Analog Testability Analysis by Determinant-Decision-Diagrams based Symbolic Analysis *

Tao Pi

Department of Electrical Engineering
University of Washington
Seattle, WA 98195-2500
e-mail: tpi@ee.washington.edu

C.-J. Richard Shi

Department of Electrical Engineering
University of Washington
Seattle, WA 98195-2500
e-mail: cjshi@ee.washington.edu

Abstract: The use of the column-rank of the sensitivity matrix as a testability measure for parametric faults in linear analog circuits was pioneered by Saeks in 1970s, and later re-discovered by several others. Its practical use has been, however, limited by how it can be calculated. Numerical algorithms suffer from inevitable round-off errors, while classical symbolic techniques can only handle very small circuits. In this paper, an innovative and efficient graph based symbolic analysis approach, called Determinant Decision Diagrams, is applied to testability measurement and selection for optimum test vectors. The new approach is promising in testability analysis of much larger analog circuits.

I. INTRODUCTION

Testing large analog and mixed-signal circuits efficiently with an optimal set of test points or test vectors has become very critical with the fast development of technology. Especially when large mixed-signal systems containing millions of transistors are integrated on a single chip, IC testing is even more difficult and the challenge becomes finding a way to reduce the testing cost.

Testability of a circuit provides the network element solvability information, which allows us to know how many internal system parameters can be uniquely determined or isolated by measuring certain I/O relations of the system. This information is very useful for both design and test engineers. This paper presents a very efficient symbolic analysis approach for quantitative testability measurement for large linear analog circuits.

Testability measurement of a circuit, defined as a measure of the solvability of fault diagnosis equations, usually involves the calculation of network transfer functions and the sensitivity matrix [1][7][8]. Symbolic analysis approach offers great advantages over a numerical method in this application, where the unavoidable truncation errors introduced by the latter one renders the obtained testability only an estimate [9]. Symbolic analysis derives network transfer functions in terms of symbolic parameters. With the circuit behavior represented in a closed analytic form, symbolic analysis not only eliminates the problem of round off errors but also provides insight into the circuit behavior. Unfortunately with the conventional symbolic methods, the manipulation and evaluation of symbolic expressions are computationally hard because the number of product terms in the symbolic expression "blow up" exponentially with the size of a circuit. Any evaluation will have both the time and space complexities exponential to the size of the circuit, which limits the applicability of symbolic analysis in the design and testing areas for large analog and mixed-signal circuits.

In this paper, a new symbolic analysis approach based on Determinant Decision Diagrams (DDDs) [3] is applied to testability analysis of linear analog systems. Using DDDs, exact network transfer functions, as well as the sensitivity matrix, can be constructed in a compact and canonical way. Since DDDs-based symbolic analysis has the time complexity proportional to the size of a DDD, and for a practical circuit, the number of DDD vertices is usually orders of magnitude less than the number of product terms, the new method offers a significant improvement over existing methods for testability analysis.

II. TESTABILITY MEASUREMENT AND A REFORMULATION

In this section, the theoretical background for testability analysis will be discussed briefly. The most useful definition for testability was introduced by Saeks et. al.[1][2], which provides a well-defined quantitative index of element value solvability for linear analog systems. For a linear analog circuit, the network transfer

*This work was sponsored by U. S. Defense Advanced Research Projects Agency (DARPA) under grant number F33615-96-1-5601 from the United States Air Force Research Laboratory, Manufacturing Technology Directorate, by Rockwell Semiconductor Systems and by NSF/Industry Center for Design of Analog-Digital Integrated Circuits (CDADIC).

function can be written in the following non-normalized form:

$$H(s,p)=\frac{\sum_{i=0}^{n} a_i(p)s^i}{\sum_{j=0}^{m} b_j(p)s^j}=\frac{N(p,s)}{D(p,s)} \qquad (1)$$

where $p=[p_1,p_2,...,p_k]^t$ is the vector of the unknown circuit parameters or potentially faulty parameters, and a_i, b_j represent, respectively, the coefficients for different power of s in the numerator and the denominator. The sensitivity matrix $\Phi(s)$ as shown in (2) is a Jacobian matrix of partial derivatives of the transfer functions with respect to the faulty parameters for different test points.

