A Multithreaded Initial Detailed Routing Algorithm Considering Global Routing Guides*

Fan-Keng Sun[1], Hao Chen[1], Ching-Yu Chen[1], Chen-Hao Hsu[1], and Yao-Wen Chang[1,2]
[1]Department of Electrical Engineering, National Taiwan University, Taipei 10617, Taiwan
[2]Graduate Institute of Electronics Engineering, National Taiwan University, Taipei 10617, Taiwan
{b03901056, b03901016, b03901152, b03505028, ywchang}@ntu.edu.tw

ABSTRACT

Detailed routing is the most complicated and time-consuming stage in VLSI design and has become a critical process for advanced node enablement. To handle the high complexity of modern detailed routing, initial detailed routing is often employed to minimize design-rule violations to facilitate final detailed routing, even though it is still not violation-free after initial routing. This paper presents a novel initial detailed routing algorithm to consider industrial design-rule constraints and optimize the total wirelength and via count. Our algorithm consists of three major stages: (1) an effective pin-access point generation method to identify valid points to model a complex pin shape, (2) a via-aware track assignment method to minimize the overlaps between assigned wire segments, and (3) a detailed routing algorithm with a novel negotiation-based rip-up and re-route scheme that enables multithreading and honors global routing information while minimizing design-rule violations. Experimental results show that our router outperforms all the winning teams of the 2018 ACM ISPD Initial Detailed Routing Contest, where the top-3 routers result in 23%, 52%, and 1224% higher costs than ours.

CCS CONCEPTS

• **Hardware → Electronic design automation**;

KEYWORDS

Physical Design, Routing, Detailed Routing, Multithread, Advance Technologies

1 INTRODUCTION

Routing is the most complicated and time-consuming stage in the VLSI design flow and has become one of the most critical process for advanced node realization. As circuit designs evolve into the deep nanometer era, complex design rules and requirements become common, making the routing process much more challenging than ever. As the technology advances, the long-standing routability issue imposes more difficult challenges and thus attracts higher attention. In order to cope with routability in the early design stages, previous works [11, 12] handled the detailed-routing-driven placement problem with routability and congestion awareness. The works [15, 16] estimated the feasibility of routing according to the information of congestion and wiring.

To handle the high complexity of modern routing, the divide-and-conquer technique is often employed to further divide the routing process into stages, each stage handles specific routing tasks, and all routing constraints and objectives are addressed and refined stage by stage to achieve a desired final routing solution.

Traditionally, the routing problem is divided into two stages: global routing and detailed routing. The goals of the two stages are balancing the routes of all routing layers and regions, and assigning connections to specific routing tracks, respectively. In the global routing stage, the whole routing region is usually divided into a set of global routing bins, called *g-cells*, and *global routing guides* (i.e. rough routing solutions for each net) are generated by the edge connections on a global routing grid graph. The global routing guide provides information for the subsequent detailed routing stage to complete the final routing.

The detailed routing stage completes the final routing considering various design constraints, while honoring the solution from global routing because the routing results generated by a global router [4, 14] are often optimized for some predefined cost metrics (e.g., routability, timing, manufacturability, reliability, and slew). Figure 1 shows an example detailed routing result with and without honoring the global routing guide. In modern circuit design, detailed routing becomes more complex and important and is even a dead-or-alive critical process because it needs to address all the design constraints and objectives to make the final routing solution. As a result, modern detailed routing is further divided into two stages: initial detailed routing and design-rule-checking (DRC) refinement. First, the initial detailed routing step handles the routing with exact geometrical constraints while honoring the results of global routing. Then the DRC refinement step is performed to eliminate the design rule violations and refine the final routing solution.

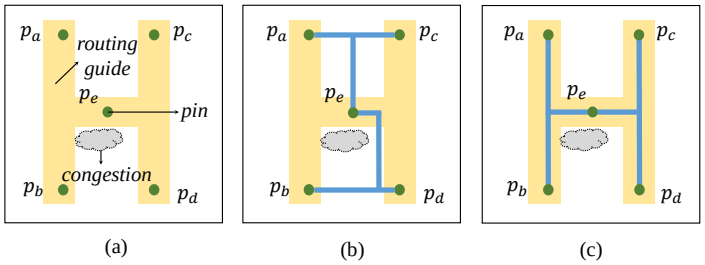

Figure 1: (a) A global routing result of a net with the source pin p_e and sink pins p_a, p_b, p_c, and p_d, with a local congestion region, which was well optimized for the timing metric. (b) A detailed routing result focusing on wirelength optimization without considering the global routing guide, where we can observe that $p_e \rightsquigarrow p_b$ is a critical path which causes a timing degradation. (c) A detailed routing result honoring the global routing guide, which optimizes timing with the shorter $p_e \rightsquigarrow p_b$ critical path, yet larger total wirelength.

However, the traditional practice of dividing the routing stage into global routing and detailed routing might be problematic. Batterywala *et al.* [3] pointed out that there exists a large gap between the global routing stage and the detailed one, due to the reason that global routing overlooks some detailed routing issues such as pin access, local net wiring, and some complicated via rules. To handle the big mismatch between global routing and detailed routing, Batterywala *et al.* [3] proposed a track assignment stage that is performed between global routing and detailed routing, so as to minimize the gap between the two traditional routing stages and reduce the high computational complexity in detailed routing. As a result, the routing flow is modified to three stages: global routing, track assignment, and detailed routing. Several frameworks have been proposed in the literature to address the track assignment problem. Wong *et al.* [20] introduced a novel negotiation-based track assignment technique to solve the problem efficiently. Shi *et al.* [18] proposed a track assignment framework with fast analysis of track congestion inside each g-cell at the global routing stage, and an effective feature of estimating the locations of vias and partial tracks inside each g-cell.

Although the track assignment stage minimizes the gap between global routing and detailed routing effectively, the complexity of detailed routing remains very high because it requires to handle so many final routing issues and clean up all tedious design constraint violations. To deal with the complex

*This work was supported by AnaGlobe, TSMC, MOST of Taiwan under Grant No's MOST 105-2221-E-002-190-MY3, MOST 106-2911-I-002-511, and MOST 107-2221-E-002-161-MY3.

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

modern detailed routing problem, as a result, the initial detailed routing process is usually applied to finish connecting all nets and minimize the DRC violations so that the later DRC refinement step will not cause significant troubles to the final routing, and the optimized metrics in the previous global routing stage is preserved.

To stimulate the research on this complex topic, the 2018 ACM ISPD held the Initial Detailed Routing Contest [2], which is the first contest on the detailed routing problem. This contest introduces practical industrial benchmarks with design rules defined in LEF files [1] as a standard for evaluating the solution quality of an initial detailed router because of the lack of unified objectives in current detailed routing works [5, 6, 13, 19, 22]. Since the benchmarks provided in this contest [2] offer only global routing solutions instead of track assignment results, we integrate a track assignment technique to our initial detailed routing framework. Besides, we propose a more robust evaluation metric to consider more reasonable manufacturing issues, while avoiding abusing the contest metric by any trick not beneficial to real design needs.

