Within the recent academic literature, a number of works address aspects of detailed routing, as reviewed in Section 2 below. However, only a few academic works even attempt to present an end-to-end detailed routing flow, and almost no works make claims to viability in the real-world IC physical design (P&R) context. Since most detailed routing research works focus on different objectives, such as crosstalk or new-technology contexts, comparison between these works is difficult. Further, direct application of academic codes to modern industrial benchmarks has many hurdles, especially given that commercial tools and industrial designs satisfy far more and complex design rules than any academic tools.

Given the above, it is a highly significant milestone for the field that the ISPD-2018 contest, on the subject of Initial Detailed Routing, has recently exposed industrial detailed routing challenges and benchmarks to the academic community [19][34]. The ISPD-2018 benchmark suite provides 10 testcases in 45nm and 32nm nodes, with up to 290K standard cells and 182K nets. These designs are industrial benchmarks – including large memory cells, off-track pin access, IO ports, and power and macro blockages – with realistic design rules offered in industry-standard input/output formats while keeping problem complexity tractable to academic researchers within the four-month contest timespan.

Based on the ISPD-2018 Initial Detailed Routing contest, the present paper describes TritonRoute, an initial detailed router for advanced VLSI technologies. Our main contribution is a parallel

mixed integer-linear programming (MILP) based routing framework, along with results that reduce the ISPD-2018 contest metric by up to 74%, and on average 50%, compared to the first-place routing solution for each testcase as reported in [34]. Highlights of our work are summarized as follows.

- We propose a MILP-based *panel routing* scheme. Our proposed scheme is capable of comprehending connectivity constraints (i.e., opens and shorts) and design rule constraints (i.e., spacing tables, end-of-line (EOL) spacing, minimum area and cut spacing).
- We propose an overall flow that orchestrates *intra-layer parallel routing* within an overall *inter-layer sequential routing* framework. Our router works in parallel within each metal layer, completes routing on one layer and then goes on to process the next upper layer.
- We evaluate our router using the official ISPD-2018 benchmark suite and evaluation script, and show that we reduce design rule violations by up to 93.58%, and overall contest metric by up to 74% and on average 50%, compared to the contest-winning solution for each testcase.

The remainder of this paper is organized as follows. Section 2 provides a brief overview of related works in the open literature. As noted above, such literature is sparse as far as it gives insight into industry routing tools and how they address modern routing challenges. Section 3 presents background information, including benchmark input file properties and our problem preprocessing techniques. Section 4 details our problem statement, overall tool flow, connectivity handling and MILP-based formulation. Section 5 presents our experimental results using the official ISPD-2018 benchmark suite and evaluator. Section 6 gives conclusions and directions for ongoing work.

2 RELATED WORKS

As surveyed in [1], previous works on detailed routing can be categorized into (i) fundamental and conventional algorithms, along with (ii) recent developments.

Fundamental and conventional algorithms. Lee [13] proposed the first maze routing algorithm, i.e., a breadth-first search that guarantees to find a minimum-cost path between two terminals if a path exists. Use of "best-first search", also known as A* search [21], sometimes in its bidirectional [22] form, enables maze-based search to focus itself toward desired targets, and reduces effort needed to find a minimum-cost feasible path. Hadlock [9] and Soukup [23] applied speedups to Lee's algorithm and others applied the line-search paradigm [11] to improve time and space efficiency compared to Lee's and A* algorithms. Specialized contexts such as channel routing [6] and switchbox routing [18], along with general frameworks such as multicommodity flow [24] and ripup-and-reroute [25], have respective sub-literatures and remain as fundamental building blocks of the detailed router today (cf. [8]).

Recent developments. More recent academic works on detailed routing focus on certain aspects of the modern routing challenge, mainly to address issues arising with advanced nodes. [14] gives an excellent summary of the academia-industry gap for detailed routing as of 2003; much of this gap remains today. Examples of focused recent works include Nieberg [20], which proposes techniques for gridless pin access in detailed routing. Xu et al. [26] propose pin-access planning and regular routing for self-aligned double patterning (SADP). The works of [4][5][7][15] address the detailed routing problem in an SADP process context. Han et al. [10] develop a framework to reduce various design rule violations in advanced nodes using multicommodity flow-based integer-linear programming. RegularRoute [27] and BonnRoute [8] are two works prominent in the recent literature that present more complete portraits of overall detailed routing solutions.

3 PRELIMINARIES

In this section, we describe the definition and the properties of a *route guide*, along with the preprocessing techniques that we apply to route guides (which are given as part of each benchmark in the ISPD-2018 contest) before performing detailed routing.

3.1 Route Guides

In the ISPD-2018 Initial Detailed Routing Contest, a global routing solution is given as route guides. A **route guide** specifies a rectangular region on a specific metal layer. A global routing solution for a net may contain several route guides on some or all of the metal layers. A detailed router needs to honor route guides, i.e., to route within route guides as much as possible. Our router assumes that the route guides for each net satisfy **inter-guide connectivity** constraints.

Inter-Guide Connectivity. All unconnected terminals of a net can be traced through connected route guides. Two guides are *connected* if (i) they are on the same metal layer with touching edges, or (ii) they are on neighboring metal layers with a nonzero vertically overlapped area. Each unconnected terminal (i.e., pin of a standard-cell instance or an IO port) should have its pin shape overlapped by a route guide.

Figure 1 shows route guides for a four-pin net. Four pins (on M1) have pin shapes in route guides on M1. The vertically overlapped area of two route guides on neighboring metal layers could imply a via. A detailed routing solution within the route guides is considered desirable, while routing segments outside of the guides are discouraged. Purple dashed lines may form one of several possible routing topologies within the route guides.

