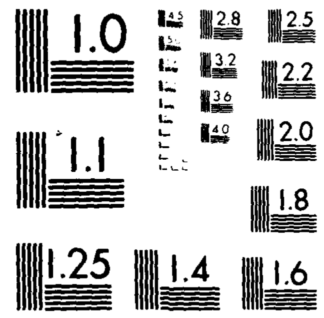


MICROCOPY RESOLUTION TEST CHART

NATIONAL

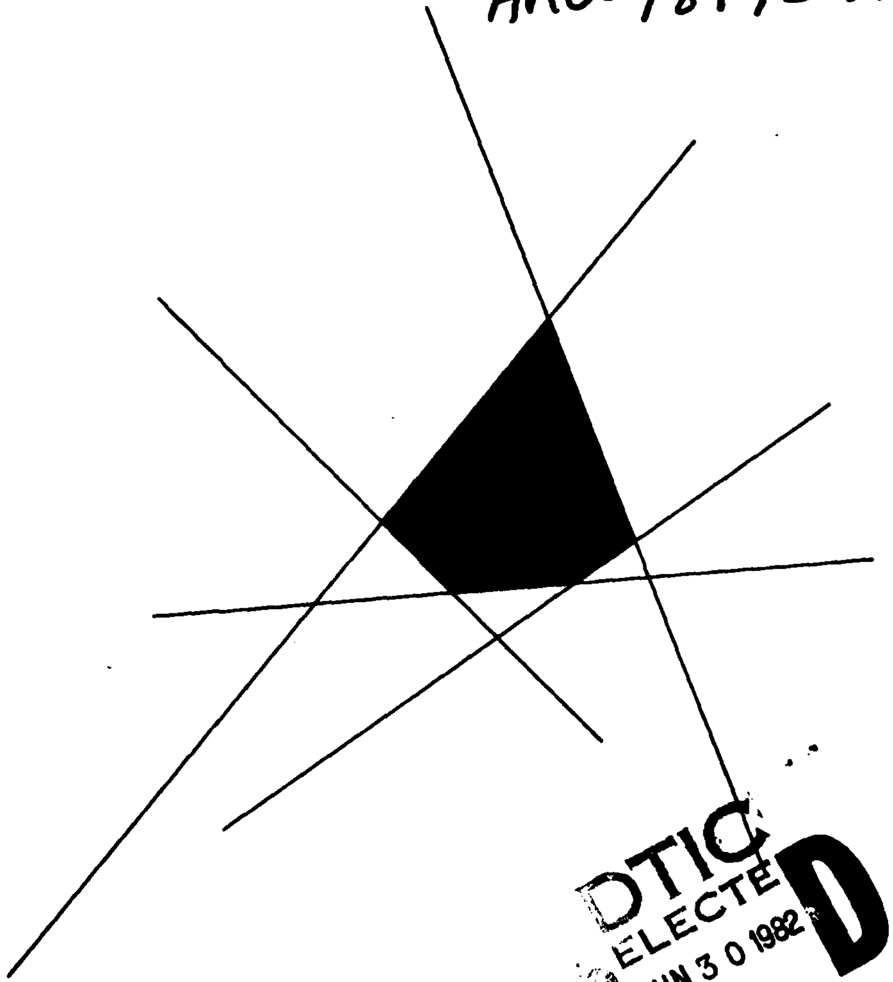
ORC 82-4
MARCH 1982

(12)

POLYGON-TO-CHAIN REDUCTIONS AND NETWORK RELIABILITY

by
A. SATYANARAYANA
and
R. KEVIN WOOD

ARO-18195.2-MA



OPERATIONS
RESEARCH
CENTER

UNIVERSITY OF CALIFORNIA · BERKELEY

POLYGON-TO-CHAIN REDUCTIONS AND NETWORK RELIABILITY[†]

Operations Research Center Research Report No. 82-4

A. Satyanarayana and R. Kevin Wood

March 1982

U. S. Army Research Office - Research Triangle Park

DAAG29-81-K-0160

Operations Research Center
University of California, Berkeley

[†]Partially supported by the Air Force Office of Scientific Research (AFSC), USAF, under Grant AFOSR-81-0122 with the University of California. Reproduction in whole or in part is permitted for any purpose of the United States Government.