1.1 Our Contributions

In this paper, we propose a complete initial detailed routing framework to consider some industrial DRC constraints defined in LEF files [1] and some routing preference metrics while honoring a given global routing guide. The main contributions of this paper are summarized as follows:

- An effective pin-access generation algorithm is proposed to avoid local congestion and DRC violations that could affect the later track assignment and detailed routing stages.
- A track assignment technique considering via locations is presented to generate an initial solution from the given global routing guide for the subsequent detailed routing stage, and further reduce its computational complexity.
- A two-stage multi-threaded negotiation-based initial detailed routing algorithm which integrates real industrial constraints is proposed. The multi-threaded technique can reduce the runtime almost proportional to the number of threads.
- Experimental results show that our algorithm outperforms all the participating routers of the 2018 ISPD Contest [17] in solution quality. Specifically, the top-3 routers of the contest result in 23%, 52%, and 1224% higher costs than ours.
- We propose to make the evaluation metric of the 2018 ISPD Contest even more robust by preventing any trick not beneficial to real design needs from abusing the metric.

The remainder of this paper is organized as follows. Section 2 describes some DRC constraints and routing preference metrics used in the contest [2] and formulates the initial detailed routing problem. Section 3 details our algorithm. Section 4 shows the experimental results, and Section 5 concludes this paper.

2 PRELIMINARIES

In this section, we introduce some real industrial DRC constraints and the routing preference metrics considered in the 2018 ISPD Contest [17], and formulate the initial detailed routing problem.

2.1 Industrial DRC constriants

Design rules for routing, defined in LEF files [1], need to be considered so that the resulting routing solutions can satisfy manufacturing requirements. We consider the routing rules defined in the ISPD 2018 Initial Routing Contest [2], which is detailed in the following.

2.1.1 Min-area Rule. The *min-area rule* indicates that the area of each polygon is required to be greater or equal to the value specified in LEF files [1]. A *patch metal* can be appended to a routed wire in order to increase the area of the wire if it is too small to meet the min-area rule. Figure 2(a) illustrates a feasible solution to fix a min-area violation by adding a patch metal to the wire whose area does not meet the requirement.

2.1.2 End-of-line (EOL) Spacing. The *end-of-line (EOL)* spacing rule specifies the minimum spacing to an edge as follows. If an *end-of-line (EOL)* edge is shorter than *eolWidth*, the end-of-line spacing rule is required to preserve a spacing greater than or equal to *eolSpace* beyond the EOL anywhere less than the *eolWithin* distance. Figure 2(b) shows an example of the end-of-line spacing rule.

2.1.3 Cut Spacing. By specifying a minimum spacing between via cuts on the same net or different nets on cut layers, the cut spacing rule guarantees that two different vias are not too close to each other.

2.1.4 Spacing Table. According to the parallel-run length of two objects, a spacing table specifies the minimum required spacing between the two objects. Figure 2(c) shows an example of the spacing table constraint.

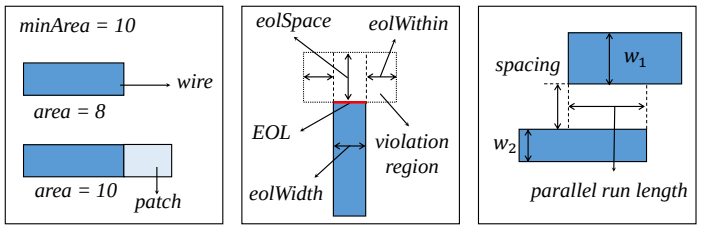

Figure 2: Examples of design rule violations. (a) The upper wire violates the min-area rule, while the lower wire with patch metal satisfies the min-area rule. (b) Example of an end-of-line violation. The dotted-line region shows the violation region. (c) The spacing between two objects should be greater than or equal to the specified value according to the parallel-run length.

2.2 Routing Preference Metrics

In addition to the DRC constraints mentioned in the previous subsection, metrics such as global routing guide honoring, wrong-way routing, and off-track routing, are also used to evaluate the quality of a detailed routing solution in the 2018 ISPD Contest [2]. Though these routing preference metrics are not hard constraints, they lead to better routing results. The following subsections give the routing preference metrics considered in the contest.

2.2.1 Global Routing Guide Honoring. Honoring the global routing guide in the initial detailed routing stage can facilitate subsequent detailed routing processes. Figure 3(a) shows an example with a routing pattern containing out-of-guide wires. If the center lines of wires or the coordinates of vias are outside the routing guide, extra penalty will be added to the evaluation score with respect to the length of the center lines and the number of vias outside the routing guide.

2.2.2 Wrong-way Routing Minimization. For each layer defined in a LEF file, there is a preferred routing direction (either horizontal or vertical), and two adjacent layers always have orthogonal preferred directions. If a wire is horizontally (vertically) routed on a layer with its preferred direction being vertical (horizontal), the wire is considered a wrong-way wire, as shown in Figure 3(b). The length of the wrong-way wire will incur an extra penalty in the evaluation score.

2.2.3 Off-track Routing Minimization. Each layer defined in a LEF file consists of horizontal and vertical track structures. An off-track wire is a routing wire not aligning with a track, and an off-track via is a via whose coordinate not aligning with a track on its bottom or top layer. Figure 3(b) shows a routing pattern with an off-track wire and an off-track via. An extra penalty in the evaluation score will be applied according to the length of off-track wires and the number of off-track vias.

2.3 Problem Formulation

We formally define the initial detailed routing problem here.

- **The Initial Detailed Routing Problem:** Given a set of k nets $N = \{n_1, \ldots, n_i, \ldots, n_k\}$ and a set of k global routing guides $G = \{g_1, \ldots, g_i, \ldots, g$ generate a routing solution for each net $n_i \in N$ considering $g_i \in G$ such that n_i is connected. The following metrics should also be optimized simultaneously: (1) the total wirelength of all nets, (2) the cost of the total used vias, (3) the number of DRC violations, and (4) the routing preference metrics.

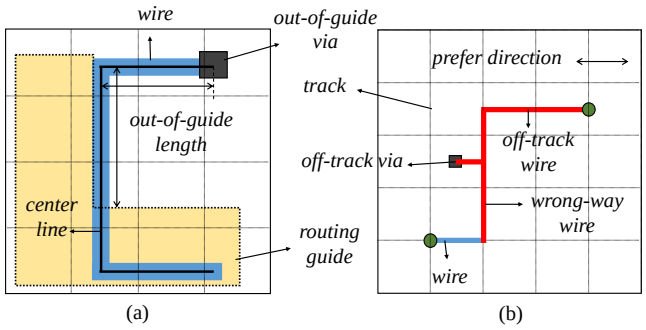

Figure 3: Examples of routing preference metrics. (a) A wire is out-of-guide if its center line lies outside the global routing guide region, and a via is out-of-guide if its coordinate is not in the guide region. (b) A wire is off-track if it is not aligned with the track, and a via is off-track if its coordinate does not lie on a track.