In the ISPD-2018 contest, route guides satisfy the following constraints: (i) the width (resp. height) of a route guide is always a multiple of a *unit* width (resp. height); (ii) the corner coordinates of a route guide, i.e., the lower-left X (resp. Y) coordinate, is always a multiple of unit width (resp. height) plus a fixed amount of offset. For example, in Figure 1(b), route guides are aligned every 2000 units (which corresponds to the *unit* width or height) with an offset of zero, and each has a width and a height that is a multiple of 2000 units. We define route guides to be **aligned** if the constraints (i) and (ii) are both satisfied.

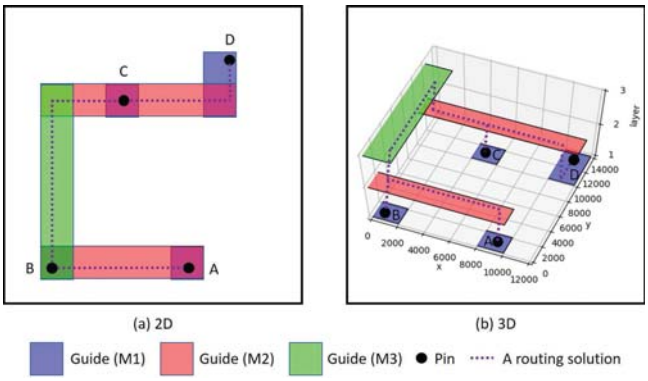

Figure 1: An example of route guides for a four-pin net: (a) 2D top-down view, and (b) 3D view. The dashed lines form one of several possible routing topologies within the route guides.

3.2 Preprocessing

In this work, we preprocess all route guides with *splitting*, *merging* and *bridging* techniques, as we now describe.

Splitting. A guide with width (orthogonal to the preferred direction) greater than unit width is split into multiple unit-width guides, as shown in Figure 2(b).

Merging. Touching guides (touching edge orthogonal to the preferred routing direction) are merged, as shown in Figure 2(c).

Bridging. Touching guides (touching edge parallel to the preferred routing direction) are bridged with additional upper (default) or lower layer guides, as shown in Figure 2(d). After bridging, inter-guide connectivity does not rely on routing topologies with non-preferred direction routing.[1]

The preprocessing is a one-time effort, resulting in all unit-width guides, with largest possible lengths in the preferred direction. We first perform splitting and merging for all route guides. We then execute the graph Steiner tree approximation [12] on this inter-guide connectivity graph to find a guide-to-guide path between the source (driving pin) and each sink. Next, we apply the bridging technique if any path implies a non-preferred direction routing topology, i.e., same-layer inter-guide wrong-way connection, and finally, we remove all redundant guides. Even though the preprocessing steps slightly change the input route guides, these steps serve to remove any possible loops in the inter-guide connectivity graph, and enforce strictly preferred-direction routing topologies that help simplify our proposed detailed routing algorithm in Section 4.

4 PARALLEL MILP-BASED ROUTING

In this section, we present our problem statement, overall flow, connectivity handling, and MILP-based routing formulation.

[1]In this work, we perform "bridging" only to simplify the connectivity, at the cost of "out-of-guide" penalty. We show in Section 5 that out-of-guide wire and vias contribute less than 2% of the total raw score obtained by TritonRoute.

4.1 Problem Statement

From the discussion above, for all nets, our preprocessed route guides satisfy the following three conditions: (i) the inter-guide connectivity graph is acyclic; (ii) topologies use only preferred routing directions; and (iii) route guides are aligned, and are of unit width. Given the preprocessed route guides, we seek to assign routing segments and vias to realize the connectivity of each net, subject to design rules, while minimizing wirelength and via count. The realization of connectivity encompasses (i) *inter-guide connectivity*, where vias are placed to realize the connectivity of route guides on neighboring metal layers that have non-zero vertically overlapped area; and (ii) *intra-guide connectivity*, where routing segments are placed to realize the connectivity between vias, unconnected terminals (pins or IO ports), and pieces of unconnected routing segments (if any). Our problem formulation can then be summarized as follows.

Inputs: LEF, DEF, preprocessed route guides.

Output: An initial detailed routing solution with optimized wirelength and via count.

Subject to: Route guide honoring, connectivity constraints, and design rules.

4.2 Overall Flow

The overall flow can be described using **intra-layer parallel routing**, and **inter-layer sequential routing**.

Intra-Layer Parallel Routing. Since route guides on the same metal layer are aligned and have unit width, routing realization can be executed in parallel as long as two route guides do not overlap (subject to design rules). We partition each metal layer into non-overlapping, unit-width **panels** such that each panel runs parallel to the preferred routing direction. Each route guide is assigned to (exactly) one panel. Routing realization of all route guides within a panel is subject to the limits of shared routing resources, i.e., track availability and various design rules. By contrast, routing realizations in different panels are almost independent, except for design rule constraints, i.e., minimum spacing. Thus, intra-layer routing can be performed in parallel, with respect to panels. In our implementation, for one metal layer, we route all even-index panels in parallel, and then route all odd-index panels in parallel so that design rules violations along panel boundaries can be optimized. Figure 3 illustrates this flow for the M3 layer.

Inter-Layer Sequential Routing. We assign routing segments and vias layer by layer, in a sequential manner, from the bottom metal layer to the top layer. Only after we have assigned all routing segments and vias for all panels on layer l, do we proceed to the next (upper) metal layer $l+1$. For example, in Figure 3, M1 is already routed. We finish routing layer M2 (3a) before we route layer M3 (3b and 3c). M4 is not yet routed.

