Hierarchical Exact Symbolic Analysis of Large Analog Integrated Circuits By Symbolic Stamps

Hui Xu, Guoyong Shi, and Xiaopeng Li
School of Microelectronics, Shanghai Jiao Tong University
Shanghai, China, 200240
e-mail: shiguoyong@ic.sjtu.edu.cn

***Abstract*—Linearized small-signal transistor models share the common circuit structure but may take different parameter values in the ac analysis of an analog circuit simulator. This property can be utilized for symbolic circuit analysis. This paper proposes to use a symbolic stamp for all device models in the same circuit for hierarchical symbolic analysis. Two levels of binary decision diagrams (BDDs) are used for maximum data sharing, one for the symbolic device stamp and the other for modified nodal analysis. The symbolic transadmittances of the device stamp share one BDD for storage saving. The modified nodal analysis (MNA) matrix formulated using symbolic stamp is of much lower dimension, hence it can be solved by a determinant decision diagram (DDD) with significantly reduced complexity. A circuit simulator is implemented based on the proposed partitioning architecture. It is able to analyze an op-amp circuit containing 44 MOS transistors *exactly* for the first time.**

I. Introduction

Symbolic analysis is capable of deriving analytical characterization of circuit behavior in terms of the circuit parameters. Comparing to a numerical simulator such as SPICE [1], symbolic simulation results are more instructive to designers for design space exploration. However, given a certain circuit size, a symbolic circuit analyzer encounters higher computational complexity than a numerical SPICE simulator.

Many traditional symbolic tools use direct enumeration schemes that require exhaustive computational memory and time even for a circuit of moderate size [2]–[5]. To deal with larger analog integrated circuit blocks, either approximation methods [6] or hierarchical methods [7] must be used.

Research on *approximate* symbolic analysis was mainly driven by the complexity of expressions and their interpretability. A number of research papers have addressed the approximate symbolic analysis techniques [6], [8]–[10].

However, if exhaustive symbolic expressions can be generated efficiently, *exact* symbolic analysis should be preferred because of the following reasons. 1) Only exact analytical formulas can provide reliable and accurate numerical results over a large range of frequency band and large scale of element variations without having to make assumptions such as certain element values are small over certain frequency range. 2) The circuit sensitivity analysis in the ac-domain can be performed based on the accurate symbolic expressions [11]–[13]. 3) Although full symbolic expressions are lengthy and not easily interpretable literally, they can be interpreted graphically by using modern graphical tools and interfaces.

The recent advances on using superior data structure such as a binary decision diagram (BDD) [14], together with a properly formulated *implicit* enumeration process, have greatly increased the efficiency of enumeration, making the exact symbolic analysis of much larger analog circuits possible. Research along this line includes an application of BDD to determinant expansion, leading to the DDD (*Determinant Decision Diagram*) algorithm [15], and an application of BDD to spanning tree enumeration, leading to the GPDD (*Graph-Pair Decision Diagram*) algorithm [16]. Both lines of research have extended the *exact* symbolic analysis capability to circuits containing about 20 transistor. Without BDD, one would have to use a mixture of approximation techniques for circuits of such a scale [6].

The BDD-based techniques not only make the enumeration *implicit*, but also improve the post-processing efficiency in many regards. For example, the numerical evaluation becomes very fast thanks to the hash-based data structure employed by a BDD. Also, the symbolic sensitivity can easily be derived even without making any modifications to an established BDD structure [12], [13].

The symbolic circuit analysis methods proposed in the literature in the past 60 years can roughly be categorized in four types. Let $H(s)$ refer to a transfer function.

Categorization of Symbolic Methods:

(*i*) Problem formulation: Using a determinant (*algebraic*) versus using a circuit graph (*topological*).
(*ii*) Symbolic $H(s)$: Exact versus approximate.
(*iii*) Hierarchy: Non-hierarchical versus hierarchical.
(*iv*) Data storage: Non-BDD versus BDD.

A performance comparison of the categorized methods is provided in Fig. 1, where the reference numbers of transistors come from the literature.

This work is devoted to a new methodology for *exactly* analyzing larger analog integrated circuits (exceeding 40 transistors). The new hierarchical scheme to be developed considers using a common symbolic stamp that is shared by all devices in their small-signal models throughout a circuit. Moreover, the BDD data structure is used for achieving the maximum data sharing in the hierarchical analysis.

