Challenges and Approaches in VLSI Routing

[1]Gracieli Posser, [2]Evangeline F.Y. Young, [3]Stephan Held, [4]Yih-Lang Li, [5]David Z. Pan
gposser@cadence.com,fyyoung@cse.cuhk.edu.hk,held@dm.uni-bonn.de,ylli@cs.nctu.edu.tw,dpan@ece.utexas.edu
[1]Cadence Design Systems Inc.
[2]The Chinese University of Hong Kong
[3]University of Bonn
[4]National Yang Ming Chiao Tung University
[5]The University of Texas at Austin

ABSTRACT

In this paper, we will first have a brief review of the ISPD 2018 and 2019 Initial Detailed Routing Contests. We will then visit a few important and interesting topics in VLSI routing that includes GPU accelerated routing, signal speed optimization in routing, PCB routing and AI-driven analog routing.

CCS CONCEPTS

• **Hardware** → *Design rules*; *Analog and mixed-signal circuit optimization.*

KEYWORDS

routing, contests, design rules; GPU, speedup, printed circuit board (PCB), digital design, analog design, machine intelligence, AI

ACM Reference Format:
[1]Gracieli Posser, [2]Evangeline F.Y. Young, [3]Stephan Held, [4]Yih-Lang Li, [5]David Z. Pan. 2022. Challenges and Approaches in VLSI Routing. In *Proceedings of the 2022 International Symposium on Physical Design (ISPD '22), March 27–30, 2022, Virtual Event, Canada.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3505170.3511477

1 INTRODUCTION

Routing has been one of the most challenging problems in physical synthesis of VLSI design. With the introduction of large scale benchmarks and practical evaluation metrics in some recent research contests on global and detailed routing, new research and progress have been resulted. Routing is a big and challenging field with many different applications. In this invited paper, we will first have a short review of the ISPD 2018 and 2019 Initial Detailed Routing Contests. The increasing design rule complexity has made detailed routing ever more challenging and time consuming in Section 1. These contests are timely in providing more information and resources for addressing the problem. We will then discuss how VLSI routing can be GPU-accelerated in Section 3. With the advancement in parallel processing hardware like GPU, it is interesting to see how routing can be benefited and massively sped up. Next, we will discuss how timing can be considered into routing effectively in Section 4. Timing is always an important and practical concern in design but it is difficult to address. There are sophisticated techniques to handle these complicated timing issues in routing. Finally, we will discuss two special routing problems, PCB routing in Section 5 and Analog routing in Section 6. The increasing diversity in semi-conductor applications have driven advancement in packaging technology and PCB routing, while analog routing with AI techniques to integrate with human knowledge in the design is an interesting direction to explore.

ISPD '22, March 27–30, 2022, Virtual Event, Canada

ACM ISBN 978-1-4503-9210-5/22/03…$15.00
https://doi.org/10.1145/3505170.3511477

2 COMPLEX ROUTING DESIGN RULES AND THE ISPD 2018/2019 INITIAL DETAILED ROUTING CONTESTS

Every new technology node brings more complexity on the design rules. The 5.8 LEF file [2] has more than thousand pages with design rule descriptions. This complexity makes the routing step more challenging and timing consuming. ISPD 2018 [50] and 2019 [45] contests addressed some of these issues on the initial detailed routing. The main goal was to provide advanced routing rules and industrial designs to academia and EDA community. So, they can conduct leading detailed routing research on the modern real designs.

The contests brought new advanced research on routing. In the 2018, we had some published works from the contest teams: [28, 63]. From the 2019 contest, where more complex designs rules were considered, we got further contributions to the literature: [9, 22, 33]. Also, the benchmarks are used on the academia not only on routing, but also to validate solutions on different Physical Design problems:

- reinforcement learning (RL) based routing [20, 39]
- partitioning tools [52],
- unified global-detailed routing [10, 29, 41]
- synchronous reinforcement learning framework [59] to search for net ordering strategies in detailed routing automatically with the objective to reduce the number of violations.

Different advanced spacing rules were considered in the contests, like: a) spacing table; b) end-of-line (EOL) spacing; c) cut spacing; d) min area rule; e) parallel run spacing; f) adjacent cut spacing; g) corner-to-corner spacing. More details about those rules can be find in [2, 45, 50].

All the nets need to be connected taking into account all the design rules to have a valid routing. Some of the big challengings on routing are presented below. The images used as example are all taken from the Cadence Innovus GUI [1]. Those are to show what a good and a bad solution solution look like.

(1) Memory and runtime control. Routing tool needs to be well architecture to be able to route all the nets in a reasonable time and been memory efficient. Multi-threading is very common on routing engines, but the big challenge is on how to keep the toll deterministic.
(2) Pin-access location selection. Figure 1 (a) shows a good example where the pins are accessed avoiding violations. Red is Metal2 and in blue is Metal1. Pins are on Metal1 and can be accessed on any place of their shape, i.e., in the blue rectangles. Routing needs to via up to Metal2 from the chosen pin access location. (b) shows that small differences on the place where the pin is access can generate several violations. In the example, two pin access have three violations.
(3) Via selection. It can be either for vertical or horizontal connections, Figure 2(a), or (b) for double cut via insertion. Double cut vias are mostly used to increase the reliability on critical nets and designs, like for automobile integrated circuits. It is a very hard problem, as inserting double cut vias can create a lot of DRCs. Figure 3 shows some some examples: (a) where a parallel run length violations happens with the wire on the right track; (b) there is a short violation as the upper track is used by another wire, so it cannot have a double cut via; (c) a well inserted double cut via, where the extra cut is using an empty track and not causing any violation,
(4) Patch wire insertion. Which is commonly used to resolve min area rule violations. When they are inserted, tool needs to be aware of designs rules to make sure the patch wires inserted are not creating new violations.
(5) Routing and instance blockages. The instances in the design have blockages that need to be considered during routing. Same for routing blockages, power and ground shapes, pre-routed wires or any other kind of metal blockages. They can create extra violations if the router is not aware of those blockages.
(6) Non-default rule (NDR) handling. Critical nets, like clock nets, can have non-default rules to improve the timing and reliability on those nets. Router needs to consider default and non-default rules and make sure the nets on each group can be routed accordingly with the rule assigned to it.

