1985

A Linear Time Algorithm for Computing K-Terminal Reliability in Series-Parallel Graphs

Satyanarayana, A.; Wood, R. Kevin

https://hdl.handle.net/10945/38415

SIAM J. COMPUT.
Vol. 14, No. 4, November 1985

A LINEAR-TIME ALGORITHM FOR COMPUTING K-TERMINAL RELIABILITY IN SERIES-PARALLEL NETWORKS*

A. SATYANARAYANA† AND R. KEVIN WOOD‡

Abstract. Let $G=(V, E)$ be a graph whose edges may fail with known probabilities and let $K \subseteq V$ be specified. The K-terminal reliability of G, denoted $R(G_K)$, is the probability that all vertices in K are connected. Computing $R(G_K)$ is, in general, NP-hard. For some series-parallel graphs, $R(G_K)$ can be computed in polynomial time by repeated application of well-known reliability-preserving reductions. However, for other series-parallel graphs, depending on the configuration of K, $R(G_K)$ cannot be computed in this way. Only exponential-time algorithms as used on general graphs were known for computing $R(G_K)$ for these "irreducible" series-parallel graphs. We prove that $R(G_K)$ is computable in polynomial time in the irreducible case, too. A new set of reliability-preserving "polygon-to-chain" reductions of general applicability is introduced which decreases the size of a graph, and conditions are given for a graph admitting such reductions. Combining all types of reductions, an $O(|E|)$ algorithm is presented for computing the reliability of any series-parallel graph irrespective of the vertices in K.

Key words. algorithms, complexity, network reliability, series-parallel graphs, reliability-preserving reductions

1. Introduction. Analysis of network reliability is of major importance in computer, communication and power networks. Even the simplest models lead to computational problems which are NP-hard for general networks [5], although polynomial-time algorithms do exist for certain network configurations such as "ladders" and "wheels" and for some series-parallel structures such as the well-known "two-terminal" series-parallel networks. In this paper, we show that a class of series-parallel networks, for which only exponentially complex algorithms were previously known [7], [8], can be analyzed in polynomial time. In doing this, we introduce a new set of reliability-preserving graph reduction of general applicability and produce a linear-time algorithm for computing the reliability of any graph with an underlying series-parallel structure.

The network model used in this paper is an undirected graph $G=(V, E)$ whose edges may fail independently of each other, with known probabilities. The reliability analysis problem is to determine the probability that a specified set of vertices $K \subseteq V$ remains connected, i.e., the K-terminal reliability of G. Computing K-terminal reliability was first shown to be NP-hard by Rosenthal [12], and it follows from Valiant [17] that the problem is #P-complete even when G is planar. Two special cases of this reliability problem are the most frequently encountered, the terminal-pair problem where $|K|=2$, and the all-terminal problem where $K=V$. These problems are also #P-complete [11], in general, although their complexities are unknown when G is planar.

In network reliability analysis, three reliability-preserving graph reductions are well-known: the series reduction, the degree-2 reduction (an extension of the series reduction for problems with $|K|>2$) and the parallel reduction. From the realiability viewpoint, we classify series-parallel graphs into two types, those which are reducible to a single edge using standard series, parallel and degree-2 reductions, and those

* Received by the editors February 23, 1982, and in final revised form June 15, 1984. This work was conducted at the Operations Research Center, University of California, Berkeley, California 94720, under Air Force Office of Scientific Research grant AFOSR-81-0122 and under U.S. Army Research Office contract DAAG29-81-K-0160.

† Department of Computer Science, Stevens Institute of Technology, Hoboken, New Jersey 07030.

‡ Department of Operations Research, Naval Postgraduate School, Monterey, California 93943.

which are not. The former type is "reducible" and the latter "irreducible." For example, the series-parallel graph of Fig. 1a is reducible if $K = \{v_1, v_2\}$, but irreducible for $K = \{v_1, v_6\}$. Thus, the reducibility of a series-parallel graph, for the purpose of reliability evaluation, depends on the nature of the vertices included in K. A more detailed exposition of this concept appears in § 2.

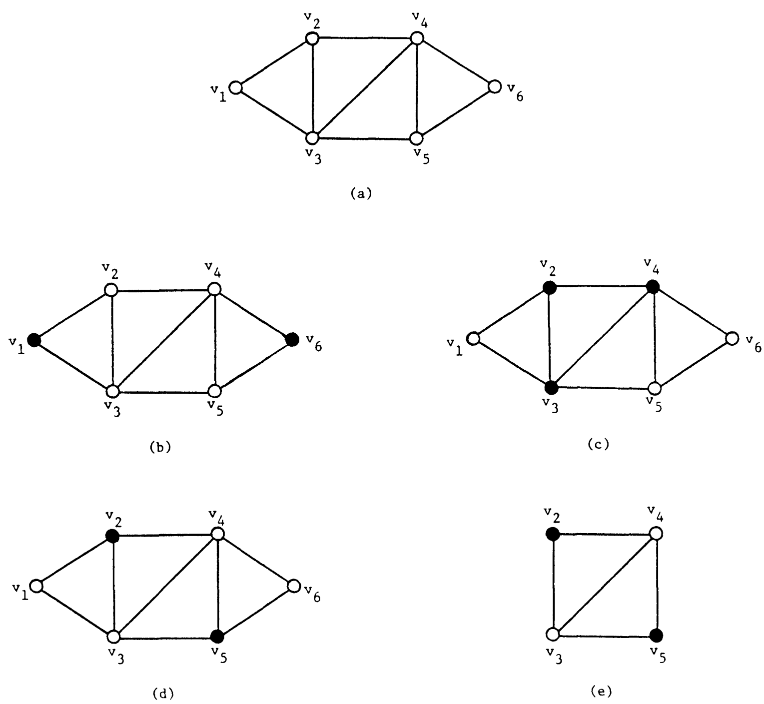


FIG. 1. *Reducible and irreducible series-parallel graphs. Note: Darkened vertices represent K-vertices.*

The K-terminal reliability of a reducible series-parallel graph can be computed in polynomial time. Several methods exist for the solution of the terminal-pair problem for such a graph, i.e., for a two-terminal series-parallel network [9], [15], and for $|K| > 2$, direct extensions of the methods can be used. However, it has been believed that computing the reliability of irreducible series-parallel graphs is as hard as the general problem. (The use of series-parallel reductions with multi-state edges [13] is applicable to this problem although this has not been recognized. We do not follow this tack because of the simplicity and generality obtained by maintaining binary-state edges.) The purpose of this paper is threefold: (1) to introduce a new set of reliability-preserving graphs reductions called "polygon-to-chain reductions," (2) to show that by using these reductions, irreducible series-parallel graphs become reducible, and (3) to give a linear-time algorithm for computing the reliability of any graph with a series-parallel structure.