Section II briefly surveys the existing representative hierarchical analysis methods proposed so far in the literature. In Section III the GPDD technique is reviewed and applied to symbolic stamp construction. The new hierarchical analysis

This research was supported in part by the National Natural Science Foundation of China (Grant No. 60876089) and by an SJTU-Synopsys Joint Research Grant (2010).

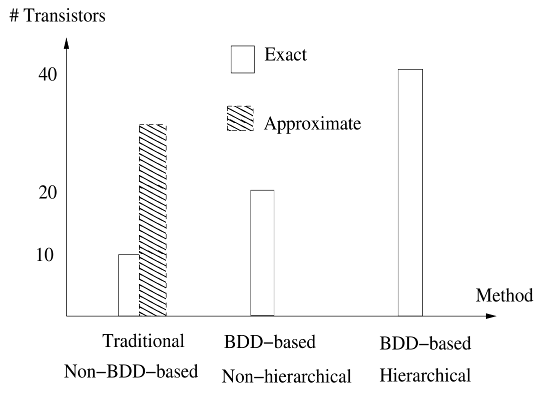


Fig. 1. Performance of the classified methods.

scheme is then presented in Section IV, where the DDD technique is briefly reviewed. Experimental results are reported in Section V. Section VI concludes the paper.

II. Existing Hierarchical Methods

A number of hierarchical symbolic techniques have been proposed in the literature [7], [10], [17]–[21]. All hierarchical methods ultimately derive symbolic formulas in *sequence of expressions* if expanding such expressions is not enforced.

Research by Starzyk and Konczykowska [17] was an early attempt on hierarchical analysis which used the Coates flowgraph representation of determinant. The method by Coates flowgraph decomposition required exhaustive enumeration of multi-connections, whose complexity of analysis would grow beyond linearly for general analog integrated circuits that do not connect in cascaded form.

Hassoun and McCarville [18] used the Mason's signal flowgraph representation of circuit and developed a hierarchical analysis method. Analogously to the work [17], exhaustive enumeration of paths and loops of the divided signal flowgraphs was required. The applications of both [17], [18] were limited to small-scale and loosely connected circuits.

An improved hierarchical method was proposed later by Hassoun and Lin [7] where the technique of circuit partitioning was used. The Gaussian elimination algorithm was used to transform a full modified nodal analysis (MNA) matrix to a so-called *Reduced Modified Nodal Analysis (RMNA)* matrix whose entries appear in sequence-of-expression (SOE) forms. Any two connected circuit blocks can be merged by eliminating the joining internal nodes. The resulting RMNA matrix is actually a symbolic stamp for the resulting merged block with the external ports remaining. Although in principle feasible, the Gaussian elimination procedure makes the intermediate expressions highly nested, meanwhile it introduces numerous *divisions* without any numerical stability control such as *pivoting*. Hence, it is left with the unpredictable numerical stability problem in the succeeding phase of numerical analysis. On the other hand, implementation of a sensitivity analysis would be nontrivial.

Many years later Pierzchala and Rodanski [22] pointed out that it was unnecessary to establish sequence-of-expressions via hierarchical decomposition, rather a direct *symbolic Gaussian elimination* from a full MNA matrix successively to a two-port matrix was sufficient to generate SOE results with equal complexity, which is quasi-linear for loosely connected filter circuits. A local pivoting scheme was used in [22] to reduce the fill-ins and in turn to reduce the number of symbolic operations. The authors of [22] noticed the problem of divisions and proposed to used a post-scaling technique.

Both works [7], [22] only tested the capability of Gaussian elimination by several filter circuit containing ideal op-amps. No experimental results on analyzing practical analog op-amp circuits were reported.

The work by Doboli and Vemuri [20] explored the structural interconnection regularity inherent in many analog circuits and the expression-level regularity that might result from Gaussian elimination of internal nodes when two blocks are interconnected. The proper functioning of the strategy requires a powerful regularity extraction program for which some heuristics are proposed in that paper. If the circuit-level regularity could be extracted efficiently, the symbolic stamp technique to be presented in this work would be applicable in the Doboli and Vemuri's framework as well for hierarchical analog synthesis.

Guerra *et al.* [10] proposed a hierarchical approach to *approximate* symbolic analysis, where circuit reduction and crossing-hierarchy error control technique were loosely discussed. The dominant symbolic terms were generated by a common-tree enumeration process based on the two-graph method and the weighted intersections of matroids [9]. Although the authors of [10] reported experimental results on large analog circuits (one containing 83 MOS transistors), the procedure presented in the paper contained too many roughly stated steps that are not easily reproducible.