$$\phi(s)=\begin{bmatrix} \frac{\partial H_1}{\partial p_1} & \frac{\partial H_1}{\partial p_2} & \cdots & \frac{\partial H_1}{\partial p_k} \\ \frac{\partial H_2}{\partial p_1} & \frac{\partial H_2}{\partial p_2} & \cdots & \frac{\partial H_2}{\partial p_k} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial H_m}{\partial p_1} & \frac{\partial H_m}{\partial p_2} & \cdots & \frac{\partial H_m}{\partial p_k} \end{bmatrix} \qquad (2)$$

The testability T of the system is equal to the maximum number of linearly independent columns or equivalently the column-rank of the sensitivity matrix $\Phi(s)$, with the circuit parameters around the nominal values.

$$T = column-rank[\phi(s)]$$

Furthermore, it is proven that if the network transfer functions are rational in circuit parameters p, the column-rank of $\Phi(s)$ is constant "almost everywhere" [1]. In other words, the testability of the system is almost independent of the circuit parameter values and can be evaluated by simply assigning arbitrary integer values to the circuit parameters.

Based on this testability definition, a simple procedure for circuit testability evaluation has been developed recently [5][6][8]. It is proven that the column-rank of the network sensitivity matrix $\Phi(s)$ coincides with the rank of a matrix B_c, where B_c is defined as in (3) for one transfer function. In multiple inputs and outputs case, B_c should include the same entries for each transfer function.

$$B_c=\begin{bmatrix} \frac{\partial a_n}{\partial p_1}b_m-\frac{\partial b_m}{\partial p_1}a_n & \frac{\partial a_n}{\partial p_2}b_m-\frac{\partial b_m}{\partial p_2}a_n & \cdots & \frac{\partial a_n}{\partial p_k}b_m-\frac{\partial b_m}{\partial p_k}a_n \\ \frac{\partial a_{n-1}}{\partial p_1}b_m-\frac{\partial b_m}{\partial p_1}a_{n-1} & \frac{\partial a_{n-1}}{\partial p_2}b_m-\frac{\partial b_m}{\partial p_2}a_{n-1} & \cdots & \frac{\partial a_{n-1}}{\partial p_k}b_m-\frac{\partial b_m}{\partial p_k}a_{n-1} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial a_0}{\partial p_1}b_m-\frac{\partial b_m}{\partial p_1}a_0 & \frac{\partial a_0}{\partial p_2}b_m-\frac{\partial b_m}{\partial p_2}a_0 & \cdots & \frac{\partial a_0}{\partial p_k}b_m-\frac{\partial b_m}{\partial p_k}a_0 \\ \frac{\partial b_{m-1}}{\partial p_1}b_m-\frac{\partial b_m}{\partial p_1}b_{m-1} & \frac{\partial b_{m-1}}{\partial p_2}b_m-\frac{\partial b_m}{\partial p_2}b_{m-1} & \cdots & \frac{\partial b_{m-1}}{\partial p_k}b_m-\frac{\partial b_m}{\partial p_k}b_{m-1} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial b_0}{\partial p_1}b_m-\frac{\partial b_m}{\partial p_1}b_0 & \frac{\partial b_0}{\partial p_2}b_m-\frac{\partial b_m}{\partial p_2}b_0 & \cdots & \frac{\partial b_0}{\partial p_k}b_m-\frac{\partial b_m}{\partial p_k}b_0 \end{bmatrix} \qquad (3)$$

By using this new method, the computation of the derivatives of the transfer functions, which is much more difficult, is avoided. In addition, since the testability value is almost independent of the parameter values, the rank of B_c can be easily calculated by using the traditional triangularization method by assigning arbitrary integer value for each unknown circuit parameter. It should be pointed out that this approach is only valid for parametric faults, which are defined as the deviation of the system parameters from their normal values, the catastrophic faults such as opens and shorts, which change the circuit topology, are not considered here.