In a graph, a chain is an alternating sequence of vertices and edges, starting and ending with vertices such that end vertices have degree greater than 2 and all internal vertices have degree 2. Two chains with the same end vertices constitute a polygon. In § 3, we show that a polygon can be replaced by a chain and that this transformation will yield a reliability-preserving reduction. We discuss the relationship between irreducible series-parallel graphs and polygons in § 4. Using the polygon-to-chain reductions in conjunction with the three simple reductions mentioned earlier, a polynomial-time procedure is then outlined which will compute the reliability of any series-parallel graph. This procedure is very simple but not of linear-time complexity, so in § 5 we develop algorithm which is shown to operate in $O(|E|)$ time. This algorithm will compute the K-terminal reliability of any graph having an underlying series-parallel structure. Finally, in § 6, we briefly discuss an extension to the algorithm to reduce a nonseries-parallel graph as far as possible so that the algorithm could be used as a subroutine in a reliability analysis program for general networks.

2. Preliminaries. Consider a graph $G=(V, E)$ in which all vertices are perfectly reliable but any edge e_i may fail with probability q_i or work with probability $p_i = 1-q_i$. All edge failures are assumed to occur independently of each other. Let K be a specified subset of V with $|K| \geqq 2$. When certain vertices of G are specified to be in K, we denote the graph G together with the set K by G_K. We will refer to the vertices of G belonging to K as the *K-vertices* of G_k. The *K-terminal reliability* of G, denoted by $R(G_K)$, is the probability that the K-vertices in G_K are connected. K-terminal reliability is a generalization of the common reliability measures, all-terminal reliability and terminal-pair reliability where $K=V$ and $|K|=2$, respectively.

Reliability of a separable graph. A *cutvertex* of a graph is a vertex whose removal disconnects the graph. A *nonseparable graph* is a connected graph with no cutvertices. A *block* of a graph is a maximal nonseparable subgraph.

Let $G=(V, E)$ be a separable graph and $v \in V$ be any cutvertex in G. G can be partitioned into two connected subgraphs $G^{(1)}=(V_1, E_1)$ and $G^{(2)}=(V_2, E_2)$ such that $V_1 \cup V_2 = V$, $V_1 \cap V_2 = v$, $E_1 \cup E_2 = E$ and $E_1 \cap E_2 = \emptyset$. Also, $E_1 \neq \emptyset$ and $E_2 \neq \emptyset$. Denote $K_1 = K \cap V_1$ and $K_2 = K \cap V_2$. If one of the K_i is null, say $K_1 = \emptyset$, then $G^{(1)}$ is *irrelevant* and $R(G_K) = R(G^{(2)}_{K_2})$. Otherwise, assuming $K_1 \neq \emptyset$ and $K_2 \neq \emptyset$, it is well known that $R(G_K) = R(G^{(1)}_{K_1 \cup v}) R(G^{(2)}_{K_2 \cup v})$. ($R(G_K) \equiv 1$ if $|K|=1$. Therefore, if $K_i = \{v\}$ then $R(G^{(i)}_{K_i \cup v}) \equiv 1$ and the above statement is still true.) Thus the reliability of a separable graph can be computed by evaluating the reliabilities of its blocks separately. For this reason, we henceforth consider only nonseparable graphs.

Simple reductions. In order to reduce the size of graph G_K, i.e. reduce $|V|+|E|$, and therefore reduce the complexity of computing $R(G_K)$, *reliability-preserving reductions* are often applied: Certain edges and/or vertices in G are replaced to obtain G'; new edge reliabilities are defined; a new set K' is defined; and a multiplicative factor Ω is defined; all such that $R(G_K) = \Omega R(G'_{K'})$. The following three reliability-preserving reductions are well known and are called *simple reductions.*

A *parallel reduction* replaces a pair of edges $e_a=(u, v)$ and $e_b=(u, v)$ with a single edge $e_c=(u, v)$ and defines $p_c = 1-q_a q_b$, $K'=K$, and $\Omega=1$.

Suppose $e_a=(u, v)$ and $e_b=(v, w)$ such that $u \neq w$, $\deg(v)=2$, and $v \notin K$. A *series reduction* replaces e_a and e_b with a single edge $e_c=(u, w)$, and defines $p_c = p_a p_b$, $K'=K$ and $\Omega=1$.

Suppose $e_a=(u, v)$ and $e_b=(v, w)$ such that $u \neq w$, $\deg(v)=2$, and $\{u, v, w\} \subseteq K$. A *degree-2 reduction* replaces e_a and e_b with a single edge $e_c=(u, w)$ and defines $p_c = p_a p_b/(1-q_a q_b)$, $K'=K-v$, and $\Omega = 1-q_a q_b$.

Series-parallel graphs. The following definition should not be confused with the definition of a "two-terminal" series parallel network in which two vertices must remain fixed. No special vertices are distinguished here. In a graph, edges with the same end vertices are *parallel edges.* Two nonparallel edges are *adjacent* if they are incident on a common vertex. Two adjacent edges are *series edges* if their common vertex is of degree 2. Replacing a pair of series (parallel) edges by a single edge is called a series (parallel) *replacement.* A series-parallel graph is a graph that can be reduced to a tree by successive series and parallel replacements. Clearly, if a series-parallel graph is nonseparable, then the resulting tree, after making all series and parallel replacements, contains exactly one edge.

We wish to clarify the subtle difference between the term "replacement" used here and the term "reduction" used with respect to simple reductions. Replacement is a strictly graph-theoretic term indicating some edges or vertices from G are removed and then replaced by other edges or vertices to create a new graph G'. A reduction is defined, on the other hand, with respect to G, K, and edge reliabilities. A reduction includes the act of replacing edges or vertices in G to create G' along with defining edge reliabilities, K', and Ω, all such that $R(G_K)=\Omega R(G'_{K'})$, i.e. reliability is preserved. For example, in graph G as shown in Fig. 1a, series replacements are possible while no (reliability-preserving) simple reductions are possible in the corresponding G_K for $K=\{v_1, v_6\}$ (Fig. 1b). Motivated by the difference between graphs which allow replacements but, with K and edge reliabilities defined, do not allow reliability-preserving simple reductions, we distinguish between graphs which can and cannot be reduced by simple reductions.