The DDD approaches to hierarchical symbolic analysis were addressed in [19], [21], where the key ideas were to make use of the Schur decomposition of an MNA matrix. Hence, these methods can be classified as DDD factorization based on matrix decomposition, whose efficiency improvement is limited in the sense that the procedures are essentially block Gaussian eliminations. No significantly improved experimental circuit sizes were reported in the work along this line.

III. GPDD for Symbolic Stamp

Two BDD-based symbolic techniques have been developed in the literature. The DDD technique [15] represents the determinant expansion by a BDD while the GPDD technique [16] represents the spanning tree enumeration by a BDD. Both DDD and GPDD can exactly analyze analog circuits of comparable sizes up to op-amp circuits containing around 20 transistors. However, it becomes hard to use either of the BDD-based techniques to exactly analyze circuits containing more than 20 transistors. Hence, a hierarchical analysis method is necessary if *exact* analysis of much larger circuits is desired.

The GPDD work presented in [16] reformulates the spanning-tree enumeration in a graph-pair reduction process and manages the subgraph sharing by a BDD. It is essentially an extension of the two-graph method to dealing with

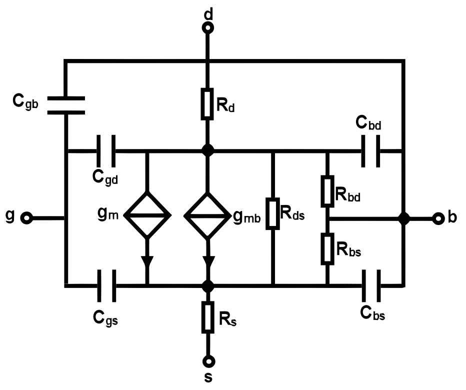


Fig. 2. MOS level 3 small signal model [23].

all four types of dependent sources. This is the first work applying BDD for *implicit* spanning-tree enumeration, making it possible to exactly analyze large analog circuits (up to 20 semiconductor devices).

A distinctive feature of GPDD is its direct one-to-one map between the simulator symbols and circuit parameters, while DDD does not have such a feature because it uses the MNA formulation and the matrix elements (treated as symbols by DDD) are composed of adjacent circuit parameters in general. We would like to construct symbolic device stamps directly in terms of the device parameters for the ease of applications, such as in sensitivity analysis [12]. It motivated us to use the GPDD technique for symbolic stamp construction.

Shown in Fig. 2 is a commonly used MOS small-signal model, which is used in SPICE ac analysis. In symbolic circuit analysis, all semiconductor devices are substituted by their small-signal models and the circuit elements appear as symbols in the analytical network functions. A typical op-amp circuit usually contains 10 to 50 (or more) semiconductor devices. Directly substituting all devices by their small-signal models would quickly increase the circuit scale, making it hard for flat (non-hierarchical) symbolic analysis.

Since the scale of the small-signal circuit as shown in Fig. 2 is small, deriving symbolic transadmittances for such a multiport modular circuit block is not a challenging task for many modern symbolic simulators. By using GPDD, the transadmittances of the four-port stamp for the small-signal model can share one GPDD.

The symbolic stamp of any circuit module can be represented in the admittance matrix form $Y_d v_d = j_d$, where the subscript d indicates the module or device. A commonly used method to determine the admittance matrix Y_d (i.e., the stamp) is to apply a unity voltage at the ith port (between the ith terminal and the ground) while shorting all other terminals to the ground. The currents flowing through all ports (directed from the ground to the terminals) are the ith column of Y_d. Repeating this procedure for all ports solves all columns of Y_d.

An m-port circuit is determined by m^2 such transadmittance analysis. GPDD solves any input-output transfer function as one of the four dependant sources (VCVS, CCCS, VCCS, and CCVS). For any entry y_{ij} of the admittance matrix Y_d, The GRASS simulator uses the CCVS (not VCCS) to model the inverse of y_{ij} (see [16]). Since GRASS uses a decision diagram (called GPDD) for representing one admittance y_{ij}, it is natural to share the m^2 admittances in one GPDD in implementation, which is a meaningful extension to the earlier GPDD work [16].