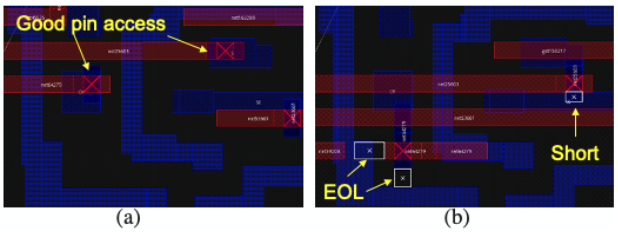

Figure 1: (a) Good pin access. (b) Bad pin access.

3 GPU ACCELERATED ROUTING

In this section, we will explore the opportunities of speeding up routing significantly using GPU parallelization. There have been

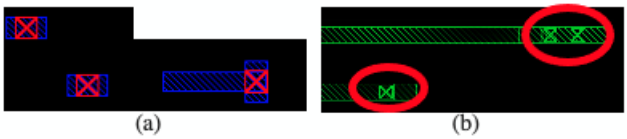

Figure 2: (a) Two most-left vias are horizontal vias, and the one on the right side is a vertical via. (b) Double cut via insertion compared to single cut.

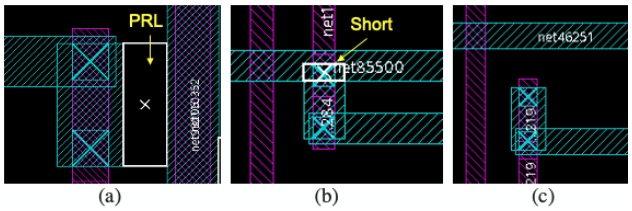

Figure 3: Double cut via insertion with (a) a parallel run length (PRL) violation; (b) with short violation and (c) a well inserted via without violations.

quite many works on CPU parallelization of routing [10, 41, 44], but GPU acceleration is still yet to be explored more. There are recent works on GPU accelerated pattern routing [43] and maze routing [35]. The former implements the dynamic programming of the pattern routing in CUGR [41] on GPU achieving a speedup of more than 10 times for the pattern routing stage. The latter proposes a parallel maze routing algorithm, called *GAMER*, that improves the original sequential maze routing algorithm significantly, and can achieve a speedup of more than 16 times when applying to the coarse grained maze routing in CUGR. Routing is usually divided into two stages, global routing and detailed routing, and maze routing is a core routing engine in both stages. Maze routing is highly flexible in handling various constraints like avoiding routing obstacles and congested regions by setting different costs on the routing edges. However, rip-up and reroute that usually calls maze routing is very often the most time-consuming stage. It is thus very beneficial if maze routing can be GPU-accelerated.

GAMER is a work on GPU-accelerated maze routing. The core of maze routing is a multi-source multi-destination shortest path searching on a grid graph. In VLSI, routing directions are rectilinear and it is preferred to have small via usage. By utilizing these two characteristics, the original sequential maze routing can be implemented as a sequence of parallel sweep operations where each sweep operation can be parallelized and efficiently performed. A horizontal (vertical) sweep is an operation to update the shortest distances using horizontal (vertical) edges only. An example is shown in Figure 4. By alternating between horizontal and vertical sweeps, the shortest distances can be correctly found. In a horizontal (vertical) sweep, different rows (columns) are independent and thus can be naturally performed in parallel. The sweeping operation itself in a single row or column can also be parallelized by reformulating the distance update as a prefix sum and a prefix min problem.

In a row of n elements, let d_i and c_i denote the current shortest distance from one end to the i-th element and the edge cost between the $i-1^{st}$ and the i^{th} elements. A row sweep can be formulated as the following operation:

$$d_i^* = \min_{0 \le j \le i} \left(d_j + \sum_{k=j+1}^{i} c_k \right) \tag{1}$$

where d_i^* is the updated shortest distance. We can define $s_i = \sum_{k=0}^{i} c_k$, which is used to replace the summation in Equation (1) to obtain the following new formulation.

$$d_i^* - s_i = \min_{0 \le j \le i} \left(d_j - s_j \right) \tag{2}$$

Equation (2) is a prefix min problem, and many parallel prefix sum algorithms can be applied to solve the prefix min problems by changing the operator from *plus* to *min*. Therefore, a row sweep can be solved on GPU in $O(\log_2 n)$ time with some well studied parallel prefix sum algorithms.

The speedup of GAMER over a well-implemented sequential maze routing method can be as high as over a hundred times depending on the size of the grid graph and the number of pins to be connected. In general, the finer the grid graph and the higher the pin count the larger the speedup can be achieved. However, the granularity of the grid graph that can be handled efficiently will also be limited by the size of the GPU local memory. Applying GAMER to only the coarse grained maze routing of the global router CUGR, which consists of a pattern routing stage to generate an initial routing solution followed by a maze routing stage that performs several rounds of rip-up and reroute based on the initial routing solution gives an overall speedup of 2.25 times.

To achieve an impressive speedup for the whole routing process, there will be a few challenges. One need to be able to accelerate every major part of the router. In general, GPU acceleration will be useful and beneficial if a problem can be resolved by repetitive applications of some systematic operations extensively. For example, dynamic programming (DP) can usually be sped up because DP involves filling up of tables in a very well-defined way. GAMER is also solving the maze routing problem by repeatedly applying the sweep operations in a very regular fashion. However, in general, the operations in a global router or a detailed router will not be very regular, and they will involve heuristics, stage-wise optimization and special cases. For example, in hierarchical routing, simple routing operations will be performed first to generate route guides for the next-level more detailed routing operations. This kind of hierarchical process is essential and cannot be replaced by a *flat* operation although we may be able to speed up the flat routing process by a hundred times. With some efforts, GAMER can be extended to handle maze routing with irregular route guides, but some systematic approaches to GPU accelerate the whole global routing process or even detailed routing are still unknown and yet to be discovered. The high degree of irregularity involving complex design rule check and the huge scale of the problem will make GPU acceleration on detailed routing a very challenging task.

4 SIGNAL SPEED OPTIMIZATION

Decreasing feature sizes and, thereby, increasing wire resistances have raised the impact of wire delays in chip timing. To mitigate this effect, heterogeneous metal stacks are used today. They have tiny wire structures at the bottom layers and increasing widths and heights in higher routing layers. With the wiring width also the spacing is increasing. Thus, the resistance gets smaller while the capacitance, which is dominated by coupling to neighboring wires, varies only marginally with higher layers. Summarizing, layers provide fast signal speeds, while lower layers are slow.

