A Novel Analog Physical Synthesis Methodology Integrating Existent Design Expertise

Po-Hsun Wu, *Student Member, IEEE*, Mark Po-Hung Lin, *Senior Member, IEEE*, Tung-Chieh Chen, Ching-Feng Yeh, Xin Li, *Senior Member, IEEE*, and Tsung-Yi Ho, *Senior Member, IEEE*

***Abstract*—Analog layout design has been a manual, time-consuming, and error-prone task for decades. To speed up layout design time for a new design, analog layout designers prefer referring to legacy designs and layouts rather than starting from scratch, or thoroughly applying placement and routing tools because legacy layouts contain pretty much design expertise. Motivated by such layout design process, this paper presents the first knowledge-based physical synthesis methodology to generate new layouts by integrating existent design expertise. The proposed approach can automatically analyze legacy design data including circuits, layouts, and constraints, extract matched sub-circuits between new and legacy designs, and generate multiple layouts for the new design by utilizing the quality-approved legacy layouts as much as possible. Experimental results show that the proposed methodology can achieve high layout reusage rate, and hence the designers' layout preference can be successfully reserved.**

***Index Terms*—Analog layout, design pattern, knowledge mining, migration, physical design, placement, routing.**

I. Introduction

MODERN system-on-chip (SoC) design usually contains both digital and analog circuits. The design of digital circuits has been extensively assisted by many design automation tools, while that of analog counterparts is still a manual, time-consuming, and error-prone task. As the time-to-market requirement of modern SoC is becoming increasingly stringent, and nanometer design rules are too complicated to be handled manually [1], it is essential to introduce new analog design methodologies to speed up the design cycle as well as to reduce the design effort.

Manuscript received June 8, 2014; revised September 15, 2014; accepted October 29, 2014. Date of publication December 10, 2014; date of current version February 16, 2015. This work was supported by the Ministry of Science and Technology of Taiwan under Grant NSC 102-2220-E-194-006, Grant NSC 102-2221-E-194-065-MY2, and Grant NSC 103-2917-I-006-086. This paper was recommended by Associate Editor N. Wong.

P.-H. Wu is with the Department of Computer Science and Information Engineering, National Cheng Kung University, Tainan 701, Taiwan.

M. P.-H. Lin and C.-F. Yeh are with the Department of Electrical Engineering and Advanced Institute of Manufacturing with High-tech Innovations, National Chung Cheng University, Chiayi 621, Taiwan (e-mail: marklin@ccu.edu.tw).

T.-C. Chen is with Synopsys, Inc., Hsinchu 300, Taiwan.

X. Li is with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213 USA.

T.-Y. Ho is with the Department of Computer Science, National Chiao Tung University, Hsinchu 300, Taiwan.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCAD.2014.2379630

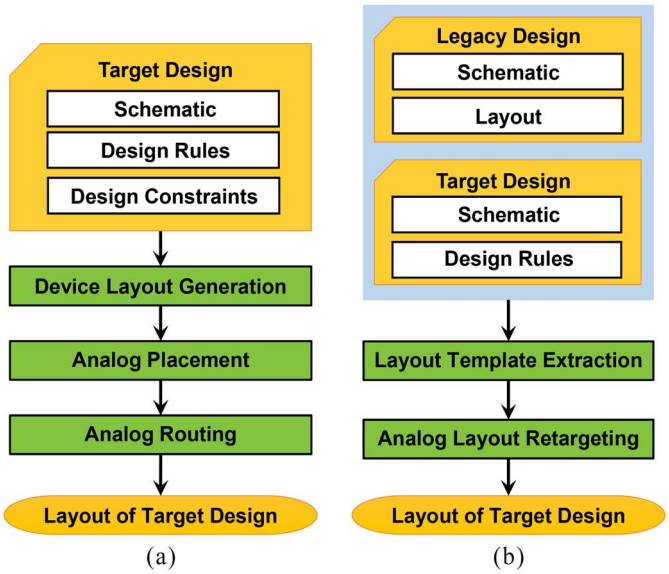

Fig. 1. Two conventional analog physical synthesis flows. Analog layout (a) generation flow and (b) migration flow.

To automatically generate the layout of a newly designed analog circuit (i.e., the target design), there have been two kinds of analog physical synthesis flows in the literature, including: 1) analog layout generation flow and 2) analog layout migration flow, as demonstrated in Fig. 1. The common inputs for both flows include the schematic and layout design rules of the target design. The analog layout generation flow additionally requires some design constraints of the target design, while the analog layout migration flow additionally requires the layout of a legacy design whose schematic topology is the same as the target design. It should be noted that the device size and layout design rules of the legacy design can be different from those of the target design based on the layout migration flow.

According to Fig. 1(a), the analog layout generation flow first constructs the layout of each device, or building block. In order to optimally place the building blocks, recent works have introduced various placement constraints, and manipulated these constraints with different topological representations and algorithms. The placement constraints include various analog layout constraints, such as boundary, common-centroid, minimum/maximum-distance, current/signal path, preplacement, proximity/range, regularity,

TABLE I
COMPARISONS OF THE PLACEMENT CONSTRAINTS HANDLED BY THE RECENT WORKS AND THE CORRESPONDING TOPOLOGICAL REPRESENTATIONS THEY APPLIED

Constraint \ Representation	B*-tree	CBL	O-tree	SP	Slicing Tree	TCG
Boundary	[2], [3]			[4]	[5]	
Common-centroid	[6]		[7]			
Min/Max-distance, proximity, range	[6], [3], [8]			[4]	[9], [10], [11]	
Signal/Current path				[12]	[13]	
Pre-placement	[3]			[4]	[9], [14]	
Regularity	[15]			[16]		
Symmetry	[17], [6], [2], [3], [8], [15], [18], [25], [26], [27], [28], [29]	[19]	[7], [20]	[4], [16], [21], [30], [31], [32]	[9], [10], [13], [22], [23]	[24]
Symmetry-island	[17], [2], [8], [18]				[13]	
Thermal	[18]	[19]				

symmetry, symmetry-island,[1] and thermal gradient, which are summarized in Table I. According to Table I, most of the recent works resort to the topological representations, including B*-trees, corner block lists, O-trees, sequence pairs, slicing trees, and transitive closure graphs, due to their flexibility and effectiveness in representing the relative locations among devices and exploring solutions with the consideration of various placement constraints. During placement optimization, most of the recent works applied the simulated annealing algorithm [33], while [6] was based on a nonstochastic approach. Although these layout constraints were designed to minimize the impact from layout-induced parasitic, the resulting layouts are sometimes unacceptable because manual layouts contain much more experts' knowledge, and designers may have their own layout preferences which cannot simply be expressed by those layout constraints.

In order to preserve all experts' knowledge, some other recent works [34]–[37] proposed to generate analog layouts based on layout migration/retargeting, as shown in Fig. 1(b). These works automatically extract a symbolic structural template from a legacy layout to preserve the layout topology, design rules, and symmetry/matching constraints. According to the extracted template, the new layout can be generated by layout compaction techniques, which solve a set of constraints with linear programming or graph-based algorithms, such that the total layout area is minimized. Although the idea of layout migration/retargeting can fully preserve experts' knowledge from design to design, such approach assumes that the legacy design and the new design must have the same schematic/netlist topology, or the same layout template, while the device size and the process technology can be different. This flow does not work well if the assumptions do not hold.

To overcome the drawbacks of both analog layout generation and analog layout migration flows, in this paper, we propose a novel knowledge-based physical synthesis methodology to produce new analog layouts by fully or partially extract experts' knowledge from the quality-approved legacy layouts in the design repository and reutilizing them as much as possible. Consequently, the designers' layout preference can be successfully reserved. The contributions of this paper are summarized in the following.

1) We present the first analog physical synthesis methodology which can generate a target layout by acquiring design expertise from multiple legacy design data. Moreover, our proposed flow can generate multiple reference layouts based on different design objectives, e.g., area maximization and fixed-outline constraint.
2) To generate the layout of a target design, we propose novel algorithms to analyze legacy design data, extract common sub-circuits between a legacy design and the target design, and reuse quality-approved placement and routing topologies from different legacy designs. Different from the traditional sub-graph identification algorithms, our proposed design pattern matching algorithms can easily consider different design constraints when solving the sub-graph identification problem.
3) As the extracted common sub-circuits, or design patterns, are not mutually exclusive, we determine the best design patterns among all the extracted ones by formulating the pattern selection problem as the maximum-weight-clique problem while maximizing the reusage rate of legacy layouts. To further reduce the complexity of maximum-weight-clique problem, several properties have been proposed to prune redundant nodes without affecting the optimality of selection results.
4) Based on the presented knowledge-based physical synthesis methodology, analog designers can maintain their own design repositories, and effectively and efficiently generate new layouts according to their own expertise. Consequently, both design cycle and design effort are reduced.

The rest of this paper is organized as follows. Section II proposes the knowledge-based physical synthesis flow. Section III introduces a graph-based approach to store the design information of legacy design. Section IV presents design matching and pattern extracting/selection algorithms to maximize the layout reusage rate. Section V describes a knowledge-based layout generation method by integrating the extracted legacy layouts.

[1]A symmetry-island constraint is more restrictive than a symmetry constraint, which makes the symmetric modules of a symmetry group form a connected placement [17].

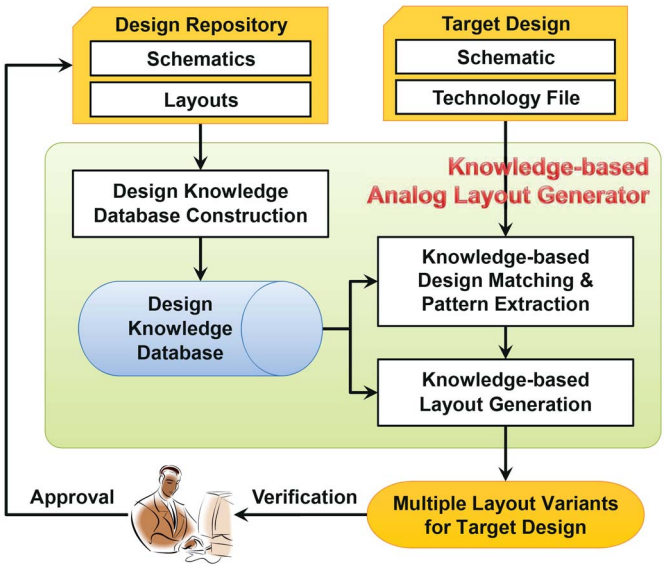

Fig. 2. Proposed knowledge-based physical synthesis methodology and flow integrating existent design expertise.

Section VI shows the experimental results, and Section VII concludes this paper.

II. Proposed Knowledge-Based Physical Synthesis Design Flow

The flow of our proposed knowledge-based analog physical synthesis methodology is demonstrated in Fig. 2, which consists of three major steps: 1) design knowledge database construction; 2) knowledge-based design matching and pattern extraction; and 3) knowledge-based layout generation. Inputting a design repository containing legacy design schematics/netlists and the corresponding legacy layouts with design expertise, the design knowledge database construction analyzes the design data of each circuit and layout, and stores the analyzed data in a design knowledge database. In order to utilize the legacy layouts in the design repository as much as possible when generating the layout of a newly designed circuit, or the target design, the knowledge-based design matching and pattern extraction matches the target design with the legacy design in the design knowledge database, extracts all common sub-circuits, or design patterns, and selects the most suitable ones among the extracted patterns. Finally, the knowledge-based layout generation further extracts the corresponding layout of each design pattern in the database, migrates the extracted layouts to the new technology if it is necessary, and generates target layouts with multiple variants by assembling the migrated layouts of all design patterns. The design knowledge database can be continuously expanded when more and more analog layouts are verified and approved by experienced layout experts, as seen in Fig. 2. It should be noted that all the design data in the design repository must be committed by designers. If the initial design repository is empty, the proposed flow could be replaced by any other existing analog layout generation approach, including manual design. For the design migration application, there should be at least one legacy design in the design repository.