For a circuit module of the scale as shown in Fig. 2, GRASS can derive each transadmittance in unnoticeable time. Even establishing the whole transadmittance matrix containing 16 transadmittances for such a four-port module does not take appreciable time by the implementation we have developed. Also, the memory consumption is managed at minimum by sharing all m^2 transadmittance in one BDD.

Note that two internal nodes are suppressed if the small-signal module is treated as a single stamp. With over 20 transistors in a circuit, the MNA matrix dimension using the 4×4 stamp for each semiconductor device is significantly less than that of a flat formulation. The computation burden of using a DDD routine to solve a reduced-dimensional MNA matrix is henceforth greatly reduced.

IV. Proposed Hierarchical Scheme

For symbolic ac analysis given a selected pair of input-output, the symbolic device stamps are assembled into an MNA matrix. The linear system $Ax = b$ is then solved by the Determinant Decision Diagram (DDD) routine [15]. At this time, the DDD routine only has to solve a low dimensional linear system with sparsity. Along with the reduced solving complexity, the requirement of finding a good *symbol order* for BDD construction becomes less critical.

Fig. 3 illustrates the hierarchical simulator structure described so far. We see that the GPDD routine runs for multiple times if several modules of different structure have to be analyzed, while the DDD routine runs only once for the assembled MNA matrix. Since both BDD-based routines only need to analyze circuit problems of smaller scale, the overall computation complexity is reduced significantly.

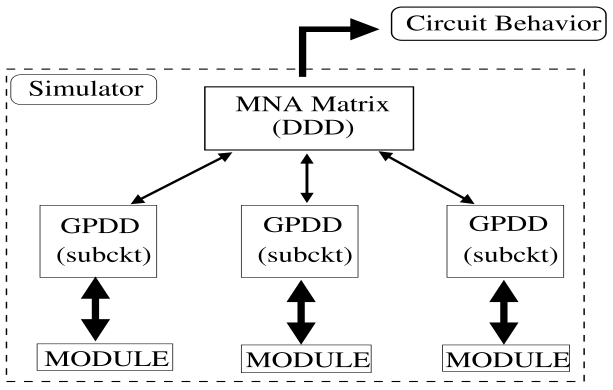
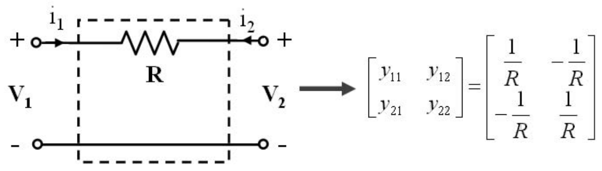


Fig. 3. The hierarchical simulator structure (HybridSim).

Proposed Hierarchical Analysis Procedure:

Step 1. Choose a small-signal model for the transistors (devices) used in the circuit.

Step 2. Run GPDD to construct the multiport symbolic stamp for the device by sharing one BDD.



i_1
i_2
R
V_1
V_2
$\begin{bmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{bmatrix} = \begin{bmatrix} \frac{1}{R} & -\frac{1}{R} \\ -\frac{1}{R} & \frac{1}{R} \end{bmatrix}$

Fig. 4. A two-port resistor block.

Step 3. Formulate MNA matrix and create a device table for stamp evaluation.
Step 4. Run DDD to construct a symbolic transfer function.
Step 5. Run numerical evaluations.

Shi and Tan [15] proposed to use a BDD for data storage in the expansion of a determinant and named the constructed data structure a *determinant decision diagram* (DDD). According to Cramer's rule, a single-input single-output transfer function of a network can be solved as the ratio of two determinants. As long as the determinant expansion can be represented compactly, the symbolic network function can be expressed compactly as well. The key mechanism that brings efficiency is the hash mechanism enforced in BDD that guarantees canonicity (uniqueness of representation). The identical minors that would appear repeatedly in determinant expansion are handled only once by implementing a hash table. This mechanism not only saves the effort of repetitive expansion and redundant memory, but also avoids exhaustive enumeration of all product terms explicitly. Hence, the enumeration time is reduced enormously. Moreover, the numerical evaluation is accelerated substantially as well.

For those who are not familiar with the BDD approaches to symbolic circuit analysis, the example presented in the sequel serves to illustrate the basic steps involved in the hierarchical scheme proposed in this paper. More technical details of the two BDD-based routines can be found in [15], [16].