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER ORC 82-4	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) POLYGON-TO-CHAIN REDUCTIONS AND NETWORK RELIABILITY		5. TYPE OF REPORT & PERIOD COVERED Research Report
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) A. Satyanarayana and R. Kevin Wood		8. CONTRACT OR GRANT NUMBER(s) AFOSR-81-0122
9. PERFORMING ORGANIZATION NAME AND ADDRESS Operations Research Center University of California Berkeley, California 94720		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 2304/A5
11. CONTROLLING OFFICE NAME AND ADDRESS United States Air Force Air Force Office of Scientific Research Bolling Air Force Base, D.C. 20332		12. REPORT DATE March 1982
		13. NUMBER OF PAGES 33
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Also supported by the U. S. Army Research Office - Research Triangle Park under Contract DAAG29-81-K-0160.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Network Reliability
Series-Parallel Graphs
Efficient Algorithm
Graph Reductions

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

(SEE ABSTRACT)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

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. Two classes of series-parallel graphs, which arise from the configuration of the vertices in K, are defined with respect to reliability computation, namely *s-p reducible* and *s-p complex*. $R(G_K)$ can be computed in polynomial time for *s-p* reducible graphs using well-known, reliability-preserving graph reductions. However, it cannot be computed in this way for the whole class of *s-p* complex graphs. Only exponential-time algorithms as used on general graphs were previously known for computing $R(G_K)$ in the *s-p* complex case, but we prove that $R(G_K)$ is computable in polynomial time in this case, too. A new set of reliability-preserving "polygon-to-chain" reductions of general applicability is introduced which decreases the size of a graph. Conditions are given for graphs which admit 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.

Polygon-to-Chain Reductions and Network Reliability

A. Satyanarayana and R. Kevin Wood

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 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. 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 $K=V$.

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 reliability viewpoint, we classify series-parallel graphs into two broad types, those which are reducible to a single edge using *series, parallel and degree-2 reductions*, and those which are not. The former type is referred to as *s-p* reducible and the latter, *s-p* complex. For example, the series-parallel graph of Figure 1a is *s-p* reducible if $K=\{v_1, v_2\}$, but is *s-p* complex 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 section 2.

The K-terminal reliability of an *s-p* reducible graph can be computed in polynomiai 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* [9,12], and for $|K|>2$, direct extensions of the methods can be used. However, it has been believed that computing the reliability of *s-p* complex graphs is as hard as the general problem. The purpose of this paper is to present an efficient, linear-time algorithm for this problem by introducing a new set of reliability-preserving graph reductions called polygon-to-chain reductions.

In a graph, a chain is an alternating sequence of vertices and edges, starting and ending with vertices such that all internal vertices have degree 2. Two chains with the same end vertices *constitute a polygon*. In section 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 *s-p* complex graphs and polygons in section 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 an *s-p* complex graph. This procedure is very simple but not necessarily linear, so in section 5 we develop, in detail, an efficient 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 section 6, we discuss how the algorithm can be extended to reduce a nonseries-parallel graph as far as possible so that it could be used as a subroutine in a reliability analysis algorithm 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 $|K|\geqslant 2$. 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 $V_1\bigcup V_2=V$, $V_1\bigcap V_2=v$, $E_1\bigcup E_2=E$ and $E_1\bigcap E_2=\varnothing$. Also, $E_1\neq\varnothing$ and $E_2\neq\varnothing$. Denoting $K_1=K\bigcap V_1$ and $K_2=K\bigcap V_2$, it is well known that $R(G_K)=R({G^{(1)}}_{K_1\bigcup v})R({G^{(2)}}_{K_2\bigcup v})$. 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 and therefore reduce the complexity of computing $R(G_K)$, the following well-known *simple reductions* are often applied:

Parallel reduction

A *parallel reduction* replaces a pair of edges $e_a=(u,v)$ and $e_b=(u,v)$ with a single edge $e_c=(u,v)$ such that $p_c=1-q_aq_b$.

Series reduction

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 $e_c=(u,w)$ such that $p_c=p_a p_b$.

If G'_{K} is the graph obtained from G_{K} after a series or parallel reduction, then $R(G_{\mathrm{K}})=R(G'_{\mathrm{K}})$. In other words, the K-terminal reliability of G_{K} remains invariant under series or parallel reductions.