Reducible and irreducible series-parallel graphs. Clearly, if G has no series or parallel edges, then for any K, G_K admits no simple reductions. If G is a series-parallel graph, then a simple reduction might or might not exist in G_K depending upon the vertices of G that are chosen to be in K. For example, consider the series-parallel graph G of Fig. 1a. The graph G_K, for $K=\{v_2, v_3, v_4\}$ as in Fig. 1c, can be reduced to a single edge by successive, simple reductions. On the other hand, for $K=\{v_1, v_6\}$, G_K admits no simple reductions (Fig. 1b). A series-parallel graph G_K is *reducible* if it can be reduced to a single edge by successive, simple reductions. If G_K is reduced to a single edge e_i using m reductions, then $R(G_K)=p_i\prod_{k=1}^{m}\Omega_k$ where Ω_k is the multiplicative factor defined by the kth reduction. Note that any series-parallel graph G is reducible for the all-terminal problem since any degree-2 vertex in G_V allows a degree-2 reduction.

It is possible for a (nonseparable) series-parallel graph to admit one or more simple reductions for a specified K and still not be completely reducible to a single edge. As an illustration, consider G_K of Fig. 1d. Two series reductions may be applied to this graph to obtain the graph of Fig. 1e, but no further simple reductions are possible. A graph G_K is an *irreducible* series-parallel graph if G_K cannot be completely reduced to a single edge using simple reductions.

Chains and polygons. In a graph, a *chain* χ is an alternating sequence of distinct vertices and edges, v_1, (v_1, v_2), v_2, (v_2, v_3), $v_3, \cdots, v_{k-1}$, (v_{k-1}, v_k), v_k, such that the internal vertices, $v_2, v_3, \cdots, v_{k-1}$, are all of degree 2 and the end vertices, v_1 and v_k, are of degree greater than 2. A chain need not contain any internal vertices, but it must contain at least one edge and two end vertices. The length of a chain is simply the number of edges it contains. A *subchain* is a connected subset of a chain beginning and ending with a vertex and containing at least one edge. Both the end vertices of a subchain may be of degree 2. The notation χ will also be used for a subchain with the usage differentiated by context.

Suppose χ_1 and χ_2 are two chains of lengths l_1 and l_2, respectively. If the two chains have common end vertices u and v, then $\Delta=\chi_1\cup\chi_2$ is a *polygon* of length l_1+l_2. In other words, a polygon is a cycle with the property that exactly two vertices of the cycle are of degree greater than 2. While this definition allows two parallel edges to constitute a polygon, we will initially require a polygon to be of length at least 3.

3. Polygon-to-chain reductions. In this section a new set of reliability-preserving reductions will be introduced which replace a polygon with a chain and always reduces $|V|+|E|$ by at least 1. Consider a graph G_K which does not admit any simple reductions but does contain some polygon Δ. In general, no such polygon need exist, but, if it does exist, then the number of possible configurations is limited.

Property 1. Let G_K be a graph which admits no simple reductions. If G_K contains a polygon, then it is one of the seven types given in the first column of Table 1.

Proof. This follows from the facts that (i) every degree-2 vertex of G_K is a K-vertex, (ii) there can be no more than two K-vertices in a chain, and (iii) the length of any chain in G_K is at most 3.

Polygon-to-chain transformations. Let Δ_j be a type j polygon in G_K, a graph which admits no simple reductions. Let u and v be the vertices in Δ_j such that $\deg(u)>2$ and $\deg(v)>2$. Then, $\Delta_j=\chi_j'\cup\chi_j''$, where χ_j' and χ_j'' are chains in G_K with common end vertices u and v. Replacing the pair χ_j' and χ_j'' by the corresponding chain χ_j, as in Table 1, is called a *polygon-to-chain transformation.*

In Theorem 1 we will prove that a polygon-to-chain transformation can be used to produce a reliability-preserving, polygon-to-chain reduction. It is useful here, however, to make the distinction between a polygon-to-chain reduction and a polygon-to-chain transformation, in the same manner that simple reductions and replacements are differentiated. A transformation is only a topological mapping of a graph G to a graph G' and ignores all considerations of reliability including K-vertices. A reduction includes the topological transformation as well as all reliability calculations and changes in K-vertices.

The proof technique of Theorem 1 requires that we first discuss the use of conditional probabilities for computing the reliability of a graph in a general context. Let $e_i=(u,v)$ be some edge of G_K and let F_i denote the event that e_i is working and $\bar{F}_i$ denote the complementary event that e_i has failed. Using rules of conditional probability, the reliability of G_K can be written as

$$R(G_K)=p_iR(G_K\,|\,F_i)+q_iR(G_K\,|\,\bar{F}_i)=p_iR(G'_{K'})+q_iR(G''_{K''}) \tag{1}$$

where

$$G'=(V-u-v+w,\ E-e_i),\qquad w=u\cup v,$$

$$K'=\begin{cases}K & \text{if } u, v\notin K,\\ K-u-v+w & \text{if } u\in K \text{ or } v\in K\end{cases}$$

and

$$G''=(V,\ E-e_i),$$

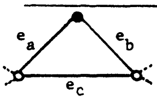
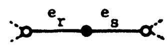
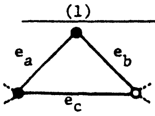
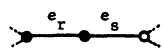
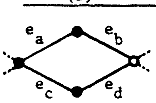
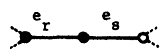
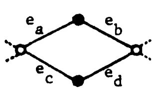
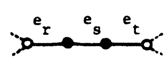
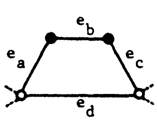
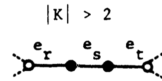
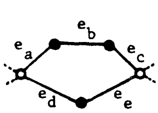
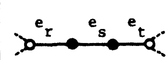
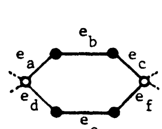
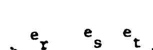
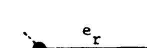
$$K''=K.$$

F_i and $\bar{F}_i$ are said to "induce" $G'_{K'}$ and $G''_{K''}$ from G_K, respectively. ("Induce" is not used in the standard graph-theoretic sense here.) $G'_{K'}$ is G_K with edge e_i contracted, and $G''_{K''}$ is G_K with edge e_i deleted.

