Analog Circuit Topological Feature Extraction with Unsupervised Learning of New Sub-Structures

Hao Li, Fanshu Jiao and Alex Doboli

Department of Electrical and Computer Engineering
State University of New York at Stony Brook, Stony Brook, NY, USA
Email: {hao.li.1, fanshu.jiao, alex.doboli}@stonybrook.edu

***Abstract*—This paper presents novel techniques to automatically extract the topological (structural) features in analog circuits. The extracted features include basic building blocks, structural templates and hierarchical structures. Finding structural features is important for tasks like circuit synthesis and sizing, design verification, design reuse, and design knowledge description, summarization and management. The paper presents algorithms for supervised feature extraction and unsupervised learning of new block connections. Experiments discuss feature extraction for a set of 34 state-of-the-art analog circuits.**

I. Introduction

Design knowledge is critical in analog circuit design. In spite of heavily relying on powerful optimization or equation solving methods, analog circuit synthesis still requires as input a significant amount of design knowledge [1], [6], [7]. This knowledge is represented by the building blocks and circuit libraries that many tools use, the constraints set on design parameters (e.g., specific ranges, value matching, etc.), the rules indicating compatible and incompatible topological structures, and the models to predict nonidealities and performance with different accuracies and at various abstraction levels. Arguably, the effectiveness of analog circuit synthesis tools is tightly connected to the amount and quality of design knowledge provided to a tool [2]. The knowledge is mostly produced manually, therefore, it not only requires substantial effort and time to be developed but tends to lag behind the state-of-the-art in circuit design.

Most efforts to automatically produce the design knowledge needed by automated synthesis tools are based on recognizing well-known circuit building blocks (BBs) and then creating block-specific design information, like design constraints [1], [2], [5]. These techniques have been shown to significantly improve the quality of circuit sizing, layout generation, and circuit topology generation. Graeb *et al.* [1] and Massier *et al.* [2], [3] propose a hierarchical library of BBs organized along four abstraction levels. Two search algorithms recognize the BBs in a circuit. Eick *et al.* use the same BB library to produce placement constraints [4], [5]. The hierarchical BB library devised by Das *et al.* [11] has self-learning capability. The circuit synthesis tool in [6] employs a library with standard and custom BBs. The genetic programming method for circuit topology synthesis in [8] utilizes pre-defined BBs to guide the evolutionary process. McConaghy *et al.* [9] [10] describe a topology synthesis approach that uses a library that expresses the electrical behaviors of popular BBs.

Existing methods for BB recognition have mainly a static character with limited or no capabilities to automatically learn new structural features as they are being proposed in the literature. Libraries contain well-known (traditional) BBs, but have much fewer customized structural features needed for high-end analog circuits. Moreover, other aspects specific to analog circuit topologies, like variable number of hierarchical levels, device and sub-structure sharing among BBs, and repetitive structures, are only partially tackled by existing algorithms. These limitations might not only incorrectly or incompletely identify the BBs in a circuit, but also reduce the type of analog circuits that can be tackled by automated tools (e.g., sized, laid out, or generated), hence, increasing the gap to state-of-the art designs.

This paper presents a novel technique to automatically recognize and extract the topological (structural) features in analog circuits. The paper presents algorithms for both supervised feature extraction and unsupervised learning of new connections between known building blocks. In supervised extraction, all structural features to be identified are stored in a library. In the unsupervised mode, new connections among known BBs are learned during the process. Experiments discuss feature extraction for 34 state-of-the-art analog circuits. Finding structural features is important for the correctness and effectiveness of design automation tasks, like circuit synthesis and sizing, design reuse, and design knowledge description and management.

The paper has five sections. Section II presents the motivation for this work. Section III describes the algorithms for topological feature recognition, and Section IV discusses the experimental results. Conclusions end the paper.

II. Motivation

The objective of the work is to identify the invariant topological structures present in analog circuits. This problem introduces elements beyond finding topological (structural) isomorphism in graphs, the problem which has been traditionally associated with identifying BBs in analog circuits [2]. In addition to finding BBs, this section explains that correctly recognizing the topological structures in analog circuits requires tackling the following aspects: (i) finding hierarchical structures, (ii) isolating generic templates (patterns), and (iii) recognizing overlaps among structures, i.e. device sharing between two BBs.