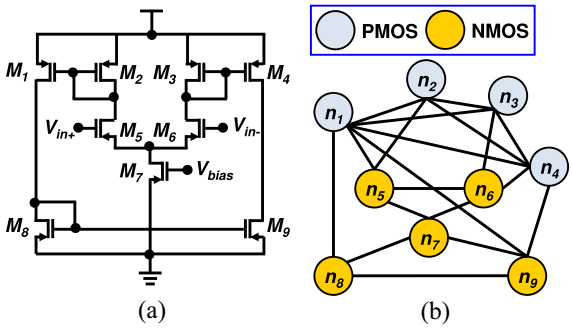

Fig. 3. CMOS cascade operational transconductance amplifier (OTA). (a) Legacy schematic. (b) Corresponding connection graph representation.

TABLE II
TCIs Between the Terminals of Two MOS Transistors, M_i and M_j, Where D, G, and S Denote Drain, Gate, and Source Terminal, Respectively

$tci_{i\to j}^{p\to q}$		Terminals of M_i		
		D	G	S
Terminals of M_j	D	1	8	64
	G	2	16	128
	S	4	32	256

III. Design Knowledge Database Construction

Given a set of legacy schematics, S_L, and legacy layouts, L_L, we first construct a design knowledge database based on the connection graph representation, as described in Section III-A. Since a basic connection graph can only represent logical information corresponding to a schematic, we further annotate some important physical information from the corresponding legacy layout into the connection graph, as illustrated in Section III-B.

A. Connection Graph Generation

For each legacy schematic, $s_L \in S_L$, the corresponding connection graph, G_{s_L}, can be derived by converting the devices and nets in s_L to respective nodes and edges. One of the device types, such as transistors, capacitors, resistors, or inductors, is tagged on each node. If there is a connection between two devices in s_L, there is also an edge between the corresponding nodes in G_{s_L}. Fig. 3 shows the transformation from a circuit to the corresponding connection graph representation. Since a device in s_L may have two or more terminals, to identify the terminal connections (TCs) of an edge in G_{s_L}, we introduce a universal coding scheme.

Definition 1: A TC, $\text{tc}_{i\to j}^{p\to q}$, is defined as the connection between the pth terminal of device d_i to the qth terminal of device d_j, and the corresponding TC index (TCI) is encoded by $\text{tci}_{i\to j}^{p\to q}$. TC denotes the set of TCs between two devices, d_i and d_j and TCI represents the set of respective TCIs of TC.

For two devices, d_i and d_j, which have N_p and N_q terminals, respectively, the interconnection code from d_i to d_j, $\Gamma_{i\to j}$, can be calculated by (1), where $\alpha_{i\to j}^{p\to q} \in \{0, 1\}$. If there is a TC between pth of d_i and qth of d_j, $\alpha_{i\to j}^{p\to q} = 1$; otherwise, $\alpha_{i\to j}^{p\to q} = 0$. The value of $\text{tci}_{i\to j}^{p\to q}$ is defined in (2). Table II shows the TCIs between the terminals of two MOS transistors,

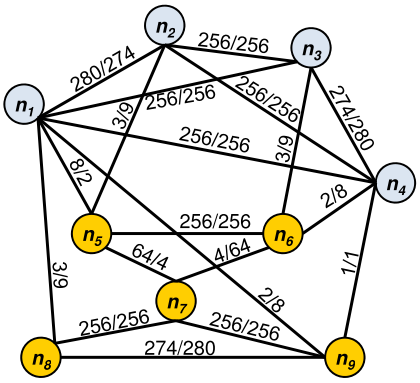

Fig. 4. Connection graph representation for the circuit in Fig. 3(a) after attaching the interconnection codes to all edges in Fig. 3(b).

M_i and M_j, where D, G, and S denote drain, gate, and source terminal, respectively

$$\Gamma_{i\to j} = \sum_{p=1}^{N_p}\sum_{q=1}^{N_q}\left(\alpha_{i\to j}^{p\to q} \times \mathrm{tci}_{i\to j}^{p\to q}\right) \quad (1)$$

$$\mathrm{tci}_{i\to j}^{p\to q} = 2^{(q+N_q(p-1)-1)}. \quad (2)$$

Based on Table II, (1), and Fig. 3(a), we can easily calculate the interconnection codes of each edge contained by the connection graph of Fig. 3(b). For example, Fig. 3(a), there are a set of terminal connections (TCs) from M_1 to M_2, $\mathrm{TC}_{1\to 2} = \{\mathrm{tc}_{1\to 2}^{G\to D}, \mathrm{tc}_{1\to 2}^{G\to G}, \mathrm{tc}_{1\to 2}^{S\to S}\}$, where $\mathrm{tc}_{1\to 2}^{G\to D}$, $\mathrm{tc}_{1\to 2}^{G\to G}$, and $\mathrm{tc}_{1\to 2}^{S\to S}$ denote the TC from the gate terminal of M_1 to the drain terminal of M_2, the terminal connection from the gate terminal of M_1 to the gate terminal of M_2, and the terminal connection from the source terminal of M_1 to the source terminal of M_2, respectively. Based on Table II, $\mathrm{tci}_{1\to 2}^{G\to D} = 8$, $\mathrm{tci}_{1\to 2}^{G\to G} = 16$, and $\mathrm{tci}_{1\to 2}^{S\to S} = 256$, respectively, so $\Gamma_{1\to 2} = 280$. Similarly, we can obtain all the other interconnection codes for different kind of TCs between any two devices. Fig. 4 shows the connection graph representation for the circuit in Fig. 3(a) after attaching the interconnection codes to all edges in Fig. 3(b). Based on the universal coding scheme, each edge in the connection graph is attached with a pair of universal codes, $\Gamma_{i\to j}/\Gamma_{j\to i}$, which can uniquely specifies the interconnection relationship between the devices, d_i and d_j. It should be noted that one edge in the connection graph represents a set of two-pin nets between two devices, d_i and d_j, connecting different terminals, e.g., drain, gate, and source of two transistors.

B. Constraint Annotation

As symmetry and proximity constraints are very common in analog layout design, it is required to annotate the constraints in the corresponding connection graph, G_{s_L}. Such constraint annotation can help to reduce the search space during design matching and pattern extraction, as described in Section IV. Fig. 5 shows the resulting connection graph after attaching a universal code on each edge, and annotating both symmetry and proximity constraints in the original connection graph in Fig. 3(b). We finally store the annotated connection graphs of all legacy designs in the design knowledge database.

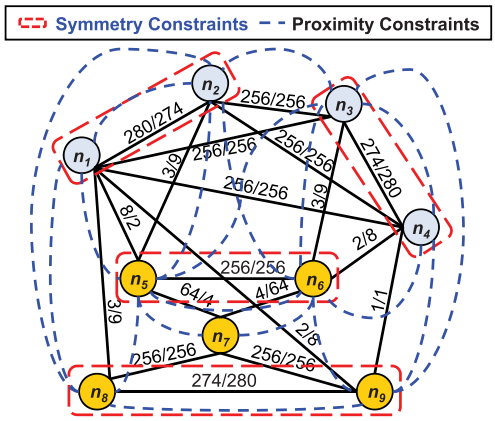

Fig. 5. Resulting connection graph after attaching the universal codes on each edge, and annotating both symmetry and proximity constraints according to the legacy schematic and layout of the circuit in Fig. 3.

IV. Knowledge-Based Design Matching and Pattern Extraction

Our problem formulation is to utilize a set of legacy schematics, S_L, and a set of legacy layouts, L_L, and reuse the placement and routing topologies in L_L as much as possible to generate the layout, l_T, of a target design, s_T such that the designers' layout preference can be successfully reserved. To generate l_T based on S_L and L_L which had been stored in the design knowledge database, we first propose a design pattern matching algorithm, as described in Section IV-A, which extracts common sub-circuits, or design patterns, between s_T and $s_L \in S_L$. As the number of design patterns can be very large, we further introduce some rules, as illustrated Section IV-B, to effective eliminate redundant design patterns. Finally, we determine which design patterns should be chosen to generate l_T for achieving the best layout reusage rate, as demonstrated in Section IV-C.

A. Design Pattern Matching

As s_L and s_T may not be fully identical, but may have common sub-circuits. Only partial of the layout in l_L corresponding to the common sub-circuits can be reused to generate l_T. Such circuit matching and layout pattern extraction problem can be formulated as a sub-circuit identification problem, as defined in Problem 1.

Problem 1 (Sub-Circuit Identification Problem): Given two circuit netlists, s_L and s_T, identify all common sub-circuits, which have the same device types, device constraints, and interconnections, in both s_L and s_T.

It should be noted that the sub-circuit identification problem is different from the sub-graph isomorphism problem. In circuit comparison, all compared circuits are represented as a graph, and then the graph-based algorithms are applied for graph comparison. Although several graph comparison algorithms have been proposed, those algorithms only verify either whether two graph are identical (i.e., graph isomorphism) [38]–[41], or whether a given sub-graph is contained in another graph (i.e., sub-graph isomorphism) [42]–[44]. Such kinds of algorithms cannot be directly adopted to extract

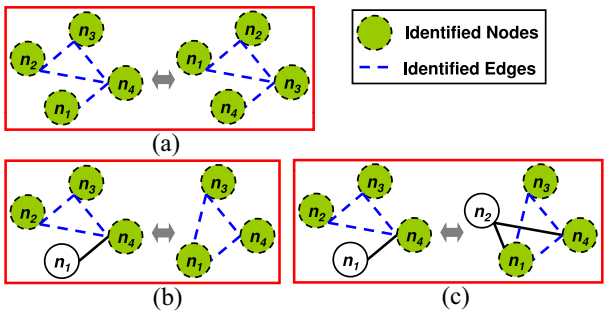

Fig. 6. Difference among graph comparison problems. Example of (a) graph isomorphism, (b) sub-graph isomorphism, and (c) sub-graph identification.

all common sub-graphs contained in two different graphs (i.e., sub-graph identification).

Recently, Ferent and Doboli [45] and Ferent *et al.* [46] developed a systematical technique to present the similarity and dissimilarity of two analog circuits based on the electrical model. Although their method can be modified to identify the sub-graphs, the electrical model becomes very inefficient when a lot of devices are compared because a great number of design variables need to be generated.

Although there are existing sub-graph identification algorithms [47]–[49], they just check if two nodes have same node degree without verifying if two nodes have the same device type, device constraints, and interconnections. As a result, the derived target design of traditional sub-graph identification algorithms may derive different functionalities of two sub-circuits. Moreover, we perform node pruning technique to keep a polynomial time complexity, as discussed in Section IV-A2, compared with traditional sub-graph identification algorithms which usually have an exponential time complexity in worst case. With the consideration of circuit functionality and efficiency, in this paper, we propose a new design matching and pattern extraction algorithm to efficiently identify all sub-graphs between the target design and the legacy design. Fig. 6 demonstrates the difference among graph isomorphism, sub-graph isomorphism, and sub-graph identification.