Equation (1) can be applied recursively on the induced graphs and simple reductions made where applicable within the recursion. After repeated applications of the formula, the induced graphs are either reduced to single edges for which the reliability is simply the probability that the edge works, or some K-vertices become disconnected, in which case the reliability of the induced graph is zero. In this way, the reliability of any general graph may be computed. This method of computing the reliablity of a graph is known as "factoring" [10], [14] and is a special case of pivotal decomposition

TABLE 1
Polygon-to-chain reductions

Note: Darkened vertices represent K-vertices

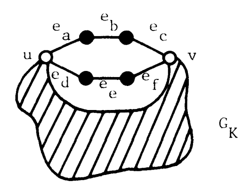
Polygon Type	Chain Type	Reduction Formulas	New Edge Reliabilities
 (1)		$\alpha = q_a p_b q_c$ $\beta = p_a q_b q_c$ $\delta = p_a p_b p_c\left(1 + \frac{q_a}{p_a} + \frac{q_b}{p_b} + \frac{q_c}{p_c}\right)$	$p_r = \frac{\delta}{\alpha + \delta}$ $p_s = \frac{\delta}{\beta + \delta}$ $\Omega = \frac{(\alpha + \delta)(\beta + \delta)}{\delta}$
 (2)		$\alpha = q_a p_b q_c$ $\beta = p_a q_b q_c$ $\delta = p_a p_b p_c\left(1 + \frac{q_a}{p_a} + \frac{q_b}{p_b} + \frac{q_c}{p_c}\right)$	
 (3)		$\alpha = p_a q_b q_c p_d + q_a p_b p_c q_d + q_a p_b q_c p_d$ $\beta = p_a q_b p_c q_d$ $\delta = p_a p_b p_c p_d\left(1 + \frac{q_a}{p_a} + \frac{q_b}{p_b} + \frac{q_c}{p_c} + \frac{q_d}{p_d}\right)$	
 (4)		$\alpha = q_a p_b q_c p_d$ $\beta = p_a q_b q_c p_d + q_a p_b p_c q_d$ $\delta = p_a q_b p_c q_d$ $\gamma = p_a p_b p_c p_d\left(1 + \frac{q_a}{p_a} + \frac{q_b}{p_b} + \frac{q_c}{p_c} + \frac{q_d}{p_d}\right)$	$p_r = \frac{\gamma}{\alpha + \gamma}$ $p_s = \frac{\gamma}{\beta + \gamma}$ $p_t = \frac{\gamma}{\delta + \gamma}$ $\Omega = \frac{(\alpha + \gamma)(\beta + \gamma)(\delta + \gamma)}{\gamma^2}$
 (5)	$\lvert K \rvert > 2$  See note	$\alpha = q_a p_b p_c q_d$ $\beta = p_a q_b p_c q_d$ $\delta = p_a p_b q_c q_d$ $\gamma = p_a p_b p_c p_d\left(1 + \frac{q_a}{p_a} + \frac{q_b}{p_b} + \frac{q_c}{p_c} + \frac{q_d}{p_d}\right)$	
 (6)		$\alpha = q_a p_b p_c q_d p_e$ $\beta = p_a q_b p_c (p_d q_e + q_d p_e)$ $\quad + p_b (q_a p_c p_d q_e + p_a q_c q_d p_e)$ $\delta = p_a p_b q_c p_d q_e$ $\gamma = p_a p_b p_c p_d p_e\left(1 + \frac{q_a}{p_a} + \frac{q_b}{p_b} + \frac{q_c}{p_c} + \frac{q_d}{p_d} + \frac{q_e}{p_e}\right)$	
 (7)		$\alpha = q_a p_b p_c q_d p_e p_f$ $\beta = p_a q_b p_c (q_d p_e p_f + p_d q_e p_f + p_d p_e q_f)$ $\quad + p_a p_b q_c p_f (p_d q_e + q_d p_e)$ $\quad + q_a p_b p_c p_d (q_e p_f + p_e q_f)$ $\delta = p_a p_b q_c p_d p_e q_f$ $\gamma = p_a p_b p_c p_d p_e p_f\left(1 + \frac{q_a}{p_a} + \frac{q_b}{p_b} + \frac{q_c}{p_c} + \frac{q_d}{p_d} + \frac{q_e}{p_e} + \frac{q_f}{p_f}\right)$	Note: For $\lvert K \rvert = 2$, new chain is  $p_r = (p_b + p_a q_b p_c p_d)/\Omega$ $\Omega = p_b + p_a q_b p_c$

of a general binary coherent system [1]. For our purposes, factoring will only be applied to the edges of a single polygon or a chain.

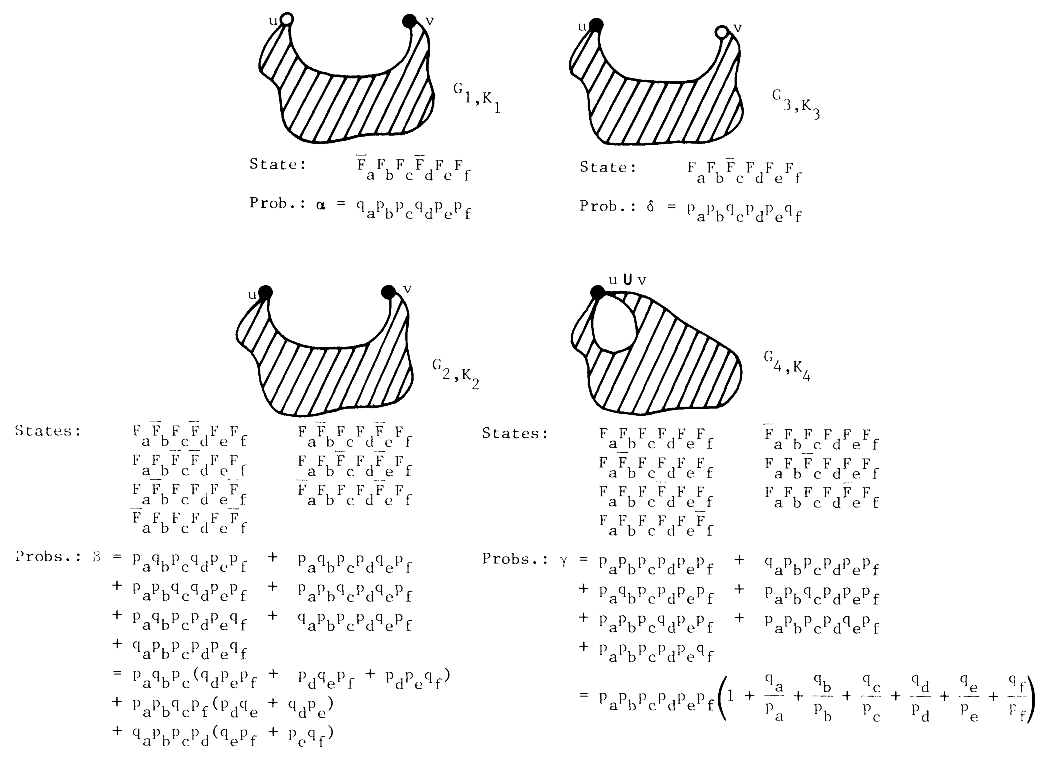
Polygon-to-chain reductions.