Example: Figure 1(a) shows a circuit with four types of BBs: folded cascode (FCO) (solid boxes), complementary inputs (CI) and cascode current sources (CCS) (dashed boxes), and

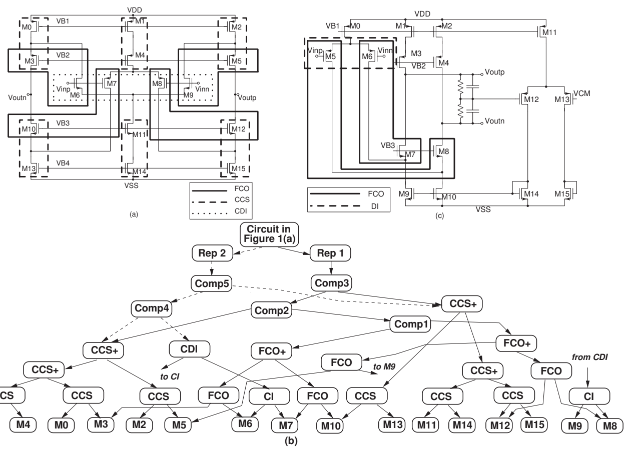

Fig. 1: (a) An analog circuit, (b) the topological structure of the circuit in (a), and (c) a circuit with overlaps

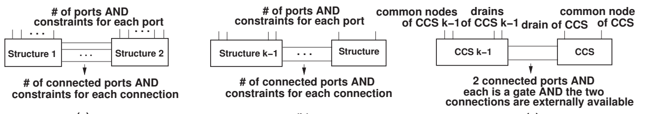
# of ports AND constraints for each port
Structure 1
Structure 2
# of connected ports AND constraints for each connection
(a)
# of ports AND constraints for each port
Structure k−1
Structure
# of connected ports AND constraints for each connection
(b)
common nodes of CCS k−1
drains of CCS k−1
drain of CCS
common node of CCS
CCS k−1
CCS
2 connected ports AND each is a gate AND the two connections are externally available
(c)

Fig. 2: (a) Structure composition, (b) structure composition for generic templates, and (c) generic template for CCS+

complementary differential inputs (CDI) (dotted boxes). The circuit has one CDI ($M6$, $M7$, $M8$ and $M9$), four FCOs, and six CCSs. BB CDI is composed of two CIs. The topological structure of the circuit is presented in Figure 1(b). Note that the circuit has two representations, labeled as $Rep1$ (solid arcs) and $Rep2$ (dashed arcs), due to the overlapping of its topological features, e.g., each CI shares components with two FCOs. The four FCOs share components with CDI and four different CCSs. The hierarchical structure of the circuit, e.g., the nodes labeled $Comp_i$, shows how the building blocks and other structural features are connected together to form the circuit. Generic patterns (templates), like $CCS+$ or $FCO+$, are repetitive structures created by applying the same connection rules. Tackling these issues is beyond finding topological isomorphism.

A. Hierarchical structures: Each analog circuit has a hierarchical structure that presents how the structural features are connected with each other to create the BBs and the circuit. For example, connecting nodes $Comp_i$ in Figure 1(b) describe the hierarchical structure of the circuit. Depending on how the connecting blocks are defined, many distinct hierarchical structures can be found for the same circuit, most of which do not actually offer insight into circuit design. For example, connections can refer to either two, three, etc. BBs that are linked together. Also, the order in which BBs are connected together can be different, hence, producing different hierarchical structures.

The following constraints were defined to avoid the ambiguity of the hierarchical structure of a circuit: (i) connections refer only to two blocks, (ii) features that are identical or have the same structural pattern (hence, form a generic template) are connected first, (iii) features are connected in the order set by the signal flow, (iv) biasing features are added to their corresponding paths after the paths are found, and (v) the final connecting blocks describe how different signal paths [12] are linked to each other.