In the rest of this paper, we will use a simple example to illustrate how Determinant-Decision-Diagrams based symbolic analysis approach can be applied to analog testability analysis. The example circuit is shown in Fig. 1. We assume three possible test points, which are, v_1, v_2 and v_3, and that six components are all potentially faulty.

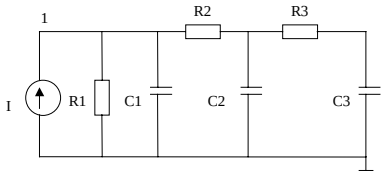

Fig. 1. An example circuit.

III. DDDs-based Transfer Function Representation

First, we introduce the concept of Determinant Decision Diagrams (DDDs). Formally, a Determinant Decision Diagram (DDD) is a signed rooted directed acyclic graph with two terminal vertices, called the *0-terminal* vertex and the *1-terminal* vertex. Each non-terminal vertex *D*, labeled as a symbol *D.label*, has a sign denoted as *D.sign*, which is either positive or negative. Each vertex *originates* two outgoing edges, namely *1-edge* and *0-edge*, pointing respectively to its two children vertices denoted as *D.left* and *D.right*. *D* is called the parent vertex. For each vertex, there is also an integer number denoted as *D.index* associated with it. This index number indicates the ordering of the set of vertices in the DDD graph. One basic rule is that the index number for the parent vertex is always larger than that of its two children vertices. A DDD graph with the root vertex *D* represents a symbolic expression *D.expr* defined recursively as follows:

1. if *D* is the *1-terminal* vertex, then *D.expr* = 1,
2. if *D* is the *0-terminal* vertex, then *D.expr* = 0,
3. if *D* is a non-terminal vertex, then

*D.expr = D.sign*D.label*(D.left).expr+(D.right).expr*

Furthermore, a path from the vertex to the *1-terminal* is called as a *1-path*. Each *1-path* in the DDD graph represents one product term, which is defined as a product

of symbols and signs of those vertices that originate all the *1-edge* along the path. The summation of all *1-paths* is exactly the symbolic expression the root vertex represents. This new graph-based data structure enables compact representation for large symbolic expressions. It exploits the sharing of the equivalent sub-graphs. The manipulation and evaluation of DDD graph have time complexities proportional to the size of the graph - the number of DDD vertices, which is usually orders of magnitude less than the number of product terms in the expression. Symbolic analysis can be performed by efficient graph-based algorithm.

Now we illustrate how to apply DDDs to the problem of testability analysis for the example circuit in Fig. 1. Based on the nodal formulation, the system equations of the circuit can be written as:

$$\begin{bmatrix} \frac{1}{R_1}+sC_1+\frac{1}{R_2} & -\frac{1}{R_2} & 0 \\ -\frac{1}{R_2} & \frac{1}{R_2}+sC_2+\frac{1}{R_3} & -\frac{1}{R_3} \\ 0 & -\frac{1}{R_3} & \frac{1}{R_3}+sC_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = \begin{bmatrix} I \\ 0 \\ 0 \end{bmatrix}$$

The circuit matrix can be rewritten as:

$$A = \begin{bmatrix} a+b+cs & d & 0 \\ e & f+g+hs & i \\ 0 & j & k+ls \end{bmatrix}$$

where $a=\frac{1}{R_1}, b=f=\frac{1}{R_2}, d=e=-\frac{1}{R_2}, g=k=\frac{1}{R_3}, i=j=-\frac{1}{R_3}$, $c=C_1, h=C_2, l=C_3$, represent the unknown or potentially faulty circuit parameters. According to Cramer's rule, the network transfer functions at three test points of the circuit can be written as shown in (4), where p is the unknown parameters, a_{11}, a_{12}, and a_{13} are the entries in circuit matrix, det(A) represents the determinant of circuit matrix A.

$$H_1(p,s)=\frac{V_1}{I}=\frac{Cofactor(A,a_{11})}{\det(A)}$$
$$H_2(p,s)=\frac{V_2}{I}=\frac{Cofactor(A,a_{12})}{\det(A)} \qquad (4)$$
$$H_3(p,s)=\frac{V_3}{I}=\frac{Cofactor(A,a_{13})}{\det(A)}$$