3 PROPOSED ALGORITHM

Detailed routing is considered the most time-consuming and the most complicated stage due to its huge problem size and the complicated DRC rules. Therefore, we propose an algorithm to achieve better trade-off between solution quality and efficiency by the following techniques: (1) a pin-access generation method identifying proper pin-access points to model complicated pin shapes and reducing the complexity of the problem, (2) a negotiation-based via-location-aware track assignment process assigning straight wires to the tracks while minimizing the overlaps on each track and the total estimated wire length, and (3) a multi-threaded two-stage detailed routing algorithm connecting all the non-fully-connected nets considering the important metrics mentioned in Section 2.3. Fig. 4 summarizes the overall flow of our proposed algorithm. The following subsections detail the three stages.

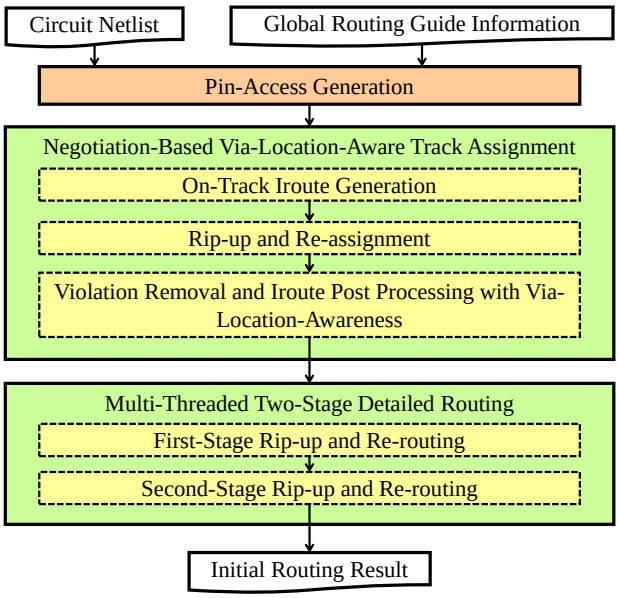

Figure 4: Overview of our algorithm.

3.1 Pin-Access Generation

In order to reduce the complexity of the initial detailed routing problem, we utilize a pin-access generation method with fast invalid point identification to model a complicated pin shape.

To properly formulate the process of pin access, we adopt the concept and the terminologies *hit points* and *valid hit points* in [21]. However, we modified the definitions of the two terminologies as follows.

Algorithm 1 *PinAccess(D, P)*

Input: D: the input design P: the set of all pins
Output: A: pin-access point set for every pin

```
1:  K_extend ⟵ 0, S_used ⟵ φ, A ⟵ φ
2:  while P is not empty
3:    for p ∈ P
4:      S ⟵ GetCandidateHitPoints(p, K_extend)
5:      for s ∈ S
6:        if s ∈ S_used or s is invalid
7:          S ⟵ S \ s
8:      if S is no empty
9:        P ⟵ P \ p
10:       S ⟵ MaxIndependentSet(S)
11:       S_used ⟵ S_used ∪ S
12:       A ⟵ A ∪ (p, S)
13:   K_extend ⟵ K_extend + 1
14: return A
```

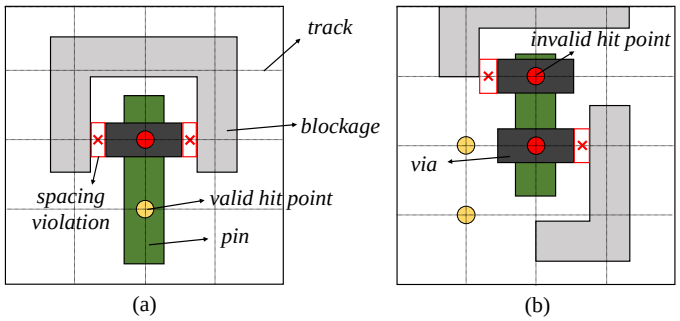

Figure 5: (a) The invalid hit point incurs two design rule violations, and the valid hit point is picked to model the pin shape. (b) The valid hit points outside the pin shape are selected for the lack of hit points inside the pin shape.

Definition 1. *A* hit point *on a layer is the intersection of tracks with preferred direction on this layer and tracks with a preferred direction on adjacent layers. Thus, all hit points on a single layer form a vertex set of a grid graph, and two hit points are considered too close if there exists an edge between them in the grid graph. In our pin-access algorithm, two pin shapes never share the same hit point, but a pin shape may have many hit points.*

Definition 2. *A hit point is* valid *if we can place at least one type of vias on that point, without incurring any design rule violation; otherwise, the hit point is considered* invalid.

Fig. 5(a) illustrates an example of valid hit points and invalid hit points.

Our pin-access algorithm is illustrated in Algorithm 1. Given a pin p, when K_{extend} is zero, we first try to find all valid hit points within the pin shape. If there exist such valid hit points, then this pin has found its own candidate pin-access points S. Otherwise, we will expand the search area of valid hit points. Since hit points are intersections of orthogonal tracks, we expand one track space at one time. Also, notice that one valid hit point can only belongs to at most one pin shape.

After candidate pin-access points are selected, we need to prune out pin-access points that are too close while retaining as more access points as possible. This step not only reduces the number of spacing violations, but also frees up possible pin-access points for other pins. We formulate this step as a maximum independent set problem. Since the grid graph is a bipartite graph, the induced subgraph of a grid graph by candidate pin-access points S is also bipartite. According to König's theorem, a maximum matching on a bipartite graph is equivalent to its minimum vertex cover, which is the complement of its maximum independent set. Thus, this step can be performed in $O((V + E)\sqrt{V})$, where V is number of vertices and E is number of edges, according to the Hopcroft-Karp algorithm [9].

3.2 Negotiation-Based Via-Location-Aware Track Assignment

After pin-access generation, partial routing solutions can be constructed based on the given global routing guide by the track assignment technique. To reduce the problem complexity for the following detailed routing stage, we integrate the track assignment stage mentioned in Section 1 to our initial detailed routing framework. The three main steps of our track assignment scheme is detailed as follows:

3.2.1 On-Track Iroute Generation. The global routing guide information given in the 2018 ISPD Contest benchmarks only specifies the desired routing region which consists of rectangular regions; however, the tree structure of a global routing net (i.e., a g-cell to g-cell connection) is not given. As a result, we extract a tree structure from the given desired routing region for each net.