In Fig. 6(a), the graph isomorphism problem is to verify whether two given graphs are identical by finding a one-to-one relation between two sets of nodes. The identified sub-graph is corresponding to the original graph, where all nodes of the identified sub-graph is represented by the dashed nodes in Fig. 6. On the other hand, the sub-graph isomorphism problem is to verify whether a given graph is a sub-graph of another graph. In Fig. 6(b), the identified sub-graph cannot be found by finding a one-to-one relation between two sets of nodes for two given graphs, so the sub-graph isomorphism problem cannot be solved by using the graph isomorphism algorithms. In our problem formulation, we propose a sub-graph identification algorithm to identify the sub-graphs contained in both given graphs as demonstrated in Fig. 6(c). Obviously, the graph isomorphism algorithms cannot be used to identify the sub-graph because the identified is not corresponding to the original graph. Besides, the sub-graph isomorphism algorithms cannot be used to identify the sub-graph because both given graphs in Fig. 6(c) are not a sub-graph of another graph. As a result, we propose a new sub-graph identification algorithm to identify all sub-graphs of two given graphs.

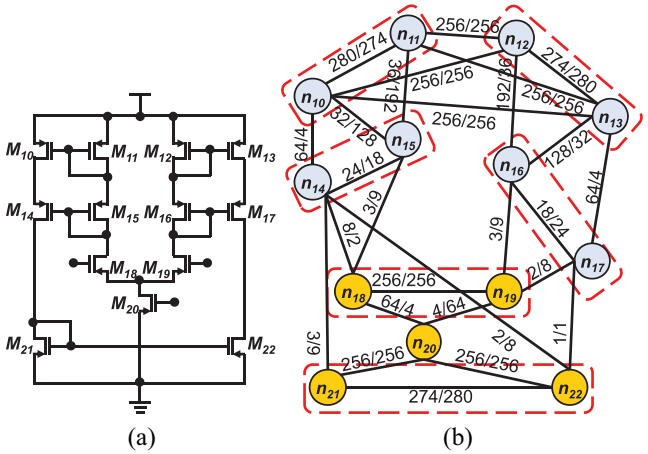

Fig. 7. Example target design. (a) Schematic of the target design. (b) Corresponding connection graph with attached universal codes and annotated symmetry constraints.

Algorithm 1 Design Pattern Matching

Input: G_{s_L} and G_{s_T};
Output: a set of design patterns, $\mathbb{P}$;
1: $\mathbb{S} \leftarrow \emptyset$; // $\mathbb{S}$ is a list of possible initial nodes.
2: Initial Node Generation();
3: **for all** $\{n_i^L \rightleftharpoons n_j^T\} \in \mathbb{S}$ **do**
4: //Explore all connected nodes for each $\{n_i^L \rightleftharpoons n_j^T\}$;
5: $\mathbb{C}_{n_i^L n_j^T} \leftarrow \emptyset$; // $\mathbb{C}_{n_i^L n_j^T}$ contains a set of connected nodes of $\{n_i^L \rightleftharpoons n_j^T\}$
6: Connected Node Exploration();
7: Connected Node Pruning();
8: **end for**
9: $\mathbb{P} \leftarrow \emptyset$;
10: **for all** $\{n_i^L \rightleftharpoons n_j^T\} \in \mathbb{S}$ **do**
11: $\mathbb{D} \leftarrow \emptyset \cup \{n_i^L \rightleftharpoons n_j^T\}$; // $\mathbb{D}$ contains a set of equivalent nodes forming current design pattern
12: $\mathbb{C} \leftarrow \emptyset \cup \mathbb{C}_{\{n_i^L \rightleftharpoons n_j^T\}}$; // $\mathbb{C}$ contains a set of connected nodes of $\mathbb{D}$
13: **while** $(\exists \{n_x^L \rightleftharpoons n_y^T\} \in \mathbb{C}) \cap (n_x^L \notin \mathbb{D}) \cap (n_y^T \notin \mathbb{D})$ **do**
14: Find $\{n_x^L \rightleftharpoons n_y^T\} \in \mathbb{C}$ such that $NR_{\{n_x^L \rightleftharpoons n_y^T\}}$ is maximum;
15: $\mathbb{D} \leftarrow \mathbb{D} \cup \{n_x^L \rightleftharpoons n_y^T\}$;
16: Update $\mathbb{C}$ with $\mathbb{C}_{\{n_x^L \rightleftharpoons n_y^T\}}$; //Replace $\{n_x^L \rightleftharpoons n_k^T\}$ $(\{n_l^L \rightleftharpoons n_y^T\}) \in \mathbb{C}$ with $\{n_x^L \rightleftharpoons n_y^T\} \in \mathbb{C}_{\{n_x^L \rightleftharpoons n_y^T\}}$ if $NR_{\{n_x^L \rightleftharpoons n_k^T\}}$ $(NR_{\{n_l^L \rightleftharpoons n_y^T\}}) < NR_{\{n_x^L \rightleftharpoons n_y^T\}}$
17: **end while**
18: $\mathbb{P} \leftarrow \mathbb{P} \cup \{\mathbb{D}\}$; // add one design pattern
19: **end for**

To identify common sub-circuits, or to extract design patterns, between a legacy and a target designs, we first construct the connection graph, G_{s_T}, of s_T, which is similar to the construction of G_{s_L}, as described in Section III. The symmetry constraints of s_T are also annotated in G_{s_T}, which can be given by designers. Fig. 7 shows the schematic of a target design and the corresponding connection graph with attached universal codes and annotated symmetry constraints.

Algorithm 2 Initial Node Generation

Input: G_{s_L} and G_{s_T};
Output: a set of initial nodes, $\mathbb{S}$;
1: **for all** $n_i^L \in G_{s_L}$ **do**
2: **for all** $(n_j^T \in G_{s_T}) \cap (dev\_type(n_i^L) = dev\_type(n_j^T)) \cap (dev\_contraint(n_i^L) = dev\_contraint(n_j^T))$ **do**
3: // The devices represented by n_i^L and n_j^T are equivalent.
4: $\mathbb{S} \leftarrow \mathbb{S} \cup \{n_i^L \rightleftharpoons n_j^T\}$; // add starting node
5: **end for**
6: **end for**

1) Design Pattern Matching Algorithm: After obtaining G_{s_T}, we then introduce Algorithm 1 to match all reusable design patterns from a legacy design represented by G_{s_L} to a target design represented by G_{s_T}. Before introducing our proposed algorithms, we define the equivalence condition of two nodes and net reusage as follows.

Definition 2: If node $n_i^L \in G_{s_L}$ has same device type and device constraints with node $n_j^T \in G_{s_T}$, then n_i^L and n_j^T are called equivalent and represented by $\{n_i^L \rightleftharpoons n_j^T\}$, where n_i^L (n_j^T) is the equivalent node of n_j^T (n_i^L).

Definition 3: For one node n_x^L connected to n_i^L and another node n_y^T connected to n_j^T, where n_x^L, $n_i^L \in G_{s_L}$ and n_y^T, $n_j^T \in G_{s_T}$, the net reusage of $\{n_x^L \rightleftharpoons n_y^T\}$ is calculated based on the number of same type of TCs with n_i^L and n_j^T as shown in (3) and is represented by $NR_{\{n_x^L \rightleftharpoons n_y^T\}}$, where $\alpha_{i\to x}^{p\to q}$, $\alpha_{j\to y}^{p\to q} \in \{0, 1\}$ and N_p (N_q) denotes the number of terminals of devices d_i and d_j (devices d_x and d_y). If there is a TC between pth of node n_x (n_y) and qth of node n_i (n_j), $\alpha_{i\to x}^{p\to q}$ ($\alpha_{j\to y}^{p\to q}$) $= 1$; otherwise, $\alpha_{i\to x}^{p\to q}$ ($\alpha_{j\to y}^{p\to q}$) $= 0$

$$NR_{\left\{n_x^L \rightleftharpoons n_y^T\right\}} = \sum_{p=1}^{N_p}\sum_{q=1}^{N_q}\left(\alpha_{i\to x}^{p\to q} \times \alpha_{j\to y}^{p\to q}\right). \tag{3}$$

In Algorithm 1, before finding different design patterns, the algorithm first traverses through all nodes in G_{s_L} and searches for all their equivalent nodes in G_{s_T} to generate all possible initial nodes, $\{n_i^L \rightleftharpoons n_j^T\}$, as demonstrated in line 2. During generating all possible initial nodes in Algorithm 2, each pair of nodes are checked if they have equivalent device types and device constraints. If two nodes have equivalent device type and device constraints, then they are stored in $\mathbb{S}$ for further reference. Within lines 3–8 of Algorithm 1, all connected nodes, $\{n_x^L \rightleftharpoons n_y^T\}$, of each initial node, $\{n_i^L \rightleftharpoons n_j^T\}$, are first explored and their net reusage are also calculated as shown in Algorithm 3, where the set of derived connected nodes are stored in $\mathbb{C}_{n_i^L n_j^T}$ for further reference. During exploring the connected nodes, each connected node is checked if they have equivalent device type, device constraints, and device connections. In Algorithm 4, to further reduce the complexity of Algorithm 1, the set of connected nodes are sorted based on the nonincreasing order of net reusage, then the connected nodes with less net reusage are removed from $\mathbb{C}_{n_i^L n_j^T}$. It should be noted that if two devices are annotated with a symmetry constraint, the net reusage of the symmetric counterparts is assigned to infinite to guarantee they would be chosen simultaneously.

Algorithm 3 Connected Node Exploration

Input: $\{n_i^L \rightleftharpoons n_j^T\}$, and their connected nodes;
Output: a set of connected nodes, $\{n_x^L \rightleftharpoons n_y^T\} \in \mathbb{C}_{n_i^L n_j^T}$;
1: **for all** (n_x^L connected to n_i^L) **do**
2: **for all** (n_y^T connected to n_j^T) $\cap$ $(dev\_type(n_x^L) = dev\_type(n_y^T)) \cap$ $(dev\_constraint(n_x^L) = dev\_constraint(n_y^T)) \cap$ $(dev\_connection(n_x^L) = dev\_connection(n_y^T))$ **do**
3: // The devices represented by n_x^L and n_y^T are equivalent.
4: Calculate $NR_{\{n_x^L \rightleftharpoons n_y^T\}}$;
5: $\mathbb{C}_{n_i^L n_j^T} \leftarrow \mathbb{C}_{n_i^L n_j^T} \cup \{n_x^L \rightleftharpoons n_y^T\}$; // add connected node
6: **end for**
7: **end for**

Algorithm 4 Connected Node Pruning

Input: a set of connected nodes, $\{n_x^L \rightleftharpoons n_y^T\} \in \mathbb{C}_{n_i^L n_j^T}$;
Output: a reduced set of connected nodes, $\mathbb{C}_{n_i^L n_j^T}$;
1: Sort all connected node, $\{n_x^L \rightleftharpoons n_y^T\} \in \mathbb{C}_{n_i^L n_j^T}$, based on the nondecreasing order of net reusage;
2: Remove $\{n_x^L \rightleftharpoons n_y^T\}$ from $\mathbb{C}_{n_i^L n_j^T}$ if $\exists$ $\{n_x^L \rightleftharpoons n_k^T\}$ ($\{n_l^L \rightleftharpoons n_y^T\}$) $\in \mathbb{C}_{n_i^L n_j^T}$ and $NR_{\{n_x^L \rightleftharpoons n_k^T\}}(NR_{\{n_l^L \rightleftharpoons n_y^T\}}) \geq NR_{\{n_x^L \rightleftharpoons n_y^T\}}$;

The main purpose of Algorithms 3 and 4 are to reduce the time of searching connected nodes in the following design pattern matching step. For example, given a pair of initial nodes, $\{n_i^L \rightleftharpoons n_j^T\}$, they both have two connected nodes $\{n_{x_1}^L, n_{x_2}^L\}$ and $\{n_{y_1}^T, n_{y_2}^T\}$, respectively. After Algorithm 3, $\mathbb{C}_{n_i^L n_j^T}$ contains $\{\{n_{x_1}^L \rightleftharpoons n_{y_1}^T\}, \{n_{x_1}^L \rightleftharpoons n_{y_2}^T\}, \{n_{x_2}^L \rightleftharpoons n_{y_1}^T\}, \{n_{x_2}^L \rightleftharpoons n_{y_2}^T\}\}$, where $NR_{\{n_{x_1}^L \rightleftharpoons n_{y_1}^T\}} = 1$, $NR_{\{n_{x_1}^L \rightleftharpoons n_{y_2}^T\}} = 2$, $NR_{\{n_{x_2}^L \rightleftharpoons n_{y_1}^T\}} = 2$, and $NR_{\{n_{x_2}^L \rightleftharpoons n_{y_2}^T\}} = 1$. By sorting all equivalent nodes based on the nondecreasing order of net reusage and removing those with less net reusage from $\mathbb{C}_{n_i^L n_j^T}$ as shown in Algorithm 4, $\mathbb{C}_{n_i^L n_j^T}$ finally contains $\{\{n_{x_1}^L \rightleftharpoons n_{y_2}^T\}, \{n_{x_2}^L \rightleftharpoons n_{y_1}^T\}\}$.