THEOREM 1. *Suppose G_K contains a type j polygon. Let $G'_{K'}$ denote the graph obtained from G_K by replacing the polygon Δ_j with the chain χ_j having appropriately defined edge probabilities, and let Ω_j be the corresponding multiplication factor, all as in Table 1. Then, $R(G_K) = \Omega_j R(G'_{K'})$.*

We prove the exactness of reduction 7 only, since reductions 1-6 may be shown in a similar fashion. Figs. 2 and 3 illustrate the proof of the theorem. To improve readability in the proof, we drop the subscript "7" on α, β, δ, γ, and Ω even though, strictly speaking, these are functions of the type of reduction.



(a) *Schematic of a graph with a type 7 polygon.*



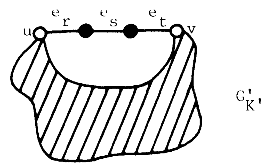
(b) *Nonfailed induced graphs.*

FIG. 2

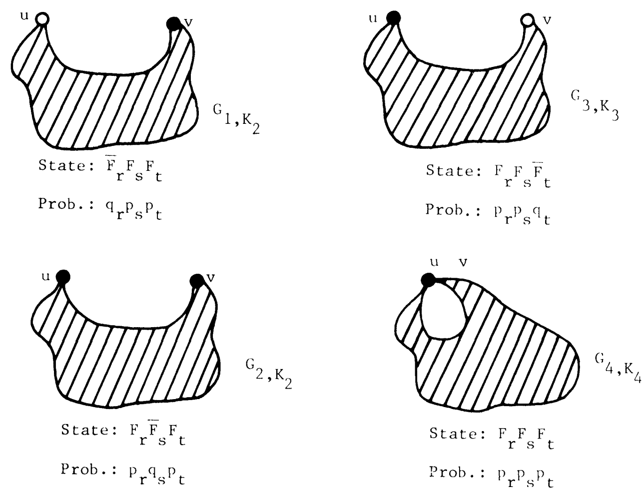
Proof of Theorem 1. Let F_i be the event that edge e_i in the polygon is working and let $\bar{F}_i$ be the event that edge e_i has failed. $\underline{F}$ denotes a compound event or state such as $F_a F_b \bar{F}_c F_d \bar{F}_e F_f$, and $\mathbf{F}$ denotes the set of all 2^6 such states. Also, $z_i = 1$ if F_i occurs and $z_i = 0$ if $\bar{F}_i$ occurs. By conditional probability and extension of (1),

$$R(G_K) = \sum_{\underline{F} \in \mathbf{F}} p_a^{z_a} q_a^{1-z_a} \cdots p_f^{z_f} q_f^{1-z_f} R(G_K \mid \underline{F}). \tag{2}$$

Only sixteen of the possible sixty-four states are nonfailed states where $R(G_K \mid \underline{F}) \neq 0$. Each nonfailed state will induce a new graph with a corresponding set



(a) *Graph of Fig.* 2 *with polygon replaced by chain.*



(b) *Nonfailed induced graphs*

FIG. 3

of K-vertices of which there only four different possibilities. Figure 2 gives these four graphs G_{i,K_i}, $i=1,2,3,4$, the states under which the graphs are induced, and the summed state probabilities in each case, α, β, δ, and γ. Thus, by grouping and eliminating terms, (2) is reduced to

$$R(G_K)=\alpha R(G_{1,K_1})+\beta R(G_{2,K_2})+\delta R(G_{3,K_3})+\gamma R(G_{4,K_4}). \tag{3}$$

Now $G'_{K'}$ is obtained from G_K by replacing the polygon with a chain u, e_r, v_1, e_s, v_2, e_t, w, and redefining K as shown in Fig. 3. Using conditional probabilities again,

$$\begin{aligned} R(G'_{K'}) &= p_rq_sp_tR(G'_{K'}|(F_r\bar{F}_sF_t))+q_rp_sp_tR(G'_{K'}|(\bar{F}_rF_sF_t)) \\ &\quad + p_tp_sq_tR(G'_{K'}|(F_rF_s\bar{F}_t))+p_rp_sq_tR(G'_{K'}|(F_rF_sF_t)) \end{aligned} \tag{4}$$

where only the nonfailed states have been written.

The four nonfailed states of $G'_{K'}$ induce the same four graphs which the nonfailed states of G_K induce. Multiplying (4) by a factor Ω, we thus have

(5)

$$\Omega R(G'_{K'})=\Omega p_rq_sp_tR(G_{1,K_1})+\Omega q_rp_sp_tR(G_{2,K_2})+\Omega p_rp_sq_tR(G_{3,K_3})+\Omega p_rp_sp_tR(G_{4,K_4}).$$

Equating, term by term, the coefficients in (3) and (5) gives

$$\alpha=\Omega q_rp_sp_t=\Omega(1-p_r)p_sp_t, \qquad \delta=\Omega p_rp_sq_t=\Omega p_rp_s(1-p_t),$$

$$\beta=\Omega p_rq_sp_t=\Omega p_r(1-p_s)p_t, \qquad \gamma=\Omega p_rp_sp_t.$$

These four equations in the four unknowns Ω, p_r, p_s, and p_t may be easily solved to obtain

$$p_r=\frac{\gamma}{\alpha+\gamma}, \qquad p_s=\frac{\gamma}{\beta+\gamma},$$
$$p_t=\frac{\gamma}{\delta+\gamma}, \qquad \Omega=\frac{(\alpha+\gamma)(\beta+\gamma)(\delta+\gamma)}{\gamma^2},$$

which are the values given in Table 1 for a type 7 polygon. The reader may verify that when these values are substituted into (4), we obtain

$$\Omega R(G'_{K'})=\alpha R(G_{1,K_1})+\beta R(G_{2,K_2})+\delta R(G_{3,K_3})+\gamma R(G_{4,K_4})=R(G_K). \qquad \square$$

It can be seen from Table 1 that polygon-to-chain reductions, like simple reductions, always reduce $|V|+|E|$ by at least 1.