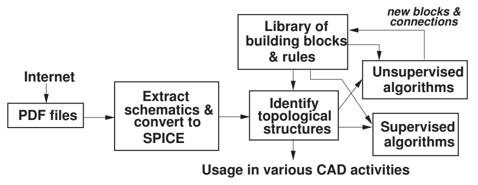
new blocks & connections
Library of building blocks & rules
Internet
Unsupervised algorithms
PDF files
Extract schematics & convert to SPICE
Identify topological structures
Supervised algorithms
Usage in various CAD activities

Fig. 3: Overall flow for topological structure recognition

To address the five constraints, connection blocks for hierarchical structures are described as shown in Figure 2(a). Blocks indicate the nature of the two structures that are linked together, the number and constraints of the ports (e.g., structure nodes) that are connected with each other (i.e. to connect the device gates in CCS), and the number and constraints of the ports (e.g., structure nodes) that are utilized to further connect the produced structure to the rest of the circuit hierarchy.

B. Generic templates (patterns): Generic templates are a special case of hierarchical structures that involve repetitive sub-structures created by induction. Templates either connect two identical topological structures, or connect two structures that follow the same topological pattern (rule). For example, in Figure 1(b), blocks $CCS+$ or $FCO+$ are generic templates. They describe repetitive structures that connect similar blocks.

Figure 2(b) presents the general case of generic templates, which are a special case of the connection blocks in Figure 2(a). Structure $k-1$ has a variable number of ports (e.g., three, four, etc. branches in a $CCS+$), but the constraints on the ports follow the same rules. For example, in Figure 2(c), the two connections between $CCS\ k-1$ and CCS are always gates, and the remaining ports are always as follows: k ports are device drains and k ports are internal nodes that connect a MOSFET drain and a MOSFET source.

C. Overlapping structures: There are three cases in which topological structures in a circuit share devices: (i) devices are shared between two topological features of the same path, like devices $M9$ or $M12$ between BBs CCS and FCO in Figure 1(c). The shared devices do not modify the structure of the circuit, e.g., BBs CCS and FCO are still found. (ii) Devices are shared between different signal paths, like feedforward and feedback paths. (iii) Device sharing that produces different representations of the circuit, e.g., in Figure 1(b). Devices $M6$, $M7$, $M8$, and $M9$ are part of both BBs CI and FCO, thus pertain to both input and output structures. Identifying only blocks $FCO/FCO+$ (or $CCS/CCS+$) generates an complete covering of the circuit, even though it does not find the input BBs, CDI and CI.

III. Algorithms

Figure 3 shows the overall flow for recognizing the topological structure of analog circuits. The flow automatically downloads papers on new circuit designs, after which the figures of the circuits in the PDF files are automatically converted to their Spice descriptions. Spice files are used for topological feature recognition, either through supervised or unsupervised algorithms. The supervised method uses a library of BBs and connection rules, while the unsupervised method also learns new BBs and rules during the recognition procedure.

The algorithm for topological structure recognition performs the following steps: First, it creates a set of all tentative blocks (called set S) that are identified in a circuit, including hierarchical structures and generic templates. Second, **Algorithm** 1 identifies overlapping structure and eliminates redundant structures. The two main steps are discussed next.

Step 1: Finding hierarchical structures and generic templates: Both hierarchical structures and generic templates are handled by similar algorithms. The methods for supervised feature identification consider that BBs and connection rules are fully specified, while the methods for unsupervised identification learn new BBs and connection rules during the identification process.

1. Supervised identification. The algorithms recognize BBs and connection rules based on the observation that context-independent grammars (CIG) [13], a well-known formalism in compiler design, are an appropriate mechanism to specify structures with alternatives, hierarchy, and repetitions. Then, parsing algorithms, like recursive descendant parsing [13], is used to identify the hierarchical structure and generic templates.

Example: A simple differential input stage is expressed by the following three CIG rules:

$AE == NMOS|PMOS$
$INTER == wire|RE|TG$

$DI == (VINN <> AE_1.G, VINP <> AE_2.G, AE_1.S <> AE_2.S)$

The rules specify that BB AE (amplification element) is a PMOS or an NMOS MOSFET. Interconnect (BB $INTER$) is a wire, a resistor (RE), or a transmission gate (TG). A differential input block (DI) includes two amplification elements (AE_i), for which the two gates are connected to V_{in+} ($VINP$) and V_{in-} ($VINN$) and their sources are connected with each other.

