SAGERoute: Synergistic Analog Routing Considering Geometric and Electrical Constraints with Manual Design Compatibility

Haoyi Zhang[1†], Xiaohan Gao[2,1†], Haoyang Luo[1], Jiahao Song[1],
Xiyuan Tang[3,1], Junhua Liu[1,5], Yibo Lin[1,4,5*], Runsheng Wang[1,4,5], Ru Huang[1,4,5]
[1]School of Integrated Circuits [2]School of Computer Science [3]Institute for Artificial Intelligence, Peking University
[4]Beijing Advanced Innovation Center for Integrated Circuits
[5]Institute of Electronic Design Automation, Peking University, Wuxi, China

***Abstract*—Routing is critical to the post-layout performance of analog circuits. As modern analog layouts need to consider both geometric constraints (e.g., design rules and low bending constraints) and electrical constraints (e.g., electromigration (EM), IR drop, symmetry, etc.), it becomes increasingly challenging to investigate the complicated design space. Most previous work has focused only on geometric constraints or basic electrical constraints, lacking holistic and systematic investigation. Such an approach is far from typical manual design practice and can not guarantee post-layout performance on real-world designs. In this work, we propose SAGERoute, a synergistic routing framework taking both geometric and electrical constraints into consideration. Through Steiner tree based wire sizing and guided detailed routing, the framework can generate high-quality routing solutions efficiently under versatile constraints on real-world analog designs.**

***Index Terms*—analog routing, electrical constraints, geometric constraints, high quality**

I. Introduction

Analog layout design heavily relies on manual efforts. Routing is one of the most tedious steps in the layout design stage, where designers have to draw wire connections with hands carefully. As Figure 1 indicates, to guarantee performance and functionality, analog routing must consider versatile constraints such as EM [1], [2] & IR drop [3] constraints on power/ground nets and critical nets, symmetry constraint, sensitive area constraint, etc. Based on the cause of the constraints, we roughly categorize routing constraints into geometric constraints (e.g., design rules) and electrical constraints (e.g., EM and IR drop constraints on power/ground nets, symmetry nets, sensitive area constraint, etc.).

To automate analog routing, the literature has explored techniques to tackle various constraints. Recent work proposes a detailed routing framework that considers multiple complex design rules [4]. To avoid mismatch, previous studies implement symmetry nets for analog layouts [5], [6], [7], [8], [9], [10], [11]. To guarantee post-layout performance and reliability, early work explores the EM and IR drop effects, which conducts wire planning and determines the wire width (wire sizing) [12], [13], [14], [15]. Despite plenty of studies, existing studies have the following limitations.

1) There is no framework that tackles multiple constraints (including both geometric constraints and electric constraints) in a uniform routing framework.
2) Most previous studies ignore the differences in resistance and capacitance caused by different net routing topologies when handling EM and IR drop constraints.

[†]Equal Contribution.
[*]Corresponding author: yibolin@pku.edu.cn

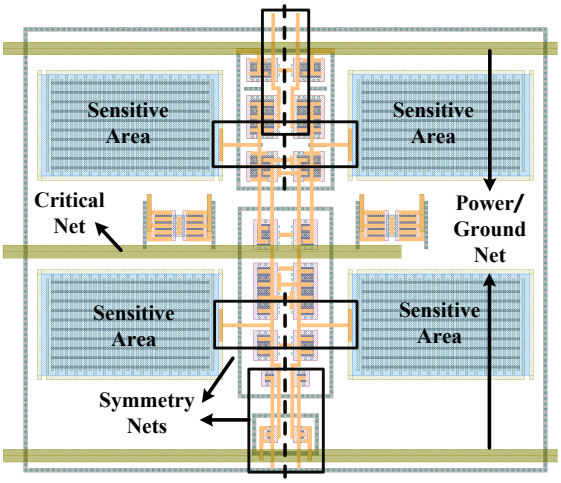

Fig. 1: A real-world layout example with multiple constraints considered.

In this paper, we propose a synergistic framework of routing named SAGERoute, for analog/mixed-signal integrated circuits. Our framework takes multiple routing constraints into consideration. In Table I, we make a comparison between the proposed framework with other routers of previous mainstream layout frameworks. Our framework is capable of handling the major geometric constraints and electrical constraints. We summarize our main contributions as follows.

- We propose SAGERoute, a synergistic routing framework, which considers both geometric and electrical constraints, and our framework is compatible with manually placed layouts.
- We design a novel Steiner tree based wire sizing scheme that considers both net routing topologies and wire sizes with accurate estimation of resistance and capacitance for EM and IR drop constraints.
- We develop a multi-constraint aware routing algorithm that can synergistically work with complicated constraints and the wire sizing scheme.
- Experimental results show that our framework can achieve competitive performance, and demonstrate advantages in real-world taped-out analog designs with manual placement.

The remainder of this paper is organized as follows. Section II outlines the preliminaries. Section III details our implementation. Section IV demonstrates the experimental results, and Section V concludes the paper.