Based on the definition of DDD, a DDD graph can be constructed to represent the transfer functions in a very compact form. Since the time complexity of DDDs-based symbolic analysis is proportional to the size of the DDDs, an efficient vertex-ordering heuristic has been developed to label entries in the circuit matrix so that the resulting DDDs has as few vertices as possible [3]. For this example, the computed vertex ordering is *a+b+cs, e, d, f+g+hs, j, i,*and *k+ls*. The recursive procedure for expanding the circuit matrix to construct DDDs is shown in Fig. 2. The resulting DDDs representing the determinant and cofactors of the circuit matrix *A* required by the three transfer functions is shown in Fig. 3. Since the complex frequency variable *s* is implicitly contained in some DDD labels, we refer to it as *complex* DDDs. Observe that each *1-path* in the graph corresponds exactly to one complex product term in the transfer function expressions.

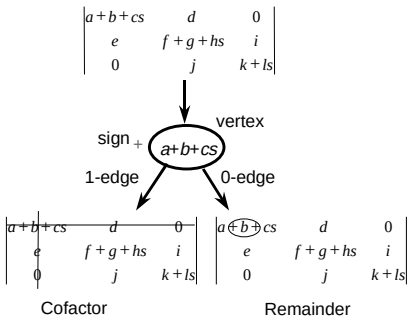

Fig. 2. Graphical representation of matrix expansion.

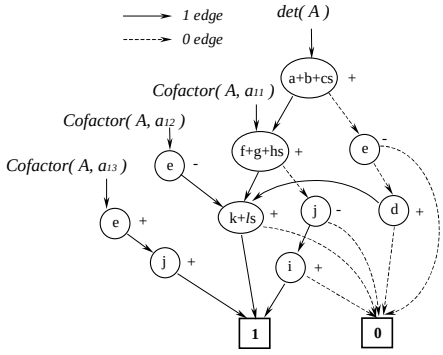

Fig. 3. Complex DDDs for network transfer functions.

IV. S-EXPANDED TRANSFER FUNCTIONS IN DDDS

In order to analyze the testability of the circuit, we need to construct testability matrix B_c, which is made of the coefficients and the derivatives of each coefficient of the transfer functions with respect to the unknown parameters. Therefore, it is necessary to derive the *s*-expanded polynomial expressions for the network transfer functions. It turns out that by using DDD and the related graph-based operations, the exact polynomial symbolic expressions can be represented by the multi-root *s*-expanded Determinant Decision Diagrams (*s*-expanded DDDs). A very efficient algorithm [4] has been developed to recursively construct *s*-expanded DDDs from the complex DDDs. It has the time complexity proportional to the size of the complex DDDs and the highest power of *s* in the *s*-expanded expression. The resulting multi-root *s*-expanded DDDs for the transfer functions are shown in

Fig. 4. Each DDD root vertex represents the symbolic expression for the coefficient of a particular power of *s* in the transfer functions, for example, a_0^1 represents the coefficient a_0 in the numerator for the transfer function H_1 and *b* represents the coefficients in the denominator. We refer to it as coefficient DDDs. Note that coefficient DDDs exploits the sharing among various coefficients of *s* polynomials in the numerators and the denominators of transfer functions.

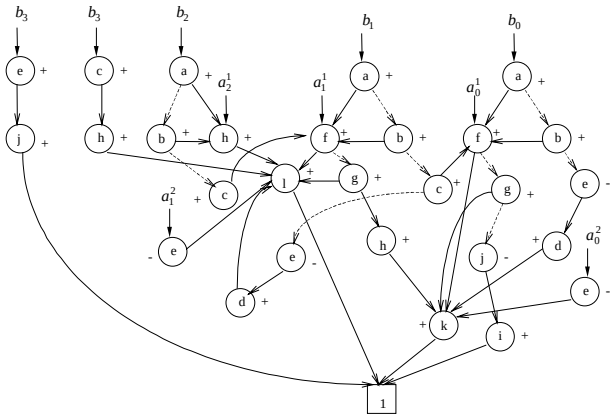

Fig. 4. *S*-expanded coefficient DDDs for network transfer functions.