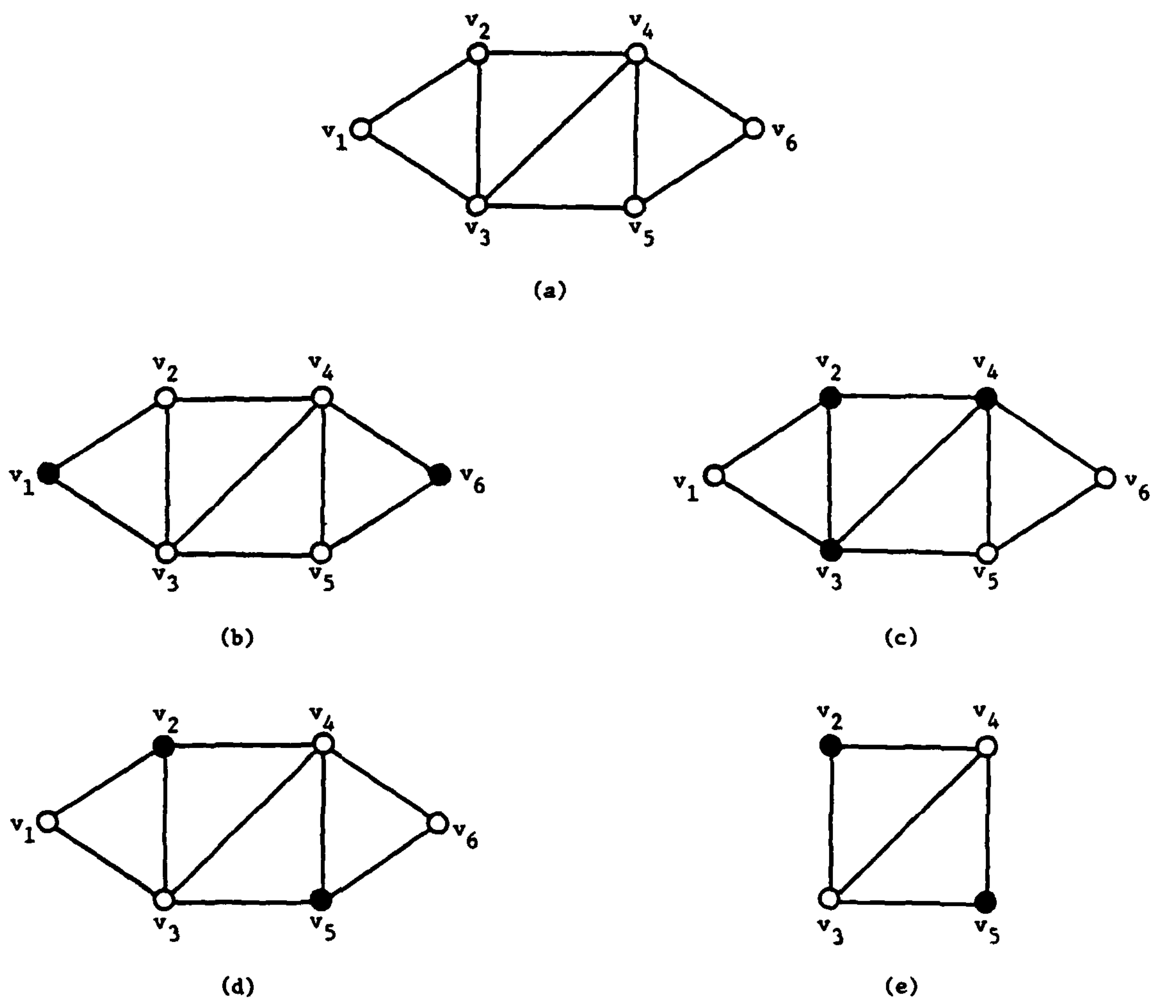
Degree-2 reduction

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 $e_c=(u,w)$ such that $p_c=p_a p_b/(1-q_a q_b)$ and $R(G_{\mathrm{K}})=(1-q_a q_b)R(G'_{\mathrm{K}-v})$, where G' is the graph obtained from G by replacing e_a and e_b with e_c.

