A Simple Approach to the Evaluation of Multistage Interconnection Network Reliability[1]

Po-Jen Chuang and Chun-Liang Kuo

Department of Electrical Engineering
Tamkang University
Tamsui, Taipei Hsien
Taiwan 25137, Republic of China

Abstract—The terminal reliability is an important performance parameter in the design of a highly reliable multistage interconnection network (MIN). In this paper we present a new method able to evaluate the terminal reliability of any MINs, with a rather simplified procedure. Implementation considerations and discussions are also provided.

I. INTRODUCTION

With the advance of VLSI technology, research on multiprocessors is drawing more and more attention. In the development of multiprocessors, the design of its interconnection networks plays quite an important role as it governs the performance of the whole system. The multistage interconnection network (MIN), such as the Omega network [1] or the Gamma network [2] (see Fig. 1), is a very significant example of such networks. It is employed in a multiprocessor to connect processors and memory modules together, using multiple stages of small crossbar switches of a fixed size. The terminal reliability between a source-destination pair in such a MIN is the probability that at least one path exists between the pair. It is a fairly important performance parameter in the design of a highly reliable MIN.

In [2], a formula is developed to calculate the exact terminal reliability of the Gamma Network, but it is suitable only for the Gamma network and can not be applied to the other MINs. In search of a method able to evaluate the exact terminal reliabilities of any MINs, we have reviewed several related algorithms in the literature, such as state enumeration [3], decomposition technique [4], graph-theoretic approach [5], and path enumeration [6,7]. Among them, path enumeration tends to be the most efficient.

In path enumeration, the terminal reliability is obtained by, first, finding the set of possible paths between a pair of source and destination nodes. A path is represented by a Boolean product of components along the path. Boolean techniques concept is utilized to convert a sum of such products expression into an equivalent sum of disjoint products (SDP) expression, and then the components are replaced by their reliabilities. The terminal reliability resulting from arithmetic sum of products is therefore obtained.

For all algorithms based on path enumeration, the most time consuming step is the process to convert the sum of products expression into its equivalent SDP expression. For such transformation, one or more operators are defined by each algorithm, with intention to reduce the computation time. By using Boolean algebraic concepts, CAREL [7] defines four operators, namely, COMpare, REDuce, CoMBine and GENerate, to be operated during the transformation. Although claimed to be more efficient than the other established methods, CAREL needs to apply the four operators repeatedly to obtain the equivalent SDP expression, involving a fairly complicated procedure.

Presented in this paper is a new method which is as efficient as CAREL but involves a much simpler procedure in converting the sum of products expression into its equivalent SDP expression. It also carries easier programming and can evaluate any networks that CAREL is able to evaluate.

This paper is organized as follows. Section II provides necessary backgrounds for our study. Section III presents our proposed new method. Implementation considerations and discussions are given in Section IV. Section V concludes the paper.

II. BACKGROUNDS

In path enumeration, the terminal reliability is obtained by first finding the set of possible paths between a pair of source and destination nodes. A path is represented by a Boolean product of components along the path. Boolean techniques concept is employed to transform a sum of such products expression into an equivalent sum of disjoint products (SDP) expression, and then an *up* or operational (*down* or failure) state of each component in the expression is replaced by its reliability (unreliability). Thus the terminal reliability is obtained from arithmetic sum of products.

[1]This work was supported in part by the National Science Council of R. O. C., under contract numbers NSC82-0102-E-032-028-T and NSC83-0408-E-032-004.

0-7803-2428-5/95 $4.00 © 1995 IEEE

To be specific, if F_i represents *up* events of path P_i, the sum of products expression F can be given by $F = \cup_{i=1}^{n} F_i$ where n denotes the number of paths between a source-destination pair in a network, and the equivalent SDP expression $F(disjoint)$ will be generated by

$$F_1 + F_2 \sim F_1 + F_3 \sim F_1 \sim F_2 + \cdots + F_n \sim F_1 \sim F_2 \cdots \sim F_{n-1} \quad (1)$$

where $\sim F_i$ denotes *down* events of path P_i. For this, various researchers have provided certain techniques to get the equivalent SDP expression and to achieve low computation time. The algorithm used in the CAREL [7] is claimed to be more efficient than the others and is based on the following technique (proposition P II in [7]). For each term F_i, $1 < i \le n$, T_i is defined to be the union of all predecessor terms F_1, F_2, $\cdots$, F_{i-1}, in which any literal that is present in both F_i and any of the predecessor terms is deleted from those predecessor terms, i.e.,