V. CIRCUIT TESTABILITY MATRIX IN DDDS

To complete the construction of testability matrix B_c, it is necessary to calculate the partial derivatives of the coefficients with respect to circuit parameters. Fig. 5 describes an algorithm called **DERIVATIVE_DDD_CONSTRUCTION** for computing all the partial derivatives. We assume that the total number of coefficients in the denominator and numerator of the transfer function is *n*, and *Qi, i=1, …, n,* represent the *n* coefficient DDD vertices that represent these *n* coefficients. We assume that unknown parameters are $p_1 \sim p_k$ where each p_j is the label of a DDD vertex that represents a faulty circuit parameter. For simplicity, a DDD label is implemented as an integer index in such a way that the index of a parent vertex is greater than that of its children vertices. *DERIV[i][j]* stores the resulting derivative DDD representing the partial derivative of the *ith* coefficient with respect to parameter p_j. The core operation is **COFACTOR***(Qi, p_j)* which calculates the derivative of a DDD vertex with respect to parameter p_j [3]. In Fig. 6, **GET_VERTEX** (*sign, symbol, D0, D1*) returns a DDD vertex with *sign* as its sign, *symbol* as its label, and the *0-edge* pointing to *D0*, and the *1-edge* pointing to *D1*. The resulting derivative DDDs are cached. If the derivative of *Qi* with respect to some specific parameter is needed again, no construction will be performed. Clearly, the time complexity of algorithm **DERIVATIVE_DDD_CONSTRUCTION** is O(*n*|DDD|), where |DDD| is the number of vertices in the coefficient DDDs.

```
DERIVATIVE_DDD_CONSTRUCTION {
    for (i = 0; i< n; i++){
        for ( j = 0; j < k; j++)
            DERIV[i][j] = COFACTOR(Qi, pj);
        }
}
```

Fig. 5. Derivative DDD construction algorithm.

```
COFACTOR (Q, pj) {
        If ( Q.index < pj)
            return 0;
        If ( Q.index = = pj)
            return Q.left;
        If (Q.index > pj)
            derivative = GET_VERTEX (Q.sign,
            Q.label, COFACTOR (Q.left, pj ),
            COFACTOR ( Q.right, pj) );
        return derivative;

    }
```

Fig. 6. **COFACTOR** operation.

For the example circuit shown in Fig. 1, circuit parameters are $p_1 = a, p_2 = c,\ p_3 = -d, -e, b, f, p_4 = h,$ $p_5 = g, k, -i, -j,\ p_6 = l$. Fig.7 shows the multi-root derivative DDDs for coefficient b_0 with respect to different faulty parameters. Observe that the sharing among derivative DDDs and coefficient DDDs is obvious. It is this inherent sharing property of DDDs that enables very efficient and compact representation for the testability matrix.

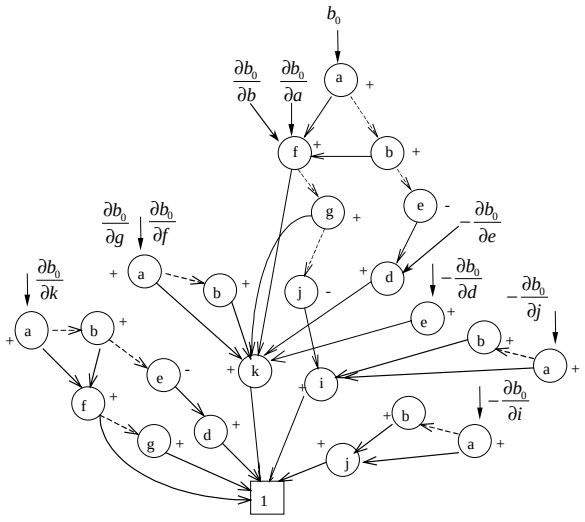

Fig. 7. Multi-root derivative DDDs for b_0 with respect to unknown parameters.