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. By reduction, we mean a reliability-preserving reduction; hence, the term is applicable only to G_{K}. On the other hand, a replacement is defined on G, irrespective of K. For example, in graph G as shown in Figure 1a, series replacements exist while no reductions are possible in the corresponding G_{K} for



Note: Darkened vertices represent K-vertices

s-p Reducible and *s-p* Complex Graphs

FIGURE 1

K$=\{v_1, v_6\}$ (Figure 1b). Motivated by this, we define an *s-p* reducible graph and an *s-p* complex graph next.

s-p reducible graphs and s-p complex 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 Figure 1a. The graph G_K, for K$=\{v_2, v_3, v_4\}$ as in Figure 1c, can be reduced to a single edge by successive, simple reductions. On the other hand, for K$=\{v_1, v_6\}$, G_K has no reductions (Figure 1b). A graph G_K is termed *s-p reducible* if it can be reduced to a single edge by successive, simple reductions.

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 Figure 1d. Two series reductions may be applied to this graph to obtain the graph of Figure 1e, but no further simple reductions are possible. A graph G_K is *s-p complex* if G is a series-parallel graph, but G_K cannot be completely reduced to a single edge using simple reductions. An *s-p* complex graph may or may not admit some 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, $v_2, v_3, \ldots, v_{k-1}$, 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 the 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 $\chi_1 \bigcup \chi_2$ 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. Consider a graph G_K which does not admit any simple reductions but does contain some polygon Δ. In general, no such Δ 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:

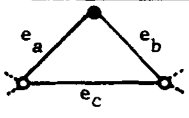
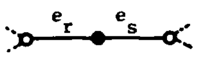
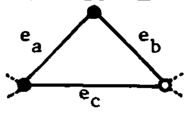
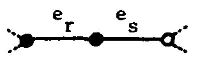
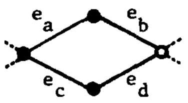
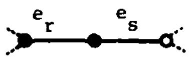
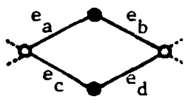
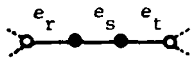
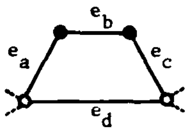
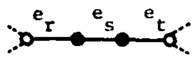
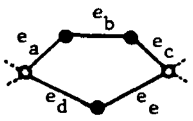
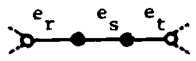
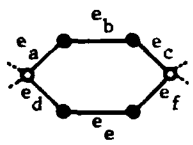
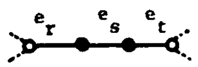
Let Δ_j be a type j polygon in G_K, a graph which admits no simple reductions. Let u and v be the vertices in Δ_j such that $\deg(u)>2$ and $\deg(v)>2$, then $\Delta_j=\chi'_j \bigcup \chi''_j$, where χ'_j and χ''_j are chains in G_K with common end vertices u and v. Replacing the pair χ'_j and χ''_j of polygon Δ_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.

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)		$\alpha = q_a p_b p_c q_d$ $\beta = p_a q_b p_c q_d$ $\delta = p_a p_b q_c q_d$ $\gamma = p_a p_b p_c p_d \left(1 + \frac{q_a}{p_a} + \frac{q_b}{p_b} + \frac{q_c}{p_c} + \frac{q_d}{p_d}\right)$	
 (6)		$\alpha = q_a p_b p_c q_d p_e$ $\beta = p_a q_b p_c (p_d q_e + q_d p_e)$ $+ p_b (q_a p_c p_d q_e + p_a q_c q_d p_e)$ $\delta = p_a p_b q_c p_d q_e$ $\gamma = p_a p_b p_c p_d p_e \left(1 + \frac{q_a}{p_a} + \frac{q_b}{p_b} + \frac{q_c}{p_c} + \frac{q_d}{p_d} + \frac{q_e}{p_e}\right)$	
 (7)		$\alpha = q_a p_b p_c q_d p_e p_f$ $\beta = p_a q_b p_c (q_d p_e p_f + p_d q_e p_f + p_d p_e q_f)$ $+ p_a p_b q_c p_f (p_d q_e + q_d p_e)$ $+ q_a p_b p_c p_d (q_e p_f + p_e q_f)$ $\delta = p_a p_b q_c p_d p_e q_f$ $\gamma = p_a p_b p_c p_d p_e p_f \left(1 + \frac{q_a}{p_a} + \frac{q_b}{p_b} + \frac{q_c}{p_c}\right.$ $\left. + \frac{q_d}{p_d} + \frac{q_e}{p_e} + \frac{q_f}{p_f}\right)$	