Many approaches address non-uniform metal stacks by performing a timing-aware layer assignment following a global routing that might be two-dimensional [37, 38, 71].

In addition, the tree topology and geometry influence the signal delay. Several approaches have been proposed to compute linear delay-constrained (but congestion-unaware) routing trees [5, 11, 24] or trees with restricted rc-delay [7, 61]. In [64] routing topologies of critical nets are adjusted inside global routing.

If nets are allowed to choose their fastest layout independent from others, they would choose the highest routing layers unless they are very short. In addition, fast geometries can exceed the minimum Steiner length considerably. High layers are also favorable to minimize power, as they allow for smaller drive strengths and larger repeater spacings. The latter also reduces the number of vias on lower layers.

To balance the conflicting goals routability, timing closure, and power, it is necessary to have a full understanding of global timing constraints as well as the power impact inside the routing model. This is particularly true for global routing, where the main structure of the interconnects is determined and global packing algorithms can be applied.

The global routing results can mostly be preserved in detailed routing by closely following the global routes and by avoiding certain notorious problems, e.g. replacing a global wire segment on fast layers by slow interconnects on low layers where they are used anyway to connect multiple neighboring pins [3].

Many approaches for timing-driven global routing employ net delay bounds, e.g. [4, 60]. Others forbid routing trees that cause too large delays on critical paths [25]. Some embed fast routing geometries into the global routing graph with respect to routing congestion [26].

A theoretically and practically attractive approach for global routing is based on the multi-commodity flows [62] and its improved variant: the min-max resource sharing problem [51]. Here, we are given a set of customers $\mathcal{C}$ and resources $\mathcal{R}$. Each customer $C \in \mathcal{C}$ needs a solution $b_c \in \mathcal{B}_C$ from the set $\mathcal{B}_C$ (block) of possible solutions for that customer. A solution b_c consumes a certain fraction $\text{usg}_r(b_c)$ from each resource $r \in \mathcal{R}$ and the task is to find a solution vector $(b_c)_{c \in C}$ minimizing

$$\max_{r \in \mathcal{R}} \sum_{C \in \mathcal{C}} usg_r(b_c).$$

Traditionally, $\mathcal{R}$ is composed of global routing graph edges and an additional resource for netlength or power. $\mathcal{C}$ is composed of the nets, where $\mathcal{B}_C$ is the set of all Steiner trees for net C. The min-max resource sharing algorithm is a variant of the multiplicative weight update method that maintains prices price_r for all resources $r \in \mathcal{R}$. As a subroutine it needs an oracle that computes solutions w.r.t. the resource prices instead of respecting them directly

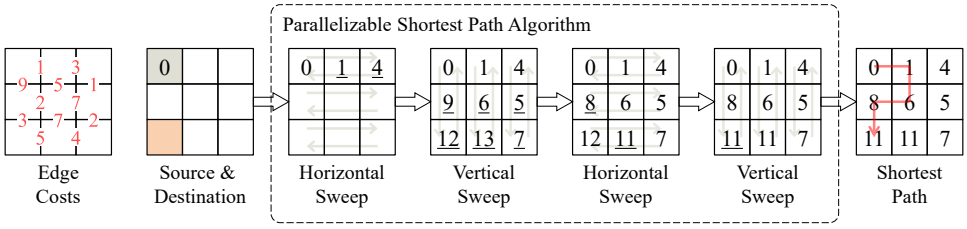

Figure 4: Shortest Path via Alternating Sweeps.

In [23] it was first shown how to integrate static timing constraints efficiently into this model. The multiplicative weight update to tackle the problem [51]. For each path P in the timing graph, a new resource P is added to $\mathcal{R}$ The fraction $\text{usg}_P(b_c)$ that a net C with two pins $p, q \in C$ and $(p, q) \in E(P)$ consume is $\text{usg}_P(b_C) := \text{delay}_{b_C}(p, q)/T$, where T is the cycle time. Then, the path delay is within the cycle time T if

$$\sum_{C \in C} \text{usg}_P(b_c) \leq 1.$$

These constraints can be represented implicitly by prices on the timing graphs which are computable in linear time [16].

With timing resources, we need to solve the cost-distance Steiner tree problem in the oracle subroutine. Let $\mathcal{R}^{GR}$ denote the set of global routing graph resources. Given a net N with source $p \in N$, congestion prices price_r for edges $r \in \mathcal{R}^{GR}$ and delay prices for all timing graph edges (p, q) $(q \in N \setminus \{p\}$ it asks for a tree b_N minimizing

$$\sum_{e \in E(T)} \sum_{r \in \mathcal{R}^{GR}} \text{price}_r \text{usg}_r(b_N) + \sum_{q \in \text{sink}(N)} \text{price}_{(p,q)} \text{delay}_{b_N}(p, q)/T. \tag{3}$$

Here, the delay model $\text{delay}_{b_N}(r, q)$ may vary from the application scenario. Before repeater insertion, a simple linear delay model may be used, later Elmore delays or more accurate models can be applied.

Given such an oracle function, we can compute near-optimum fractional solutions in near-linear time, which we can round randomly and post-optimize.

Theorem 4.1. *([23, 51]) Given a σ-approximate oracle algorithm for (3) and an $\epsilon > 0$, using $\tilde{O}((|C| + |\mathcal{R}|)/\epsilon^2)$ calls to the oracle and random selections from the computed solutions, we achieve a solution where the maximum expected relative usage exceeds the maximum usage of an optimum solution by at most a factor $(1 + \epsilon)\sigma$.*

For a constant number of sinks the oracle function can be solved to optimality for linear delays or approximated accurately for RC delays [23]. Unfortunately, no approximation better than $\wr(\log \log n)$ appears to be possible [15] for general n. Obtaining better practical and theoretical approximation approximation for (3) remains a major research challenge.

Having integrated timing constraints it is also easy to integrate the repeater insertion into global routing [17]. Using resources representing placement tiles, local placement congestion can be addressed natively.

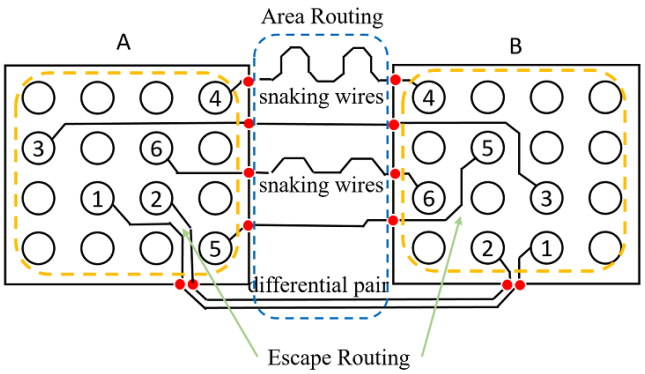

Figure 5: An Example of One-Layer Escape Routing and Area Routing for PCB Design.