The primitive elements include BBs AE, $INTER$, RE, CE (capacitive element), $VSOURCE$ (voltage source), $ISOURCE$ (current source), $INPUT$ (inputs), $OUTPUT$ (outputs), $BIAS$ (bias nodes), and $SUPPLY$ (supply). More complex blocks are built using these BBs and connection rules.

The following two rules describe the hierarchical structure of complementary differential input (CDI) blocks:

$CI == (AE_1.G <> AE_2*.G, INPUT <> AE_1.G)$

$CDI == CI(VINN <> CI_1.G, VINP <> CI_2.G, CI_1.S1 <> CI_2.S1, CI_1.S2 <> CI_2.S2)$

Each block CDI has two blocks CI, denoted as CI_1 and CI_2, with their terminals (ports) connected as described in the list: their gates are connected to $VINN$ (V_{in-}) and $VINP$ (V_{in+}), and their sources are linked together. The rule for block CI indicates that a complementary input (CI) block has one NMOS and one PMOS ("*" indicates the different nature of the devices) with their terminals connected as shown in the rule.

Tackling generic templates (patterns) is a special case of handling hierarchical structures. Generic templates are repetitive structures produced by applying the same rule to sub-structures of increasing size. For example, the next two rules describe the generic templates ($PDI+$) for pseudo-differential input (PDI) blocks with variable number of devices:

$PDI == (VINN <> AE_1.G, VINP <> AE_2.G, AE_1.D <> INTER/OUTPUT, AE_1.S <> SUPPLY/BIAS/INTER, AE_2.D <> INTER/OUTPUT, AE_2.S <> SUPPLY/BIAS/INTER)$

$PDI+ == PDI(PDI_1.G1 <> PDI_2.G1, PDI_1.G2 <> PDI_2.G2)| PDI+ PDI(VINN <> PDI.G1, VINN <> PDI+.G1, VINP <> PDI.G2, VINP <> PDI+.G2)$

The rules state that template $PDI+$ is produced either by connecting two BBs PDI, or by connecting block $PDI+$ with a block PDI. The latter description is recursive in terms of $PDI+$, thus specifying repetitive structures.

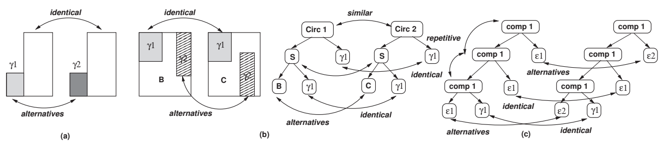

Fig. 4: Unsupervised learning

2. Unsupervised learning. The algorithm learns new sub-structures, like rules to connect known BBs, as they are encountered for new circuits (see Figure 3). As detailed for supervised learning, the rules describing block connections (hierarchical structures) and generic templates are of the following two types: (i) $BB_y == BB_m BB_n(compositionrule...)$ (hierarchical composition of blocks BB_m and BB_n to form new block BB_y) and (ii) $BB_z == BB_z BB_k(compositionrule...)|BB_p BB_q(compositionrule...)$ (generic template BB_z is formed by hierarchically connecting blocks BB_p and BB_q, or by the recursive composition of block BB_z and BB_k). Unsupervised learning learns the rules pertaining to the two types.