In order to generate the initial track assignment, we first construct a graph $H(U, E)$, where the vertex set U represents the set of all rectangular regions, and the edge set $E = \{(u_1, u_2) | u_1, u_2 \in U,$ and u_1 is adjacent to $u_2\}$, while u_1 is adjacent to u_2 if the two rectangular regions can be connected by a via or a single horizontal (vertical) wire. Then we assign the weight of each edge $e \in E$ according to the three types of edges: a normal wire, a wrong-way wire, and a via, with the weight being δ, δ_{ww}, and δ_{via}, respectively.

Since the utilization of wrong-way wires should be minimized, δ_{ww} should be higher than δ. Similarly, using a via incurs a higher cost than using a normal wire; therefore, δ_{via} is desired to be higher than δ. In our implementation, δ, δ_{ww}, and δ_{via} are set to 1, 2, and 4, respectively.

Definition 3. *An* iroute *is a straight wire that passes through one or more g-cells.*

After the graph H is constructed, a minimum spanning tree algorithm is applied to generate a tree structure of the global routing guide. An *iroute*, extended along the preferred direction of the layer, is placed on a track inside the desired routing region belonging to a vertex of the minimum spanning tree. The length of each iroute is extended to the boundary of the region where it was placed. Figure 6(a) shows an example of an initial track assignment result.

3.2.2 Rip-up and Re-assignment. After the initial assignment, we adopt negotiation-based track assignment [20] with the modified cost function to rip-up and re-assign iroutes to minimize total wirelength and overlaps. The original cost function for re-assignment described in [20] is

$$C_t = \beta_{wl} \cdot C_{wl} + \beta_{ol} \cdot C_{ol} + \beta_{bk} \cdot C_{bk} + \beta_{his} \cdot C_{his}, \quad (1)$$

where C_t is the total cost of track t, C_{wl} is the wirelength cost, C_{ol} is the overlap length, C_{bk} is the blockage cost, C_{his} is the history cost, and β_{wl}, β_{ol}, β_{bk}, and β_{his} are weighted parameters which vary in each iteration of rip-up and re-assignment. We adopt the definitions of the overlap cost, the blockage cost, and the history cost from the work [20]. However, because the original cost function does not consider via location, we modify the wirelength cost in [20] to the sum of the total estimated length of trimmed iroutes. In our work, an iroute will be trimmed according to the positions of its adjacent iroutes. Fig. 6(b) demonstrates an example of trimmed iroutes.

3.2.3 Via-Location-Aware Violation Removal and Iroute Post Processing.

Definition 4. *A* net component *is a set of connected wires, vias, and/or pins.*

In this stage, all iroutes overlapping with blockages or other iroutes will be removed. It is desirable to maximize the total length of remaining iroutes because the longer an iroute is, the more possible to detour in the following detailed routing stage. To properly remove the overlapping wires, we apply the maximum weight independent set (MWIS) algorithm on an interval graph [10], which has been proved to be optimal with $O(n \log n)$ time complexity. According to [10], the weight of an iroute is defined as its length. Moreover, if two iroutes of adjacent layers cross in the view of a plane, a via will be inserted to connect them. Finally, the connected pins, vias, and wires of the same net are grouped together into a net component, and all redundant wires are trimmed away, which is illustrated in Fig. 6(b).

3.3 Multi-Threaded Two-Stage Detailed Routing

In this subsection, we detail our multi-threaded two-stage detailed routing algorithm by introducing some definitions first.

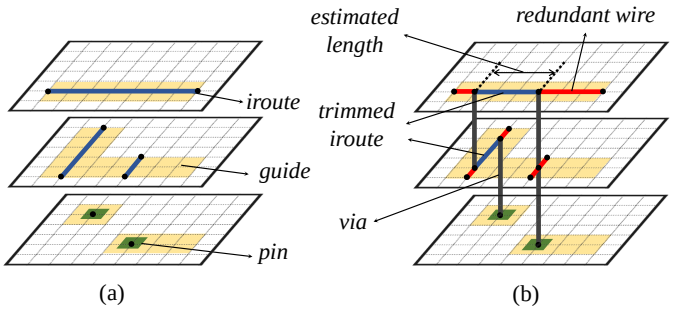

Figure 6: (a) An initial track assignment result. (b) A final track assignment result with via location determined.

Definition 5. *A net is* fully connected *if and only if its number of net components is one.*

Definition 6. *A net is* disjoint *if and only if its number of net components is greater than one.*

In order to connect a disjoint net, one of the net components is selected as target and the other components are sources. Then we connect the sources and the target by a path-searching algorithm, which is a core step of detailed routing. If a net has n net components, then it requires at most $(n-1)$ times of path searching to route a net completely. In our work, we apply an interval-based A^* search algorithm [8] because it is shown to be the most efficient path search algorithm [7] on modern VLSI designs with the track structure. Moreover, it also supports a multi-source single-target shortest path algorithm which can be applied to our work.

Negotiation-based routing techniques are utilized in global routing [14] and track assignment [20] and are shown to be efficient and effective to enhance the routability. The basic concept of a negotiation-based algorithm is its *history cost* which increases the cost in congestion regions. We adopt a similar technique in our detailed routing algorithm. If wires are overlapped after a routing iteration on a track, we increase the cost of the overlapped interval according to the number of wires on the interval. By doing so, an interval with a higher history cost will have less chance to be routed, which helps to reduce the congestion. The history cost somehow reflects the congestion condition. That is, regions with higher history costs are required by more nets to pass through. In contrast, regions with lower history costs are less congested.

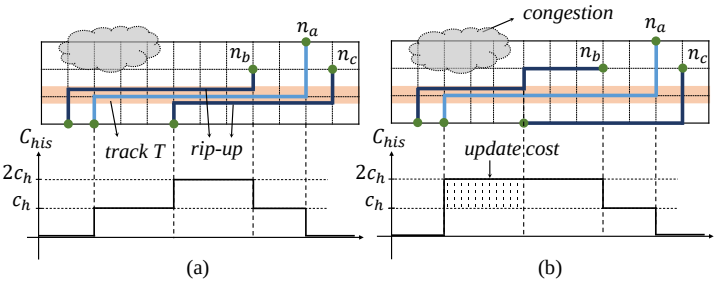

Figure 7: (a) At the end of an iteration, the history cost will be updated because nets n_a, n_b, and n_c overlap with one another on track T. Besides, net n_b and net n_c need to be ripped up after because the W_{mwis} of net n_a is the maximum among the three nets. (b) At the end of an iteration, the history cost of track T should be updated because net n_a and net n_b overlap again after n_b and net n_c are re-routed. (The increment of the history cost C_{his} on T from (a) to (b) is labeled by the dotted region.)

The multi-threaded technique can accelerate the program significantly, but a fundamental difficulty is to minimize the dependency of individual tasks. To divide the routing problem into independent tasks, we propose a routing scheme that enables parallel routing. In a routing iteration, all disjoint nets are routed in parallel without knowing the routing results of other nets, and therefore it may incur wire overlaps between different nets in some congestion regions. Once an iteration is completed, we update the history cost before entering the next iteration (Fig. 7(a)).