978-3-9819263-7-8/DATE23/© 2023 EDAA

TABLE I: COMPARISON OF EXISTING ANALOG ROUTING TOOLS

Features	Geometric Constraints			Electrical Constraints			
	Parallel-run spacing	End-of-line spacing	Low Bending	Electromigration	IR drop	Symmetry	Layout sensitive area
MAGICAL [16], [4], [17]	✓	✓	✗	✗	✓	✓	✗
ALIGN [18], [19], [20]	✓	✓	✗	✗	✗	✓	✗
LAYGENII [21], [22], [23]	✓	✓	✗	✗	✗	✓	✓
SAGERoute	✓	✓	✓	✓	✓	✓	✓

II. PRELIMINARIES

In this section, we introduce the basic background of analog routing and formulate the problem.

A. Analog Routing Methodology

A typical approach for analog routing is to construct a 3-D grid graph [18], [17], where each edge represents the routing resources using metal wires or vias. Pins of real devices are modeled as a set of vertices on the graph. Routing essentially needs to connect pins with a set of edges with the lowest costs.

B. Electrical Constraints

Figure 2 summarizes the electrical constraints and geometric constraints considered in this paper. Electrical constraints include Electromigration (EM), IR drop, symmetry, and layout-sensitive area.

Electromigration is the migration of metal wire caused by the gradual movement of ions. EM can lead to unexpected dilemmas such as metal openings and metal shorts, which further may lead to malfunction or even failure in analog design. EM effects are mainly affected by current density and temperature as well as can be diminished by increasing the wire width. EM effects constrain the minimum wire width by:

$$w_{EM} = \frac{I_{peak}}{t_{metal} J_{max}(T_{ref})} \tag{1}$$

where I_{peak} is the current, t_{metal} denotes the thickness of the corresponding metal layer, and $J_{max}(T_{ref})$ is the maximum allowed current density at the reference temperature T_{ref}.

IR drop refers to a voltage drop that appears at the resistive component of any impedance, causing circuit performance degradation. We can calculate the static IR drop by accumulating the IR drop ir per wire segment as follows.

$$IRdrop_{static} = \sum ir \tag{2}$$

Symmetry is a regular constraint in analog layout. The proposed routing approach considers not only general mirror-symmetry and central symmetry [24], [25], but also partial symmetry. Partial symmetry is more practical in real-world analog design.

Layout-sensitive area refers to the intent to protect a specified device group from circuit parasitic effects, such as signal cross-talk, bias deviation, etc. For example, a typical sensitive area in analog layout is the oxide-diffusion area. During the routing procedure, our framework transforms the layout-sensitive area into routing obstacles and avoids wires crossing the obstacles. The framework is flexible to handle other sensitive areas once specified.

C. Geometric Constraints

Aiming at promoting automated analog routing towards practice, we not only consider basic geometric constraints like minimum width/spacing/area rules, but also more complicated constraints like low bending, parallel run spacing, and end-of-line spacing, as shown in Figure 2.

Parallel run length spacing guarantees the spacing $S_{1,2}$ between a pair of parallel run wires with specified wire width W_1, W_2 and parallel run length $L_{1,2}$. *End-of-line (EOL) spacing* guarantees the spacing at each wire end. Figure 2 demonstrates more details. On both sides of the EOL region, two yellow parallel edge regions with *parSpace* * *parWidth* are defined. If other metal wires overlap with the light yellow region, the EOL spacing rule demands no metal wires overlap with the orange region. *Low bending* constraint keeps routing free from jagging.

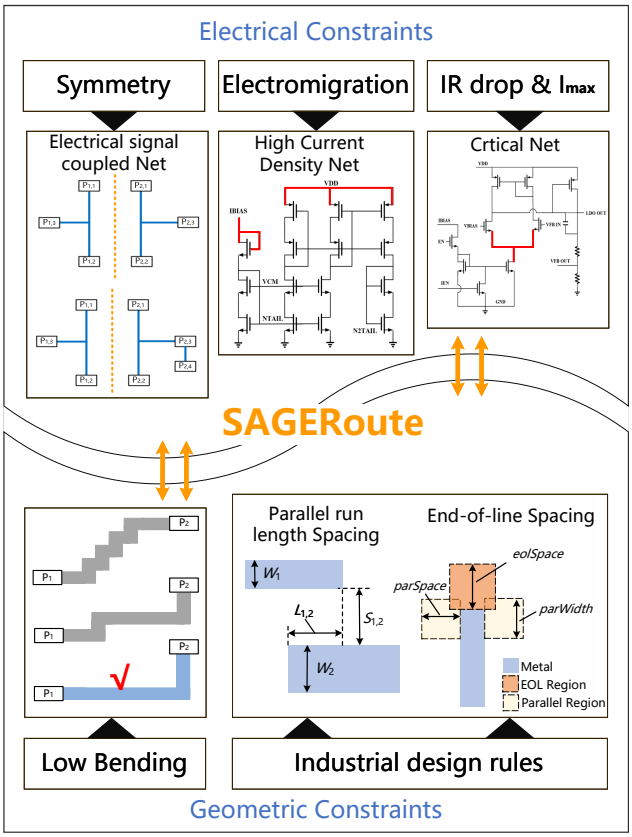

Fig. 2: Multiple constraints for SAGERoute.