In our implementation, vias between a lower layer and an upper layer are assigned when we perform the upper-layer panel routing. We describe in Section 4.3 how we handle both intra-guide and inter-guide connectivity in our MILP-based panel routing. Note that in this flow, we do not consider pre-routed power-ground (PG) mesh. Also, since we do not perform iterated inter-panel and inter-layer improvements, we currently do not consider routing detours to help achieve DRC convergence. Support of the above is part of our ongoing work, as described in Section 6.

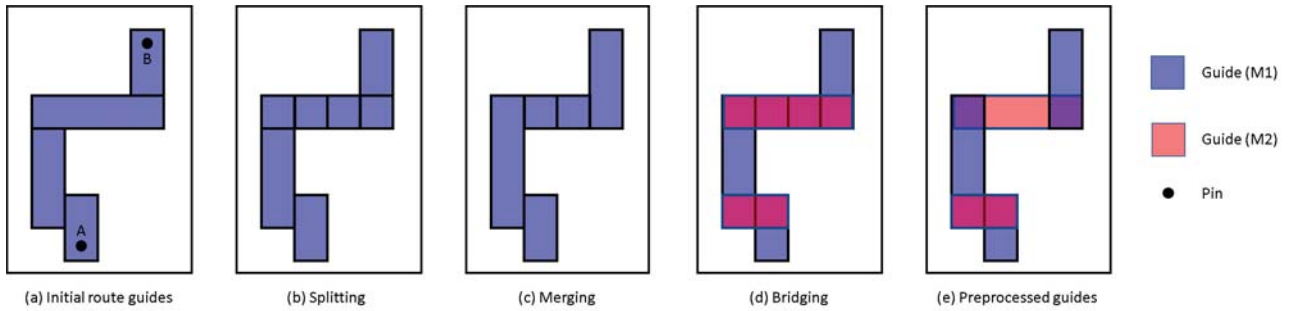

Figure 2: Preprocessing: (a) initial route guides; (b) splitting; (c) merging; (d) bridging and (e) preprocessed guides. The preferred direction for M1 is vertical, and horizontal for M2.

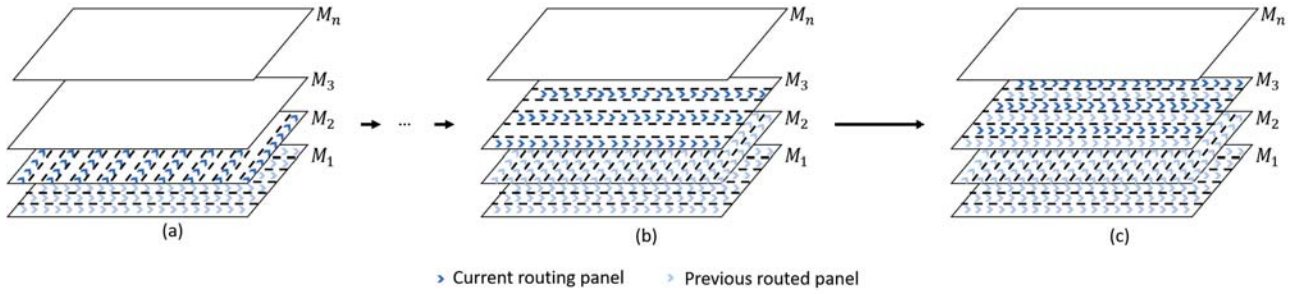

Figure 3: Overall flow: (a) parallel routing of panels on M2; (b) parallel routing of even (i.e., even-index) panels on M3; and (c) parallel routing of odd panels on M3.

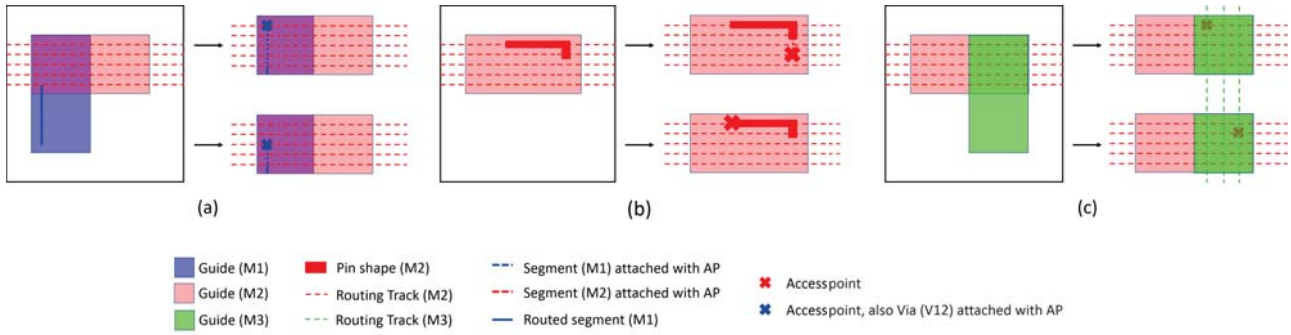

Figure 4: Illustration of access points: (a) to a lower-layer segment; (b) to a pin shape; and (c) to upper layer.

4.3 Handling Connectivity

Before we describe how we handle the inter- and intra-guide connectivity, we first define an *access point*.

Access Point. An *access point* (AP) is an on-grid point on the metal layer of the route guide, and is used to connect to lower-layer segments, upper-layer segments, pins or IO ports. An access point may also contain certain detailed routing segments and vias as needed.