Algorithm 2 *MultiNegoDR*(D, N)

Input: D: the input design, N: the set of nets in D
Output: R_N: the routing result of N
$N_{disjoint}$: a set of disjoint nets
I_1: the number of iterations for the first stage
I_2: the number of iterations for the second stage
1: $R_N \longleftarrow \phi$, $N_{disjoint} \longleftarrow Disjoint(N)$, $i \longleftarrow 0$
2: **while** $N_{disjoint} \neq \phi$ and $i \leq I_1 + I_2$
3: $i \longleftarrow i + 1$
4: **if** $i \leq I_1$
5: $R_N \longleftarrow R_N \cup$ *MultiRouteAll_Stage1*$(N_{disjoint}, D)$
6: **else**
7: $R_N \longleftarrow R_N \cup$ *MultiRouteAll_Stage2*$(N_{disjoint}, D)$
8: *UpdateHistoryCost*(R_N, D)
9: $N_{disjoint} \longleftarrow$ *RemoveOverlaps*(R_N, D)
10: **return** R_N

Our proposed two-stage algorithm imposes different short constraints in two stages.

3.3.1 First-Stage Rip-up and Re-routing. In this stage, all disjoint nets are routed regarding the history cost on the tracks without considering the wires of other nets, and thus the paths of a net may overlap with other nets during path search. In other words, even if a wire is placed on the interval I in the $(i-1)^{th}$ iteration, I is still available for routing in the i^{th} iteration. Furthermore, after an iteration is performed, we increase the history cost of a track by the number of overlaps. A higher history cost indicates that the corresponding region is more congested, and therefore it will be more difficult for a net to go through this region afterwards. Then we rip-up the overlapping wires and its adjacent ones by the maximum weight independent set algorithm [10] on an interval graph with the weighted function:

$$W_{mwis} = \alpha_{wl} \cdot L_{wl} + \alpha_{pin} \cdot K_{pin} + \alpha_{fail} \cdot K_{fail}, \tag{2}$$

where L_{wl} is the wirelength of the wire itself and its adjacent wires, K_{pin} is the number of pins of the corresponding net, K_{fail} is the number of continuous failures in the rip-up phase, and α_{wl}, α_{pin}, and α_{fail} are the weighted constants. In order to deal with the spacing constraint, we extend a wire according to spacing rules in order to prevent any DRC violation. Moreover, to provide more chance for a net to explore new paths, we rip up the overlapping wires and the adjacent ones instead of the wires alone. Besides, routing a net consisting of more pins is much more difficult, and thus we tend to rip up the wires belonging to nets with fewer pins. Furthermore, we observe that if a net is continuously ripped up in several iterations, it is better not to rip it up in avoidance of getting stuck in the same situation. This procedure is illustrated in Fig. 7(a). And an example of re-routing nets is shown in Fig. 7(b).

3.3.2 Second-Stage Rip-up and Re-routing. Different from the first stage, in this stage, it is not allowed to route a net through a region where it is routed by other nets in the previous iteration. The main task in this stage is to explore paths for disjoint nets without shorts to other nets. The routing solution can converge fast because shorts with routed wires are prohibited in this stage.

Algorithm 2 illustrates the steps of our algorithm. In a multi-threaded routing iteration, all disjoint nets will be routed in parallel by the interval-based A^* search algorithm. After an iteration is done, we update the history cost of conflicting intervals and remove the overlapping wires. If the set of disjoint nets is not empty and the number of iterations does not exceed the limit, we repeat the routing process until the termination condition is met. The number of iterations of the first stage I_1 and that of the second stage I_2 affect the solution quality. We observe that the short area is reduced if I_1 increases. Therefore, in our work, we set I_1 and I_2 as 11 and 2, respectively. Experimental results show that our two-stage algorithm can reduce the congestion in critical regions effectively. The percentage of fully connected nets can reach over 95% before the second stage.

4 EXPERIMENTAL RESULTS

To evaluate the proposed initial detailed routing algorithm, we implemented our algorithm in the C++ programming language. All experiments were performed on the same Linux workstation with 64 2.7GHz Intel Xeon CPUs with 128GB memory. We conducted experiments on the benchmarks suite (including the hidden cases) from the 2018 ISPD Initial Detailed Routing Contest [17]. Table 2 lists the benchmarks statistics.

In order to rank the solution quality of all participating routers based on the metrics described in Section 2, the 2018 ISPD Initial Detailed Routing Contest defines a total cost function, which is essentially the weighted sum of all metrics:

$$C_{total} = \alpha_l \cdot \frac{L_{total}}{P_{m_2}} + \alpha_v \cdot V_{total} + C_{drc} + C_{pref}, \tag{3}$$

where C_{drc} and C_{pref} are defined as:

$$C_{drc} = \alpha_{sm} \cdot \frac{A_{sm}}{P_{m_2}^2} + \alpha_{spc} \cdot K_{spc} + \alpha_{ma} \cdot K_{ma}, \tag{4}$$

$$\begin{aligned} C_{pref} =& \alpha_{ogl} \cdot \frac{L_{og}}{P_{m_2}} + \alpha_{ogv} \cdot V_{og} \\ &+ \alpha_{otl} \cdot \frac{L_{ot}}{P_{m_2}} + \alpha_{otv} \cdot V_{ot} + \alpha_{ww} \cdot \frac{L_{ww}}{P_{m_2}}. \end{aligned} \tag{5}$$

The meaning and value of notations used in Equations (3), (4), and (5) are listed in Table 1.

symbol	description	value
C_{total}	total cost	defined by Equation (3)
C_{drc}	cost of the DRC constraint specified in Section 2	defined by Equation (4)
C_{pref}	cost of the routing preference metrics specified in Section 2	defined by Equation (5)
L_{total}	total wirelength	depends on solution
V_{total}	total number of vias	depends on solution
A_{sm}	total area of short metals	depends on solution
K_{spc}	number of spacing violations, including rules in Sections 2.1.2, 2.1.3, and 2.1.4	depends on solution
K_{ma}	number of min-area violations described in Section 2.1.1	depends on solution
P_{m_2}	length of metal2 pitch	depends on testcase
L_{og}	off-guide wirelength	depends on solution
L_{ot}	off-track wirelength	depends on solution
V_{og}	number of off-guide vias	depends on solution
V_{ot}	number of off-track vias	depends on solution
L_{ww}	wrong-way wirelength	depends on solution
α_t	weight on total wirelength	0.5
α_v	weight on total number of vias	2
α_{sm}	weight on area of short metals	500
α_{ma}	weight on min-area violations	500
α_{spc}	weight on number of spacing violations	500
α_{ogl}	weight on off-guide wirelength	1
α_{ogv}	weight on off-guide vias	1
α_{otl}	weight on off-track wirelength	0.5
α_{otv}	weight on off-track vias	1
α_{ww}	weight on wrong-way wirelength	1

Table 1: Notations for the evaluation metric in the 2018 ISPD Initial Detailed Routing Contest.