D. Problem Formulation

Problem 1. Given a placement result PL which consists of nets $N = \{n_i | 1 \leq i \leq |N|\}$ and device pin locations $P = \{p_i | 1 \leq i \leq |P|\}$, a set of constraints $C = \{c_i | 1 \leq i \leq |C|\}$, generate a routing solution $R = \{R_i | 1 \leq i \leq |R|\}$ for each net $n_i \in N$ considering $c_j \in C$ such that all nets are routed without any violations of constraints.

III. ALGORITHM

Overview of the synergistic analog routing framework is depicted in Figure 3. Circuit netlist as well as technology files including DRC rules, electromigration parameters, and metal layer parameters are necessary for the proposed framework. Design configurations for IR

drop (e.g., voltage margin and critical nets) and EM (e.g., operating temperature and critical nets) are also necessary. Before the major flow starts, schematic simulation is performed to generate current information for the specified circuit netlist. The major flow consists of two phases: 1. Electrical constraints processing; 2. Multi-constraint aware routing. Phase 1 involves Steiner tree based wire sizing, symmetry type recognition, and sensitive area detection. The former determines wire width, the middle determines symmetry constraints and the latter generates routing obstacles, all of which will be used to guide the multi-constraint aware routing in phase 2.

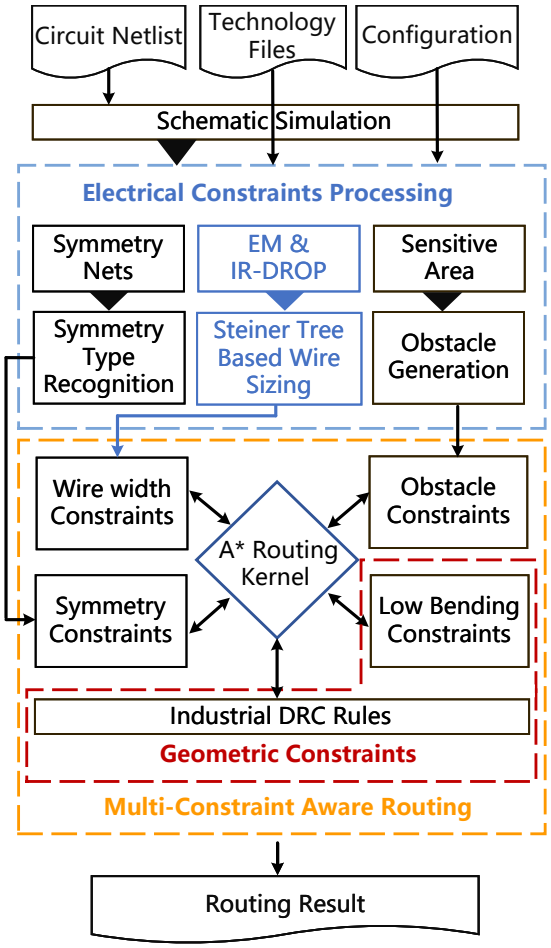

Fig. 3: Overview of SAGERoute framework.

A. Wire Sizing for Electrical Constraints

This step will generate a Steiner tree for a net and determine the wire sizing of each segment. Both the Steiner tree and the wire sizing solutions serve as guidance for the follow-up routing step. In this step, we first generate a specific Steiner tree as the net routing topology. Then, the follow-up algorithm solves a linear programming problem to get the wire width based on the net topology and the electrical constraints.

1) Steiner Tree Generation: In view of Equation 2, IR drop can be interpreted as a constraint on the sum of $I \times R$ over several wire segments. To meet the IR drop requirement, the intuition is to make the wire segments with higher current (I) have lower resistance (R), which means routing as shortest and direct as possible and increasing the wire width. In order to route shorter paths for higher currents and estimate wire width, we propose a current-weighted wirelength metric

Algorithm 1 Steiner Tree Generation Inspired by Huffman Coding

Require: A net n, and its pins $\{p_i\}$ with coordinates $\{(x_{p_i}, y_{p_i})\}$, and the current $\{i_{p_i}\}$ from each pin.
Ensure: A tree topology T.

function ESTIMATECOST(S_k, S)
  balance point (x_{S_k}, y_{S_k}) for S_k, $(x_{S\setminus S_k}, y_{S\setminus S_k})$ for $S\setminus S_k$
  cost $c_{S_k} \leftarrow \sum_{v\in S_k}$ IRCOST$(x_v, y_v, x_{S_k}, y_{S_k}, i_v)$
  cost $c_{S_k,S\setminus S_k} \leftarrow$ IRCOST$(x_{S_k}, y_{S_k}, x_{S\setminus S_k}, y_{S\setminus S_k}, \sum_{v\in S_k} i_v)$
  cost $c_{S\setminus S_k} \leftarrow \sum_{v\in S\setminus S_k}$ IRCOST$(x_v, y_v, x_{S\setminus S_k}, y_{S\setminus S_k}, i_v)$
  return $c \leftarrow c_{S_k} + c_{S,S\setminus S_k} + c_{S\setminus S_k}$
end function
function TREEGENERATION(x)
  Set of nodes $S \leftarrow \{v_{\{p_i\}}\} \cup \{v_{\{\emptyset\}}\}$
  Tree $T \leftarrow \emptyset$
  while $|S| > 1$ **do**
    for any k nodes $S_k = \{v_{s_1}, \cdots, v_{s_k}\}$ of set S **do**
      ESTIMATECOST(S_k, S)
    end for
    pick k nodes $S_k = \{v_{s_1}, \cdots, v_{s_k}\}$ with the lowest cost
    $S.erase(v_{s_1}, \cdots, v_{s_k})$
    new node v_{s_m} with $s_m = s_1 \cup \cdots \cup s_k$
    calculate current $i_{v_{s_m}} = \sum_{v_{s_i}} i_{v_{s_i}}$ for v_{s_m}
    $S.insert(v_{s_m})$
    $T.add\_edge(v_{s_i}, v_{s_m})$ for any $v_{s_i} \in \{v_{s_1}, \cdots, v_{s_k}\}$
  end while
  return T
end function