From lines 9–19, one initial node, $\{n_i^L \rightleftharpoons n_j^T\}$, is selected to start including more equivalent nodes and expanding a design pattern. In each iteration of expanding current design pattern ($\mathbb{D}$) it finds and selects the connected nodes, $\{n_x^L \rightleftharpoons n_y^T\}$, with maximum net reusage, where $\mathbb{D}$ is a set used to store all derived equivalent nodes, $\{n_x^L \rightleftharpoons n_y^T\}$ and is expanded when more equivalent nodes are included. To verify whether the interconnection of a device represented by n_x^L in G_{s_L} is equivalent to that of a device represented by n_y^T in G_{s_T}, we shall examine the equivalence condition of the edges connecting among n_x^L and $\forall n_i^L \in \mathbb{D}$ and the edges connecting among n_y^T and $\forall n_j^T \in \mathbb{D}$, which will be discussed later in Section IV-A3. Moreover, $\mathbb{C}$ is a set to store all connected nodes with maximum net reusage for different equivalent nodes in $\mathbb{D}$ which is updated by removing the connected nodes with less net reusage after new equivalent node is added to $\mathbb{D}$. If no equivalent nodes can be further added to $\mathbb{D}$, the process of expanding $\mathbb{D}$ is terminated. Then, $\mathbb{D}$ is stored to design pattern set, $\mathbb{P}$, and

another initial node is selected to start forming new design pattern. The procedure of design pattern matching terminates when all initial nodes are checked.

For example, we are given the connection graph of a legacy design, as shown in Fig. 5, and a target design, as seen in Fig. 7. After applying line 2 of Algorithm 1, a set of 9×13 initial nodes can be derived, which is $\mathbb{S} = \{n_1^L \rightleftharpoons n_{10}^T, n_1^L \rightleftharpoons n_{11}^T, \ldots, n_9^L \rightleftharpoons n_{21}^T, n_9^L \rightleftharpoons n_{22}^T\}$. Assuming that $n_7^L \rightleftharpoons n_{20}^T$ are first selected, after applying line 6 of Algorithm 1, a set of connected nodes, $\mathbb{C}_{n_7^L n_{20}^T} = \{n_5^L \rightleftharpoons n_{18}^T, n_5^L \rightleftharpoons n_{19}^T, n_6^L \rightleftharpoons n_{18}^T, n_6^L \rightleftharpoons n_{19}^T, n_8^L \rightleftharpoons n_{21}^T, n_8^L \rightleftharpoons n_{22}^T, n_9^L \rightleftharpoons n_{21}^T, n_9^L \rightleftharpoons n_{22}^T\}$, can be obtained. By pruning several nodes, as seen in line 7 of Algorithm 1, the set of connected nodes can be reduced, which is $\mathbb{C}_{n_7^L n_{20}^T} = \{n_5^L \rightleftharpoons n_{18}^T, n_6^L \rightleftharpoons n_{19}^T, n_8^L \rightleftharpoons n_{21}^T, n_9^L \rightleftharpoons n_{22}^T\}$. Finally, by performing lines 11–18 of Algorithm 1, the corresponding design patterns can be obtained, which is $\mathbb{D}_{n_7^L n_{20}^T} = \{n_5^L \rightleftharpoons n_{18}^T, n_6^L \rightleftharpoons n_{19}^T, n_7^L \rightleftharpoons n_{20}^T, n_8^L \rightleftharpoons n_{21}^T, n_9^L \rightleftharpoons n_{22}^T$.

Since different target designs may be created for different design purposes, it may be desirable to reuse the design expertise from the legacy design with the same design objective to create the target design. Based on this consideration, only the sub-circuits from the legacy design with the same design objective of target design are considered and reused to generate the target design when performing our proposed design pattern matching algorithm. Consequently, the derived target layout can have expected electrical characteristics.

2) Time Complexity Analysis: In Algorithm 2, it takes $O(V_n \times V_m)$ iterations to finish the generation process, where V_n and V_m are the number of nodes in Gs_L and Gs_T, respectively. Since line 4 of Algorithm 2 can be executed in constant time, it takes $O(V_n \times V_m)$ to complete line 2 of Algorithm 1. In Algorithm 3, it takes $O(V'_n \times V'_m)$ iterations to explore all connected nodes. Since both lines 4 and 5 can be executed in constant time, assuming that there are V'_n and V'_m nodes connected to n_i^L and n_j^T, respectively, then it takes $O(V'_n \times V'_m)$ to complete line 6 of Algorithm 1. In Algorithm 4, assuming that there are E connected nodes, it takes $O(E \times \log E)$ to sort all E connected nodes and needs $O(E)$ to remove $\{n_x^L \rightleftharpoons n_y^T\}$ with less net reusage from $\mathbb{C}_{n_i^L n_j^T}$. Therefore, assume that there are total V initial nodes generated by Algorithm 2, it takes $O(V \times \mathrm{Max}(V'_n \times V'_m, E \times \log E))$ to complete lines 3–8 of Algorithm 1, where $\mathrm{Max}(V'_n \times V'_m, E \times \log E)$ denotes the maximum between $(V'_n \times V'_m)$ and $(E \times \log E)$. In general, $O(V'_n \times V'_m)$ is bounded by $O(V_n \times V_m)$ and $O(E \times \log E)$ is bounded by $O(V_m \times \log V_m)$. Moreover, V_m can be treated as a multiple of V_n, i.e., $\log V_m$ equals to $\beta + \log V_n$. As a result, $O(V \times \mathrm{Max}(V'_n \times V'_m, E \times \log E))$ is bounded by $O(V \times V_n^2)$.

In Algorithm 1, lines 9, 11, 12, 15, and 18 can be executed in constant time, line 13 needs at most $\mathrm{Min}(V_n, V_m)$ iterations, and lines 14 and 16 takes $O(V_m)$, where $\mathrm{Min}(V_n, V_m)$ denotes the minimum between V_n and V_m. Therefore, it takes $O(V \times \mathrm{Min}(V_n, V_m) \times V_m)$ to complete lines 9–19. By the aforementioned assumption, V_m can be treated as a multiple of V_n, then $O(V \times \mathrm{Min}(V_n, V_m) \times V_m)$ can be bounded by $O(V \times V_n^2)$. To sum up, the overall time complexity of Algorithm 1 is $O(V \times V_n^2)$.

3) Equivalence Condition Adjustment: For the efficiency of verifying whether two sub-circuits are identical based on the connection graph, the TC between any two devices is represented by a universal code which is attached on the respective edge in the connection graph. Given s_L, as shown in Fig. 3(a), and s_T, as shown in Fig. 7(a), the corresponding G_{s_L} and G_{s_T} are constructed, as shown in Figs. 5 and 7(b), respectively. Assuming that $n_{20} \in G_{s_T}$ and $n_7 \in G_{s_L}$ are selected as the initial nodes, the resulting design pattern containing five MOS transistors, $M_5^L \rightleftharpoons M_{18}^T$, $M_6^L \rightleftharpoons M_{19}^T$, $M_7^L \rightleftharpoons M_{20}^T$, $M_8^L \rightleftharpoons M_{21}^T$, and $M_9^L \rightleftharpoons M_{22}^T$, can be derived based on Algorithm 1 if the equivalence checking of device connections in line 2 of Algorithm 3 is simply verified by the following condition:

$$\Gamma_{i\to x}^L = \Gamma_{j\to y}^T \quad \Big(\text{i.e., } \mathrm{TC}_{i\to x}^L = \mathrm{TC}_{j\to y}^T\Big). \tag{4}$$

Based on such condition in (4), it may not effectively utilize more reusable layout patterns from l_L to generate l_T when the sub-circuits of s_L and s_T are not exactly the same. According to our observation, we found that the layout of the sub-circuit containing the devices represented by n_j and n_y can also be generated by utilizing the layout of the sub-circuit containing the devices represented by n_i and n_x if the following condition is satisfied:

$$\mathrm{TC}_{j\to y}^T \subseteq \mathrm{TC}_{i\to x}^L. \tag{5}$$

In order to effectively extract more reusable layout patterns from l_L, we replace the equivalent condition of the interconnection between two devices in (4) with that in (5). However, it should be noted that some redundant connections, $\{\mathrm{TC}_{i\to x}^L - \mathrm{TC}_{j\to y}^T\}$, must be removed from l_L to obtain an identical circuit structure for both l_T and s_T. Assuming that $n_{20} \in G_{s_T}$ and $n_7 \in G_{s_L}$ in Figs. 7(b) and 5, respectively, are selected as the initial nodes, the design patterns resulting from Algorithm 1 with (5) will contain four more devices, including $M_1^L \rightleftharpoons M_{14}^T$, $M_2^L \rightleftharpoons M_{15}^T$, $M_3^L \rightleftharpoons M_{16}^T$, and $M_4^L \rightleftharpoons M_{17}^T$, in addition to $M_5^L \rightleftharpoons M_{18}^T$, $M_6^L \rightleftharpoons M_{19}^T$, $M_7^L \rightleftharpoons M_{20}^T$, $M_8^L \rightleftharpoons M_{21}^T$, and $M_9^L \rightleftharpoons M_{22}^T$.

Table III further lists all matched design patterns, including the numbers of matched devices and nets in each design pattern, between the legacy design in Fig. 3 and the target design in Fig. 7 resulting from Algorithm 1.

B. Design Pattern Pruning

As the number of design patterns can be very large when the a legacy or a target design contains too many devices, we further propose a graph-based approach to effectively prune the unwanted design patterns without deteriorating the solution quality, and also to reduce the complexity of the succeeding design pattern selection procedure. Given a list of design patterns, as seen in Table III, we construct the corresponding pattern graph, $G_{P_{L\rightleftharpoons T}}$, which is defined as follows.

Definition 4: A pattern graph, $G_{P_{L\rightleftharpoons T}}$, represents the relationship among all design patterns between each legacy design and the target design, where each node $v_i \in G_{P_{L\rightleftharpoons T}}$ represents a design pattern, and there is an edge, $e_{ij} \in G_{P_{L\rightleftharpoons T}}$ connecting v_i and $v_j \in G_{P_{L\rightleftharpoons T}}$ if and only if v_i and v_j have no common device in the target design, s_T.