Figure 4 illustrates the conceptual mechanism for unsupervised learning of the two types of rules. The procedure clusters circuits based on their maximum topological similarity. For example, in Figure 4, two circuits (shown as shapes) are identical except the smaller rectangles filled with different patterns and marked as γ_1 and γ_2. Assuming that the two circuits have same functionality (i.e. both are OpAmps) then a new rule can be extracted to formulate the structural alternative as using either BB γ_1 or γ_2 in a circuit. The rule has the form $BB_{new} == \gamma_1|\gamma_2$. Rules describing the connection of the devices in γ_1 and γ_2 are also extracted. Figure 4(b) illustrates the learning of new hierarchical rules. For two circuits having the shown structural similarities, their topological structure corresponds to the two trees in the figure. The unsupervised method first associates the identical features, so that the unmatched parts describe alternatives. The rules that emerge are as follows: $Circ == S\gamma_1$, $S == E\gamma_1$, and $E == B|C$. Finally, Figure 4(c) shows the learning of generic templates. As shown in the Figure, repetitive structures are indicated by repeatedly using the same connecting rule ($comp_1$) in the structure. Alternatives are found by associating BBs as described in the previous step. The extracted rules are as follows: $comp_1 == comp_1\epsilon|\epsilon\gamma_1$ and $\epsilon == \epsilon_1|\epsilon_2$.

Algorithm 1: Identifying overlapping blocks

```
Input: Set $S$ of tentative building blocks
Output: Final set $FS$ of building blocks
$FS = \emptyset$;
for each building block $b \in S$, from level $l = L_{max}$ to 1 do
    for each $b \in S$ in decreasing order of its number of BBs do
        $FS = FS \cup \{b\}$;
        $S = S - \{b\}$;
        for each building block $b' \in S$ do
            if $b' \subset b$ then
                $S = S - \{b'\}$;
```

Step 2: Identifying overlapping blocks: Building blocks are identified according to the rule that states that feature recognition should find all, including the largest, topological structures, so that each structure is justified either by the correct circuit operation or by performance improvement.

Algorithm 1 presents the procedure to identify overlapping during BB recognition. The set of tentative BBs (set S found

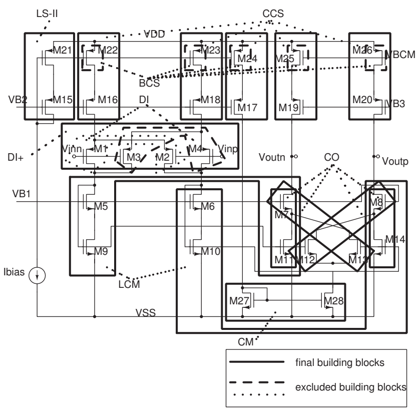

Fig. 5: The building blocks of a complex analog circuit

by Step 1) includes all BBs that can be identified for a circuit. If simple BBs were to be found first then it is possible that larger blocks, which include the simpler blocks, are not found, as the devices of the simpler blocks are assigned only to them. For instance, as block CCS includes block BCS, if the devices forming block BCS are also part of another block, then these devices might be associated only to that block, hence, BCS is not found, and block CCS is not recognized either. To avoid such situations, set S of tentative BBs is examined starting from the higher levels of the hierarchy down to level 1. BBs that are composed of simpler BBs are found first. Device sharing among BBs is allowed.

IV. Experimental Results

The structural features were recognized for a set of 34 modern analog circuits [7]. Experiments show that hierarchical BBs are frequent, as they occur in almost every circuit. The number of levels is mostly two, but it can go up to four levels. There are numerous devices shared by BBs, hence finding overlapping BBs, a feature not present in previous methods, is important. Some BBs are frequent. For example, for biasing: BCS appears 91 times and CCS - 66 times, for current mirrors: CM appears 73 times and LCM - 16 times, for input stage: DI is used 13 times and template $DI+$ - 8 times, for output stage: CO and FCO are used 15 and 17 times, respectively. Hence, circuits tend to include, understandably, simpler BBs instead of more complex BBs. Rare BBs are two types of cross-coupled outputs used to improve linearity, and class AB structures to improve power efficiency. As these structures are more complex, it is unlikely that they are utilized unless linearity or power efficiency is main design requirement. Figure 5 shows a complex circuit with six types of recognized BBs. BBs are highlighted in solid boxes. The BBs that are included in larger, hierarchical BBs are shown in dashed or dotted boxes. The five CCSs are composed of five different BCSs. Also, the two DIs ($M1$, $M2$ and $M3$, $M4$) are components of the generic template $DI+$.