VI. TESTABILITY EVALUATION AND TEST-POINT SELECTION

As discussed in section II, the testability of a circuit is almost independent of the component values. Therefore, a numerical matrix B_c' as shown in (5) is constructed from the testability matrix B_c by simply assigning arbitrary integer value to each parameter. The rank of B_c' is equal to the testability value *T*. In this case, $T = rank\,[\,B_c'\,] = 6$, which indicates that all six parameters in the example circuit can be uniquely determined with the measurements at those three test points.

$$B_c' = \left[\begin{array}{cccccc} 0 & -1 & 1 & -1 & 1 & -1 \\ \hline 0 & -1 & 1 & -1 & 0 & 0 \\ 0 & -1 & 1 & -1 & 1 & -1 \\ \hline 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & -3 & 1 & -2 & 2 & -1 \\ 0 & -1 & 1 & -1 & 1 & -1 \\ \hline 1 & -2 & 2 & -2 & 2 & -1 \\ 3 & -5 & 4 & -4 & 5 & -3 \\ 1 & -1 & 1 & -1 & 1 & -1 \end{array}\right] \begin{array}{l} (v_3) \\ \\ \\ (v_2) \\ \\ \\ (v_1) \\ \\ \\ \end{array} \quad (5)$$

A set of circuits including the example circuit ***rc***, a RLC network ***rlctest***, a two-stage miller compensated MOS opamp ***miller*** [10] shown in Fig. 8 and the Tow-Thomas active RC filter ***ttfilter*** [7] shown in Fig. 9, have been tested using the proposed algorithm. All the parameters in the circuits are considered potentially faulty. The experimental results are reported in Table 1.

In Table 1, column 1 lists, for each circuit, the number of circuit nodes *#node* and the number of circuit parameters *#para*. Column 2 gives three possible test point combinations for each circuit. Column 3 ~ column 5 list the statistics for complex DDDs, coefficient DDDs and derivative DDDs respectively, where *#node* is the number of DDD vertices and *#path* is the number of *1-paths* in the graph. *Total |DDD|* column shows the total number of vertices in the coefficient DDDs and derivative DDDs representing the testability matrix. *T* column lists the testability value obtained from evaluating the rank of the testability matrix. The last column reports the total CPU time in seconds on a 450MHz Intel Pentium-II workstation for testability analysis for each circuit.

We observe from Table 1 that fully expanded coefficient DDDs and the derivative DDDs of the coefficients are able to represent huge symbolic expressions with a relatively small number of vertices. Especially for large circuits, the size of the DDDs could be several orders of magnitude smaller than the numbers of product terms, which makes DDD-based symbolic analysis especially suitable for testability analysis of large analog circuits. Looking into the total number of DDD vertices *Total |DDD|*, it's less than the sum of coefficient DDDs *#node* and derivative DDDs *#node*, which indicates the sharing property of the graph-based data structure.

The testability information of different test points for each circuit provides quantitative criteria for optimum test point selection. For example, for circuit ***rc***, to ensure the maximum testability value 6, that is, we want to determine all six parameters, the test points should be chosen as v1, v2 and v3. This information is very important to designers who must know which nodes to make accessible for testing, and to test engineers who must plan tests and know how many components can be uniquely isolated by these tests [5].

Table 1. Experimental results

circuit	**Test points**	**complex |DDD|**		**coefficent |DDD|**		**derivative |DDD|**		**Total |DDD|**	**T**	**CPU (s)**
		# node	# path	# node	# path	# node	# path	# node		
rc	v3	9	4	25	24	49	71	52	4	0.01
#node: 8	v3,v2	10	5	27	26	50	75	54	5	0.02
#para: 6	v3,v2,v1	10	5	27	33	50	89	54	6	0.02
rlctest	v9	135	656	344	83760	3970	670080	3856	6	0.33
#node: 10	v3,v7,v10	179	784	439	98320	5263	786560	5115	11	0.45
#para: 21	v2,v4,v6, v8	127	704	318	167200	4079	1.337e+6	3977	8	0.53
ttfilter	v7	44	54	163	17638	1120	100164	1107	5	0.1
#node: 8	v3,v5,v7	56	64	208	19052	1398	108408	1379	7	0.14
#para: 8	v2,v3,v5, v7	57	72	211	21216	1432	121152	1415	7	0.16
miller	v5	10	9	47	84	90	252	105	5	0.02
#node: 5	v5,v4	12	10	57	96	100	288	119	8	0.03
#para: 14	v5,v4,v3	13	13	63	124	111	372	135	12	0.04