5 PRINTED CIRCUIT BOARD (PCB) ROUTING

Printed circuit boards (PCBs) constitute the most common method of assembling electronic components. The continual miniaturization of semiconductor devices and the increasing diversity of semiconductor applications have increased the complexity of electronic components and have driven advancements in packaging technology, which has also increased the pin density of PCB designs. Therefore, successful modern PCB design requires many available routing layers. These trends in PCB design have raised the difficulty of PCB routing. For simplification, the PCB routing problem is traditionally partitioned into escape routing and area routing, and each of these components is investigated separately. The escape points along the boundary of each component of a PCB serve as ideal locations for linking escape routing and area routing. Accordingly, escape routing entails connecting the pins of a component to the escape points, whereas area routing entails connecting the escape points to all components. Figure 5 depicts an example of a one-layer escape routing and area routing solution.

Escape routing is categorized into unordered escape routing (UER) and ordered escape routing (OER) depending on whether the escape order of the nets to be routed is given. UER involves identifying a legal routing path from a pin to an escape point for each net without a restricted escape order. OER entails restricting the escape order of the nets along each component's boundary. In area routing, if the escape order of the nets is not restricted, many vias may be required to expel wire crossings, thereby lowering the PCB's performance and increasing the required number of layers. Therefore, because OER involves an artificial arrangement

or restriction of the escape order, it is more suitable for modern PCB design.

Boolean satisfiability (SAT)-based algorithms have been employed to resolve the OER problem [47, 48]. In the algorithm presented in [47], the OER problem is modeled as the SAT problem. Owing to the intrinsically high complexity of the SAT problem, a large-scale design is partitioned into several subregions for individual processing. The SAT-based algorithm presented in [47] includes the condition that when the boundary of a component is traversed in a clockwise order starting from the top-left corner, the first encountered net must be net 1; therefore, the SAT formula was improved to support cyclic ordering rather than linear ordering. Furthermore, the researchers in [47] adopted bus-based clustering instead of area-based clustering to implement a more practical design.

Integer linear programming (ILP) is another approach that has been applied to the OER problem [19, 27, 34]. For example, in [19], an ILP method was proposed for resolving the preassignment RDL routing problem to obtain an optimal routing solution. Moreover, in [27], a minimum-cost multi-commodity flow problem involving three constraints (non-cross, ordering, and capacity) was resolved using an ILP solver to achieve wirelength reduction in OER. In [34], new variables were introduced to construct an injective map in which each solution of an OER problem can be represented by only one model solution, thus engendering a compact model. For sequential OER, in [31], restrictions were imposed on monotonic routing paths, and a monotonic via assignment method was implemented to guarantee the feasibility of monotonic OER for two-layer ball grid array packages.

For PCB routing, neither UER nor OER offers a promising solution because an escape ordering favoring escape routing may complicate area routing, and vice versa. By contrast, simultaneous escape routing (SER) optimizes the escape routing of two connected components as well as the corresponding crossings concurrently. In [55], the researchers proposed 16 routing patterns for realizing the escape routing of all nets. In the proposed patterns, each node represents a combination of two patterns of a net for two components. Accordingly, an arc passing from one node of net i to another node of net j implies that the associated routing patterns of nets i and j will not induce a crossing inside the two components; moreover, the escape order of these two nets will be consistent at the two boundaries. Subsequently, the researchers in [55] used a digraph to describe the SER problem; specifically, they modeled the problem as that of finding the longest path avoiding forbidden pairs of vertices. In their subsequent work [57], the researchers presented more general rather than straight patterns to address high-speed constraints, such as length-matching and adjacency constraints, and differential pair routes. However, such patterns are not sufficient to resolve routing problems for dense PCBs. In [46], a novel boundary routing approach that considers six types of routing was proposed to address the dense PCB routing problems. In this approach, the escape order between two components is maintained by updating the net ordering and boundary routing type during the backtracking procedure.

Although the aforementioned studies can optimize the escape routing of two components while also ensuring the feasibility of the area routing of such components, optimizing the routing process on one layer does not result in a reduction in the number of routing layers required [36]. A hierarchical multilayer SAT-based methodology was presented in [36] to address multilayer optimization and protracted SAT problems. To remedy the high computational complexity of the SAT problem, the mentioned study first applied a hierarchical SAT-based methodology to concurrently determine the feasible routes at each hierarchy. All previous works in SER do not tackle the length-matching constraints since the routing resources in an escape area is limited such that the optimal escape order to favor escape routing and area routing is its foremost objective.

Area routing includes the routing of data, addresses, clock and control signals, and is the essential stage to fulfill the imposed length-matching constraints of a PCB design. Studies on data routing [54, 69] have primarily focused on satisfying the length-matching constraint for each group of data. In this length-matching constraint, a tolerance value δ and a list of nets are defined such that the difference between the maximum wirelength and the minimum wirelength for the component-to-component routing of all nets in the list does not exceed δ. In [54], the Lagrangian relaxation method was applied to allocate routing resources based on the current wirelength and length slack. A directed acyclic graph with a certain number of vertices was constructed; through this graph, the minimum-cost path was identified to match the desired length in the PCB design. To execute general length-matching routing for a given topology, a study [69] applied a bounded-sliceline grid (BSG) structure to embed the components, pins, nets, and net topology; inflated the cells of the BSG containing the net topology according to the matched length; and then established the route with the desired length. The cell inflation problem was modeled as a mathematical non-convex programming problem with only linear and quadratic constraints.