TABLE III
ALL MATCHED DESIGN PATTERNS BETWEEN THE LEGACY AND TARGET DESIGNS IN FIGS. 3 AND 7 RESULTING FROM ALGORITHM 1

Pattern #	Matched device list	# of devices	# of nets
1	$M_1^L \rightleftharpoons M_{10}^T$, $M_2^L \rightleftharpoons M_{11}^T$, $M_3^L \rightleftharpoons M_{12}^T$, $M_4^L \rightleftharpoons M_{13}^T$	4	6
2	$M_1^L \rightleftharpoons M_{14}^T$, $M_2^L \rightleftharpoons M_{15}^T$, $M_3^L \rightleftharpoons M_{16}^T$, $M_4^L \rightleftharpoons M_{17}^T$, $M_5^L \rightleftharpoons M_{18}^T$, $M_6^L \rightleftharpoons M_{19}^T$, $M_7^L \rightleftharpoons M_{20}^T$, $M_8^L \rightleftharpoons M_{21}^T$, $M_9^L \rightleftharpoons M_{22}^T$	9	15
3	$M_4^L \rightleftharpoons M_{14}^T$, $M_3^L \rightleftharpoons M_{15}^T$, $M_2^L \rightleftharpoons M_{16}^T$, $M_1^L \rightleftharpoons M_{17}^T$, $M_6^L \rightleftharpoons M_{18}^T$, $M_5^L \rightleftharpoons M_{19}^T$, $M_7^L \rightleftharpoons M_{20}^T$	7	9
4	$M_8^L \rightleftharpoons M_{18}^T$, $M_9^L \rightleftharpoons M_{19}^T$	2	1
5	$M_5^L \rightleftharpoons M_{20}^T$	1	0

Based on Definition 4, a pattern graph has the following property.

Property 1: Given a pattern graph, $G_{P_{L\rightleftharpoons T}}$, if $e_{ij} \notin G_{P_{L\rightleftharpoons T}}$, the corresponding design patterns of v_i and v_j cannot be simultaneously selected for layout generation of the target design.

Proof: According to the construction of $G_{P_{L\rightleftharpoons T}}$, if there exists no edge between $v_i \in G_{P_{L\rightleftharpoons T}}$ and $v_j \in G_{P_{L\rightleftharpoons T}}$, there will be at least one common device $d_k \in s_T$ in both design patterns represented by v_i and v_j. If both design patterns are selected to be reused in the target layout of s_T, the layout of d_k is duplicated. Therefore, the design patterns represented by v_i and v_j cannot be selected together if $e_{ij} \notin G_{P_{L\rightleftharpoons T}}$. ■

Before introducing how to eliminate the redundant nodes in $G_{P_{L\rightleftharpoons T}}$, we first define the candidate set and reusability of selecting a design pattern, or a node $v_i \in G_{P_{L\rightleftharpoons T}}$, as follows.

Definition 5: Given a pattern graph, $G_{P_{L\rightleftharpoons T}}$, the candidate set of a node $v_i \in G_{P_{L\rightleftharpoons T}}$, denoted by CanSet_{v_i}, is the set of nodes which have an edge connecting to v_i.

Definition 6: Given a pattern graph, $G_{P_{L\rightleftharpoons T}}$, the reusability of a node $v_i \in G_{P_{L\rightleftharpoons T}}$, denoted by reusability(v_i), refers to the number of devices, denoted by device_num(v_i), and the number of nets, denoted by net_num(v_i), contained in the corresponding design pattern. For any two nodes, $v_i \in G_{P_{L\rightleftharpoons T}}$ and $v_j \in G_{P_{L\rightleftharpoons T}}$, if reusability($v_i$) > reusability($v_j$), one of the following conditions must be satisfied.

1) device_num(v_i) > device_num(v_j).
2) device_num(v_i) = device_num(v_j) ∩ net_num(v_i) > net_num(v_j).

Base on the above definitions, we can derive the following property which indicates the redundant nodes in a pattern graph.

Property 2: Given a pattern graph, $G_{P_{L\rightleftharpoons T}}$, a node, $v_i \in G_{P_{L\rightleftharpoons T}}$, is a redundant node if there exists another node, $v_j \in G_{P_{L\rightleftharpoons T}}$, which satisfies all the following conditions.

1) $e_{ij} \notin G_{P_{L\rightleftharpoons T}}$.
2) $\text{CanSet}_{v_i} \subseteq \text{CanSet}_{v_j}$.
3) reusability(v_i) < reusability(v_j).

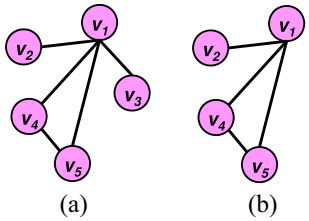

Fig. 8. (a) Example pattern graph which is constructed according to the design patterns in Table III. (b) Reduced pattern graph after removing v_3 in (a) for eliminating redundant patterns in Table III.

Proof: According to Property 1, since $e_{ij} \notin G_{P_{L\rightleftharpoons T}}$, v_i and v_j cannot be simultaneously selected during pattern selection procedure, which implies that either v_i or v_j might be redundant. $\text{CanSet}_{v_i} \subseteq \text{CanSet}_{v_j}$ implies that if $\forall v_k \in G_{P_{L\rightleftharpoons T}}$ can be selected together with v_i, v_k can also be selected together with v_j. According to Definition 6, reusability(v_j) > reusability(v_i) means that selecting the design pattern represented by v_j results in reusing more placement and routing topologies of s_L than selecting the design pattern represented by v_i when generating the layout of s_T. If a node, $v_k \in \text{CanSet}_{v_i} \cap \text{CanSet}_{v_j}$ can be selected together with either v_i or v_j, selecting v_j is always preferred for higher layout reusage rate. Consequently, v_i is regarded as a redundant node, and it can be removed from $G_{P_{L\rightleftharpoons T}}$ such that the complexity of the succeeding pattern selection procedure is reduced. ■

Based on Property 2, all redundant nodes can be effectively eliminated by checking all pairs of nodes without an edge connection in $G_{P_{L\rightleftharpoons T}}$ such that the number of nodes in $G_{P_{L\rightleftharpoons T}}$ is minimized. Fig. 8(a) shows an example pattern graph, $G_{P_{L\rightleftharpoons T}}$, which is constructed based on the design patterns in Table III. Since $e_{23} \notin G_{P_{L\rightleftharpoons T}}$, $\text{CanSet}_{v_2} = \text{CanSet}_{v_3} = \{v_1\}$, and reusability($v_2$) > reusability($v_3$) according to Table III and Definition 6, v_3 is a redundant node. By removing v_3 from the pattern graph in Fig. 8(a), the resulting pattern graph with minimized node number is shown in Fig. 8(b).

C. Design Pattern Selection

Based on the reduced pattern graph, where all the redundant design patterns had been eliminated, we formulate the design pattern selection problem as the maximum-weight-clique problem. The pattern selection problem is defined as follows.

Problem 2 (Pattern Selection Problem): Given a set of design patterns, $\mathbb{P}$, between each legacy design and the target design, select a subset of $\mathbb{P}$ resulting in maximum reusage of placement and routing topologies of devices and nets in legacy designs for layout generation of the target design.

According to the definition and property of pattern graph in Definition 4 and Property 1, a subset of design patterns can be selected only when the corresponding nodes in the reduced pattern graph form a clique. To achieve the maximum reusage of legacy layouts for layout generation of the target design, such pattern selection problem can be formulated as the maximum-weight-clique problem by assigning a weight to each node in the pattern graph. The weight of a node, v_i, can be calculated by the weighted summation of the numbers

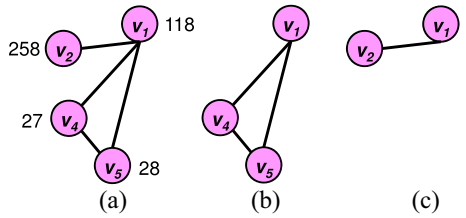

Fig. 9. Example of design pattern selection. (a) Reduced pattern graph in Fig. 8(b). (b) Clique derived from the pattern graph in (a). (c) Maximum weight clique derived from (a).

of devices and nets in v_i, as seen in (6). In analog design, a device usually has larger layout area than a routing net, so it is more important to reuse more devices than nets. In order to emphasize the importance of device_num(v_i), a user-defined parameter, α, is set to device_num(v_i). Moreover, to guarantee device_num(v_i) has always higher priority than net_num(v_i), α is defined as total number of nets in the target design plus one (because net_num(v_i) $\leq$ total number of nets)

$$\text{weight}(v_i) = \alpha \times \text{device\_num}(v_i) + \text{net\_num}(v_i). \quad (6)$$

Based on the formulation, the maximum-weight-clique problem can be solved by the existing approaches, such as [50]. Fig. 9 shows an example of formulating the design pattern selection problem as a maximum-weight-clique problem. In Fig. 9(a), each node is first annotated with the corresponding weight according to Table III and (6). Fig. 9(b) and (c) shows two different cliques extracted from the pattern graph in Fig. 9(a). Based on the existing maximum-weight-clique solver, the maximum weight clique in Fig. 9(c) can be obtained. Consequently, the first two design patterns in Table III are selected for target layout generation due to better reusage of placement and routing topologies in the legacy design.

V. Knowledge-Based Layout Generation

After selecting the best design patterns from the existent legacy designs for the target design, the layouts of all selected design patterns are first extracted from l_L and migrated to the new device parameters and process technology for l_T, as described in Section V-A. Once the layouts of all design patterns are migrated, they are further assembled to generate the final layout of the target design, which is detailed in Section V-B.

A. Layout Extraction and Migration for Design Patterns

As mentioned in [51], the physical information of a legacy layout, l_L, can be obtained by utilizing OpenAccess API, so the layout of each selected design pattern can be easily extracted. For example, given a legacy layout, as shown in Fig. 10(a), the corresponding layout of a design pattern containing five MOS devices, M_1, M_2, M_5, M_6, and M_{11}, as well as their internal routing in Fig. 10(a) can be automatically extracted, as shown from Fig. 10(b).

Since the process technology and device parameters, such as width, length, etc., of the target design may be different from those of the legacy designs in the design repository, it is required to perform some adjustments on the extracted layouts before reusing them. Such layout migration process can be well manipulated by the existent layout migration methods and algorithms [51], [52].

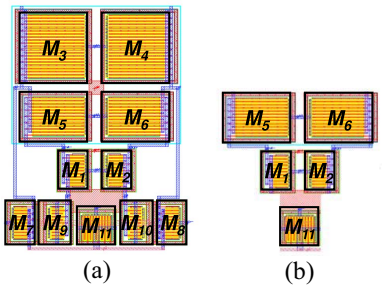

Fig. 10. (a) Example legacy layout. (b) Extracted layout of a design pattern containing five MOS devices, M_1, M_2, M_5, M_6, and M_{11}, as well as their internal routing in (a).

B. Integration of Design Pattern Layouts

Once the layout of each design pattern is extracted from the design repository and migrated to the new process technology, we shall complete the layout of the target design by integrating all the design pattern layouts and connecting the unconnected nets among different design patterns. It should be noted that the shape of the extracted layout of each design pattern might not be rectangular, but rectilinear. To integrate rectilinear shapes of different design pattern layouts while considering other placement constraints among different design patterns, the hierarchical B*-tree representation [3], [17] is applied such that the integrated layout can be compacted with the consideration of rectilinear contours. As the number of design pattern layouts is much less than the number of devices in the target design, the hierarchical B*-trees can be exhaustively searched to explore all placement variants of the integrated layout.

In some cases, the design patterns, or matched sub-circuits, might not be found in the legacy design repository. The layouts of the devices in those sub-circuits will be mapped to some primitive cells with different layout variants, which is similar to the "device layout generation" in the conventional analog layout generation flow in Fig. 1(a). Our approach can effectively integrate the layouts of both matched and unmatched sub-circuits for complete analog layouts.