Given a two-port network, let its transfer admittance matrix be

$$\begin{pmatrix} i_1 \\ i_2 \end{pmatrix} = \begin{pmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}. \tag{1}$$

We need to compute symbolically the four transadmittance functions y_{ij}.

For example, a single resistor can be modeled as a two-port shown in Fig. 4. The port transadmittance matrix is just the nodal analysis stamp shown aside in Fig. 4. Four runs of GPDD can derive the four transadmittances in the 2×2 stamp.

GPDD computes a transadmittance by representing the circuit with specified input and output in a pair of graphs and starts a reduction process according to a set of rules [16].

Since the four transadmittances come from the same network, many subexpressions among them can be shared. Our implementation uses only one hash table (i.e., one GPDD) for storing all transadmittances of a multiport module.

Assembling a number of symbolic stamps into an MNA matrix is illustrated next. Suppose we have a circuit composed of three two-port modules as shown in Fig. 5. Each two-port has its admittance matrix given by

$$\begin{pmatrix} i_1^\alpha \\ i_2^\alpha \end{pmatrix} = \begin{pmatrix} y_{11}^\alpha & y_{12}^\alpha \\ y_{21}^\alpha & y_{22}^\alpha \end{pmatrix} \begin{pmatrix} v_1^\alpha \\ v_2^\alpha \end{pmatrix}. \tag{2}$$

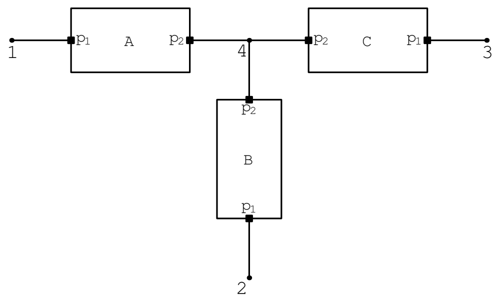


Fig. 5. A network composed by three sub-blocks.

where the superscript 'α' stands for 'A', 'B' or 'C', each labeling one two-port. The MNA matrix assembling the three two-port stamps together becomes the following array

$$\begin{array}{ccccc} & node_1 & node_2 & node_3 & node_4 \\ node_1 & y_{11}^A & & & y_{12}^A \\ node_2 & & y_{11}^B & & y_{12}^B \\ node_3 & & & y_{11}^C & y_{12}^C \\ node_4 & y_{21}^A & y_{21}^B & y_{21}^C & y_{22}^A + y_{22}^B + y_{22}^C \end{array} \tag{3}$$

Given an input source, the output can be solved from the MNA system $Ax = b$ by the DDD routine. Note that the symbols manipulated by DDD are composite expressions as seen from (3). For example, the $(4, 4)$ entry in (3), $y_{22}^A + y_{22}^B + y_{22}^C$, is treated as one independent symbol in DDD.

Remark 1: Since an m-port network is characterized by m^2 transadmittances, to control the number of GPDD runs for computing all m^2 transadmittances, it is recommended to use circuit modules with the number of ports no more than four. This restriction does not cause too much problem for application to analog integrated circuits.

V. Experimental Results

Based on the proposed hierarchical paradigm, a symbolic simulator (code-named *HybridSim*) was implemented in C++. For the purpose of comparison, a simulator based on GPDD (code-named *GRASS* – Graph Reduction Analog Symbolic Simulator [24]) was developed, meanwhile, a DDD simulator using a new implementation method [25] also were constructed. The test run results were collected on an AMD Athlon64 2.20GHz processor with 2GB memory.

Two large benchmark circuits used in this work are:

- Benchmark 1: A two-stage Miller MOSFET amplifier (with a folded-cascode first stage), containing 24 transistors (Fig. 6).
- Benchmark 2: A MOSFET operational amplifier, containing 44 transistors (Fig. 7).

A $0.18\mu m$ model library was used for dc analysis. The level-3 small-signal model (containing 12 symbols) shown in Fig. 2 (see [23]) was used for all MOS transistors in both circuits. In our experiment, each MOS transistor was treated as a four-port module. the GPDD routine derives all the 16 transadmittances for the four-port transistor small-signal module and shares them in one BDD.

The experimental results are summarized in Table I. The column "*#Device*" lists the number of MOS transistors in each

TABLE I
PERFORMANCE OF THE HIERARCHICAL METHOD.