Addressing BB overlapping, unjustified BBs, and generic templates are important in avoiding errors in BB recognition, e.g., finding BBs that are redundant or inexistent, or missing certain BBs. If BB overlapping is not considered then certain BBs are not recognized, e.g., a BB DI was found, but not BB FCO. Also, incorrect BBs may be found, if generic templates are not considered. For instance in Figure 5, two DIs are recognized instead of $DI+$, even though the DIs are part of the same structure. For the set of 34 circuits, the total number of eliminated errors would be as follows: redundant BBs for 15 circuits, unfound BBs for 20 circuits, and incorrect BBs for 14 circuits. Such errors have significant consequences in a synthesis tool. Finding inexistent BBs produces invalid design-related constraints, i.e. for sizing or layout. Finding redundant BBs is likely to waste computing resources as unnecessary constraints are added. Not recognizing all BBs produces sub-optimal results since useful design constraints are missed.

We run an experiment in which the rules for connecting BBs were not provided. Unsupervised learning learned 15 out of the 30 rules needed to connect the BBs of the circuits. Unsupervised learning did not identify sub-structures that occur in a single circuit or include unknown BBs.

The method in [2] finds for the circuit in Figure 1 six cascode pairs and two differential inputs. The proposed technique finds the six cascode pairs as cascode current sources (CCS), and the two differential inputs are combined as a complementary differential input (CDI). It also finds four folded cascode outputs (FCO) which overlap with the other BBs.

V. Conclusions

This paper presents a novel method to automatically recognize and extract the topological (structural) features in analog circuits. The paper presents algorithms for both supervised and unsupervised feature extraction. The method addresses the following main aspects: (i) finding hierarchical structures, (ii) isolating repetitive structures and (iii) recognizing overlapping among BBs. Experiments discuss feature extraction for 34 state-of-the-art analog circuits. Finding structural features is important for the correctness and effectiveness of tasks like circuit synthesis, sizing, and design knowledge description.

References

[1] H. Graeb, et al. "The sizing rules method for analog integrated circuit design", Proc. ICCAD, 2001.
[2] T. Massier, et al. "The sizing rules method for CMOS and bipolar analog integrated circuit synthesis", IEEE TCAD, 12, 2008, 2209-2222.
[3] T. Massier, H. Graeb, and U. Schlichtmann. "Sizing rules for bipolar analog circuit design", Proc. DATE, 2008.
[4] M. Eick and H. Graeb. "Mars: Matching-driven analog sizing." IEEE Trans. CADICS, 8 (2012): 1145-1158.
[5] M. Eick, et al. "Comprehensive generation of hierarchical placement rules for analog integrated circuits", IEEE TCAD, 2, 2011, 180-193.
[6] G. Van der Plas, et al. "AMGIE-A synthesis environment for CMOS analog integrated circuits", IEEE TCAD, 9, 2001, 1037-1058.
[7] F. Jiao, S. Montano, C. Ferent, A. Doboli, S. Doboli, "Analog Circuit Design Knowledge Mining: Discovering Topological Similarities and Uncovering Design Reasoning Strategies", IEEE TCAD, 34(7), 2015, 1045-1059.
[8] T. Sripramong, et al. "The invention of CMOS amplifiers using genetic programming and current-flow analysis", IEEE TCAD, 11, 2002, 1237-1252.
[9] T. McConaghy, et al. "Variation-aware structural synthesis of analog circuits via hierarchical building blocks and structural homotopy", IEEE TCAD, 9, 2009, 1281-1294.
[10] T. McConaghy, et al. "Trustworthy genetic programming-based synthesis of analog circuit topologies using hierarchical domain-specific building blocks", IEEE Trans. Evol. Comp., 4, 2011, 557-570.
[11] A. Das, and R. Vemuri. "Topology synthesis of analog circuits based on adaptively generated building blocks", Proc. DAC, 2008.
[12] Y. Wei, A. Doboli, "Systematic Development of Analog Circuit Structural Macromodels through Behavioral Model Decoupling", Proc. DAC, 2005.
[13] A. Aho, M. Lam and R. Sethi. "Compilers: Principles, Techniques, and Tools", 2006.