Figure 4 shows three examples of access points. In Figure 4(a), we show two route guides on M1 and M2, respectively. For the guide on M1, there is one routed segment within the guide, and we can now perform routing on M2. To realize the inter-guide connectivity from M1 to M2, we can propose up to five access points on M2, two of which are shown in the figure. In this way, the inter-guide connectivity problem is transformed into the intra-guide connectivity problem, i.e., any routing segments touching one of these access points can be seen to actually have a physical connection to the routed segments on the lower layer. To realize the physical connection to lower layers, a unique routing pattern, i.e., routing segment on the lower layer, plus a via, is generated along with each access point. Figure 4(b) shows access points for pins. Again, each access point comes with a necessary routing pattern,

i.e., routing segments to the pin shapes. Figure 4(c) shows access points for inter-guide connectivity to upper-layer guides, two of 15 APs are shown in the figure. Since vias are assigned following the upper-layer routing, the physical connection to routing segments on the upper layer is not handled here.

In our implementation, if access points are used for inter-guide connectivity to lower-layer or upper-layer guides, we generate all on-grid access points within the guides' overlapped area. If access points are used for pins or IO ports, we generate all on-grid access points along the preferred routing direction up to 15 tracks apart. We note that only access points with unique routing patterns in the preferred direction are generated. We discard all access points using non-preferred routing except for pin access. We do not generate off-grid access points in our implementation. However, our flow supports the co-existence of on-grid and off-grid access points, including off-grid routing patterns.

Access Point Cluster. An access point cluster (APC) is the union of all access points derived from the same lower-layer segment, upper-layer guide, a pin or an IO port.

For example, the two APs in Figure 4(a) are of the same APC since they are both used to connect to the same lower-layer segment. Similarly, two APs in Figure 4(b) form an APC, and two APs in Figure 4(c) form an APC. Clearly, two APs in the same APC are electrically connected through their attached vias and/or routing segments to the same lower-layer (resp. upper-layer) segments, pin or IO port. However, two APs in different APCs are electrically open (i.e., not electrically connected). Since inter-guide connectivity is transformed into intra-guide connectivity, satisfying the connectivity requirement is equivalent to making physical connections among all APCs in the same route guide.

Given a route guide with n APCs, we need at least $n-1$ routing *segments* to electrically connect them. We use a rectilinear minimum spanning tree (MST) construction to determine an optimized routing topology, i.e., a set of logical APC-to-APC segments that minimizes estimated wirelength.

Algorithm 1 details the procedure of determining the topology of an MST in a given route guide for a given net. Lines 1-5 initialize the edges of the complete graph with vertices of all APCs in the route guide. The cost of each APC_i to APC_j edge is determined using the smallest preferred-direction distance between any AP in APC_i and any AP in APC_j that share the same track. If APC_i and APC_j do not have any APs that are on the same track, then the cost is infinite. Line 6 applies the MST algorithm (we use an $O(n^2)$ implementation of Prim's algorithm) on the complete graph with vertices of *APCs* in a route guide and with edge cost vector *COSTs*. Line 7 returns the $n-1$ edges (logical segments) in the MST.

Algorithm 1 Optimization of Routing Topology

1: **for all** $i = 1$ to $n-1$ **do**
2: **for all** $j = i+1$ to n **do**
3: $cost_{i,j} \leftarrow dist(APC_i, APC_j)$
4: **end for**
5: **end for**
6: $T \leftarrow MST(APCs, COSTs)$
7: **Return** $e_{i,j} \in T$

4.4 MILP-based Formulation

Table 1: Notations.

Notation	Meaning
$G(V, E)$	conflict graph
V	set of vertices
$v_{i,j}$	a vertex representing the j^{th} candidate of the i^{th} segment
E	set of undirected edges
$e_{i,j}^{i',j'}$	undirected edge indicating conflict between $v_{i,j}$ and $v_{i',j'}$
$b_{i,j}$	binary indicator of whether $v_{i,j}$ is used in the routing
$w_{i,j}$	weight of $v_{i,j}$

Table 1 describes notations used in our routing formulation. In each panel, we seek to realize routing for each logical APC-to-APC segment by selecting proper routing candidates using a MILP-based optimization. For each logical APC-to-APC segment, a *candidate* is a unidirectional routed segment connecting any of the APs in the two APCs. We generate all candidates on all available tracks between any pair of APs in both APCs. The problem can be summarized as follows.

Inputs: Route guides and set of APs in APCs for each net in the given panel.

Output: An initial detailed routing solution such that APCs in each route guide are electrically connected.

Constraints: Design rules, i.e., metal shorts and spacing rules.

We formulate a maximum weighted independent set (MWIS) problem to realize the optimized routing topologies in a panel. Similar in spirit to [27], we use a *conflict graph* $G = (V, A)$ to represent routing segment candidates and conflicts between candidate pairs. Each vertex $v_{i,j} \in V$ represents a routing segment candidate with associated weight $w_{i,j}$. Each edge $e_{i,j}^{i',j'}$ indicates a conflict, i.e., metal shorts or design rule violations if candidates $v_{i,j}$ and $v_{i',j'}$ are chosen at the same time.

$$\textbf{Maximize: } \sum w_{i,j} \cdot b_{i,j}$$
$$\textbf{Subject to: } b_{i,j} + b_{i',j'} < 1,\ \forall e_{i,j}^{i',j'} \in E \tag{1}$$

The objective is to maximize the weighted sum of $b_{i,j}$'s, i.e., total weight of a DRC violation-free set of routing segment candidates. The constraint ensures that the solution of the ILP does not contain conflicting candidates. Two vertices may conflict due to (i) uniqueness constraints, or (ii) DRC constraints as shown in Figure 5(a). The uniqueness constraint states that at most one candidate can be chosen for a given segment, while the DRC constraints enforce design rules. Figure 5(b) shows the corresponding conflict graph of the scenario illustrated in Figure 5(a).