$$T_i = \cup_{j=1}^{i-1} F_j \mid_{each\ literal\ of\ F_i \rightarrow 1}.$$

Consider $F^1 = F_1$, and define $F^i = F_i \sim T_i$. The $F(disjoint)$ expression is then given by $F(disjoint) = \cup_{j=1}^{n} F^j$. To give an example, consider the bridge network shown in Fig. 2. For the (S, D) node pair, four (minimal) paths can be found and F_1 through F_4 are determined to be *ab, cd, aed,* and *ceb* respectively. Then, we have $F^1 = ab$; $T_2 = ab \mid_{c=d=1} = ab$, and $F^2 = F_2 \sim T_2 = cd \sim (ab)$; $T_3 = ab + cd \mid_{a=d=e=1} = b + c$, and $F^3 = F_3 \sim T_3 = aed \sim b \sim c$; and similarly we obtain $F^4 = ceb \sim a \sim d$. Finally, we have $F(disjoint) = ab + \sim(ab)cd + \sim b \sim cade + \sim a \sim dbce$. Thus the terminal reliability of the bridge network is equal to 0.97848, assuming that the component reliabilities are all equal to 0.9.

In CAREL, Boolean algebraic concepts are used to define four operators, COMpare, REDuce, CoMBine and GENerate. The COM operator generates T_i's and the RED operator is used to remove the redundant Boolean product terms generated from $\sim T_i$'s. For example, $\sim T_i = \sim(bd + bcd) = \sim(bd) \sim (bcd) = \sim(bd)$, that is, term $\sim(bcd)$ is redundant here. The CMB operator examines each $\sim T_i$ that results from the RED operator and creates the equivalent SDP expression by considering the following six Boolean algebraic formula [7]:

$(abcX)(\sim(abc)) = \varnothing$,
$(abX)(\sim(abcde)) = (abX)(\sim(cde))$,
$(abX)(\sim(cd)) = ab \sim (cd)X$,
$(\sim(abc))(\sim(ab)X) = \sim(ab)X$,
$(\sim(abc)X)(\sim(ab)) = \sim(ab)X$, and
$(\sim(abe)X)(\sim(abcd)) = \sim(ab)X + (ab \sim eX)(\sim(cd))$,

where X is any Boolean expression. The last operator GEN then gives each disjoint product F^i from $F_i \sim T_i$, where $\sim T_i$ results from the CMB operator. Thus, to obtain the equivalent SDP expression, CAREL needs to apply all of the four operators to each term except the first one in (1), involving a fairly complicated procedure. To disentangle the procedure, a new method with a much simpler procedure will be introduced and discussed in the following section.

III. PROPOSED NEW METHOD

As has been mentioned, to obtain the equivalent SDP expression, CAREL needs to apply all of the four operators to each term except the first one in (1), making the procedure rather complex. To simplify it, instead of obtaining $F(disjoint)$ from the $n-1$ terms in (1), we look for $\sim F(disjoint)$ to get terminal *un*reliability first. Terminal *un*reliability between a source-destination pair is, by our definition, the probability that *no* paths exist between the pair. It can be gained from $\sim F$ with only one term

$$\sim F_1 \sim F_2 \sim F_3 \cdots \sim F_n. \quad (2)$$

To gather the terminal unreliability, the equivalent SDP expression $\sim F(disjoint)$ is pursued and the components in it are replaced by their reliabilities. Thus we have the terminal unreliability (resulting from arithmetic sum of products), and hence the terminal reliability (from 1 − *terminal unreliability*). To give an example, with the same four paths found for the (S, D) node pair from the bridge network mentioned in the previous section, $\sim F = \sim F_1 \sim F_2 \sim F_3 \sim F_4 = \sim(ab) \sim (cd) \sim (aed) \sim (ceb)$. The above Boolean product term can be converted into the equivalent SDP expression by using the six Boolean algebraic formula and yields $\sim F(disjoint)$ as $\sim a \sim c + \sim a \sim (be)c \sim d + a \sim b \sim d + a \sim b \sim cd \sim e$, that is, the terminal *un*reliability equals to 0.02152. The terminal reliability of the bridge network can be obtained from (1 − 0.02152), i.e., 0.97848, exactly the same as that shown in the previous section.

It is clear from the above example that to generate final disjoint terms $\sim F(disjoint)$ from $\sim F$, we need to apply only *one* operation on only *one* term, that is, $\sim F_1 \sim F_2 \sim F_3 \cdots \sim F_n$. The only operation employed here is similar to the CoMBine operation in CAREL which uses the six Boolean algebraic formula to convert a Boolean product term into the equivalent SDP expression. Fig. 1 gives another example to illustrate the simplicity and correctness of our new method. As can be seen, there are five paths from source 1 to destination 0. Components (switches only) are ordered as literals along the paths shown in Fig. 3. Note that link faults are negligible compared with switches in a MIN and are hence ignored as they are in [2,8]. F_1 through F_5, according to the five paths, are determined as $abdfh$, *acdfh*, *acefh*, *acegh*, and *acegh* respectively and $\sim F = \sim(abdfh)\sim(acdfh)\sim(acefh)\sim(acegh)\sim(acegh)$. Again, $\sim F(disjoint)$ can be obtained from $\sim F$ by applying only one operation on only one term in $\sim F$ as follows:

$$
\begin{aligned}
&\sim(abdfh)\sim(acdfh)\sim(acefh)\sim(acegh)\sim(acegh)\\
&= \sim(abdfh)\sim(acdfh)\sim(acefh)\sim(acegh)\\
&=(\sim(adfh)+adfh\sim b\sim c)\sim(acefh)\sim(acegh)\\
&=(\sim(adfh)\sim(acefh)+adfh\sim b\sim c\sim(acefh))\sim(acegh)\\
&=(\sim(afh)+afh\sim d\sim(ce)+afhd\sim b\sim c\sim(ce))\sim(acegh)\\
&= \sim(afh)\sim(acegh)+afh\sim d\sim(ce)\sim(acegh)\\
&\qquad +afhd\sim b\sim c\sim(ce)\sim(acegh)\\
&= \sim(ah)+ah\sim f\sim(ceg)+ahf\sim d\sim(ce)\sim(ceg)\\
&\qquad +ahfd\sim b\sim c\sim(ce)\sim(ceg)\\
&= \sim(ah)+ah\sim f\sim(ceg)+ahf\sim d\sim(ce)+ahfd\sim b\sim c.
\end{aligned}
$$

The terminal *un*reliability can be obtained from replacing the literals in $\sim F(disjoint)$ with switch reliabilities. It can be verified that the terminal reliability (from $1-$ *terminal unreliability*) is the same as that gathered from the formula developed in [2] — assuring the correctness of our method.

IV. IMPLEMENTATION CONSIDERATIONS AND DISCUSSIONS

The illustration given in the previous section demonstrates clearly that our new method enjoys a much simpler procedure. It also enjoys an easier programming at the same time. Note that, to implement our new method, adopting a representation for each Boolean product expression is important. A set of symbols $\{-\beta^l, 0, 1\}$ which is used in [7] is also adopted in our implementation. $-\beta^l$, 0, or 1 in the position of the variable respectively represents a complemented, absent, or uncomplemented variable, and a "$-\beta^l$" represents complements of β number of variables which are grouped together, where l is used to distinguish groups. The advantage of such a representation lies, as mentioned in [7], in its uniqueness in handling complemented variables that are grouped together. For example, the product term $\sim(abd)\sim(ce)g\sim h$ is denoted as $(-3^1\ -3^1\ -2^2\ -3^1\ -2^2\ 0\ 1\ -1^3)$.

Our method has been programmed to evaluate various MINs, especially MINs with redundant paths, such as Gamma networks [2] and REGINs (modified Gamma networks) [8] for their exact terminal reliabilities. It is worth attention that the collected results indicate the same terminal reliability figures as that obtained by using the formula established in [2] — assuring the correctness of our method and its implementation as well. In addition, our new method is observed to be as efficient as CAREL. To give a specific example, the number of disjoint terms generated in the SDP expression (that is, $F(disjoint)$ for CAREL and $\sim F(disjoint)$ for our new method) is a performance parameter [7]. Listed in Table I are such numbers for a Gamma network and a REGIN. As can be seen from the table, the numbers of terms generated are almost identical for every tag, a revelation that our method performs as well as CAREL. Nevertheless, the advantage still goes to our method in that it harvests as good performance through a much simpler procedure and easier programming, and that it can evaluate whatever networks CAREL is able to evaluate, with the focus at present set on all kinds of MINs.

V. CONCLUDING REMARKS

The terminal reliability is an important performance parameter in the design of a highly reliable multistage interconnection network (MIN). In our effort to find the exact terminal reliability for any MINs, we have come up with a simple but efficient new method which is able to calculate the exact terminal reliability for many networks, including all kinds of MINs. Instead of converting the sum of products expression F into the SDP expression $F(disjoint)$, our new method converts $\sim F$ into $\sim F(disjoint)$ to find the terminal unreliability first, and then obtain the terminal reliability by ($1-$ *terminal unreliability*). It applies only one operator on only one term to get the same result that CAREL pursues by applying four operators on $n-1$ terms (where n is the number of paths). Our approach not only enjoys a much simpler procedure and hence an easier programming when compared with CAREL, which involves a rather complicated procedure, it also provides as good performance as CAREL. The correctness of our method and its implementation is assured by the identical results as are obtained from [2]. Nevertheless, [2] calculates only the terminal reliabilities for Gamma networks while our method can be used to calculate terminal reliabilities for any MINs. It can actually evaluate whatever networks CAREL is able to evaluate, though the focus at present is set on MINs.