Theorem 1 can be extended to give a result which can be useful for computing the reliability of a general graph. In a nonseparable graph, a *separating pair* is a pair of vertices whose deletion disconnects the graph. For example, vertices u and v in Fig. 2 are a separating pair. Using the same conditioning arguments as in the proof of Theorem 1, it can be shown that any subgraph between a separating pair can be replaced by a chain of 1, 2, or 3 edges to yield a reliability-preserving reduction. For two special cases, it has been shown that a subgraph between a separating pair can be replaced by a single edge [6]. The first case occurs when the subgraph including the separating pair has no K-vertices, and the second case occurs when the separating pair belongs to K. The fact that a chain can always be used to replace any subgraph, irrespective of the K-vertices, greatly increases the generality of any algorithm which uses this reduction.

4. Properties of series-parallel graphs. In this section we set down some properties of series-parallel graphs with respect to topology and reliability. We prove that a series-parallel graph must admit a polygon-to-chain reduction if all simple reductions have first been performed. Thus, every series-parallel graph is reducible irrespective of the vertices in K. Using this fact, we then outline a simple polynomial-time procedure for computing the reliability of such graphs.

The following property is a simple extension of the definition of a series-parallel graph.

Property 2. Let G' be the graph obtained from G by applying one or more of the following operations:

a series replacement;

a parallel replacement;

an inverse series replacement (replace an edge by two edges in series);

an inverse parallel replacement (replace an edge by two edges in parallel).

Then, G' is a series-parallel graph if and only if G is series-parallel.

Proof of Property 2 may be found in [3]. The next two properties show that the series-parallel structure of a graph is not altered by simple or polygon-to-chain reductions.

Property 3. Let G' be the graph obtained by a polygon-to-chain transformation on G. Then G' is a series-parallel graph if and only if G is series-parallel.

Proof. G' may be obtained from G by one or more series replacements, a parallel replacement, and one or more inverse series replacements, in that order. Thus, this property follows directly from Property 2. $\square$

Property 4. Let $G'_{K'}$ be the graph obtained from G_K by applying a simple reduction or a polygon-to-chain reduction on G_K. Then, G' is a series-parallel graph if and only if G is series-parallel.

Proof. A series or degree-2 reduction implements a series replacement, a parallel reduction implements a parallel replacement, and a polygon-to-chain reduction implements a polygon-to-chain transformation on G. Hence, by Properties 2 and 3, G' is a series-parallel graph if and only if G is a series-parallel. □

By next proving that every series-parallel graph G_K admits a simple reduction or a polygon-to-chain reduction, it will be possible to show that $R(G_K)$ can be computed in polynomial time for such graphs.

Property 5. Let G_K be a series-parallel graph. Then, G_K must admit either a simple reduction or one of the seven types of polygon-to-chain reductions given in Table 1.

Proof. If G_K admits a simple reduction, then we are done. If G_K has no simple reductions, then by Property 1, any polygon of G_K must be one of the seven types given in Table 1. Hence, we need only show that G contains a polygon. Let G' be the graph obtained by replacing all chains in G with single edges. If G' contains a pair of parallel edges, then the two chains in G corresponding to this pair of edges constitute a polygon. We argue that G' must contain a pair of parallel edges. If G' has no parallel edges, no simple reductions are possible in G' since all vertices in G' have degree greater than 2. Thus, G' and hence G are not series-parallel graphs, which is a contradiction. □

One simple procedure for computing $R(G_K)$ can now be outlined as follows: (1) Make all simple reductions; (2) find a polygon and make the corresponding reduction; and (3) repeat steps 1 and 2 until G_K is reduced to a single edge. If G_K is originally series-parallel, then Properties 4 and 5 guarantee that the above procedure eventually reduces G_K to a single edge. The reliability is calculated by initializing $M \leftarrow 1$, letting $M \leftarrow M\Omega_j$ whenever a polygon-to-chain reduction of type j is made, and letting $M \leftarrow M\Omega$, for $\Omega = 1 - q_a q_b$, whenever a degree-2 reduction is made on some edges e_a and e_b. At the end of the algorithm with a single remaining edge e_i, the reliability of the original graph is given by $R(G_K) = Mp_i$.

The total number of parallel and polygon-to-chain reductions executed by this procedure, before the graph is reduced to a single edge, is exactly $|E|-|V|+1$. This is because the number of fundamental cycles in a connected graph is $|E|-|V|+1$, and a parallel or polygon-to-chain reduction deletes exactly one such cycle [2]. The complexity of steps (1) and (2) above can be linear in the size of G, and thus, the running time of the whole procedure is at best quadratic in the size of G. In order to develop a linear-time algorithm, we have found it necessary to move the parallel reduction from the domain of simple reductions to the domain of polygon-to-chain reductions. Indeed, a parallel reduction is a trivial case of a polygon-to-chain reduction with a multiplier $\Omega = 1$. We will henceforth consider two parallel edges to be the type 8 polygon and the parallel reduction to be the type 8 polygon-to-chain reduction.

5. An $O(|E|)$ algorithm for computing the reliablity of any series-parallel graph. The objective here is to develop an efficient, linear-time algorithm for computing the reliability of any series-parallel graph. All results needed to present this algorithm have been established; however, some additional notation and definitions must be given.

If u and v are the end vertices of a chain χ, then u and v are said to be *chain-adjacent.* When it is necessary to distinguish these vertices, we will use the notation $\chi(u, v)$. A subchain with end vertices u and v will also be denoted $\chi(u, v)$ but in this case u and

v cannot be said to be chain-adjacent. The algorithm is presented next, followed by a proof of its validity and linear complexity. The algorithm reduces G_K to two edges in parallel and prints $R(G_K)$ if G is initially series-parallel (We stop at two edges in parallel instead of a single edge because these edges do not form a polygon by our definition; their end vertices do not have degrees greater than 2.), or prints a message that G is not series-parallel. Comments are enclosed in square brackets.

ALGORITHM.

Input: A nonseparable graph G with vertex set V, $|V| \geqq 2$, edge set E, $|E| \geqq 2$, and set $K \subseteq V$, $|K| \geqq 2$. Edge probabilities p_i for each edge $e_i \in E$.