The Cadence physical design tool Innovus was used to detect DRC violations and report the detailed statistics of the evaluation metrics, and the evaluation script provided by the contest organizers was used to compute the total cost subsequently.

Under the evaluation metrics of the contest [2], the top-3 routers result in 23%, 52%, and 1224% higher total costs than our router. Table 3 summarizes the routing results of our algorithm and the top three teams (Team21, Team7, and Team5) of the contest [2].

Besides, we propose a more robust evaluation metric to consider more practical issues, while avoiding abusing the contest metric by any trick not beneficial to real design needs. Based on the DRC verification scheme of the

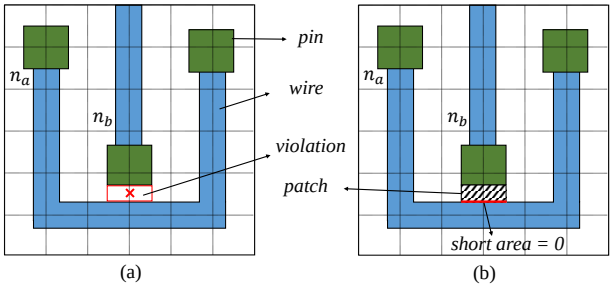

Figure 8: An example illustrating the patch metal insertion technique to remove a spacing violation. (a) A spacing rule violation is induced because net n_a and net n_b are too close. (b) A patch metal is inserted to eliminate the spacing violation between net n_a and net n_b while the short area is zero.

Cadence P&R tool Innovus, a spacing violation can be removed by inserting a patch metal between two violating objects, and the added patch metal will lead to a short with the short area equal to zero. However, the cost of patch metals is not considered in the original evaluation metrics. Figure 8 shows how a patch metal is inserted to eliminate a spacing violation.

To remedy the vulnerability in the evaluation metrics, we suggest that two extra metrics should be added: (1) the total number of short metals and (2) the cost of a patch metal with respect to its area.

Table 4 compares the wirelength of our router *without patch metal insertion* with those of the top-3 routers. The experimental results show that our algorithm achieves significant reductions in out-of-guide wirelength (L_{og}), off-track wirelength (L_{ot}), and wrong-way wirelength (L_{ww}), with only a small degradation of the total wirelength (L_{total}). Table 5 compares the number of used vias of our router with the top-3 routers. The experimental results show that our total numbers of out-of-guide vias (V_{og}) are smaller than those of all the top-3 routers, and that our router can complete all benchmarks without using any off-track via. According to the experiment results shown in Table 4 and Table 5, we conclude that our router can closely follow the routing preference metrics.

Table 6 compares the DRC violations of our router *without patch metal insertion* with those of the top-3 routers based on the following data: (1) K_{sm} (total number of short metals), (2) A_{sm} (total area of short metals), (3) K_{ma} (total number of min-area violations), and (4) K_{spc} (total number of spacing violations). The experimental results show that our router has the best performance on minimizing the short area, implying that our router handles the important routability issue best, and meanwhile the total numbers of min-area violations and spacing violations are the least. Since the total area of patch metals is not provided by the contest organizers, we do not list this information in Table 6. We observe from Table 6 that the number of short metals of the first-place winner might not be reasonable because the values of short areas of their results are smaller. All the above results were provided by the contest organizers [17].

To evaluate the effectiveness of our multi-threaded scheme, we further conducted experiments by using different numbers of threads, and recorded the corresponding runtimes. In Figure 9, the runtime of our router *without patch metal insertion* is plotted as a function of the number of threads. We can observe that our multi-threaded scheme is effective in that the runtime is inversely proportional to the number of threads used. The experimental results show that our algorithms are effective and efficient.

5 CONCLUSIONS

This paper has presented an effective initial detailed routing algorithm handling modern circuit designs with real industrial constraints. The algorithm consists of three main steps: (1) a pin-access generation method modeling a complex pin shape in modern designs with valid points on the routing track, (2) a via-aware track assignment technique embedded to the initial detailed routing framework to reduce the computational complexity and problem size, and (3) a multi-threaded initial detailed routing algorithm utilizing a negotiation-based rip-up and re-route scheme, which honors the global routing guide while minimizing design-rule violations effectively. In addition, two metrics supplementing the original evaluation metrics provided by the

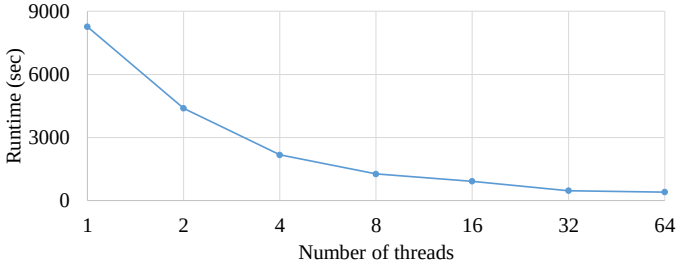

Figure 9: The runtime of *ispd_test3* is plotted as a function of the number of threads, where the runtime of our program is almost inversely proportional to the number of threads.

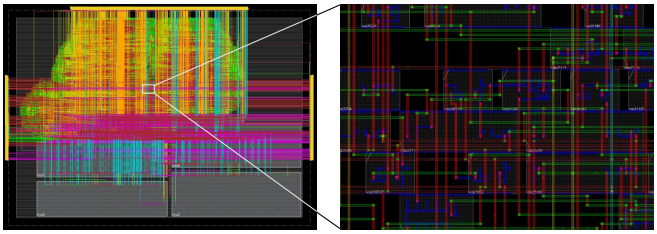

Figure 10: (a) The routing solution for ispd_test3 obtained by our router. (b) The partly enlarged routing solution of (a) gives more details.

contest organizers have also been proposed. Compared with the winning routers of the 2018 ISPD Initial Detailed Routing Contest, experimental results have shown that our router obtains the best overall score; for example, the first-place router of the contest results in a 23% higher cost than our router.