for the rectilinear Steiner tree. The generated tree topology provides a rough wirelength estimation for the next stage, to help determine the wire width based on the constraint on resistance. The current-weighted wire length is defined as follows:

Definition 1 (*Current-Weighted wire length*)**.** Given a net n and its routing tree with pins and Steiner points $\{p_i\}$. The routing tree is represented with edges $E = \{(p_i, p_j)\}$ which has a current $i_{(p_i,p_j)}$ flowing through. We define the current-weighted wirelength as: $WL^{cw} = \sum_{(p_i,p_j)\in E}(|i_{(p_i,p_j)}| + c) * ManhattanDistance(p_i, p_j)$, where c is a positive constant to avoid zero-current wire segment to have unlimited length.

Rectilinear Steiner minimal tree generation is an NP-Complete problem [26]. Generating a rectilinear Steiner tree with minimal current-weighted wire length can not be simpler than the rectilinear Steiner minimal tree. We propose an algorithm inspired by Huffman coding [27] to approximate the minimum current-weighted wire length. Huffman coding assigns a shorter code length for a symbol with higher frequency and *vice versa*. We make an analogy between the frequency of a symbol and the current through a wire segment, and develop an algorithm that puts wire segments with higher currents closer to the root of the tree. Algorithm 1 details the steps of bottom-up Steiner tree generation. To explain the algorithm, we define the following concepts. $IRCost$ is used to measure the cost c when adding a new Steiner point.

Definition 2 (*Tree Node*)**.** A tree node represents a pin or an inserted Steiner point. Each tree node maintains a set that records all the pins in its subtree and tracks the current flowing out of itself.

Definition 3 (*Balance Point*)**.** We define the balance point as the current-weighted version of the mass center.

Definition 4 (*IRCost*)**.** We define a function to estimate the cost of a

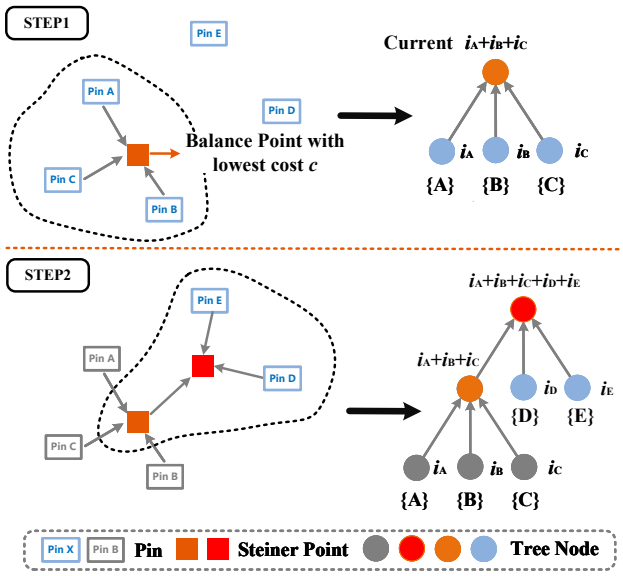

Fig. 4: An example of the Steiner tree generation.

current flow through a path. Given two points (x_s, y_s), (x_e, y_e), and current i, define $\text{IRCost}(x_s, y_s, x_e, y_e, i) = |i|(|x_s - x_e| + |y_s - y_e|)$.

The basic idea is to build a Steiner tree bottom-up by always inserting a new Steiner point among a group of tree nodes which has the lowest total $IRCost$. Algorithm 1 constructs a k-ary Steiner tree. First, the algorithm initializes a queue S with tree nodes of pins $\{v_{\{p_i\}}\}$. As we expect the tree to be fully near the tree root, the algorithm is supposed to insert $[k-1-(|\{p_i\}|-1)] mod (k-1)$ empty nodes to the queue (line 8). Then the algorithm computes a cost for every k nodes. The cost combines three parts estimated by IRCost. The first part c_{S_k} estimates the IRCost inside the chosen k nodes S_k (line 3). The algorithm calculates the balance points for the chosen S_k and the left $S \backslash S_k$ (line 2). The second part $c_{S_k, S \backslash S_k}$ estimates the IRCost between the balance points (line 4). The third part $c_{S \backslash S_k}$ estimates the IRCost inside the left points (line 5). Then, we choose the k nodes with minimal cost. Figure 4 demonstrates an example of this generation process, in which we assign $k = 3$. The first chosen k nodes with lowest cost are Pin A, B, C. Then, a new Steiner point with pin set $\{A, B, C\}$ and current $i_A + i_B + i_C$ is inserted and edges are generated between the pins and the Steiner point. Note that the current $i_A + i_B + i_C$ is calculated based on Kirchhoff's law. After a node representing the new Steiner point is inserted to the queue, the tree generation repeats the process until there is only one node in the queue (line 10). The Steiner tree generation terminates at a tree root with all pins in its subtree.