Current trends towards dense pins and large-scale design still engender challenges in PCB design automation. Previous works resolve the length-matching constraints only in area routing. However, if the length slacks of the nets in a matching group vary considerably, length-matching-aware area routing algorithms may not be applicable. Length-matching-aware escape routing algorithms that route the nets of a length-matching group with similar wirelengths can help achieve routing closure in area routing. Another means of satisfying length-matching constraints is to make more room for snaking wires by minimizing the number of vias used. The existence of vias usually leads to failure in the insertion of snaking wires because such wires generally require a continuous free space. In [36], instead of minimizing the number of crossings in area routing, an SER scheme to realize a cross-free area routing was implemented for all layers, thus increasing the amount of free space for snaking wires. The advantage of this scheme is that more free space is reserved for snaking wires; the disadvantage is that SER is more difficult. Sophisticated PCB designers implement a via such that inserting a necessary snaking wire around the via remains feasible. Creating a more general and powerful length-matching-aware area router to address the routing of groups of signals, differential pairs, and buses can help advance the automation of PCB design. To push the advancement of PCB routing research, the PCB designs used in [36] will be released in the GitHub (https://github.com/boris-li/NYCU-PCB-benchmark) as the public benchmarks.

6 INTEGRATING HUMAN AND MACHINE INTELLIGENCE INTO ANALOG ROUTING

Analog ICs pose unique challenges for routing, In addition to the usual objective of wirelength minimization, analog routing often needs to consider other analog-design specific constraints and objectives such as current balancing, parasitics matching, signal coupling, etc. However, directly considering these complicated constraints during routing together with design rule checking, parasitic extractions, and simulations is simply not possible and practical, as even routing itself is mostly done sequentially, i.e., one net at a time. Usually these design-specific objectives are abstracted and considered through geometrical layout constraints, based on human designer experiences early on and/or generated by machine intelligence recently. These analog routing constraints include symmetric constraints, common-centroid constraints, topology matching, length matching, and so on.

For analog routing, layout constraints can be manually generated by experienced circuit designers, as they know what constraints could correlate well with the final analog circuit performance, yield, etc. However, this kind of approach heavily depends on the experiences and skills of circuit designers. It does not scale well and the process can be tedious and error-prone in particular when the design becomes complicated. Among all analog layout constraints, the symmetry constraints are particularly important, as analog designs frequently use differential typologies to reject common-mode noise and layout symmetry helps reduce mismatches and improves circuit performance. There have been a lot of studies on how to automatically extract these geometric constraints. Conventionally, the sensitivity analysis methods [8, 14, 49] and heuristic methods are used to extract the constraints [6, 18, 30, 40, 65, 66, 68, 72]. Recently, statistic [42] and deep graph neural network methods [13, 21, 32] are proposed to automatically generate the analog constraints. A survey of machine learning based analog constraint generation can be found in [73].

Once the constraints are generated, the routing engine will try its best to honor these layout constraints. In [58, 67], maze routing algorithms are extended to support the mirror symmetry constraints. In [56, 70], the length-matching routing approaches for general routing topologies are presented. In [53], an integer linear programming (ILP) formulation for analog routing is presented to simultaneously consider symmetry, common-centroid, topology-matching, and length-matching. But the ILP formulation is not scalable. Furthermore, for real-world designs, besides the straightforward mirror symmetry constraints, some variants of symmetry constraints may be needed or adopted to describe more sophisticated matching structures of nets. For example, [12] extends the conventional symmetry constraints into four variants: mirror-, cross-, self-, and partial-symmetry, as shown in Fig. 6.

The analog routing constraints are not limited to the simple geometric ones. In practice, many detailed design techniques can be incorporated for performance improvement. However, such design strategies are often design-specific and hard to transfer into a generalized geometric constraint. GeniusRoute [74] proposes a systematic methodology to bridge the gap between human design expertise and automated routing guidance generation. It uses machine learning (ML) models to extract the layout patterns and infer

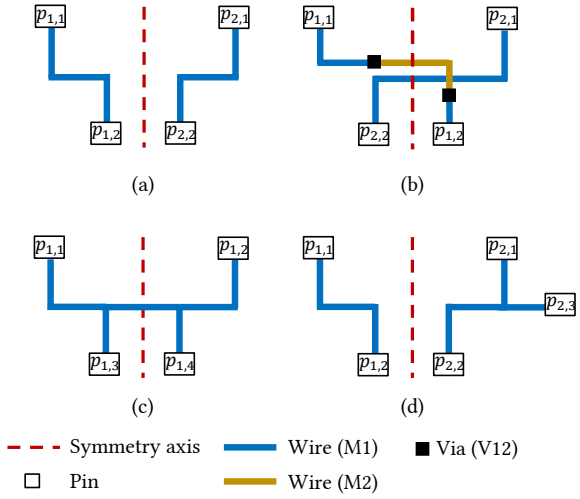

Figure 6: Examples of symmetry constraint variants: (a) Mirror-symmetry. (b) Cross-symmetry. (c) Self-symmetry. (d) Partial-symmetry [12].

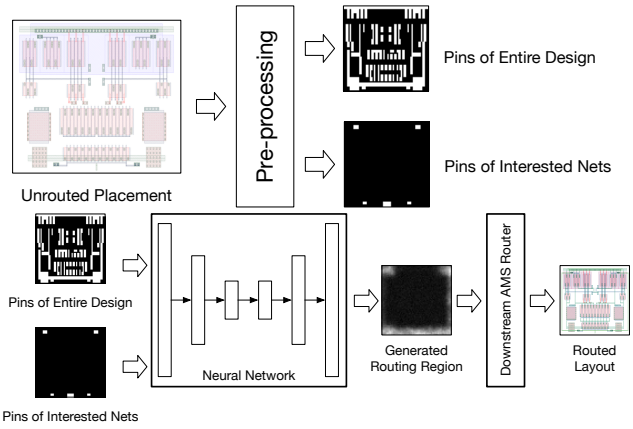

Figure 7: The GeniusRoute inference flow [74].

the best human behavior on routing. The ML models implicitly summarize design expertise and automatically extract constraints from existing layouts for analog routing. Essentially it builds a template library automatically from machine learning models and prior best layout practices. The GeniusRoute framework first extracts several images from the placed layouts and feed them into a trained neural network model. After training, the ML model can predict the best routing region for the given nets and guide the downstream automated analog detailed router, as shown in Fig. 7. Then the detailed routing will follow the machine learning generated guidance while honoring other geometric constraints such as the symmetry constraints. Experimental results show that GeniusRoute produces high-quality layouts with significant performance improvement over prior analog routing [74].

Building machine learning models for analog routing optimization still needs a good set of training data, e.g., as that in GeniurRoute [74]. It shall be noted that analog IC designs are very diverse, thus it may be hard to have one-model-good-for-all. How to generate good training dataset itself is a major research topic and it needs community efforts. Ultimately, we expect to see a combination of techniques levering both human and machine intelligence to tackle analog routing, including constraint-driven routing heuristics, supervised learning, semi-supervised learning, and reinforcement learning.