REFERENCES

[1] *Cadence LEF/DEF 5.3 to 5.7 Exchange Format 2009.* www.si2.org/openeda.si2.org/projects/lefdef.
[2] *ISPD 2018 Contest on Initial Detailed Routing.* http://www.ispd.cc/contests/18/index.htm#intro.
[3] S. Batterywala, N. Shenoy, W. Nicholls, and H. Zhou. Track assignment: A desirable intermediate step between global routing and detailed routing. In *Proc. of ICCAD*, 2002.
[4] H. Y. Chen, C. H. Hsu, and Y. W. Chang. High-performance global routing with fast overflow reduction. In *Proc. of ASPDAC*, 2009.
[5] Y. Ding, C. Chu, and W. K. Mak. Self-aligned double patterning lithography aware detailed routing with color preassignment. *IEEE Tran. on CAD*, 36(8):1381–1394, 2017.
[6] M. Gester, D. Müller, T. Nieberg, C. Panten, C. Schulte, and J. Vygen. Bonnroute: Algorithms and data structures for fast and good vlsi routing. *ACM Tran. on DAES*, 18(2):32:1–32:24, 2013.
[7] S. M. M. GonÃğalves, L. S. da Rosa, and F. d. S. Marques. A survey of path search algorithms for vlsi detailed routing. In *Proc. of ISCAS*, 2017.
[8] A. Hetzel. A sequential detailed router for huge grid graphs. In *Proc. of DATE*, 1998.
[9] J. E. Hopcroft and R. M. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing*, 2(4):225–231, 1973.
[10] J. Y. Hsiao, C. Y. Tang, and R. S. Chang. An efficient algorithm for finding a maximum weight 2-independent set on interval graphs. *Inf. Process. Lett.*, 43(5):229–235, 1992.
[11] M. K. Hsu, Y. F. Chen, C. C. Huang, S. Chou, T. H. Lin, T. C. Chen, and Y. W. Chang. Ntuplace4h: A novel routability-driven placement algorithm for hierarchical mixed-size circuit designs. *IEEE Tran. on CAD*, 33(12):1914–1927, 2014.
[12] C. C. Huang, H. Y. Lee, B. Q. Lin, S. W. Yang, C. H. Chang, S. T. Chen, Y. W. Chang, T. C. Chen, and I. Bustany. Ntuplace4dr: A detailed-routing-driven placer for mixed-size circuit designs with technology and region constraints. *IEEE Tran. on CAD*, 37(3):669–681, 2018.
[13] I.-J. Liu, S.-Y. Fang, and Y.-W. Chang. Overlay-aware detailed routing for self-aligned double patterning lithography using the cut process. In *Proc. of DAC*, 2014.
[14] 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 Tran. on CAD*, 32(5):709–722, 2013.
[15] W. H. Liu, Y. L. Li, and C. K. Koh. A fast maze-free routing congestion estimator with hybrid unilateral monotonic routing. In *Proc. of ICCAD*, 2012.
[16] W. H. Liu, Y. Wei, C. Sze, C. J. Alpert, Z. Li, Y.-L. Li, and N. Viswanathan. Routing congestion estimation with real design constraints. In *Proc. of DAC*, 2013.
[17] S. Mantik, G. Posser, W.-K. Chow, Y. Ding, and W.-H. Liu. ISPD 2018 initial detailed routing contest and benchmarks. In *Proc. of ISPD*, 2018.
[18] D. Shi and A. Davoodi. Trapl: Track planning of local congestion for global routing. In *Proc. of DAC*, 2017.
[19] Y.-H. Su and Y.-W. Chang. Nanowire-aware routing considering high cut mask complexity. In *Proc. of DAC*, 2015.
[20] M. P. Wong, W. H. Liu, and T. C. Wang. Negotiation-based track assignment considering local nets. In *Proc. of ASPDAC*, 2016.
[21] X. Xu, B. Cline, G. Yeric, B. Yu, and D. Z. Pan. Self-aligned double patterning aware pin access and standard cell layout co-optimization. In *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2015.
[22] Y. Zhang and C. Chu. Regularroute: An efficient detailed router with regular routing patterns. In *Proc. of ISPD*, 2011.

Circuit	#stds	#blks	#nets	#fix pins	#layers	Die Size	Tech
ispd_test1	8879	0	3153	0	9	0.20x0.19mm^2	45nm
ispd_test2	35913	0	36834	1211	9	0.65x0.57mm^2	45nm
ispd_test3	35973	4	36700	1211	9	0.99x0.70mm^2	45nm
ispd_test4	72094	0	72401	1211	9	0.89x0.61mm^2	32nm
ispd_test5	71954	0	72394	1211	9	0.93x0.92mm^2	32nm
ispd_test6	107919	0	107701	1211	9	0.86x0.53mm^2	32nm
ispd_test7	179865	16	179863	1211	9	1.36x1.33mm^2	32nm
ispd_test8	191987	16	179863	1211	9	1.36x1.33mm^2	32nm
ispd_test9	192911	0	178857	1211	9	0.91x0.78mm^2	32nm
ispd_test10	290386	0	182000	1211	9	0.91x0.87mm^2	32nm

Table 2: Benchmark characteristics. #stds, #blks, #nets, #fix pins, #layers, Die Size, and Tech denote the numbers of placed standard cells, blockages, nets, fix pins, layers, the die size of the design, and the technology node of the design, respectively [17].

	Team21 (1st Place)		Team7 (2nd Place)		Team5 (3rd Place)		Ours		
Circuit	C_{total}	C_{norm}	C_{total}	C_{norm}	C_{total}	C_{norm}	C_{total}	C_{norm}	*Runtime* (sec)
ispd_test1	386188.07	1.08	731640.91	2.05	5761603.15	16.13	357169.57	1.00	33.79
ispd_test2	5636273.56	1.02	7987888.08	1.44	82542695.72	14.90	5538011.26	1.00	346.00
ispd_test3	8645534.12	1.29	12126876.16	1.81	178905476.37	26.63	6718070.16	1.00	460.15
ispd_test4	67978777.21	1.83	47890529.35	1.29	733474117.68	19.78	37081079.91	1.00	1210.25
ispd_test5	65227108.33	1.28	73762264.00	1.45	735559666.33	14.42	50997107.05	1.00	1451.04
ispd_test6	89303689.90	1.22	106519261.03	1.45	352317181.43	4.80	73386130.44	1.00	3168.50
ispd_test7	N/A	N/A	193575674.83	1.55	N/A	N/A	124661899.77	1.00	8586.44
ispd_test8	161426598.31	1.28	188495138.68	1.50	N/A	N/A	125778356.25	1.00	8295.57
ispd_test9	144221466.20	1.22	188979325.95	1.60	603223377.02	5.12	117858984.47	1.00	5657.37
ispd_test10	193867714.04	0.83	258480435.33	1.10	973977612.19	4.16	234017327.32	1.00	6553.18
Ratio		1.23		1.52		13.24		1.00	

Table 3: Resulting total costs (C_{total}) and the normalized scores (C_{norm}) of the top-3 routers of the contest [2] and ours, where C_{total} is the raw total score, C_{norm} the normalized score, and *Runtime* the real runtime of our router with patch metal insertion.