After integrating the layouts of all design patterns, only the nets connecting different design patterns need to be routed. Such interpattern routing problem is much simpler than the conventional analog routing problems which had been studied in [53]–[55]. Consequently, the routing approaches proposed in those works can easily handle the interpattern routing problem.

VI. Experimental Results

We implemented the proposed knowledge-based analog physical synthesis methodology in C++ programming language on a 2.9 GHz Linux machine with 32 GB memory. To show the effectiveness of our proposed synthesis methodology, we experimentally tested our algorithm on two sets of experiments as demonstrated in Sections VI-A and VI-B.

TABLE IV
COMPARISONS OF LAYOUT REUSAGE RATES WITH FOUR DIFFERENT SETS OF LEGACY DESIGNS TO GENERATE A MILLER OP-AMP

Target design	Legacy designs	Layout reusage rate of devices	Layout reusage rate of nets	Runtime (sec)
Miller Op-Amp (15 devices;28 nets)	{∅}	0 / 15 = 0 %	0 / 28 = 0 %	-
	{Diff Amp}	11 / 15 = 73.33 %	11 / 28 = 39.29 %	0.00
	{OTA}	13 / 15 = 86.67 %	14 / 28 = 50.00 %	0.00
	{Diff Amp, OTA}	15 / 15 = 100.00 %	16 / 28 = 57.14 %	0.00

TABLE V
PERFORMANCE COMPARISONS BETWEEN MANUAL LAYOUT AND THIS PAPER FOR MILLER OP-AMP

Target layout	Gain (dB)	Gain bandwidth (GBW) (MHz)	Phase margin (PM) (deg)	Power (mW)
Manual	77.43	977.90	9.93	5.32
This work	76.16	977.43	7.91	5.08
Comparisons	↓ 1.64%	↓ 0.05%	↓ 20.34%	↓ 4.49%

TABLE VI
COMPARISONS OF LAYOUT REUSAGE RATES WITH FIVE DIFFERENT SETS OF LEGACY DESIGNS TO GENERATE A WIDE-SWING DIFF AMP

Target design	Legacy designs	Layout reusage rate of devices	Layout reusage rate of nets	Runtime (sec)
Wide-swing Diff Amp (37 devices;80 nets)	{∅}	0 / 37 = 0 %	0 / 80 = 0 %	-
	{Diff Amp}	28 / 37 = 75.67 %	20 / 80 = 25.00 %	0.01
	{OTA}	31 / 37 = 83.78 %	25 / 80 = 31.25 %	0.05
	{Diff Amp, OTA}	37 / 37 = 100.00 %	30 / 80 = 37.50 %	0.14
	{Diff Amp, OTA, Miller Op-Amp}	37 / 37 = 100.00 %	48 / 80 = 60.00%	1.55

TABLE VII
DESIGN PATTERNS DERIVED FROM DIFF AMP AND OTA

Pattern #	All devices involved in each design pattern
1*	$M_4^L \rightleftharpoons M_3^T$, $M_5^L \rightleftharpoons M_4^T$, $C_1^L \rightleftharpoons C_1^T$, $C_2^L \rightleftharpoons C_2^T$
2	$M_1^L \rightleftharpoons M_1^T$, $M_2^L \rightleftharpoons M_2^T$, $M_3^L \rightleftharpoons M_{10}^T$, $M_4^L \rightleftharpoons M_{11}^T$
3	$M_8^L \rightleftharpoons M_5^T$, $M_9^L \rightleftharpoons M_6^T$
4	$M_5^L \rightleftharpoons M_7^T$, $M_6^L \rightleftharpoons M_8^T$, $M_7^L \rightleftharpoons M_9^T$ $M_8^L \rightleftharpoons M_{12}^T$, $M_9^L \rightleftharpoons M_{13}^T$

*This design pattern is derived from "Diff Amp", and all the other design patterns are derived from "OTA".

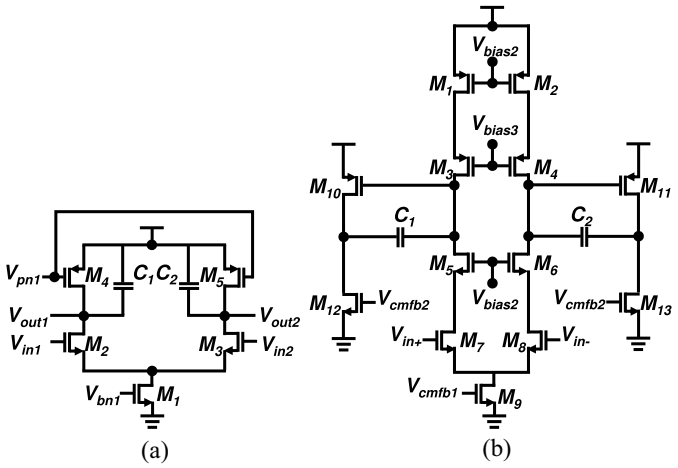

Fig. 11. (a) nMOS differential amplifier, which is a legacy design. (b) Two-stage Miller-compensated operational amplifier, which is a target design.

A. Two-Stage Operational Amplifier With Miller Compensation

In the first set of the experiments, we automatically generated the layout of the target design, a two-stage operational amplifier with Miller compensation ("Miller Op-Amp"), as shown in Fig. 11(b), based on two different legacy designs, including both schematics and layouts, in the design repository: 1) an nMOS differential amplifier ("Diff Amp"), as shown in Fig. 11(a) and 2) a CMOS cascade OTA, as shown in Fig. 3(a). To demonstrate the effectiveness of the proposed methodology, we compared the layout reusage rates with four different sets of legacy designs in the design repository, including {∅}, {Diff Amp}, {OTA}, and {Diff Amp, OTA}, as shown in Table IV.

According to Table IV, if none of the legacy designs is available in the design repository, the layout of the target design should be designed from scratch. If the design repository contains only one legacy design, Diff Amp, the layout of 73.33% devices (11 out of 15 devices) and 39.29% nets (11 out of 28 nets) in the target design, Miller Op-Amp, can be generated according to the legacy layout of Diff Amp. If the design repository contains only one legacy design, OTA, the layout reusage rate of devices can be increased from 73.33% to 86.67%, and the layout reusage rate of nets can be increased from 39.29% to 50.00%. The higher layout reusage rates indicate that the legacy design and the target design have higher similarity. Finally, if the design repository contains both legacy designs, Diff Amp and OTA, the layout reusage rate of devices can achieve 100%, while the layout reusage rate of nets can also be further improved from 50.00% to 57.14%. When both legacy designs, Diff Amp and OTA, are used, all derived design patterns are demonstrated in Table VII, where pattern 1 is derived from Diff Amp and all the other patterns are derived from OTA. Fig. 12(a) and (b) shows the layouts of both legacy designs, Diff Amp and OTA, respectively.

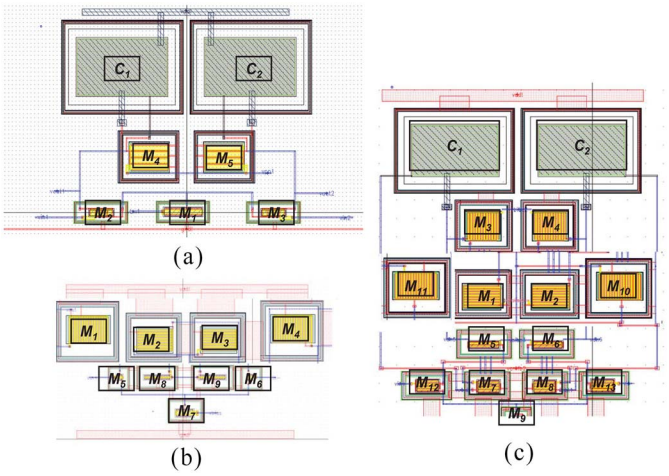

Fig. 12. (a) Layout of the legacy design in Fig. 11(a). (b) Layout of the legacy design in Fig. 3(a). (c) Layout of the target design in Fig. 11(b) utilizing both legacy layouts in (a) and (b) based on the proposed methodology.

Fig. 12(c) shows the layout of the target design, which is the Miller Op-Amp in Fig. 11(b). The placement and the internal routing topology of the devices, C_1, C_2, M_3, and M_4, in Fig. 12(c) were extracted and migrated from the layout of Diff Amp in Fig. 12(a), while the placement and the internal routing topology of the rest of devices in Fig. 12(c) were extracted and migrate from the layout of OTA in Fig. 12(b).

Table V shows the comparisons of the performance metrics, including gain, gain bandwidth, phase margin, and power, for manual layout and the layout generated by our approach. Compared with manual layout, the layout generated by our approach results in better gain, gain bandwidth, and power consumption, but much worse phase margin. The reasons is that when generating the layout of Miller Op-Amp, there are only two legacy design data in the design repository. Consequently, the target layout generated by our approach will have fewer layout variants, and the resulting layout quality is not comparable to the manual layout.

B. Fully-Differential Wide-Swing Folded-Cascode CMOS Operational Amplifier

According to Fig. 2, the design knowledge database in our proposed synthesis methodology can be continuously expanded when more and more analog layouts are verified and approved by experienced analog design experts. In the second set of our experiments, we further expanded the design repository by adding the previously generated circuit, Miller Op-Amp. We automatically generated the layouts of another target design, a fully-differential wide-swing folded-cascode CMOS operational amplifier ("Wide-swing Diff Amp"), as shown in Fig. 13, based on five different sets of legacy designs in the design repository, including {Ø}, {Diff Amp}, {OTA}, {Diff Amp, OTA}, and {Diff Amp, OTA, Miller Op-Amp}, as shown in Table VI.

According to Table VI, if none of the legacy designs is available in the design repository, the layout of the target design should be designed from scratch. If the design repository contains only one legacy design, Diff Amp, the layout

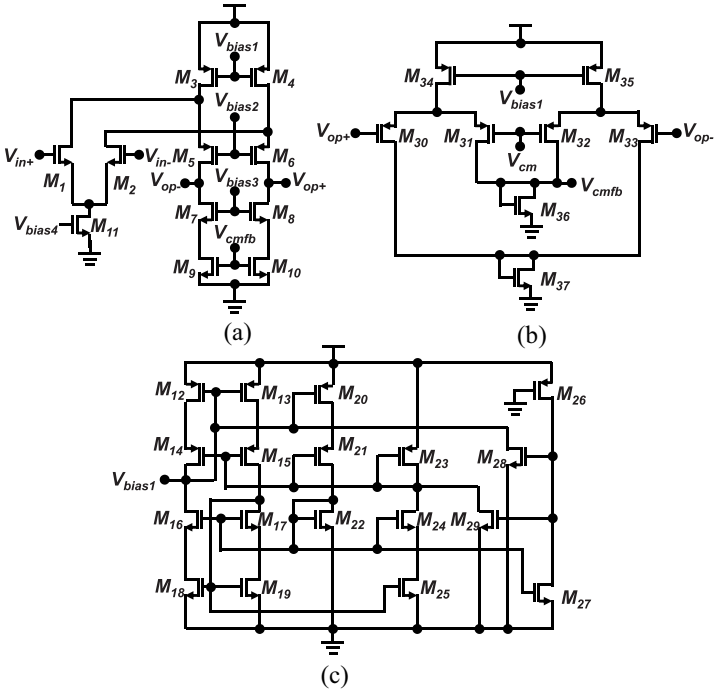

Fig. 13. Schematic of a fully-differential wire-swing folded-cascode CMOS operational amplifier, which includes (a) fully-differential folded-cascode CMOS operational amplifier, (b) continuous-time common-mode feedback circuit, and (c) wide-swing bias circuit.

of 75.67% devices (28 out of 37 devices) and 25.00% nets (20 out of 80 nets) in the target design, Wide-swing Diff Amp, can be generated according to the legacy layout of Diff Amp. If the design repository contains only one legacy design, OTA, the layout reusage rate of devices can be increased from 75.67% to 83.78%, and the layout reusage rate of nets can be increased from 25.00% to 31.25%. If the design repository contains both legacy designs, Diff Amp and OTA, the layout reusage rate of devices can achieve 100%, while the layout reusage rate of nets can also be further improved from 31.25% to 37.50%. Finally, if the design repository contains all the three legacy designs, Diff Amp, OTA, and Miller Op-Amp, the layout reusage rate of devices can achieve 100.00%. Compared with the design repository containing only Diff Amp and OTA, the layout reusage rate of nets can be significantly improved from 37.50% to 60.00%. Table IX shows all the derived design patterns when the design repository contains all the three legacy designs, Diff Amp, OTA, and Miller Op-Amp, where pattern 1 is derived from Diff Amp, patterns 2–7 are from OTA, and patterns 8–13 are from Miller Op-Amp.