REFERENCES

[1] D. H. Lawrie, "Access and Alignment of Data in An Array Processor," *IEEE Trans. on Computers,* Vol. C-24, No. 12, pp. 1145-1155, Dec. 1975.

[2] D. S. Parker and C. S. Raghavendra, "The Gamma Network," *IEEE Trans. on Computers,* Vol. C-33, No. 4, pp. 367-373, Apr. 1984.

[3] S. Arnborg, "A Reduced State Enumeration — Another Algorithm for Reliability Evaluation," *IEEE Trans. on Reliability,* Vol. R-27, pp. 101-105, June 1978.

[4] A. Satyanarayana and M. K. Chang, "Network Reliability and Factoring Theorem," *Networks,* Vol. 13, pp. 107-120, Spring 1983.

[5] R. Johnson, "Network Reliability and Acyclic Orientation," *Networks*, Vol. 14, pp. 489-505, Winter 1984.

[6] S. Hariri and C. S. Raghavendra, "SYREL: A Symbolic Reliability Algorithm Based on Path and Cutset Methods," *IEEE Trans. on Computers*, Vol. C-36, No. 10, pp. 1224-1232, Oct. 1987.

[7] S. Soh and S. Rai, "CAREL: Computer Aided Reliability Evaluator for Distributed Computing Networks," *IEEE Trans. on Parallel and Distributed Systems*, Vol. 2, No. 2, pp. 199-213, Apr. 1991.

[8] N.-F. Tzeng, P.-J. Chuang, and C.-H. Wu, "Creating Disjoint Paths in Gamma Interconnection Networks," *IEEE Trans. on Computers*, Vol. 42, No. 10, pp. 1247-1252, Oct. 1993.

Table I. The number of disjoint terms generated in the SDP expression for every tag in a network with size 16

Tag	Gamma		REGIN	
	I	II	I	II
0	1	1	4	4
1	4	4	11	6
2	3	3	10	10
3	5	5	6	5
4	2	2	5	5
5	5	6	8	6
6	3	3	6	4
7	4	4	4	4
8	1	1	2	2
9	4	4	7	4
10	3	3	7	7
11	5	5	7	5
12	2	2	8	7
13	5	6	13	11
14	3	3	10	10
15	4	4	5	4

I: CAREL, II: Our new method.

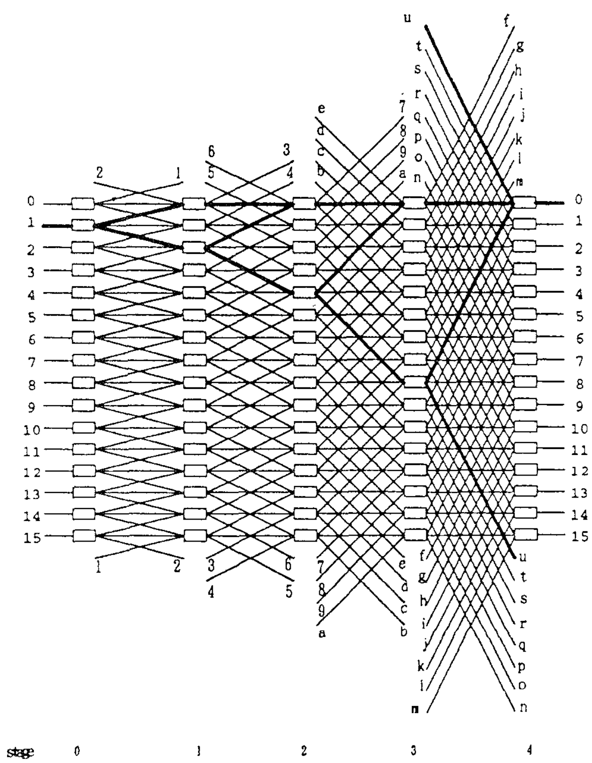

Fig. 1. A Gamma network with size 16.
(Bold lines indicate paths between (1, 0)).

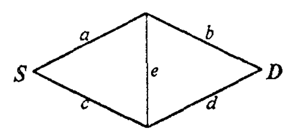

Fig. 2. Bridge Network.

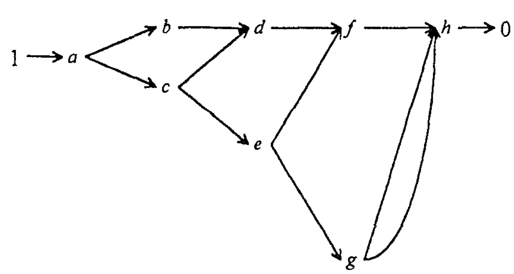

Fig. 3. The paths between (1, 0) in Fig. 1.