	Team21 (1st Place)				Team7 (2nd Place)				Team5 (3rd Place)				Ours*			
Circuit	L_{total}	L_{og}	L_{ot}	L_{ww}	L_{total}	L_{og}	L_{ot}	L_{ww}	L_{total}	L_{og}	L_{ot}	L_{ww}	L_{total}	L_{og}	L_{ot}	L_{ww}
ispd_test1	0.19	2.50	1.41	1.40	0.17	1.78	0.24	0.73	0.19	2.85	0.14	4.24	0.20	0.44	0.07	0.69
ispd_test2	3.26	28.67	8.16	7.29	3.13	43.87	3.25	7.43	3.28	120.54	1.17	30.45	3.33	13.63	1.11	7.16
ispd_test3	3.63	27.67	13.39	7.55	3.49	72.63	3.24	7.51	3.61	363.49	0.82	25.54	3.69	6.48	0.77	6.67
ispd_test4	5.50	48.05	30.19	44.94	5.23	150.06	16.41	23.32	5.54	668.50	0.76	54.29	5.54	11.32	0.87	11.22
ispd_test5	5.88	68.54	9.37	38.81	5.53	105.43	10.08	21.31	5.84	1100.32	1.94	9.34	5.90	26.03	3.15	21.58
ispd_test6	7.64	94.20	30.24	56.21	7.11	143.78	14.12	30.00	7.50	252.28	2.39	30.27	7.62	38.16	6.13	36.60
ispd_test7	N/A	N/A	N/A	N/A	12.98	342.09	23.99	54.03	N/A	N/A	N/A	N/A	13.81	90.76	10.45	60.44
ispd_test8	13.91	201.25	75.05	91.16	13.04	383.98	23.73	53.47	N/A	N/A	N/A	N/A	13.81	93.79	10.64	61.45
ispd_test9	11.76	162.75	66.35	89.29	10.90	257.94	23.69	52.99	11.48	434.02	3.02	41.67	11.66	67.32	9.21	60.25
ispd_test10	14.45	282.87	125.06	95.33	13.55	505.20	24.02	54.50	14.22	934.63	3.52	41.14	14.33	104.78	12.19	71.69
Ratio	0.99	3.18	12.54	1.76	0.94	5.60	4.41	1.05	0.98	24.33	0.83	2.70	1.00	1.00	1.00	1.00

Table 4: Resulting total wirelength ($L_{total}(\times 10^9)$), out-of-guide wirelength ($L_{og}(\times 10^6)$), off-track wirelength ($L_{ot}(\times 10^6)$), and wrong-way wirelength ($L_{ww}(\times 10^6)$) of the top-3 routers of the contest [2] and ours* (without patch metal insertion).

	Team21 (1st Place)			Team7 (2nd Place)			Team5 (3rd Place)			Ours*		
Circuit	V_{total}	V_{og}	V_{ot}	V_{total}	V_{og}	V_{ot}	V_{total}	V_{og}	V_{ot}	V_{total}	V_{og}	V_{ot}
ispd_test1	41.64	1.39	0.12	36.80	0.84	0.00	33.54	0.17	0.00	42.57	0.50	0.00
ispd_test2	409.55	13.45	1.36	367.56	9.09	0.00	335.99	3.61	0.00	403.54	5.69	0.00
ispd_test3	427.41	2.45	1.22	375.77	6.58	0.00	318.13	6.37	0.00	398.14	5.99	0.00
ispd_test4	858.22	8.84	1.01	757.95	16.35	0.00	632.05	21.90	0.00	822.66	28.60	0.00
ispd_test5	1158.95	31.39	10.51	941.06	35.31	0.00	865.65	38.38	0.00	1072.81	13.39	0.00
ispd_test6	1800.29	42.71	17.55	1442.42	56.49	0.00	1415.90	14.23	0.00	1641.88	21.81	0.00
ispd_test7	N/A	N/A	N/A	2341.15	80.99	0.00	N/A	N/A	N/A	2656.80	39.37	0.00
ispd_test8	2929.58	82.48	22.29	2353.76	83.85	0.00	N/A	N/A	N/A	2621.05	39.30	0.00
ispd_test9	2920.26	67.37	22.92	2351.87	93.61	0.00	2306.40	19.25	0.00	2621.86	39.50	0.00
ispd_test10	3110.16	81.83	27.39	2516.31	108.84	0.00	2439.38	27.95	0.00	2791.06	47.66	0.00

Table 5: Resulting total numbers of used vias ($V_{total}(\times 10^3)$), out-of-guide vias ($V_{og}(\times 10^3)$), and off-track vias ($V_{ot}(\times 10^3)$) of the top-3 routers of the contest [2] and ours* (without patch metal insertion).

	Team21 (1st Place)				Team7 (2nd Place)				Team5 (3rd Place)				Ours*			
Circuit	K_{sm}	A_{sm}	K_{ma}	K_{spc}	K_{sm}	A_{sm}	K_{ma}	K_{spc}	K_{sm}	A_{sm}	K_{ma}	K_{spc}	K_{sm}	A_{sm}	K_{ma}	K_{spc}
ispd_test1	4.22	0.12	0	0.11	0.05	2.94	10	0.84	3.06	667.18	0	6.72	0.10	0.70	0	0.77
ispd_test2	36.60	15.19	1	1.16	0.29	139.43	36	5.48	42.32	14763.41	0	62.51	1.22	27.52	0	6.80
ispd_test3	46.97	782.62	0	1.39	1.96	1213.99	148	5.88	61.76	44770.71	0	65.74	3.92	218.43	0	8.48
ispd_test4	349.60	2117.88	6	50.96	9.13	1576.54	51	25.31	141.58	52663.70	13264	99.57	5.97	330.93	126	45.66
ispd_test5	431.91	1137.15	28	66.74	3.39	413.22	151	104.24	120.91	49746.73	26694	156.93	7.04	307.26	37	96.58
ispd_test6	628.78	1249.10	15	100.20	5.16	805.35	261	149.41	89.36	14188.80	80081	223.80	10.38	488.55	48	113.05
ispd_test7	N/A	N/A	N/A	N/A	16.04	2346.17	529	249.45	N/A	N/A	N/A	N/A	19.80	838.45	108	179.19
ispd_test8	1058.14	3071.58	48	161.23	14.97	2033.74	433	246.41	N/A	N/A	N/A	N/A	20.94	904.07	103	182.49
ispd_test9	1051.11	2263.23	40	158.31	11.74	1560.74	405	271.20	142.45	25072.71	124830	383.34	19.25	717.96	74	185.27
ispd_test10	1289.36	4838.64	33	177.43	19.86	6341.78	656	274.02	177.50	53462.78	128049	392.66	35.22	8861.14	55	218.48

Table 6: Resulting DRC violations including the number of short metals ($K_{sm}(\times 10^3)$), the area of short metals ($A_{sm}(\times 10^6)$), the number of min-area violations (K_{ma}), and the number of spacing violations ($K_{spc}(\times 10^3)$) of the top-3 routers of the contest [2] and ours* (without patch metal insertion).