7 CONCLUSION

In this paper, we briefly reviewed the ISPD Initial Detailed Routing Contests and discussed a few challenging topics in VLSI routing. Routing is a complex and time consuming step in physical design. With the rapid advancement in hardware and software, routing is undergoing gigantic changes and we expect to see unprecedented progress in VLSI routing.

8 ACKNOWLEDGMENTS

David Z. Pan's work on analog routing is supported in part by NSF under grants 1704758 and 2112665, and by DARPA under the IDEA program.

REFERENCES

[1] 2019. Cadence PR Tool Innovus. https://www.cadence.com/en_US/home/tools/digital-design-and-signoff/soc-implementation-and-floorplanning/innovus-implementation-system.html
[2] 2021. LEF/DEF 5.3 to 5.8 exchange format. https://si2.org/oa-tools-utils-libs/
[3] Markus Ahrens, Dorothee Henke, Stefan Rabenstein, and Jens Vygen. 2022. Faster Goal-Oriented Shortest Path Search for Bulk and Incremental Detailed Routing. In *Proc.23rd IPCO conference, 2022.*
[4] Christoph Albrecht, Andrew B Kahng, Ion Mandoiu, and Alexander Zelikovsky. 2002. Floorplan evaluation with timing-driven global wireplanning, pin assignment, and buffer/wire sizing. In *Proc. ASP-DAC/VLSI Design.* 580–587.
[5] Charles J Alpert, Wing-Kai Chow, Kwangsoo Han, Andrew B Kahng, Zhuo Li, Derong Liu, and Sriram Venkatesh. 2018. Prim-Dijkstra revisited: Achieving superior timing-driven routing trees. In *Proc. ISPD.* 10–17.
[6] Sambuddha Bhattacharya, N. Jangkrajarng, Roy Hartono, and C.-J.R. Shi. 2004. Hierarchical extraction and verification of symmetry constraints for analog layout automation. In *Proc. ASPDAC.*
[7] Kenneth D Boese, Andrew B Kahng, Bernard A McCoy, and Gabriel Robins. 1995. Near-optimal critical sink routing tree constructions. *IEEE TCAD* 14, 12 (1995), 1417–1436.
[8] E. Charbon, E. Malavasi, and A. Sangiovanni-Vincentelli. 1993. Generalized constraint generation for analog circuit design. In *Proc. ICCAD.*
[9] Gengjie Chen, Chak-Wa Pui, Haocheng Li, Jingsong Chen, Bentian Jiang, and Evangeline FY Young. 2019. Detailed routing by sparse grid graph and minimum-area-captured path search. In *Proceedings of the 24th Asia and South Pacific Design Automation Conference.* 754–760.
[10] Gengjie Chen, Chak-Wa Pui, Haocheng Li, and Evangeline F. Y. Young. 2020. Dr. CU: Detailed Routing by Sparse Grid Graph and Minimum-Area-Captured Path Search. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39, 9 (2020), 1902–1915. https://doi.org/10.1109/TCAD.2019.2927542
[11] Gengjie Chen and Evangeline FY Young. 2019. Salt: provably good routing topology by a novel steiner shallow-light tree algorithm. *IEEE TCAD* 39, 6 (2019), 1217–1230.
[12] Hao Chen, Keren Zhu, Mingjie Liu, Xiyuan Tang, Nan Sun, and David Z. Pan. 2020. Toward Silicon-Proven Detailed Routing for Analog and Mixed-Signal Circuits. In *Proc. ICCAD.* 1–8.
[13] Hao Chen, Keren Zhu, Mingjie Liu, Xiyuan Tang, Nan Sun, and David Z. Pan. 2021. Universal Symmetry Constraint Extraction for Analog and Mixed-Signal Circuits with Graph Neural Networks. In *Proc. DAC.*
[14] U. Choudhury and A. Sangiovanni-Vincentelli. 1990. Constraint generation for routing analog circuits. In *Proc. DAC>.*
[15] Julia Chuzhoy, Anupam Gupta, Joseph Naor, and Amitabh Sinha. 2008. On the approximability of some network design problems. *ACM Transactions on Algorithms* 4, 2 (2008), 1–17.
[16] Siad Daboul, Nicolai Hähnle, Stephan Held, and Ulrike Schorr. 2018. Provably fast and near-optimum gate sizing. *IEEE TCAD* 37, 12 (2018), 3163–3176.
[17] Siad Daboul, Stephan Held, Bento Natura, and Daniel Rotter. 2019. Global interconnect optimization. In *Proc. ICCAD.* 1–8.
[18] Michael Eick, Martin Strasser, Kun Lu, Ulf Schlichtmann, and Helmut E. Graeb. 2011. Comprehensive Generation of Hierarchical Placement Rules for Analog Integrated Circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 30, 2 (2011), 180–193. https://doi.org/10.1109/TCAD.2010.2097172
[19] Jia-Wei Fang, Chin-Hsiung Hsu, and Yao-Wen Chang. 2009. An integer linear programming based routing algorithm for flip-chip design. *IEEE TCAD* 28, 1 (2009), 98–110.
[20] Upma Gandhi, Ismail Bustany, William Swartz, and Laleh Behjat. 2019. A Reinforcement Learning-Based Framework for Solving Physical Design Routing Problem in the Absence of Large Test Sets. In *2019 ACM/IEEE 1st Workshop on Machine Learning for CAD (MLCAD).* 1–6. https://doi.org/10.1109/MLCAD48534.2019.9142109
[21] Xiaohan Gao, Chenhui Deng, Mingjie Liu, Zhiru Zhang, David Z. Pan, and Yibo Lin. 2021. Layout Symmetry Annotation for Analog Circuits with Graph Neural Networks. In *Proc. ASPDAC.*
[22] Stèphano MM Gonçalves, Leomar S Rosa, and Felipe S Marques. 2019. DRAPS: A design rule aware path search algorithm for detailed routing. *IEEE Transactions on Circuits and Systems II: Express Briefs* 67, 7 (2019), 1239–1243.
[23] Stephan Held, Dirk Müller, Daniel Rotter, Rudolf Scheifele, Vera Traub, and Jens Vygen. 2017. Global routing with timing constraints. *IEEE TCAD* 37, 2 (2017), 406–419.
[24] Stephan Held and Daniel Rotter. 2013. Shallow-light Steiner arborescences with vertex delays. In *Proc. 16th IPCO.* Springer, 229–241.
[25] Xianlong Hong, Tianxiong Xue, Jin Huang, Chung-Kuan Cheng, and Ernest S Kuh. 1997. TIGER: an efficient timing-driven global router for gate array and standard cell layout design. *IEEE TCAD* 16, 11 (1997), 1323–1331.
[26] Jiang Hu and Sachin S Sapatnekar. 2002. A timing-constrained simultaneous global routing algorithm. *IEEE TCAD* 21, 9 (2002), 1025–1036.
[27] Fengxian Jiao and Sheqin Dong. 2016. Ordered escape routing for grid pin array based on min-cost multi-commodity flow. In *Proc. ASPDAC.* IEEE, 384–389.
[28] Andrew B Kahng, Lutong Wang, and Bangqi Xu. 2018. Tritonroute: An initial detailed router for advanced vlsi technologies. In *2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD).* IEEE, 1–8.
[29] Andrew B Kahng, Lutong Wang, and Bangqi Xu. 2021. TritonRoute-WXL: The Open Source Router with Integrated DRC Engine. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (2021).
[30] M.E. Kole, J. Smit, and O.E. Herrmann. 1994. Modeling symmetry in analog electronic circuits. In *IEEE International Symposium on Circuits and Systems (ISCAS).*
[31] Yukiko Kubo and Atsushi Takahashi. 2006. Global routing by iterative improvements for two-layer ball grid array packages. *IEEE TCAD* 25, 4 (2006), 725–733.
[32] K. Kunal, P. Poojary, T. Dhar, M. Madhusudan, R. Harjani, and S. Sapatnekar. 2020. A General Approach for Identifying Hierarchical Symmetry Constraints for Analog Circuit Layout. In *Proc. ICCAD.*
[33] Haocheng Li, Gengjie Chen, Bentian Jiang, Jingsong Chen, and Evangeline FY Young. 2019. Dr. cu 2.0: A scalable detailed routing framework with correct-by-construction design rule satisfaction. In *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD).* IEEE, 1–7.
[34] Zhaopo Liao and Sheqin Dong. 2020. A constraint-driven compact model with partition strategy for ordered escape routing. In *Proc. GLSVLSI.* ACM, 393–398.
[35] Shiju Lin, Jinwei Liu, and Martin D.F. Wong. 2021. GAMER: GPU Accelerated Maze Routing. In *2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD).* 1–8. https://doi.org/10.1109/ICCAD51958.2021.9643563
[36] Shih-Ting Lin, Hung-Hsiao Wang, Chia-Yu Kuo, Yolo Chen, and Yih-Lang Li. 2021. A complete PCB routing methodology with concurrent hierarchical routing. In *Proc. DAC.* IEEE, 1141–1146.
[37] Derong Liu, Bei Yu, Salim Chowdhury, and David Z Pan. 2017. TILA-S: Timing-driven incremental layer assignment avoiding slew violations. *IEEE TCAD* 37, 1 (2017), 231–244.
[38] Genggeng Liu, Xinghai Zhang, Wenzhong Guo, Xing Huang, Wen-Hao Liu, Kai-Yuan Chao, and Ting-Chi Wang. 2021. Timing-Aware Layer Assignment for Advanced Process Technologies Considering Via Pillars. *IEEE TCAD* (2021), 1–1. https://doi.org/10.1109/TCAD.2021.3100296
[39] Jinwei Liu, Gengjie Chen, and Evangeline FY Young. 2021. REST: Constructing Rectilinear Steiner Minimum Tree via Reinforcement Learning. In *Proc. 58th DAC.* 1135–1140.
[40] Jiayi Liu, Sheqin Dong, Xianlong Hong, Yibo Wang, Ou He, and Satoshi Goto. 2008. Symmetry constraint based on mismatch analysis for analog layout in SOI technology. In *Proc. ASPDAC.*
[41] Jinwei Liu, Chak-Wa Pui, Fangzhou Wang, and Evangeline F. Y. Young. 2020. CUGR: Detailed-Routability-Driven 3D Global Routing with Probabilistic Resource Model. In *2020 57th ACM/IEEE Design Automation Conference (DAC).* 1–6. https://doi.org/10.1109/DAC18072.2020.9218646