Op-amp circuit	#Device (T)	#Symb for GPDD	#Symb for DDD	MNA Matrix Size	$\|GPDD\|$ (vertices)	$\|DDD\|$ (vertices)	Time (sec.)	Memory (MB)
Benchmark 1	24	12	104	18×18	481	70,129	1.81	70
Benchmark 2	44	12	140	28×28	481	45,716	1.50	91

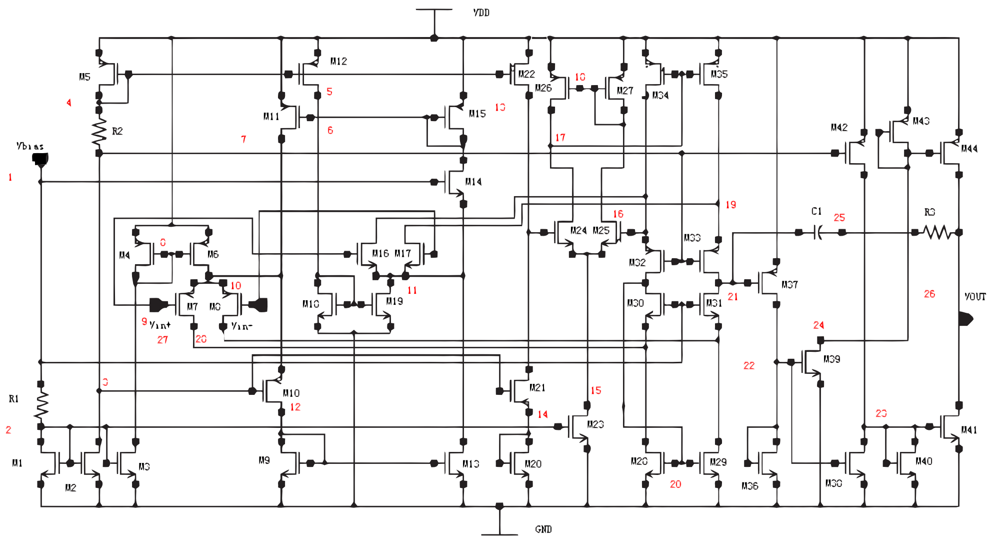


Fig. 7. Benchmark 2: A MOSFET operational amplifier containing 44 transistors [26].

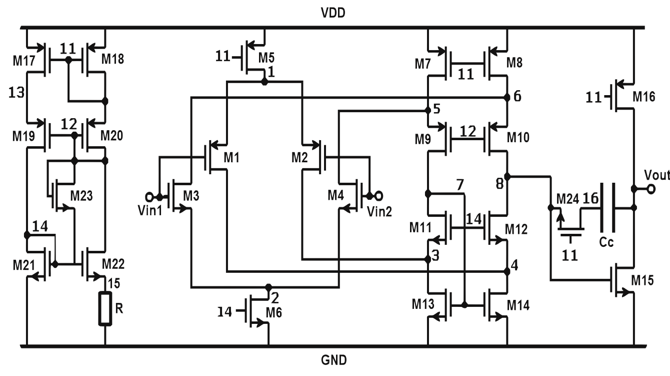


Fig. 6. Benchmark 1: A rail-to-rail Miller MOSFET amplifier containing 24 transistors.

benchmark. The column "*#Symb for GPDD*" lists the total number of symbols coming from the small-signal subcircuit analyzed by GPDD. The small-signal device module has 12 symbols for GPDD, regardless of Benchmarks 1 or 2. The column "*#Symb for DDD*" lists the number of nonzero entries in the MNA matrix analyzed by DDD. The MNA matrix size is listed in the column "*MNA Matrix Size*". The column $|GPDD|$ lists the number of GPDD vertices created for the small-signal module and the column $|DDD|$ lists the number of DDD vertices created for the assembled MNA matrix. The column "*Time*" lists the total time elapsed for analyzing one benchmark circuit, including circuit parsing, two levels of decision diagram construction, and one round of ac analysis with 100 frequency points. We found that the construction time for GPDD and DDD was mild, while the ac evaluation time was relatively more, which depends on the number of frequency points. The column "*Memory*" lists the memory consumption for running one benchmark.