Output: $R(G_K)$ if G is series-parallel or a message that G is not series-parallel.

```
Begin
M ← 1.
Perform all series reductions.
Perform all degree-2 reductions letting M ← MΩ for each such reduction.
Construct list, T ← {v | v ∈ V and deg (v) > 2} marking all such v "onlist."
Mark all v ∉ T "offlist."
While T ≠ ∅ and |E| > 2 do
  Begin
  Remove v from T.
  i ← 1. [Index of the next chain out of v to be searched]
  Until i > 3 or v is deleted or deg (v) = 2 do
    Begin
    Search the ith chain out of v.
    i ← i + 1.
    If a polygon Δ(v, w) is found then do
      Begin
      Apply the appropriate type j polygon-to-chain reduction to Δ(v, w)
      to obtain χ(v, w), and let M ← MΩ_j.
      i ← i − 1.
      If deg (v) = 2 or deg (w) = 2 then do
        Begin
        Apply all possible series and degree-2 reductions on the chain
        (or cycle) containing subchain χ(v, w) to obtain completely
        reduced chain χ(x, y) (or parallel edges (x, y) and (x, y)), letting
        M ← MΩ for each degree-2 reduction.
        If y ≠ v and y is "offlist" then mark y "onlist" and add y to T.
        If x ≠ v and x is "offlist" then mark x "onlist" and add x to T.
        End
      End
    End
  End
If |E| = 2 then print ("R(G_K) is" M(1 − q_a q_b)) [for E = {e_a, e_b}]
  else print ("G is not series-parallel").
End.
```

The key to the algorithm is the way in which the "until" loop operates. This loop says: "Sequentially search chains incident to v reducing any polygons which are found and making any subsequent series and degree-2 reductions until either (a) v is shown to be chain-adjacent to three distinct vertices, or (b) v is completely deleted from G through the reductions, or (c) v becomes a degree-2 vertex through the reductions.

No chain is ever searched more than once each time this loop is entered. The correctness of the algorithm is not hard to show. Arguments similar to those presented here may be found in [16] where the problem is the recognition of two-terminal series-parallel directed graphs.

Suppose firstly that G consists of a single cycle. The initial series and degree-2 reductions will reduce G_K to two edges in parallel, T will be empty, and the algorithm therefore gives $R(G_K)$ correctly at the final step of the algorithm. Next, suppose that G does not consist of a single cycle, in which case T will not be empty and an initial search for a polygon will begin. Since all initial series and degree-2 reductions were performed, by Property 5, any polygon found must be one of the eight specified types. If a polygon is found and reduced, the resulting chain may, in fact, be a subchain. If this happens, some new series and degree-2 reductions may be admitted on the chain (or cycle) containing that subchain but nowhere else. All such reductions are made when applicable. Thus, every time the "until" loop of the algorithm is entered or iterated, the graph admits no series or degree-2 reductions, and only polygons of the eight given types can exist.

Vertices are continually removed from the stack T and replaced, at most two at a time, only when polygon-to-chain reductions are made. At most $|E|-|V|$ polygon-to-chain reductions can ever be made since each polygon-to-chain reduction removes exactly one of the $|E|-|V|+1$ fundamental cycles of G and the final reduced graph must retain at least one fundamental cycle. Therefore, at most $|V|+2(|E|-|V|)=2|E|-|V|$ vertices can ever pass through T before T becomes empty and the "while" loop must terminate. If $|E|=2$ at that point, then $R(G_K)$ is correctly given at the last step of the algorithm since only reliabilty-preserving series, degree-2, and polygon-to-chain reductions are ever performed. Property 4 proves that the original graph must have been series-parallel.

If $|E|>2$ when T becomes empty, then we must show that the reduced graph is not series-parallel and that the original graph was not series parallel. In this case, every vertex v with $\deg(v)>2$ is chain-adjacent to at least three distinct vertices. This is true since (i) every vertex v with $\deg(v)>2$ is initially put in the list T and its chain-adjacent vertices checked in the "until" loop and (ii) whenever the chain-adjacency of a vertex or vertices is altered (this can occur to at most two vertices at a time) after a polygon-to-chain reduction, then this vertex or vertices are returned to the list T if not already there. The following property proves that a graph with the given chain-adjacency structure is not series-parallel.

Property 6. Let G be a nonseparable graph such that all vertices v with $\deg(v)>2$ are chain-adjacent to at least three distinct vertices. Then, G is not a series-parallel graph.

Proof. Let G' be the graph obtained from G by first replacing all chains with single edges in a sequence of series replacements and then removing any parallel edges in a sequence of parallel replacements. By Property 2, G is a series-parallel if and only if G' is a series-parallel. Now, every vertex $v \in V'$ has $\deg(v)>2$ and there are no parallel edges in E'. Thus, G' admits no series or parallel replacements and cannot be series-parallel. Therefore, G cannot be series-parallel. □

This proves that if the algorithm terminates with $|E|>2$, the reduced graph is not series-parallel, and Property 4 proves that the original graph could not have been series-parallel either. This establishes the validity of the algorithm. We now turn our attention to its computational complexity.

In order to show that the algorithm is linear in the size of G, we use a multi-linked adjacency list to represent G. In this representation, for each vertex a doubly-linked list of adjacent vertices corresponding to incident edges is kept together with the associated edge probabilities. Every edge is represented twice since we are dealing with an undirected graph, and additional links are kept between both representations of each edge. Such an adjacency list can be initialized in $O(|V|+|E|)$ time for any graph. Using the above representation, any series, degree-2, or polygon-to-chain reduction can be carried out in constant time. Also, none of the reductions ever require the use of more vertices or edges after the reduction than before. This means that if any new edges or vertices must be defined, old ones can be reused and the size of the graph representation is never increased.