In the example shown in Figure 5(b), the number on each segment candidate indicates the corresponding weight of the candidate. *Segment 1* (resp. *Segment 2*) has two candidates $v_{1,1}$ and $v_{1,2}$ (resp. $v_{2,1}$ and $v_{2,2}$). According to the uniqueness constraint, we build edge $e_{1,1}^{1,2}$ (resp. $e_{2,1}^{2,2}$) for *Segment 1* (resp. *Segment 2*). We also build edge $e_{1,2}^{2,1}$ to avoid metal shorts between *Segment 1* and *Segment 2*. Considering the edges and the weight of each vertex, the solution of

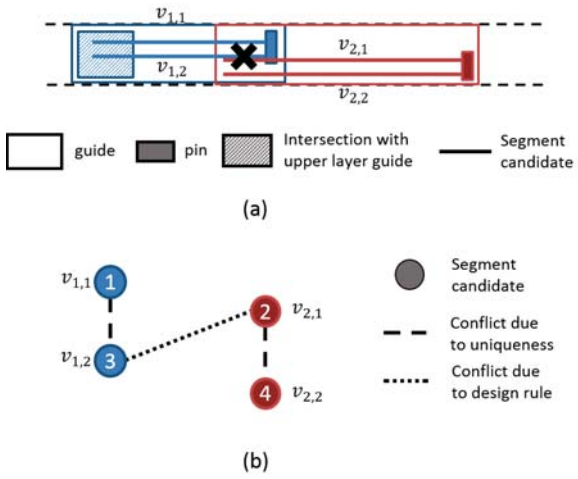

Figure 5: (a) Inter-net conflict in a panel and (b) conflict graph.

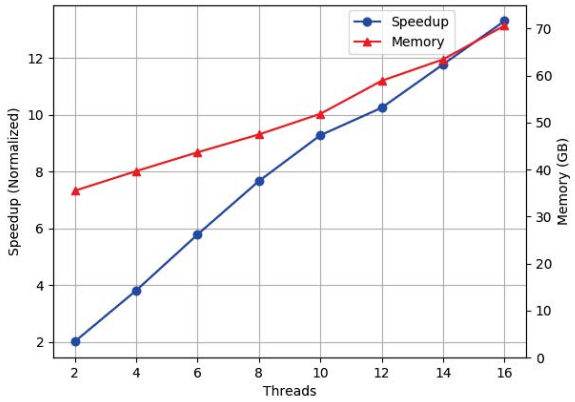

Figure 6: Sensitivities of runtime and peak memory usage to the number of threads.

this example consists of $v_{1,2}$ and $v_{2,2}$, which yields the maximum weighted independent set with weight of 7. In our implementation, the weight for each candidate is determined by the sum of its track-to-AP distances, i.e., a smaller track-to-AP distance usually indicates a smaller wirelength for wire segments on the lower and upper layers, and thus the candidate is given a larger weight. If a segment cannot be routed on layer l, we use layer $l + 2$ for this segment, and we always guarantee at least a minimum-area segment and at least one legal via location for each layer from l to $l + 2$.[2]

5 EXPERIMENTS

We implement our router in C++ with LEF/DEF reader/writer parsers [31], Boost C++ libraries 1.66.0 [28], CPLEX 12.8.0 [30] as our MILP solver, and OpenMP [32] to enable thread-level parallelism. We perform experiments using the ISPD-2018 benchmark suite [19]. Testcase information is summarized in Table 3. All experiments are performed on a 2.6GHz Intel Xeon dual-CPU server. Experimental results shown in Table 4 and Table 5 are from experiments performed with 8 threads.

To assess the scalability of our detailed router, we sweep the number of threads and study the impact on runtime and the peak memory usage. We sweep the number of threads from 2 to 16 with a step size of 2 and perform detailed routing on the largest contest testcase *ispd18_test10*. We normalize the runtime speedup to the single-thread case. Figure 6 shows the results of this scalability study. We observe that both the runtime speedup and peak memory usage increase linearly as the number of threads increases.

In the contest, a routing solution is treated as valid only if the memory and runtime usage are under 64GB and 12 hours, respectively, without any open nets. All of our reported solutions meet these requirements. The quality of result is measured by a raw score (sum of weighted metrics). The weights, reproduced from the ISPD-2018 contest specification, are summarized in Table 2.

[2] After routing all layers and all panels, we greedily perform shortest-path routing for the open nets, at the cost of design rule violations.

Table 2: Evaluation metrics.

Metric	Weight
Short metal area	500
#spacing violations	500
#min-area violations	500
Total length of wires outside of routing guides	1
Total #vias outside of routing guides	1
Total length of off-track wires	0.5
Total #off-track vias	1
Total length of wrong-way wires	1
Total #vias	2
Total length of wires	0.5

We compare our TritonRoute's detailed routing solutions to the winning routing solution for each benchmark design in the ISPD-2018 contest. Two contest-winning solutions (ispd18_test4 and ispd18_test7) come from Dr. CU, and all the other eight solutions come from TritonRoute (ISPD contest version). Experimental results are summarized in Table 4 and Table 5. We report all metrics used in ISPD-2018 contest, for which lower values are better. Metrics can be divided into three categories including *routing, guides and tracks obedience* and *design rule violations*.