2) Wire Sizing Based on Steiner Tree: We further determine the sizing of every wire segment based on the generated Steiner tree and the electrical constraints.

Each edge (v_i, v_j) of the generated tree corresponds to a wire segment with width w and length l as its variables. We formulate the sizing problem as determining each width w given length l, subject to the electrical constraints defined in Section II-B. In order to estimate the electrical constraints, we first calculate the resistance and the capacitance of each wire segment. We employ the widely-used rectangle wire resistance model and the parallel plate capacitance model. The resistance R is proportional to $\frac{1}{w}$:

$$R = R_{sg} \frac{l}{w}, \quad R_{sg} = \frac{c}{t_{metal}} \tag{3}$$

where c is a constant related to the material and t_{metal} is the thickness. The capacitance C is proportional to w:

$$C = c_{psm} \cdot w \cdot l \tag{4}$$

where c_{psm} is the capacitance per square micron.

We uniformly consider all electrical constraints as linear inequalities on α. Here α denotes $\frac{1}{w}$.

$$\begin{aligned} \min \sum_{\forall \alpha} & i * \alpha \\ & R_{sg} \cdot l \cdot \alpha \le R_{thres} && (resistance) \\ & C_{psm} \cdot l / C_{thres} \le \alpha && (capacitance) \\ & \alpha \le 1 / w_{EM} && (EM) \\ & \sum_{\alpha \in IRdrop} i \cdot \alpha \le V_{thres} && (IRdrop) \end{aligned} \tag{5}$$

where R_{thres}, C_{thres}, and V_{thres} are the resistance threshold, capacitance threshold, and maximum tolerant IR drop respectively. Equation (5) composes a linear programming problem. The width w of each wire segment can be determined by solving the LP.

B. Multi-Constraint Aware Routing

Guided by the net routing topologies and wire sizing solution from the previous step , we perform multi-constraint aware routing.

1) Self-Guided Symmetry Routing: Net symmetry is an essential constraint to avoid mismatch in analog layouts.

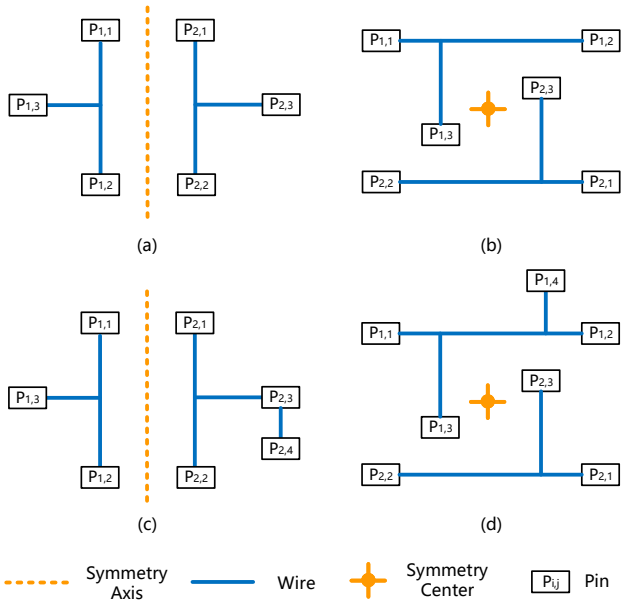

Fig. 5: Typical symmetry constraints: (a) mirror symmetry, (b) central symmetry, (c) partial mirror symmetry, and (d) partial central symmetry.

Figure 5 sketches four typical symmetry constraints, i.e., mirror symmetry, central symmetry, partial mirror symmetry, and partial central symmetry. To handle various symmetric net topologies in practice, we propose a self-guided symmetry routing algorithm, which is more flexible than conventional methods that only consider symmetry as hard constraint [17], [18].

Figure 6 gives an example of the symmetric routing strategy. The basic idea is to route one of the two nets first and then construct the symmetric counterpart as the topology guidance for the other net. The

procedure of the algorithm is illustrated in Algorithm 2. We route one net first and construct the topology guidance of another net (lines 2-5). We generate two-pin subnets for the unrouted net and minimize the distance between subnets and the topology guidance (lines 7-9). The minimum distance is achieved by considering the distance (*D1, D2*) between each searched objective node and the topology guidance in the routing cost. As Figure 6(c) illustrated, *D1* and *D2* are the manhattan distances between the routing objective node and the guidance paths. The final routing solution avoids the obstacle and follows the guidance as Figure 6(d) shown.