Table VIII shows the comparisons of the performance metrics, including gain, gain bandwidth, phase margin, and power, for manual layout and the layout generated by our approach. The results shown that all the performance metrics are within 1% difference. When comparing the difference of performance metrics between manual layouts and the layouts generated by the proposed approach in Tables V and VIII, the difference of performance metrics for Miller Op-Amp is larger than that of Wide-swing Diff Amp. It is because the number of available

TABLE VIII
PERFORMANCE COMPARISONS BETWEEN MANUAL LAYOUT AND THIS PAPER FOR WIDE-SWING DIFF OP

Target layout	Gain (dB)	Gain bandwidth (GBW) (MHz)	Phase margin (PM) (deg)	Power (mW)
Manual	70.40	208.05	55.41	2.77
This work	70.29	208.60	55.32	2.78
Comparisons	$\downarrow 0.15\%$	$\uparrow 0.26\%$	$\downarrow 0.16\%$	$\uparrow 0.31\%$

TABLE IX
DESIGN PATTERNS DERIVED FROM DIFF AMP, OTA, AND MILLER OP-AMP

Pattern #	All devices involved in each design pattern
1*	$M_2^L \rightleftharpoons M_{30}^T$, $M_3^L \rightleftharpoons M_{33}^T$
2**	$M_8^L \rightleftharpoons M_{28}^T$, $M_9^L \rightleftharpoons M_{29}^T$
3**	$M_1^L \rightleftharpoons M_{12}^T$, $M_2^L \rightleftharpoons M_{13}^T$
4**	$M_1^L \rightleftharpoons M_{20}^T$, $M_2^L \rightleftharpoons M_{26}^T$
5**	$M_8^L \rightleftharpoons M_{36}^T$
6**	$M_8^L \rightleftharpoons M_{37}^T$
7**	$M_1^L \rightleftharpoons M_{21}^T$, $M_8^L \rightleftharpoons M_{22}^T$
8***	$M_4^L \rightleftharpoons M_{23}^T$, $M_6^L \rightleftharpoons M_{24}^T$, $M_8^L \rightleftharpoons M_{25}^T$
9***	$M_1^L \rightleftharpoons M_{34}^T$, $M_2^L \rightleftharpoons M_{35}^T$, $M_3^L \rightleftharpoons M_{31}^T$, $M_4^L \rightleftharpoons M_{32}^T$
10***	$M_9^L \rightleftharpoons M_{27}^T$, $M_{12}^L \rightleftharpoons M_{18}^T$, $M_{13}^L \rightleftharpoons M_{19}^T$
11***	$M_1^L \rightleftharpoons M_3^T$, $M_2^L \rightleftharpoons M_4^T$, $M_3^L \rightleftharpoons M_5^T$ $M_4^L \rightleftharpoons M_6^T$, $M_5^L \rightleftharpoons M_7^T$, $M_6^L \rightleftharpoons M_8^T$
12***	$M_7^L \rightleftharpoons M_1^T$, $M_8^L \rightleftharpoons M_2^T$, $M_9^L \rightleftharpoons M_{11}^T$ $M_{12}^L \rightleftharpoons M_9^T$, $M_{13}^L \rightleftharpoons M_{10}^T$
13***	$M_3^L \rightleftharpoons M_{14}^T$, $M_4^L \rightleftharpoons M_{15}^T$, $M_5^L \rightleftharpoons M_{16}^T$, $M_6^L \rightleftharpoons M_{17}^T$

*This design pattern is derived from "Diff Amp".
**This design pattern is derived from "OTA".
***This design pattern is derived from "Miller Op-Amp".

legacy design for Miller Op-Amp is less than that for Wide-swing Diff Amp. When more legacy designs are available in the design repository, our approach will be able to generate more layout variants, and the resulting layout quality can be more close to manual layout. In summary, it is essential to utilize the design expertise contained in legacy designs when generating the new layout of a target design.

VII. CONCLUSION

In this paper, we have proposed a novel knowledge-based analog physical synthesis methodology which integrates existent design expertise in the design repository. In order to utilize the quality-approved legacy layouts as much as possible, we have also presented new problem formulations and algorithms to maximize the reusage of the placement and routing topologies. Experimental results have shown that the proposed methodology can achieve high layout reusage rate, and hence the designers' layout preference can be successfully reserved and the design effort can be greatly reduced.

REFERENCES

[1] A. Abdulghany, R. Fathy, L. Capodieci, and S. Malik, "Yield enhancement flow for analog and full custom designs reliability-rules automatic application," in *Proc. IEEE Int. Design Test Workshop*, Beirut, Lebanon, 2011, pp. 74–77.

[2] C.-W. Lin, J.-M. Lin, C.-P. Huang, and S.-J. Chang, "Performance-driven analog placement considering boundary constraint," in *Proc. ACM/IEEE Design Autom. Conf.*, Anaheim, CA, USA, 2010, pp. 292–297.

[3] H.-F. Tsao *et al.*, "A corner stitching compliant B*-tree representation and its applications to analog placement," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, San Jose, CA, USA, pp. 507–511, 2011.

[4] Q. Ma, L. Xiao, Y.-C. Tam, and E. F. Y. Young, "Simultaneous handling of symmetry, common centroid, and general placement constraints," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 30, no. 1, pp. 85–95, Jan. 2011.

[5] F. Y. Young, D. F. Wong, and H. H. Yang, "Slicing floorplans with boundary constraints," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 18, no. 9, pp. 1385–1389, Sep. 1999.

[6] M. Strasser, M. Eick, H. Gräb, U. Schlichtmann, and F. M. Johannes, "Deterministic analog circuit placement using hierarchically bounded enumeration and enhanced shape functions," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, San Jose, CA, USA, 2008, pp. 306–313.

[7] L. Xiao and E. F. Y. Young, "Analog placement with common centroid and 1-D symmetry constraints," in *Proc. IEEE/ACM Asia South Pac. Design Autom. Conf.*, Yokohama, Japan, 2009, pp. 353–360.

[8] P.-H. Lin and S.-C. Lin, "Analog placement based on hierarchical module clustering," in *Proc. ACM/IEEE Design Autom. Conf.*, Anaheim, CA, USA, 2008, pp. 50–55.

[9] T. H. Abthoff and F. M. Johannes, "TINA: Analog placement using enumerative techniques capable of optimizing both area and net length," in *Proc. IEEE Eur. Design Autom. Conf.*, Geneva, Switzerland, 1996, pp. 398–403.

[10] J. Rijmenants, J. B. Litsios, T. R. Schwarz, and M. G. R. Degrauwe, "ILAC: An automated layout tool for analog CMOS circuits," *IEEE J. Solid-State Circuits*, vol. 24, no. 2, pp. 417–425, Apr. 1989.

[11] F. Y. Young, D. F. Wong, and H. H. Yang, "Slicing floorplans with range constraint," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 19, no. 2, pp. 272–278, Feb. 2000.

[12] D. Long, X. Hong, and S. Dong, "Signal-path driven partition and placement for analog circuit," in *Proc. IEEE/ACM Asia South Pac. Design Autom. Conf.*, Yokohama, Japan, 2006, pp. 694–699.

[13] P.-H. Wu *et al.*, "Performance-driven analog placement considering monotonic current paths," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, San Jose, CA, USA, 2012, pp. 613–619.

[14] F. Y. Young and D. F. Wong, "Slicing floorplans with pre-placed modules," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, San Jose, CA, USA, 1998, pp. 252–258.

[15] P.-Y. Chou, H.-C. Ou, and Y.-W. Chang, "Heterogeneous B*-trees for analog placement with symmetry and regularity considerations," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, San Jose, CA, USA, 2011, pp. 512–516.

[16] S. Nakatake *et al.*, "Regularity-oriented analog placement with diffusion sharing and well island generation," in *Proc. IEEE/ACM Asia South Pac. Design Autom. Conf.*, Taipei, Taiwan, 2010, pp. 305–311.

[17] P.-H. Lin, Y.-W. Chang, and S.-C. Lin, "Analog placement based on symmetry-island formulation," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 28, no. 6, pp. 791–804, Jun. 2009.

[18] M. P.-H. Lin, H. Zhang, M. D. F. Wong, and Y.-W. Chang, "Thermal-driven analog placement considering device matching," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 30, no. 3, pp. 325–336, Mar. 2011.

[19] J. Liu, S. Dong, Y. Ma, D. Long, and X. Hong, "Thermal-driven symmetry constraint for analog layout with CBL representation," in *Proc. IEEE/ACM Asia South Pac. Design Autom. Conf.*, Yokohama, Japan, 2007, pp. 191–196.

[20] Y. Pang, F. Balasa, K. Lampaert, and C.-K. Cheng, "Block placement with symmetry constraints based on the O-tree non-slicing representation," in *Proc. ACM/IEEE Design Autom. Conf.*, Los Angeles, CA, USA, 2000, pp. 464–467.

[21] F. Balasa and K. Lampaert, "Symmetry within the sequence-pair representation in the context of placement for analog design," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 19, no. 7, pp. 721–731, Jul. 2000.

[22] M. P.-H. Lin *et al.*, "Augmenting slicing trees for analog placement," in *Proc. IEEE Int. Conf. Synth. Model. Anal. Simulat. Methods Appl.*, Seville, Spain, 2012, pp. 57–60.

[23] J. A. Prieto, A. Rueda, J. M. Quintana, and J. L. Huertas, "A performance-driven placement algorithm with simultaneous Place&Route optimization for analog ICs," in *Proc. IEEE Eur. Design Test Conf.*, Paris, France, 1997, pp. 389–394.

[24] J.-M. Lin, G.-M. Wu, Y.-W. Chang, and J.-H. Chuang, "Placement with symmetry constraints for analog layout design using TCG-S," in *Proc. IEEE/ACM Asia South Pac. Design Autom. Conf.*, Shanghai, China, 2005, pp. 1135–1138.

[25] F. Balasa, "Modeling non-slicing floorplans with binary trees," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, San Jose, CA, USA, 2000, pp. 13–16.

[26] F. Balasa, S. C. Maruvada, and K. Krishnamoorthy, "Efficient solution space exploration based on segment trees in analog placement with symmetry constraints," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, San Jose, CA, USA, 2002, pp. 497–502.

[27] F. Balasa, S. C. Maruvada, and K. Krishnamoorthy, "Using red-black interval trees in device-level analog placement with symmetry constraints," in *Proc. IEEE/ACM Asia South Pac. Design Autom. Conf.*, Kitakyushu, Japan, 2003, pp. 777–782.

[28] F. Balasa, S. C. Maruvada, and K. Krishnamoorthy, "On the exploration of the solution space in analog placement with symmetry constraints," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 23, no. 2, pp. 177–191, Feb. 2004.