For the winning detailed routing solutions from the ISPD-2018 contest, design rule violations take 58.08% of the overall score on average. Our router achieves up to 93.85% reduction in design rule violation metric, and on average 63.71% reduction in design rule violation metric, versus the ISPD-2018 winning solutions.

Overall, our router achieves up to 73.83%, and on average 50.80%, reduction in raw scores compared to the winning solutions. Also, design rule violations account for only 21.29% of the overall score on average.

6 CONCLUSION

In this work, we present TritonRoute, an initial detailed router for the ISPD-2018 contest. We describe our parallel, MILP-based routing flow, and demonstrate that we achieve up to 94% reduction in design rule violations, and up to 74% total raw score reduction, compared to the first-place contest solutions of all contest benchmarks. Our ongoing works include: (i) refinement of overall flow with better handling of routing hotspots, including consideration of routing detours; (ii) optimization of routing topology to reduce wirelength and via count; (iii) improvement of pin accessibility; (iv) runtime and memory usage optimizations; (v) support of more design rules towards complete handling of detailed routing in sub-16/14nm advanced foundry nodes; and (vi) support of pre-routed power-ground (PG) mesh.

Finally, we show a small "preview" of academic P&R capability for the foundry 16/14nm technology node – i.e., utilizing academic placers and routers. The overall flow is shown in Figure 7(a). We synthesize an Arm Cortex-M0 using *Synopsys Design Compiler L-2016.03-SP4 [33]* and perform floorplanning using *Cadence Innovus Implementation System v17.1 [29]*. We then perform place-and-route using purely academic P&R tools including RePlAce [3][17] global placer (with embedded NTUplace3 [2] detailed placer), NCTU-GR [16] global router, and TritonRoute detailed router. Figure 7(b) shows the layout of M0 after detailed routing. It is well-understood that academic tools will not (and, should not) reproduce the broad and well-seasoned functionality of commercial P&R tools. However, the rapidly improving feasibility of such academic P&R flows validates how well the ISPD-2018 contest and its predecessors have driven researchers to develop tooling that is "close" to being able to handle real-world design instances in production design enablements.

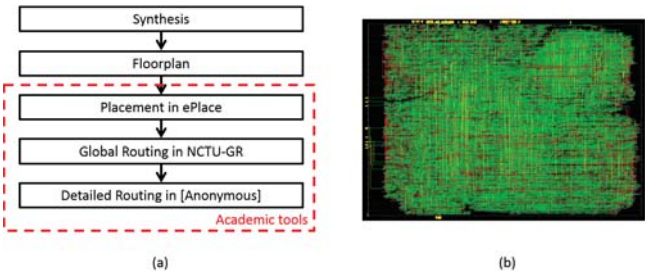

Figure 7: (a) IC design flow including academic P&R tools; and (b) Layout of Cortex-M0.

ACKNOWLEDGMENTS

We thank the authors of [2], [3], [16] and [17] for providing executables of their tools that enabled demonstration of the "preview" academic flow in Section 6 above. Research support from Qualcomm, Samsung, NXP Semiconductors, and NSF (CCF-1564302) is gratefully acknowledged.