The 16 symbolic transadmittances for the four-port small-signal model were managed to share one GPDD consisting of 481 vertices with an arbitrarily chosen symbol order. All MOS devices appearing in one circuit were evaluated by using the same BDD. The hierarchical treatment results in a 18×18 MNA matrix containing 104 symbols (nonzeros) for Benchmark 1 and a 28×28 MNA matrix containing 140 symbols (nonzeros) for Benchmark 2. Our self-implemented DDD routine [25] used the *Greedy Order* [15] for determinant expansion. The DDD routine created 70,129 vertices for the MNA matrix of Benchmark 1 and 45,716 vertices for the MNA matrix of Benchmark 2, respectively. Note that when the matrix size grows to a certain number, the DDD size $|DDD|$ (total number of BDD vertices) is very sensitive to the symbol ordering and the matrix sparsity pattern. Therefore, a larger matrix size ending up with a smaller DDD size, as seen in Table I, often means that the running symbol order was good.

Solving the two benchmark circuits by our non-hierarchical DDD and non-hierarchical GPDD simulators (GRASS) failed due to memory exhaustion. The scale of the two benchmark circuits exceeded the capability of all existing non-hierarchical symbolic programs, even by using the most powerful BDD technique with good ordering heuristics. Clearly, our proposed hierarchical method can effectively suppress the symbolic analysis complexity by maximally exploiting the existing structural repetitions in analog integrated circuits.

It was observed that the proposed hierarchical analysis method was less sensitive to the number of transistor devices. The two benchmark circuits have quite different number of devices, but their analysis times and memory costs do not differ vastly as seen from Table I.

Finally, a comparison of the performances of the published works on hierarchical symbolic analysis is listed in Table II, where the maximum circuit sizes are cited from the publications. Except for the work [10] which addresses *approximate* hierarchical analysis, our work is able to solve *exactly* a large circuit containing the largest number of MOS transistors for the first time.

TABLE II
PERFORMANCE COMPARISON OF SOME REPRESENTATIVE HIERARCHICAL METHODS.

Publication	Max Ckt Size (#Transistors)	Method	Accuracy
[19]	20 (BJT)	DDD + Schur decomp.	Exact
[20]	26 (BJT)	Regularity + Sharing	Exact
[10]	83 (MOS)	Ckt reduction + Two-graph	Approx.
[21]	26 (BJT)	DDD + De-cancellation	Exact
This work	44 (MOS)	DDD + GPDD	Exact

VI. Conclusion

A novel hierarchical symbolic circuit analysis method is presented. Differently from the existing hierarchical approaches, the proposed method considers the modular duplicative circuit structures existing in analog integrated circuits and introduces the notion of symbolic device/module stamp for hierarchical analysis. The two previously published BDD-based symbolic analysis tools, GPDD and DDD, are utilized as a hierarchical data structure for representing a symbolic network function, aiming at the maximal data sharing and ease of data manipulation. The remarkable efficiency and capacity improvement have been observed in the reported experiments. A meaningful subsequent research work would be to extend the proposed data structure to the sensitivity analysis to the semiconductor device parameters.

References

[1] L. W. Nagel, "SPICE2: A computer program to simulate semicoductor circuits," Ph.D. dissertation, University of California, Berkeley, CA, May 1975.

[2] P. Lin, *Symbolic Network Analysis*. New York: Elsevier, 1991.

[3] G. Gielen and W. Sansen, *Symbolic Analysis for Automated Design of Analog Integrated Circuits*. Norwell, MA: Kluwer Academic Publishers, 1991.

[4] F. Fernández, A. Rodríguez-Vázquez, J. Huertas, and G. Gielen, *Symbolic Analysis Techniques – Applications to Analog Design Automation*. New York: IEEE Press, 1998.

[5] P. Wambacq, G. E. Gielen, and W. Sansen, "Symbolic network analysis methods for practical analog integrated circuits: a survey," *IEEE Trans. on Circuits and Systems – II: Analog and Digital Signal Processing*, vol. 45, no. 10, pp. 1331–1341, 1998.

[6] Q. Yu and C. Sechen, "A unified approach to the approximate symbolic analysis of large analog integrated circuits," *IEEE Trans. on Circuits and Systems – I: Fundamental Theory and Applications*, vol. 43, no. 8, pp. 656–669, 1996.

[7] M. M. Hassoun and P. M. Lin, "A hierarchical network approach to symbolic analysis of large-scale networks," *IEEE Trans. on Circuits and Systems – I: Fundamental Theory and Applications*, vol. 42, no. 2, pp. 201–211, 1995.

[8] P. Wambacq, R. Fernández, G. E. Gielen, W. Sansen, and A. Rodriguez-Vźquez, "Efficient symbolic computation of approximated small-signal characteristics," *IEEE J. Solid-State Circuit*, vol. 30, no. 3, pp. 327–330, 1995.