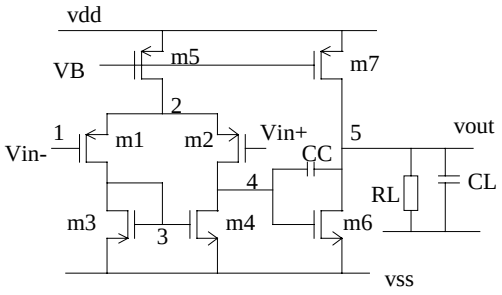

Fig. 8. Two-stage miller compensated MOS opamp.

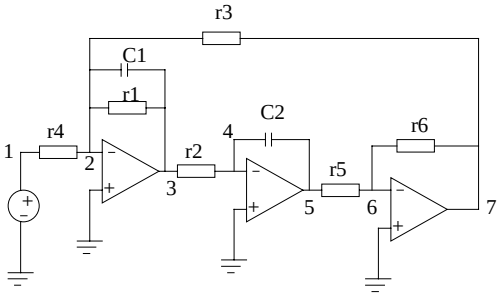

Fig. 9. Tow-Thomas filter.

VII. CONCLUSIONS

In this paper, efficient graph-based representations (DDDs) for symbolic expressions are applied to testability analysis for linear analog circuits. By exploiting sharing and sparsity in a canonical manner, *s*-expanded symbolic network transfer functions and testability matrix are constructed and manipulated very efficiently. Testability evaluation can be performed in time complexity proportional to the number of DDD vertices.

REFERENCES

[1] N. Sen and R. Saeks, "Fault diagnosis for linear systems via multifrequency measurement," *IEEE Trans. Circuits and Systems*, vol. CAS-26, pp. 457-465, July 1979.

[2] H. M. S. Chen and R. Saeks, "A search algorithm for the solution of multifrequency fault diagnosis equations", *IEEE Trans. Circuits and Systems*, vol. CAS-26, pp. 589-594, July 1979.

[3] C.-J. Shi and X.-D. Tan, "Symbolic analysis of large analog circuits with determinant decision diagrams", pp. 366-373 in *Proc. IEEE/ACM International Conference on Computer-Aided Design, (ICCAD'97)*, San Jose, CA, 1997.

[4] C.-J. Shi and X.-D. Tan, "Efficient derivation of exact s-expanded symbolic expressions for behavioral modeling of analog circuits", pp. 463-466 in *Proc. IEEE Custom Integrated Circuits Conference*, San Diego, CA, May 12-14,1998.

[5] R. Carmassi, M. Catelani, G. Iuculano, A. Liberatore, S. Manetti and M. Marini, "Analog network testability measurement: A symbolic formulation approach", *IEEE Trans. Instrumentation and Measurement*, vol. 40, no. 6, pp 930-935, December, 1991.

[6] G. Fedi, R. Giomi, A. Luchetta, S. Manetti and M.C. Piccirilli, "On the application of symbolic techniques to the multiple fault location in low testability analog circuits", *IEEE Trans. Circuits and Systems*, vol. 45, no. 10, pp. 1383-1388, October, 1998.

[7] A. Liberatore and S. Manetti, "Network sensitivity analysis via symbolic formulation", *Proc. 1989 IEEE Int. Symp. Circuits Syst.*, Portland, OR, pp. 705-708, May, 1989.

[8] S. Manetti and A. Liberatore, Chapter 13, "Symbolic analysis techniques-applications to analog design automation", IEEE Press, 1998.

[9] G. Fedi, and et.al., "A new symbolic method for analog circuit testability evaluation", *IEEE Trans. Instrumentation and Measurement*, vol.47, no.2, pp. 554-565, April, 1998.

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