REFERENCES

[1] H.-Y. Chen and Y.-W. Chang, "Global and Detailed Routing", Chapter 12 in Wang, Chang and Cheng (Eds.), *Electronic Design Automation: Synthesis, Verification, and Test*, Morgan Kaufmann, 2009, pp. 687-749. http://cc.ee.ntu.edu.tw/ ywchang/Courses/PD_Source/EDA_routing.pdf
[2] T.-C. Chen, Z.-W. Jiang, T.-C. Hsu, H.-C. Chen and Y.-W. Zhang, "NTUplace3: An Analytical Placer for Large-Scale Mixed-Size Designs with Preplaced Blocks and Density Constraints", *IEEE Trans. on CAD* 27(7) (2008), pp. 1228-1240.
[3] C.-K. Cheng, A. B. Kahng, I. Kang and L. Wang, "RePlAce: Advancing Solution Quality and Routability Validation in Global Placement", *IEEE Trans. on CAD*, to appear. DOI: 10.02109/TCAD.2018.2859220
[4] Y. Ding, C. Chu and W.-K. Mak, "Self-Aligned Double Patterning Lithography Aware Detailed Routing with Color Preassignment", *IEEE Trans. on CAD* 36(8) (2017), pp. 1381-1394.
[5] Y. Du, Q. Ma, H. Song, J. Shiely, G. Luk-Pat, A. Miloslavsky and M. D. F. Wong, "Spacer-is-Dielectric-Compliant Detailed Routing for Self-Aligned Double Patterning Lithography", *Proc. DAC*, 2013, pp. 1-6.
[6] A. Feller, "Automatic Layout of Low-Cost Quick-Turnaround Random-Logic Custom LSI Devices", *Proc. DAC*, 1976, pp. 79-85.
[7] G.-R. Gao and D. Z. Pan, "Flexible Self-Aligned Double Patterning Aware Detailed Routing with Prescribed Layout Planning", *Proc. ISPD*, 2012, pp. 25-32.
[8] M. Gester, D. Muller, T. Nieberg, C. Panten, C. Schulte and J. Vygen, "BonnRoute: Algorithms and data structures for fast and good VLSI routing", *ACM TODAES* 18(2) (2013), pp. 32:1-32:24.
[9] F. O. Hadlock, "A Shortest Path Algorithm for Grid Graphs", *Networks* 7(4) (1977), pp. 323-334.
[10] K. Han, A. B. Kahng and H. Lee, "Evaluation of BEOL Design Rule Impacts Using an Optimal ILP-Based Detailed Router", *Proc. DAC*, 2015, pp. 68:1-68:6.
[11] D. W. Hightower, "A Solution to Line-Routing Problems on the Continuous Plane", *Proc. DAC*, 1969, pp. 1-24.
[12] L. Kou, G. Markowsky and L. Berman, "A Fast Algorithm for Steiner Trees", *Acta Informatica* 15(2) (1981), pp. 141-145.
[13] C. Y. Lee, "An Algorithm for Path Connections and Its Applications", *IRE Trans. on Electro. Comp.* 10(3) (1961), pp. 346-365.
[14] H. K.-S. Leung, "Advanced Routing in Changing Technology Landscape", *Proc. ISPD*, 2003, pp. 118-121.
[15] I.-J. Liu, S.-Y. Fang and Y.-W. Chang, "Overlay-Aware Detailed Routing for Self-Aligned Double Patterning Lithography Using the Cut Process", *IEEE Trans. on CAD* 35(9) (2016), pp. 1519-1531.
[16] W.-H. Liu, W.-C. Kao, Y.-L. Li and K.-Y. Chao, "NCTU-GR 2.0: Multithreaded Collision-Aware Global Routing with Bounded- Length Maze Routing", *IEEE Trans. on CAD* 32(5) 2013, pp. 709-722.
[17] J. Lu, H. Zhuang, P. Chen, H. Chang, C.-C. Chang, Y.-C. Wong, L. Sha, D. Huang, Y. Luo, C.-C. Teng and C.-K. Cheng, "ePlace-MS: Electrostatics-Based Placement for Mixed-Size Circuits", *IEEE Trans. on CAD* 34(5) 2015, pp. 685-698.
[18] W. K. Luk, "A Greedy Switch-Box Router", *Integration: The VLSI Journal* 3(2) (1985), pp. 129-149.
[19] S. Mantik, G. Posser, W.-K. Chow, Y. Ding and W.-H. Liu, "ISPD2018 Initial Detailed Routing Contest and Benchmarks", *Proc. ISPD*, 2018, pp. 140-143.
[20] T. Nieberg, "Gridless Pin Access in Detailed Routing", *Proc. DAC*, 2011, pp. 170-175.
[21] N. J. Nilsson, "State-Space Search Methods", in *Problem-Solving Methods in Artificial Intelligence*, McGraw-Hill Book Co., 1971, pp. 43-79.
[22] I. Pohl, "Bi-Directional Search", *Machine Intelligence* (1971), pp. 127-140.
[23] J. Soukup, "Fast Maze Router", *Proc. DAC*, 1978, pp. 100-102.
[24] E. Shragowitz and S. Keel, "A Global Router Based on a Multicommodity Flow Model", *Integration: The VLSI Journal* 5(1) (1987), pp. 3-16.
[25] P.-S. Tzeng, and C. H. Sequin, "Codar: A Congestion-Directed General Area Router", *Proc. ICCAD*, 1988, pp. 30-33.
[26] X. Xu, B. Yu, J. R. Gao, C.-L. Hsu and D. Z. Pan, "PARR: Pin-Access Planning and Regular Routing for Self-Aligned Double Patterning", *ACM TODAES* 21(3) (2016), pp. 42:1-42:21.
[27] Y. Zhang and C. Chu, "RegularRoute: An Efficient Detailed Router Applying Regular Routing Patterns", *IEEE Trans. on VLSI* 21(9) (2013), pp. 1655-1668.
[28] B. Schäling, *The Boost C++ Libraries, 2nd ed.*, XML Press, 2014.
[29] Cadence Innovus User Guide, http://www.cadence.com
[30] IBM ILOG CPLEX. www.ilog.com/products/cplex/
[31] LEF/DEF reference 5.7. http://www.si2.org/openeda.si2.org/projects/lefdefnew
[32] OpenMP Architecture Review Board, "OpenMP Application Program Interface, Version 4.0".
[33] Synopsys Design Compiler User Guide, http://www.synopsys.com
[34] W.-H. Liu, "ISPD 2018 Initial Detailed Routing Contest and Benchmarks" *presentation slides*, http://www.ispd.cc/slides/2018/s7_3.pdf

Table 3: Benchmark information.

Benchmark	#std	#blk	#net	#pin	#layer	Die size	Tech. node
ispd18_test1	8879	0	3153	0	9	$0.20\times0.19mm^2$	$45nm$
ispd18_test2	35913	0	36834	1211	9	$0.65\times0.57mm^2$	$45nm$
ispd18_test3	35973	4	36700	1211	9	$0.99\times0.70mm^2$	$45nm$
ispd18_test4	72094	0	72401	1211	9	$0.89\times0.61mm^2$	$32nm$
ispd18_test5	71954	0	72394	1211	9	$0.93\times0.92mm^2$	$32nm$
ispd18_test6	107919	0	107701	1211	9	$0.86\times0.53mm^2$	$32nm$
ispd18_test7	179865	16	179863	1211	9	$1.36\times1.33mm^2$	$32nm$
ispd18_test8	191987	16	179863	1211	9	$1.36\times1.33mm^2$	$32nm$
ispd18_test9	192911	0	178857	1211	9	$0.91\times0.78mm^2$	$32nm$
ispd18_test10	290386	0	182000	1211	9	$0.91\times0.87mm^2$	$32nm$

Table 4: Comparison of total wire length, total via count, out-of-guide (OOG) wire, out-of-guide vias, off-track (OT) wire, off-track via, and wrong-way (WW) wire between TritonRoute (column A) and the first-place ISPD-2018 contest solution (column B). % represents the proportion in the total raw score. Columns in guides and tracks obedience are not noticeable contributors to the total raw score. Thus, we only show their percentage contributions to maintain readability of the table.