[9] ——, "A family of matroid intersection algorithms for the computation of approximated symbolic network functions," in *Proc. Int'l Symposium on Circuits and Systems*, 1996, pp. 806–809.

[10] O. Guerra, E. Roca, F. V. Fernández, and A. Rodríguez-Vázquez, "Approximate symbolic analysis of hierarchically decomposed analog circuits," *Analog Integrated Circuits and Signal Processing*, vol. 31, pp. 131–145, 2002.

[11] J. Y. Lee, X. Huang, and R. A. Rohrer, "Pole and zero sensitivity calculation in asymptotic waveform evaluation," *IEEE Trans. on Computer-Aided Design*, vol. 11, no. 5, pp. 586–597, May 1992.

[12] G. Shi and X. Meng, "Variational analog integrated circuit design by symbolic sensitivity analysis," in *Proc. International Symposium on Circuits and Systems (ISCAS)*, Taiwan, China, May 2009, pp. 3002–3005.

[13] D. Ma, G. Shi, and A. Lee, "A design platform for analog device size sensitivity analysis and visualization," in *Proc. Asia Pacific Conference on Circuits and Systems (APCCAS)*, Malaysia, Dec. 2010, to appear.

[14] R. E. Bryant, "Graph-based algorithms for boolean function manipulation," *IEEE Trans. on Computers*, vol. C-35, no. 8, pp. 677–691, 1986.

[15] C.-J. R. Shi and X. D. Tan, "Canonical symbolic analysis of large analog circuits with determinant decision diagrams," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, no. 1, pp. 1–18, January 2000.

[16] G. Shi, W. Chen, and C.-J. R. Shi, "A graph reduction approach to symbolic circuit analysis," in *Proc. Asia South-Pacific Design Automation Conference (ASPDAC)*, Yokohama, Japan, Jan. 2007, pp. 197–202.

[17] J. A. Starzyk and A. Konczykowska, "Flow graph analysis of large electronic networks," *IEEE Trans. on Circuits and Systems*, vol. CAS-33, no. 3, pp. 302–315, 1986.

[18] M. M. Hassoun and K. McCarville, "Symbolic analysis of large-scale networks using a hierarchical signal flowgraph approach," *J. Analog VLSI Signal Processing*, vol. 3, pp. 31–42, Jan. 1993.

[19] X. D. Tan and C.-J. R. Shi, "Hierarchical symbolic analysis of analog integrated circuits via determinant decision diagrams," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, no. 4, pp. 401–412, April 2000.

[20] A. Doboli and R. Vemuri, "A regularity-based hierarchical symbolic analysis methods for large-scale analog networks," *IEEE Trans. on Circuits and Systems – II: Analog and Digital Signal Processing*, vol. CAS-48, no. 11, pp. 1054–1068, 2001.

[21] S. X. D. Tan, W. Guo, and Z. Qi, "Hierarchical approach to exact symbolic analysis of large analog circuits," in *Proc. Design Automation Conference*, 2004, pp. 860–863.

[22] M. Pierzchala and B. Rodanski, "Generation of sequential symbolic network functions for large-scale networks by circuit reduction to a two-port," *IEEE Trans. on Circuits and Systems-I: Fundamental Theory and Applications*, vol. 48, no. 7, pp. 906–909, 2001.

[23] A. Vladimirescu and S. Liu, "The simulation of MOS integrated circuits using SPICE2," EECS Department, University of California, Berkeley, Tech. Rep. UCB/ERL M80/7, 1980. [Online]. Available: http://www.eecs.berkeley.edu/Pubs/TechRpts/1980/9610.html

[24] W. Chen and G. Shi, "Implementation of a symbolic circuit simulator for topological network analysis," in *Proc. Asia Pacific Conference on Circuits and Systems (APCCAS)*, Singapore, Dec. 2006, pp. 1327–1331.

[25] G. Shi, "A simple implementation of determinant decision diagram," in *Proc. International Conf. on Computer-Aided Design (ICCAD)*, San Jose, CA, USA, Nov. 2010, to appear.

[26] T. McConaghy and G. G. E. Gielen, "Globally reliable variation-aware sizing of analog integrated circuits via response surfaces and structural homotopy," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 11, pp. 1627–1640, Nov. 2009.