Algorithm 2 Self-Guided Symmetry Routing

Require: Two nets n_i and n_j waiting for symmetric routing.
Ensure: Symmetric routing solutions R_i and R_j for nets n_i and n_j.
1: **function** SELF-GUIDED SYMMETRY ROUTING(n_i, n_j)
2: $R_i \leftarrow$ ROUTE SINGLE NET(n_i)
3: $Guidance \leftarrow R_i$
4: $SymType \leftarrow$ DETERMINE SYMMETRY TYPE(n_i, n_j)
5: $Guidance \leftarrow$ SYM TRANSFORM($Guidance, SymType$)
6: $sns_j \leftarrow$ GENERATE SUBNET(n_j)
7: **for** $sn_i \in sns_i$ **do**
8: $Routingcost+ =$ DISTANCE COMPUTE($sn_i, Guidance$)
9: $R_{j,k} \leftarrow$ MINIMIZE($sn_i, Routingcost$)
10: **end for**
11: $R_j \leftarrow$ SUM($R_{j,k2}$)
12: **end function**

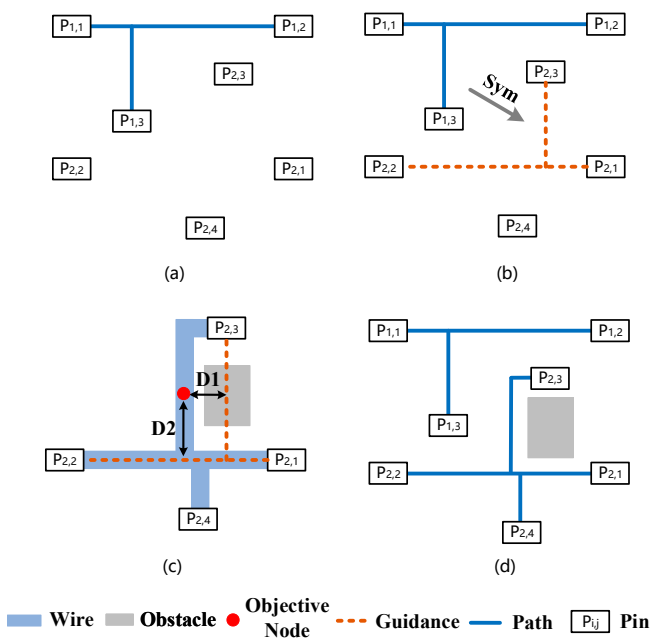

Fig. 6: An example of self-guided symmetry routing: (a) route a single net, (b) generate symmetry guidance, (c) routing subject to guidance, (d) final symmetric routing solution.

2) Sensitive Area Aware Routing: A real-world analog layout often contains sensitive areas, which should be used for routing. Sensitive areas are universal and require special attention among real-world analog layouts. Our framework is capable of avoiding the specified obstacles which can be determined by sensitive areas.

Given that the oxide-diffusion area is one of the most common sensitive areas in analog layout, the oxide-diffusion area obstacles are automatically positioned for routing. Users are also able to assign other desired obstacles.

3) Low Bending Routing: Low bending constraint is a fundamental requirement for silicon-proven analog routing. Jagged wires often lead to a great deal of DRC violations and extremely high overhead for post-processing. To implement low bending routing, our routing algorithms monitors routing directions during path searching, and adds additional cost to objective nodes that result in bending. In the experiments, this additional cost is set to five times of the half-perimeter wirelength of a net.

TABLE II: BENCHMARK STATISTICS.

Benchmark	Placement Type	Technology Node	Die Size	Post-layout Simulation Time
OTA	Automatic	TSMC40	$67.4 \times 75.7\mu m^2$	30 seconds
LDO	Automatic	TSMC40	$53.3 \times 71.7\mu m^2$	20 seconds
FIA	Manual	TSMC28	$75.8 \times 92.1\mu m^2$	2 hours
SAR-ADC	Manual	TSMC65	$240.6 \times 192.7\mu m^2$	4-5 hours

TABLE III: COMPARISON ON BENCHMARKS WITH AUTOMATIC PLACEMENT.

Benchmark		Schematic	MAGICAL [17]	SAGERoute w/o Wire Sizing	SAGERoute
OTA	Gain (dB)	38.63	**38.44**	37.84	**38.49**
	UGB (MHz)	6.85	**5.10**	4.97	**5.34**
	CMRR (dB)	–	**55.7**	52.8	**54.3**
	PM (degree)	70.98	**70.43**	**78.13**	67.47
LDO	Gain (dB)	73.69	**73.06**	72.32	**73.60**
	Current (uA)	16.82	**16.18**	**16.17**	16.19
	PM (degree)	89.69	**89.61**	89.50	**89.64**
	VOD (mV)	539.6	**1937.0**	2773.0	**877.3**
	VOU (mV)	540.2	**1422.0**	2235.0	**626.7**

1. Data bolded in black denotes the best, and data bolded in gray denotes the second best.

IV. EXPERIMENTAL RESULTS

Our framework is implemented in C++ programming language. We adopts lpsolve as our LP solver in wire sizing. Major experiments are conducted on a Linux server with Intel Xeon Gold 6230 CPU @ 2.10GHz. We perform experiments on real-world analog designs including two primary circuits (i.e., an OTA and an LDO) and two advanced circuits (i.e., a floating inverted-based amplifier or FIA, and a SAR-ADC) that have been taped out. Table II summarizes the benchmark statistics. It can be seen that the benchmarks come from three widely-used technology nodes for analog design, i.e., 65nm, 40nm, and 28nm. The two advanced circuits are rather complicated and take hours for post-layout simulation. We support routing on both automatically generated placement and manually-drawn placement, and finish routing within 20 seconds on each benchmark, which is much faster than manual process.