[42] M. Liu, W. Li, K. Zhu, B. Xu, Y. Lin, L. Shen, X. Tang, N. Sun, and D. Z. Pan. 2020. S^{3}DET: Detecting System Symmetry Constraints for Analog Circuits with Graph Similarity. In *Proc. ASPDAC*.

[43] Siting Liu, Peiyu Liao, Rui Zhang, Zhitang Chen, Wenlong Lv, Yibo Lin, and Bei Yu. 2022. FastGR: Global Routing on CPU-GPU with Heterogeneous Task Graph Scheduler. In *2022 Design, Automation Test in Europe Conference Exhibition (DATE)*.

[44] Wen-Hao Liu, Wei-Chun Kao, Yih-Lang Li, and Kai-Yuan Chao. 2013. NCTU-GR 2.0: Multithreaded Collision-Aware Global Routing With Bounded-Length Maze Routing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 32, 5 (2013), 709–722. https://doi.org/10.1109/TCAD.2012.2235124

[45] Wen-Hao Liu, Stefanus Mantik, Wing-Kai Chow, Yixiao Ding, Amin Farshidi, and Gracieli Posser. 2019. ISPD 2019 Initial Detailed Routing Contest and Benchmark with Advanced Routing Rules. In *Proceedings of the 2019 International Symposium on Physical Design* (San Francisco, CA, USA) *(ISPD '19)*. 147–151.

[46] Lijuan Luo, Tan Yan, Qiang Ma, Martin D. F. Wong, and Toshiyuki Shibuya. 2011. A new strategy for simultaneous escape based on boundary routing. *IEEE TCAD* 30, 2 (2011), 205–214.

[47] Lijuan Luo Luo and Martin D. F. Wong. 2008. Ordered escape routing based on Boolean satisfiability. In *Proc. ASPDAC*. IEEE, 244–249.

[48] Lijuan Luo Luo and Martin D. F. Wong. 2009. On using SAT to ordered escape problems. In *Proc. ASPDAC*. IEEE, 594–599.