Benchmark	Routing								Guides and tracks obedience									
	Total wire length				Total via count				OOG wire		OOG vias		OT wire		OT via		WW wire	
	A		B		A		B		A	B	A	B	A	B	A	B	A	B
	(μm)	%	(μm)	%	#	%	#	%	%	%	%	%	%	%	%	%	%	%
ispd18_test1	463518	61.1%	472032	61.1%	38777	20.4%	41641	21.6%	1.7%	1.6%	0.4%	0.4%	0.0%	0.5%	0.0%	0.0%	0.0%	0.9%
ispd18_test2	8097031	72.0%	8150587	72.3%	385111	13.7%	409551	14.5%	1.1%	1.3%	0.2%	0.2%	0.0%	0.2%	0.0%	0.0%	0.0%	0.3%
ispd18_test3	9013949	64.6%	9086138	52.5%	389718	11.2%	427410	9.9%	0.6%	0.8%	0.0%	0.0%	0.1%	0.2%	0.0%	0.0%	0.0%	0.2%
ispd18_test4	27165618	52.6%	26127059	27.3%	847643	6.6%	757949	3.2%	1.0%	1.6%	0.2%	0.0%	0.3%	0.1%	0.0%	0.0%	0.0%	0.2%
ispd18_test5	29163974	67.4%	29415618	22.5%	1098466	10.2%	1158945	3.6%	0.9%	0.5%	0.1%	0.0%	0.0%	0.0%	0.0%	0.0%	0.1%	0.3%
ispd18_test6	37844297	65.2%	38191983	21.4%	1700072	11.7%	1800286	4.0%	1.0%	0.5%	0.2%	0.0%	0.0%	0.1%	0.1%	0.0%	0.1%	0.3%
ispd18_test7	68562327	67.7%	64907309	16.8%	2756017	10.9%	2341148	2.4%	1.0%	0.9%	0.2%	0.0%	0.1%	0.0%	0.0%	0.0%	0.1%	0.1%
ispd18_test8	68872750	69.4%	69559381	21.5%	2767035	11.2%	2929578	3.6%	1.0%	0.6%	0.2%	0.1%	0.1%	0.1%	0.0%	0.0%	0.1%	0.3%
ispd18_test9	58157640	69.6%	58803452	20.4%	2760482	13.2%	2920259	4.0%	1.1%	0.6%	0.2%	0.0%	0.1%	0.1%	0.0%	0.0%	0.1%	0.3%
ispd18_test10	71528779	63.5%	72244024	18.6%	2940555	10.4%	3110163	3.2%	1.4%	0.7%	0.1%	0.0%	0.2%	0.2%	0.0%	0.0%	0.1%	0.2%

Table 5: Comparison of area of metal shorts, #min-area violations, #spacing violations, runtime and total raw score between TritonRoute (column A) and the first-place ISPD-2018 contest solution (column B). % represents the proportion in the total raw score.

Benchmark	Design rule violations												Total				
	Area of metal shorts				#min-area violations				#spacing violations				Runtime		Raw score		
	A		B		A		B		A		B		A	B	A	B	A/B
	(μm^2)	%	(μm^2)	%	#	%	#	%	#	%	#	%	(sec)	(sec)			
ispd18_test1	1.25	0.2%	0.74	0.1%	0	0.0%	0	0.0%	123	16.2%	107	13.9%	121	207	379530	386188	0.98×
ispd18_test2	36.25	0.3%	94.93	0.8%	1	0.0%	1	0.0%	1419	12.6%	1158	10.3%	797	1514	5626234	5636273	0.99×
ispd18_test3	1507.19	10.8%	4891.36	28.3%	0	0.0%	0	0.0%	1755	12.6%	1387	8.0%	1718	2019	6971806	8645534	0.80×
ispd18_test4	17088.59	33.1%	39413.62	41.4%	54	0.1%	51	0.1%	3130	6.1%	25309	26.4%	6795	326	25825192	47890529	0.53×
ispd18_test5	1730.21	4.0%	28428.68	21.8%	83	0.2%	28	0.0%	7350	17.0%	66742	51.2%	2757	1914	21629234	65227108	0.33×
ispd18_test6	1731.71	3.0%	31227.46	17.5%	65	0.1%	15	0.0%	10763	18.6%	100196	56.1%	4098	3107	29007819	89303689	0.32×
ispd18_test7	8998.97	8.9%	58654.34	15.2%	85	0.1%	529	0.1%	11266	11.1%	249453	64.4%	7057	1050	50654788	193575674	0.26×
ispd18_test8	6747.99	6.8%	76789.58	23.8%	74	0.1%	48	0.0%	11028	11.1%	161229	49.9%	5648	6262	49588075	161426598	0.30×
ispd18_test9	2416.86	2.9%	56580.75	19.6%	112	0.1%	40	0.0%	10668	12.8%	1158305	54.9%	6570	5128	41806853	144221466	0.28×
ispd18_test10	12729	11.3%	120966.00	31.2%	137	0.1%	33	0.0%	14328	12.7%	177426	45.8%	7877	5554	56291381	193867714	0.29×