[29] S. C. Maruvada, K. Krishnamoorthy, S. Annojvala, and F. A. Balasa, "Placement with symmetry constraints for analog layout using red-black trees," in *Proc. IEEE Int. Symp. Circuits Syst.*, Bangkok, Thailand, 2003, pp. 489–492.

[30] F. Balasa and S. C. Maruvada, "Using non-slicing topological representations for analog placement," *IEICE Trans. Fund. Electron. Commun. Comput. Sci.*, vol. E84-A, pp. 2785–2792, Nov. 2001.

[31] F. Balasa and K. Lampaert, "Module placement for analog layout using the sequence-pair representation," in *Proc. ACM/IEEE Design Autom. Conf.*, New Orleans, LA, USA, 1999, pp. 274–279.

[32] K. Krishnamoorthy, S. C. Maruvada, and F. Balasa, "Topological placement with multiple symmetry groups of devices for analog layout design," in *Proc. IEEE Int. Symp. Circuits Syst.*, New Orleans, LA, USA, 2007, pp. 2032–2035.

[33] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, May 1983.

[34] S. Bhattacharya, N. Jangkrajarng, and C.-J. Shi, "Multilevel symmetry-constraint generation for retargeting large analog layouts," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 25, no. 6, pp. 945–960, Jun. 2006.

[35] R. Martins, N. Lourenço, and N. Horta, "LAYGEN II—Automatic layout generation of analog integrated circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 32, no. 11, pp. 1641–1654, Nov. 2013.

[36] S. Wang, X. Jia, A. B. Yeh, and L. Zhang, "Analog layout retargeting using geometric programming," *ACM Trans. Design Autom. Electron. Syst.*, vol. 16, pp. 50:1–50:11, Oct. 2011.

[37] E. Yilmaz and G. Dundar, "Analog layout generator for CMOS circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 28, no. 1, pp. 32–45, Jan. 2009.

[38] C. Ebeling, "Gemini II: A second generation layout validation program," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, 1988, pp. 322–325.

[39] C. Ebeling and O. Zajicek, "Validating VLSI circuit layout by wirelist comparison," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, 1983, pp. 172–173.

[40] S. Fankhauser, K. Riesen, H. Bunke, and P. Dickinson, "Suboptimal graph isomorphism using bipartite matching," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 26, pp. 1250013-1–1250013-26, Dec. 2012.

[41] H. Shang, Y. Gao, and J. Zhu, "An optimized circuit simulation method for the identification of isomorphic disconnected graphs," *Circuits Syst. Signal Process.*, vol. 32, pp. 2469–2473, Oct. 2013.

[42] N. Dahm, H. Bunke, T. Caelli, and Y. Gao, "Topological features and iterative node elimination for speeding up subgraph isomorphism detection," in *Proc. IEEE Int. Conf. Pattern Recognit.*, Tsukuba, Japan, 2012, pp. 1164–1167.

[43] M. Ohlrich, C. Ebeling, E. Ginting, and L. Sather, "SubGemini: Identifying subcircuits using a fast subgraph isomorphism algorithm," in *Proc. ACM/IEEE Design Autom. Conf.*, Dallas, TX, USA, 1993, pp. 31–37.

[44] M. Weber, M. Liwicki, and A. Dengel, "Faster subgraph isomorphism detection by well-founded total order indexing," *Pattern Recognit. Lett.*, vol. 33, pp. 2011–2019, Nov. 2012.

[45] C. Ferent and A. Doboli, "Symbolic matching and constraint generation for systematic comparison of analog circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 32, no. 4, pp. 616–629, Apr. 2013.

[46] C. Ferent, A. Doboli, and S. Doboli, "An axiomatic model for concept structure description and its application to circuit design," *Knowl.-Based Syst.*, vol. 45, pp. 114–133, Jun. 2013.

[47] J. McGregor, "Backtrack search algorithms and the maximal common subgraph problem," *Softw. Pract. Exp.*, vol. 12, pp. 23–34, 1982.

[48] V. Carletti, P. Foggia, and M. Vento, "Performance comparison of five exact graph matching algorithms on biological databases," in *Proc. IEEE Int. Conf. Image Anal. Process.*, Naples, Italy, 2013, pp. 409–417.

[49] T. Fober, G. Klebe, and E. Hüllermeier, "Local clique merging: An extension of the maximum common subgraph measure with applications in structural bioinformatics," in *Proc. IEEE/ACM Algorithms Nat. Life*, 2013, pp. 279–286.

[50] P. R. J. Östergård, "A new algorithm for the maximum-weight clique problem," *Nord. J. Comput.*, vol. 8, pp. 424–436, Dec. 2001.

[51] Y.-P. Weng *et al.*, "Fast analog layout prototyping for nanometer design migration," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, San Jose, CA, USA, 2011, pp. 517–522.

[52] L. Zhang, N. Jangkrajarng, S. Bhattacharya, and C.-J. R. Shi, "Parasitic-aware optimization and retargeting of analog layouts: A symbolic-template approach," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 27, no. 5, pp. 791–802, May 2008.

[53] M. M. Ozdal and R. F. Hentschke, "An algorithmic study of exact route matching for integrated circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 30, no. 12, pp. 1842–1855, Dec. 2011.

[54] 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. IEEE/ACM Int. Conf. Comput.-Aided Design*, San Jose, CA, USA, 2012, pp. 620–626.

[55] L. Xiao, E. F. Y. Young, X. He, and K.-P. Pun, "Practical placement and routing techniques for analog circuit designs," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, San Jose, CA, USA, 2010, pp. 675–679.

Po-Hsun Wu (S'11) received the M.S. degree in computer science and information engineering from the National Cheng Kung University, Tainan, Taiwan, in 2010, where he is currently pursuing the Ph.D. degree from the Graduate Institute of Computer Science and Information Engineering.

His current research interests include analog design automation and physical design.

Mark Po-Hung Lin (S'07–M'09–SM'13) received the B.S. and M.S. degrees in electronics engineering from National Chiao Tung University, Hsinchu, Taiwan, and the Ph.D. degree from the Graduate Institute of Electronics Engineering, National Taiwan University, Taipei, Taiwan.

He has been with the Department of Electrical Engineering, National Chung Cheng University, Chiayi, Taiwan, since 2009, where he is currently an Associate Professor. From 2000 to 2007, he was with SpringSoft, Inc., (now Synopsys, Inc.), Hsinchu, as an Engineer, a Senior Engineer, an Associate Manager, and a Technical Manager. He was a Visiting Scholar with the University of Illinois at Urbana-Champaign, Urbana, IL, USA, from 2007 to 2009, and a Humboldt Research Fellow with the Technical University of Munich, Munich, Germany, from 2013 to 2015.

Dr. Lin was the recipient of the IEEE Tainan Section Macronix Award, the Humboldt Research Fellowship for Experienced Researchers, the Distinguished Young Scholar Award of Taiwan IC Design Society, the Outstanding Young Electrical Engineer Award of the Chinese Institute of Electrical Engineering, and the Distinguished Young Faculty Award of National Chung Cheng University.

Tung-Chieh Chen received the B.S. and Ph.D. degrees in electrical engineering from the National Taiwan University (NTU), Taipei, Taiwan, in 2003 and 2008, respectively.

He was a Visiting Scholar at the University of Texas at Austin, Austin, TX, USA, in 2007. He joined SpringSoft, Hsinchu, Taiwan, in 2008 and Synopsys in 2012 by acquisition. He is currently a Research and Development Manager with Design Group, Synopsys, Inc., Hsinchu. His current research interests include VLSI physical design, custom layout design automation, floorplanning, and placement.

Dr. Chen is the author of the mixed-size placement tool NTUplace that won third place in the 2006 ACM International Symposium on Physical Design (ISPD) Placement Contest. He was the recipient of the First Place Award in the 2007 ACM SIGDA CADathlon Contest, the Best Dissertation Award from the Graduate Institute of Electronics Engineering (GIEE) at NTU in 2008, and the Outstanding Research Award from GIEE at NTU in 2007 and 2008. He was a Technical Program Committee Member of ISPD from 2011 to 2013 and ICCAD from 2012 to 2014.

Ching-Feng Yeh is currently pursuing the M.S. degree in electrical engineering from the National Chung Cheng University, Chiayi, Taiwan.

His current research interests include analog layout automation.

Xin Li (S'01–M'06–SM'10) received the B.S. and M.S. degrees in electronics engineering from Fudan University, Shanghai, China, in 1998 and 2001, respectively, and the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, USA, in 2005.

From 2009 to 2012, he was an Assistant Director with the FCRP Focus Research Center for Circuit and System Solutions (C2S2), Carnegie Mellon University, Pittsburgh, PA, USA. In 2005, he co-founded Xigmix, Inc., Pittsburgh, to commercialize his Ph.D. research, and served as the Chief Technical Officer until 2007. He is currently an Associate Professor with the Department of Electrical and Computer Engineering, Carnegie Mellon University. Since 2014, he has been an Assistant Director for the Center for Silicon System Implementation. His current research interests include integrated circuit and signal processing.

Dr. Li was the recipient of the NSF Faculty Early Career Development Award (CAREER) in 2012, the IEEE Donald O. Pederson Best Paper Award in 2013, the best paper award from Design Automation Conference (DAC) in 2010, and the two IEEE/ACM William J. McCalla International Conference on Computer-Aided Design best paper awards in 2004 and 2011. He was an Associate Editor of the *ACM Transactions on Design Automation of Electronic Systems*, the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, and the *Journal of Low Power Electronics*. He served on the ACM/SIGDA Outstanding Ph.D. Dissertation Award Selection Committee, the IEEE Outstanding Young Author Award Selection Committee, the Technical Program Committee of DAC, and the Technical Program Committee of International Conference on Computer-Aided Design.

Tsung-Yi Ho (M'08–SM'12) received the Ph.D. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 2005.

He is a Professor with the Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan. From 2007 to 2014, he was with the National Cheng Kung University, Tainan, Taiwan. His current research interests include design automation for microfluidic biochips and nanometer integrated circuits. He has presented eight tutorials and contributed five special sessions in ACM/IEEE conferences, all in design automation for microfluidic biochips.

Prof. Ho was the recipient of several research awards, such as the Dr. Wu Ta-You Memorial Award of the National Science Council of Taiwan, the Junior Research Investigators Award of the Academia Sinica, the Distinguished Young Scholar Award of the Taiwan IC Design Society, the Outstanding Young Electrical Engineer Award of the Chinese Institute of Electrical Engineering, the Delta Electronics K. T. Li Research Award, the ACM Taipei Chapter Young Researcher Award, the IEEE Tainan Chapter Gold Member Award, the Invitational Fellowship of the Japan Society for the Promotion of Science, the Humboldt Research Fellowship by the Alexander von Humboldt Foundation, and the Hans Fischer Fellow by the Institute of Advanced Study of the Technical University of Munich. He currently serves as an ACM Distinguished Speaker, a Distinguished Visitor of the IEEE Computer Society, the Chair of the IEEE Computer Society Tainan Chapter and the ACM SIGDA Taiwan Chapter, and an Associate Editor of the *ACM Journal on Emerging Technologies in Computing Systems* and the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, the Guest Editor of the IEEE DESIGN AND TEST OF COMPUTERS. He serves on the Technical Program Committees of major conferences, including Design Automation Conference, International Conference on Computer-Aided Design, Design Automation and Test in Europe, Asia and South Pacific Design Automation Conference, ISPD, International Conference on Computer Design, VLSI Design, International Symposium on VLSI Design, Automation, and Test, International Conference on Very Large Scale Integration, International System-On-Chip Conference.