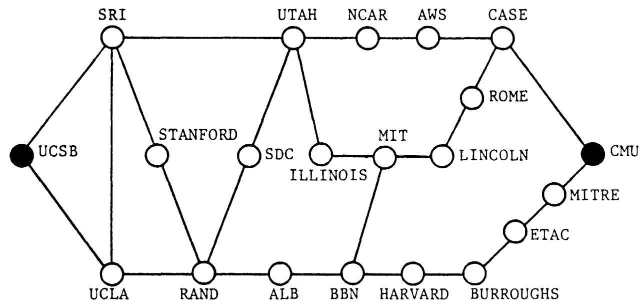
Now, initial series and degree-2 reductions are performed on $O(|V|)$ time only once and, consequently, may be ignored for purposes of complexity analysis. Consider the "until" loop of the algorithm. Each time chains emanating from the current vertex v are searched here, and l polygons are found and reduced, the maximum amount of work which can be performed is bounded by C_1+C_2l, where C_1 is a constant bounding the amount of work required to find three chains with distinct end vertices, and C_2 is a constant bounding the amount of work required to perform a polygon-to-chain reduction and any subsequent series and degree-2 reductions. That C_1 is, in fact, a constant is obvious. C_2 is a constant because there are only eight types of polygons to recognize and reduce, and because after reduction of $\Delta(v, w)$ to $\chi(v, w)$, any chain $\chi(x, y)$ containing $\chi(v, w)$ can have length at most 9. Thus $\chi(x, y)$ would require at most 8 series and degree-2 reductions to be completely reduced. This worst case could occur if $\deg(v)=\deg(w)=2$ after the polygon-to-chain reduction and the subchains $\chi(x, v)$, $\chi(v, w)$, and $\chi(w, y)$, which were proper chains before the reduction, are at their maximum possible lengths of 3. (In the case that G is a cycle after a polygon-to-chain reduction, the maximum length of such a cycle is 6, and reduction of the cycle to two edges in parallel requires at most 4 series and degree-2 reductions.) Since at most $2|E|-|V|$ vertices ever pass through T, and since at most $|E|-|V|$ polygon-to-chain reductions will ever be performed, the work performed by the algorithm is bounded by $C_1(2|E|-|V|)+C_2(|E|+|V|)$. Under the connectivity assumptions $|E| \geqq |V|$, and we have therefore proven the following theorem:

THEOREM 2. *Let G be a nonseparable series-parallel graph. Then, for any K, $R(G_K)$ can be computed in $O(|E|)$ time.*

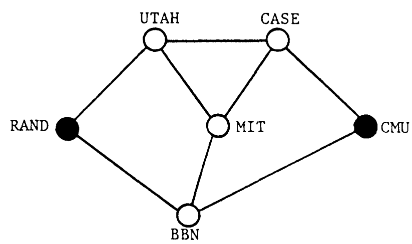
6. Extension to the algorithm. The algorithm of § 5 can be extended to make all possible simple and polygon-to-chain reductions in a nonseries-parallel graph. In this way, the extended algorithm can be used as a subroutine in a more general network reliability algorithm for computing $R(G_K)$ when G is not series-parallel. The complexity of computing $R(G_K)$ can often be reduced to some degree by this device.

Suppose the reduction algorithm of § 5 starts with a nonseries-parallel graph G. After termination of the algorithm, G_K may or may not have been partially reduced. From the proof of Property 6, the only possible remaining reductions are polygon-to-chain reductions. Each such polygon-to-chain reduction would correspond to a parallel edge replacement used to obtain the graph G' of that proof. Therefore, G_K can be totally reduced by first applying the algorithm and then finding and reducing any remaining polygons. This can easily be done by searching all chains emanating from all vertices v with $\deg(v)>2$. In the worst case, each chain, and thus each edge, must be searched twice. Parallel chains can be recognized in constant time, and therefore, the added computation is $O(|E|)$ and the algorithm with the extension remains $O(|E|)$.

To illustrate the usefulness of the extended algorithm for a general graph, let us consider the ARPA computer network configuration as shown in Fig. 4a [4]. Suppose



(a) ARPA *computer network.*



(b) *Reduced network.*

FIG. 4

we are interested in the terminal-pair reliability between UCSB and CMU. Application of the extended algorithm yields a reduced network as shown in Fig. 4b with redefined edge reliabilities and an associated multiplier. The original reliability problem is now equivalent to computing the terminal-pair reliability between RAND and CMU in the reduced network. In linear time the size of the network has been reduced considerably and, because computing the reliability of a general network is exponential in its size, a significant computational advantage should be gained.

REFERENCES

[1] R. E. BARLOW AND F. PROSCHAN, *Statistical Theory of Reliability*, Holt, Rinehart and Winston, New York, 1975.

[2] N. DEO, *Graph Theory with Applications to Engineering and Computer Science*, Prentice-Hall, Englewood Cliffs, NJ, 1974.

[3] R. J. DUFFIN, *Topology of series-parallel networks*, J. Math. Analysis Appl., 10 (1965), pp. 303–318.

[4] L. FRATTA AND U. MONTANARI, *A Boolean algebra method for computing the terminal reliability in a communication network*, IEEE Trans. Circuit Theory, CT-20 (1973), pp. 203–211.

[5] M. R. GAREY AND D. S. JOHNSON, *Computers and Intractability: A Guide to the Theory of* NP-*Completeness*, W. H. Freeman, San Francisco, 1979.

[6] J. N. HAGSTROM, *Combinatoric tools for computing network reliability*, Ph.D. Thesis, Dept. of IEOR, Univ. California, Berkeley, 1980.

[7] E. HANSLER, G. K. MCAULIFE, R. S. WILKOV, *Exact calculation of computer network reliability*, Networks, 4 (1974), pp. 95–112.

[8] P. M. LIN, B. J. LEON, T. C. HUANG, *A new algorithm for symbolic system reliability analysis*, IEEE Trans. Reliability, R-25 (1976), pp. 2–15.

[9] K. B. MISRA, *An algorithm for the reliability evaluation of redundant networks*, IEEE Trans. Reliability, R-19 (1970), pp. 146–151.

[10] F. MOSKOWITZ, *The analysis of redundancy networks*, AIEE Trans. (Commun. Electron.), 77 (1958), pp. 627–632.

[11] J. S. PROVAN AND M. O. BALL, *The complexity of counting cuts and of computing the probability that a graph is connected*, working Paper MS/S 81-002, College of Business and Management, Univ. Maryland, College Park, 1981.

[12] A. ROSENTHAL, *Computing reliability of complex systems*, Ph.D. Thesis, Dept. of EECS, Univ. California, Berkeley, 1974.

[13] ———, *Series and parallel reductions for complex measures of network reliability*, Networks, 11 (1981), pp. 323–334.

[14] A. SATYANARAYANA AND M. K. CHANG, *Network reliability and the factoring theorem*, Networks, to appear; Also, ORC 81-12, Operations Research Center, Univ. California, Berkeley, 1981.

[15] J. SHARMA, *Algorithm for reliability evaluation of a reducible network*, IEEE Trans. Reliability, R-25 (1976), pp. 337–339.

[16] J. VALDES, R. E. TARJAN AND E. L. LAWLER, *The recognition of series parallel digraphs*, this Journal, 11 (1982), pp. 297–313.

[17] L. G. VALIANT, *The complexity of enumeration and reliability problems*, this Journal, 8 (1979), pp. 410–421.