[49] E. Malavasi, E. Charbon, E. Felt, and A. Sangiovanni-Vincentelli. 1996. Automation of IC layout with analog constraints. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 15, 8 (1996), 923–942. https://doi.org/10.1109/43.511572

[50] Stefanus Mantik, Gracieli Posser, Wing-Kai Chow, Yixiao Ding, and Wen-Hao Liu. 2018. ISPD 2018 initial detailed routing contest and benchmarks. In *Proceedings of the 2018 International Symposium on Physical Design*. 140–143.

[51] Dirk Müller, Klaus Radke, and Jens Vygen. 2011. Faster min–max resource sharing in theory and practice. *Mathematical Programming Computation* 3, 1 (2011), 1–35.

[52] Isadora Oliveira, Marcelo Danigno, Paulo F. Butzen, and Ricardo Reis. 2021. Benchmarking Open Access VLSI Partitioning Tools. In *2021 IEEE 12th Latin America Symposium on Circuits and System (LASCAS)*. 1–4. https://doi.org/10.1109/LASCAS51355.2021.9459131

[53] Hung-Chih Ou, Hsing-Chih Chang Chien, and Yao-Wen Chang. 2014. Nonuniform Multilevel Analog Routing with Matching Constraints. *IEEE Trans. on CAD* 33, 12 (2014), 1942–1954.

[54] Muhammet Mustafa Ozdal and Martin D. F. Wong. 2003. Length-matching routing for high-speed printed circuit boards. In *Proc. ICCAD*. IEEE, 394–400.

[55] Muhammet Mustafa Ozdal and Martin D. F. Wong. 2006. Algorithms for simultaneous escape routing and layer assignment of dense PCBs. *IEEE TCAD* 25, 8 (2006), 1510–1522.

[56] Muhammet M. Ozdal and Martin D. F. Wong. 2006. A Length-Matching Routing Algorithm for High-Performance Printed Circuit Boards. *IEEE Trans. on CAD* 25, 12 (2006), 2784–2794.

[57] Muhammet Mustafa Ozdal, Martin D. F. Wong, and Philip S. Honsinger. 2008. Simultaneous escape-routing algorithms for via minimization of high-speed boards. *IEEE TCAD* 27, 1 (2008), 84–95.

[58] Po-Cheng Pan, Hung-Ming Chen, Yi-Kan Cheng, Jill Liu, and Wei-Yi Hu. 2012. Configurable Analog Routing Methodology via Technology and Design Constraint Unification. In *Proc. ICCAD*. 620–626.

[59] Tong Qu, Yibo Lin, Zongqing Lu, Yajuan Su, and Yayi Wei. 2021. Asynchronous Reinforcement Learning Framework for Net Order Exploration in Detailed Routing. In *2021 Design, Automation Test in Europe Conference Exhibition (DATE)*. 1815–1820. https://doi.org/10.23919/DATE51398.2021.9474007

[60] Radhamanjari Samanta, Adil I Erzin, Soumyendu Raha, Yuriy V Shamardin, Ivan I Takhonov, and Vyacheslav V Zalyubovskiy. 2015. A provably tight delay-driven concurrently congestion mitigating global routing algorithm. *Appl. Math. Comput.* 255 (2015), 92–104.

[61] Rudolf Scheifele. 2017. Steiner trees with bounded RC-delay. *Algorithmica* 78, 1 (2017), 86–109.

[62] Eugene Shragowitz and S Keel. 1987. A global router based on a multicommodity flow model. *Integration* 5, 1 (1987), 3–16.

[63] Fan-Keng Sun, Hao Chen, Ching-Yu Chen, Chen-Hao Hsu, and Yao-Wen Chang. 2018. A multithreaded initial detailed routing algorithm considering global routing guides. In *2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 1–7.

[64] Peishan Tu, Chak-Wa Pui, and Evangeline FY Young. 2019. Simultaneous reconnection surgery technique of routing with machine learning-based acceleration. *IEEE TCAD* 39, 6 (2019), 1245–1257.

[65] Po-Hsun Wu, Mark Po-Hung Lin, Tung-Chieh Chen, Ching-Feng Yeh, Xin Li, and Tsung-Yi Ho. 2015. A Novel Analog Physical Synthesis Methodology Integrating Existent Design Expertise. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 34, 2 (2015), 199–212. https://doi.org/10.1109/TCAD.2014.2379630

[66] Po-Hsun Wu, Mark Po-Hung Lin, and Tsung-Yi Ho. 2015. Analog layout synthesis with knowledge mining. In *European Conference on Circuit Theory and Design (ECCTD)*.

[67] Linfu Xiao, Evangeline F. Y. Young, Xiaoyong He, and K. P. Pun. 2010. Practical Placement and Routing Techniques for Analog Circuit Designs. In *Proc. ICCAD*. 675–679.

[68] B. Xu, K. Zhu, M. Liu, Y. Lin, S. Li, X. Tang, N. Sun, and D. Z. Pan. 2019. MAGICAL: Toward Fully Automated Analog IC Layout Leveraging Human and Machine Intelligence. In *Proc. ICCAD*.

[69] Tan Yan and Martin D. F. Wong. 2008. BSG-route: A length-matching router for general topology. In *Proc. ICCAD*. IEEE, 499–505.

[70] Tan Yan and Martin D. F. Wong. 2008. BSG-Route: A Length-Matching Router for General Topology. In *Proc. ICCAD*. 499–505.

[71] Xinghai Zhang, Zhen Zhuang, Genggeng Liu, Xing Huang, Wen-Hao Liu, Wenzhong Guo, and Ting-Chi Wang. 2020. Minidelay: Multi-strategy timing-aware layer assignment for advanced technology nodes. In *Proc. DATE*. IEEE, 586–591.

[72] Zhe Zhou, Sheqin Dong, Xianlong Hong, Qingsheng Hao, and Song Chen. 2005. Analog constraints extraction based on the signal flow analysis. In *International Conference on ASIC (ASICON)*.

[73] Keren Zhu, Hao Chen, Mingjie Liu, and David Z. Pan. 2022. Automating Analog Constraint Extraction: From Heuristics to Learning. In *Proc. ASPDAC*.

[74] Keren Zhu, Mingjie Liu, Yibo Lin, Biying Xu, Shaolan Li, Xiyuan Tang, Nan Sun, and David Z. Pan. 2019. GeniusRoute: A New Analog Routing Paradigm Using Generative Neural Network Guidance. In *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*.