We perform post-layout simulation with Cadence Spectre and Ultra APS to evaluate the quality of routing solutions. Calibre PEX is used to extract parasitic resistance, parasitic capacitance, and coupling capacitance (R+C+CC).

OTA and LDO are under technology node TSMC40, which is compatible with MAGICAL [17]. The comparison results are listed in Table III. SAGERoute indicates the final layout performance of the proposed routing framework, while SAGERoute w/o Wire Sizing represents the results without wire sizing. SAGERoute achieves better results in Gain and UGB, while MAGICAL does better in phase margin (PM) and CMRR in OTA. It can be seen that wire sizing plays an important role in LDO case which obtains 54% and 56 % reduction of VOD and VOU respectively, and better gain with close power consumption, compared with the result generated by MAGICAL.

TABLE IV: COMPARISON ON BENCHMARKS WITH MANUAL PLACEMENT.

Benchmark		Schematic	Manual	SAGERoute w/o Wire Sizing	SAGERoute
FIA	Gain (dB)	24.55	**23.97**	23.32	**23.57**
	Noise (nV)	61.03	54.83	**46.58**	**21.44**
SAR-ADC	Delay (ns)	20.43	**21.37**	**21.46**	21.47
	SINAD (dB)	65.59	**65.74**	65.46	**65.70**
	ENOB (bit)	10.60	**10.63**	10.58	**10.62**
	Pcore (uW)	206.6	**231.6**	**231.3**	231.9
	FoM (fJ/conv)	5.321	**5.862**	6.041	**5.896**

1. Data bolded in black denotes the best, and data bolded in gray denotes the second best.

FIA is a kind of high-performance amplifier that is strict with the noise level. Compared with manual layout, SAGERoute obtains 60% reduction of noise with slightly lower gain.

The analog part of a 12-bit SAR-ADC is the major routing object, while the digital part including CDAC (capacitor DAC) and control logic is excluded. We perform transient simulation with 1024 sampling points on the 12-bit SAR-ADC for accuracy. FoM represents power consumption per conversion, which is the smaller, the better. As Table IV depicts, SAGERoute obtains a decent performance which is very close to the manual layout. Figure 7 shows the final layout of SAR-ADC.

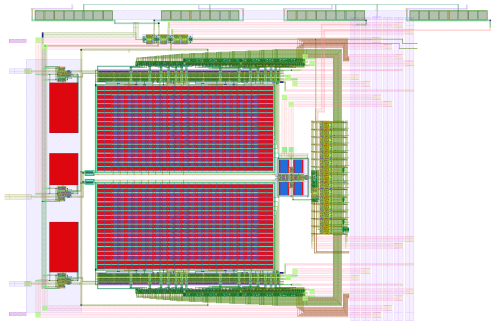

Fig. 7: SAR-ADC Final Layout

V. CONCLUSIONS

In this paper, we propose a synergistic routing framework considering geometric and electrical constraints. The framework is able to perform wire sizing and routing, subject to various constraints like EM, IR drop, layout sensitive area, symmetry, low bending, parallel run length spacing, EOL spacing, etc. Experimental results on real-world analog designs in 65nm, 40nm, and 28nm technology nodes demonstrate the effectiveness of the framework and its compatibility with manual taped-out designs.

ACKNOWLEDGEMENT

This work was supported in part by the National Science Foundation of China (Grant No. 62141404, 62125401), Zhejiang Provincial Key R&D program (Grant No. 2020C01052), and the 111 project (B18001).

REFERENCES

[1] J. Black, "Electromigration—A brief survey and some recent results," *IEEE TED*, vol. 16, no. 4, pp. 338–347, 1969.

[2] J. Lienig and G. Jerke, "Electromigration-aware physical design of integrated circuits," in *Proc. VLSI Design*, 2005, pp. 77–82.

[3] Z. Chai, Y. Zhao, Y. Lin, W. Liu, R. Wang, and R. Huang, "Circuitnet: An open-source dataset for machine learning applications in electronic design automation (eda)," *SCIENCE CHINA Information Sciences*, vol. 65, no. 12, pp. 227 401–, 2022.

[4] H. Chen, K. Zhu, M. Liu, X. Tang, N. Sun, and D. Z. Pan, "Toward silicon-proven detailed routing for analog and mixed-signal circuits," in *Proc. ICCAD*, 2020, pp. 1–8.

[5] U. Choudhury and A. Sangiovanni-Vincentelli, "Constraint-based channel routing for analog and mixed analog/digital circuits," *IEEE TCAD*, vol. 12, no. 4, pp. 497–510, 1993.

[6] E. Malavasi, E. Charbon, E. Felt, and A. Sangiovanni-Vincentelli, "Automation of ic layout with analog constraints," *IEEE TCAD*, vol. 15, no. 8, pp. 923–942, 1996.

[7] L. Xiao, E. F. Y. Young, X. He, and K. P. Pun, "Practical placement and routing techniques for analog circuit designs," in *Proc. ICCAD*, 2010, pp. 675–679.

[8] P.-C. Pan, H.-M. Chen, Y.-K. Cheng, J. Liu, and W.-Y. Hu, "Configurable analog routing methodology via technology and design constraint unification," in *Proc. ICCAD*, 2012, p. 620–626.

[9] M. M. Ozdal and R. F. Hentschke, "Maze routing algorithms with exact matching constraints for analog and mixed signal designs," in *Proc. ICCAD*, 2012, pp. 130–136.

[10] R. Martins, N. Lourenço, and N. Horta, "Routing analog ics using a multi-objective multi-constraint evolutionary approach," *Analog Integrated Circuits and Signal Processing*, vol. 78, no. 1, pp. 123–135, Jan 2014.

[11] H.-C. Ou, H.-C. C. Chien, and Y.-W. Chang, "Nonuniform multilevel analog routing with matching constraints," *IEEE TCAD*, vol. 33, no. 12, pp. 1942–1954, 2014.

[12] T. Adler and E. Barke, "Single step current driven routing of multiterminal signal nets for analog applications," in *Proc. DATE*, 2000, pp. 446–450.

[13] J. Lienig and G. Jerke, "Current-driven wire planning for electromigration avoidance in analog circuits," in *Proc. ASPDAC*, 2003, pp. 783–788.

[14] R. Martins, N. Lourenço, A. Canelas, and N. Horta, "Electromigration-aware and ir-drop avoidance routing in analog multiport terminal structures," in *Proc. DATE*, 2014, pp. 1–6.

[15] M. Torabi and L. Zhang, "Electromigration- and Parasitic-Aware ILP-Based Analog Router," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 10, pp. 1854–1867, 2018.

[16] B. Xu, K. Zhu, M. Liu, Y. Lin, S. Li, X. Tang, N. Sun, and D. Z. Pan, "Magical: Toward fully automated analog ic layout leveraging human and machine intelligence: Invited paper," in *Proc. ICCAD*, 2019, pp. 1–8.

[17] H. Chen, M. Liu, B. Xu, K. Zhu, X. Tang, S. Li, Y. Lin, N. Sun, and D. Z. Pan, "MAGICAL: An Open- Source Fully Automated Analog IC Layout System from Netlist to GDSII," *IEEE Design & Test*, vol. 38, no. 2, pp. 19–26, 2021.

[18] T. Dhar, K. Kunal, Y. Li, Y. Lin, M. Madhusudan, J. Poojary, A. K. Sharma, S. M. Burns, R. Harjani, J. Hu, P. Mukherjee, S. Yaldiz, and S. S. Sapatnekar, "The ALIGN open-source analog layout generator: V1.0 and beyond," in *Proc. ICCAD*, 2020, pp. 1–2.

[19] T. Dhar, K. Kunal, Y. Li, M. Madhusudan, J. Poojary, A. K. Sharma, W. Xu, S. M. Burns, R. Harjani, J. Hu, D. A. Kirkpatrick, P. Mukherjee, S. Yaldiz, and S. S. Sapatnekar, "Align: A system for automating analog layout," *IEEE Design & Test*, vol. 38, no. 2, pp. 8–18, 2021.

[20] K. Kunal, M. Madhusudan, A. K. Sharma, W. Xu, S. M. Burns, R. Harjani, J. Hu, D. A. Kirkpatrick, and S. S. Sapatnekar, "Invited: Align – open-source analog layout automation from the ground up," in *Proc. DAC*, 2019, pp. 1–4.

[21] R. Martins, N. Lourenço, and N. Horta, "Multi-objective multi-constraint routing of analog ics using a modified nsga-ii approach," in *Proc. SMACD*, 2012, pp. 65–68.

[22] N. Lourenco, M. Vianello, J. Guilherme, and N. Horta, "LAYGEN - Automatic Layout Generation of Analog ICs from Hierarchical Template Descriptions," in *IEEE Ph.D. Research in Microelectronics and Electronics*, 2006, pp. 213–216.

[23] R. Martins, N. Lourenco, and N. Horta, "LAYGEN II—Automatic Layout Generation of Analog Integrated Circuits," *IEEE TCAD*, vol. 32, no. 11, pp. 1641–1654, 2013.

[24] Q. Ma, L. Xiao, Y. Tam, and E. F. Y. Young, "Simultaneous handling of symmetry, common centroid, and general placement constraints," *IEEE TCAD*, vol. 30, no. 1, pp. 85–95, 2011.

[25] V. Borisov, K. Langner, J. Scheible, and B. Prautsch, "A novel approach for automatic common-centroid pattern generation," in *Proc. SMACD*, 2017, pp. 1–4.

[26] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-completeness*, ser. Mathematical Sciences Series. W. H. Freeman, 1979.

[27] D. A. Huffman, "A method for the construction of minimum-redundancy codes," *Proceedings of the IRE*, vol. 40, no. 9, pp. 1098–1101, 1952.