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 the conditional probability rules, the reliability of G_K can be written as

$$R(G_K) = p_i R(G_K | F_i) + q_i R(G_K | \bar{F}_i) \qquad (1)$$

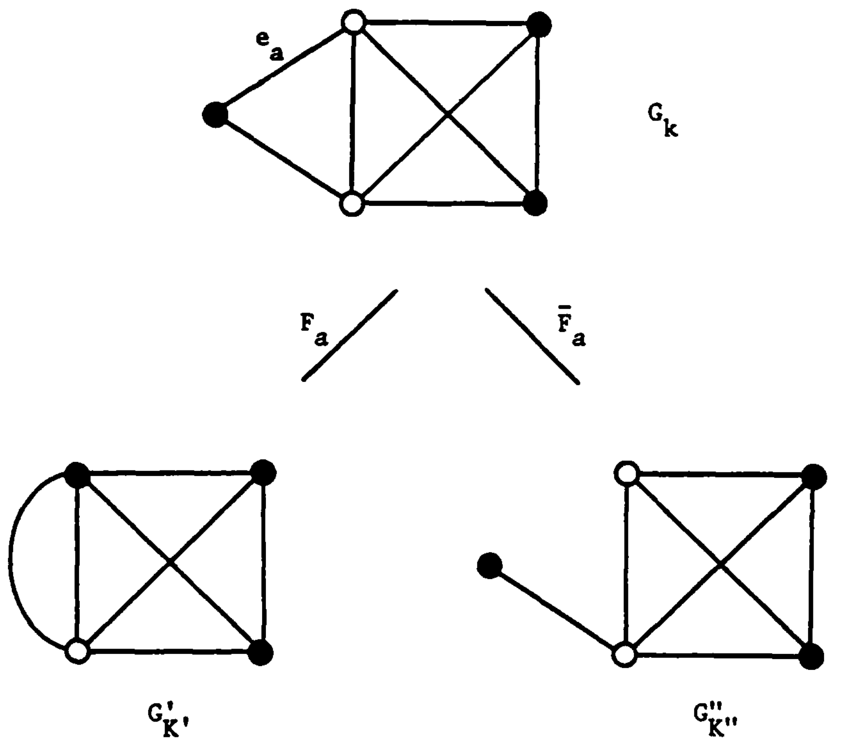
$G_K | F_i$ actually defines a new graph (say $G'_{K'}$) in which vertices u and v are known to be connected. Thus, in $G'_{K'}$, edge e_i may be deleted and u and v merged into a single supervertex $w=u \bigcup v$. If either $u \epsilon K$ or $v \epsilon K$ then $w \epsilon K'$ and thus $G'_{K'}$ is defined by

$$G' = (V-u-v+w, E-e_i)$$
$$K' = \begin{cases} K & \text{if } u,v \notin K \\ K-u-v+w & \text{if } u \epsilon K \text{ or } v \epsilon K \end{cases}$$

Similarly $G_K | \bar{F}_i$ defines a new graph $G''_{K''}$ where $G''=(V,E-e_i)$ and $K''=K$. Figure 2 illustrates how these two graphs are induced.

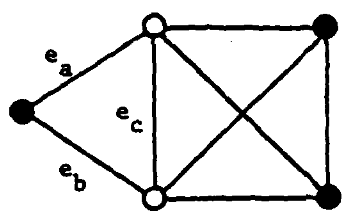
Equation 1 can be applied recursively on the induced graphs and simple reductions made where applicable within the recursion. Eventually the induced graphs are either reduced to single edges for which the reliability is simply the probability that the edge works, or some K-vertices become disconnected, 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 reliability of a graph is known as "factoring" [10,11] and is a special case of pivotal decomposition 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. To illustrate, consider the graph G_K of Figure 3a which contains a type 1 polygon with 3 edges, e_a, e_b, and e_c. Let $\underline{F}$ denote a compound event or "state" such as $F_a \bar{F}_b F_c$ and let F be the set containing all 2^3 possible states. Also, let $z_i=1$ if the event F_i occurs in $\underline{F}$ and let $z_i=0$ if $\bar{F}_i$ occurs. In other words, z_i is an indicator variable which is 1 if e_i works and is 0 if e_i has failed. Equation 1 can be extended now to

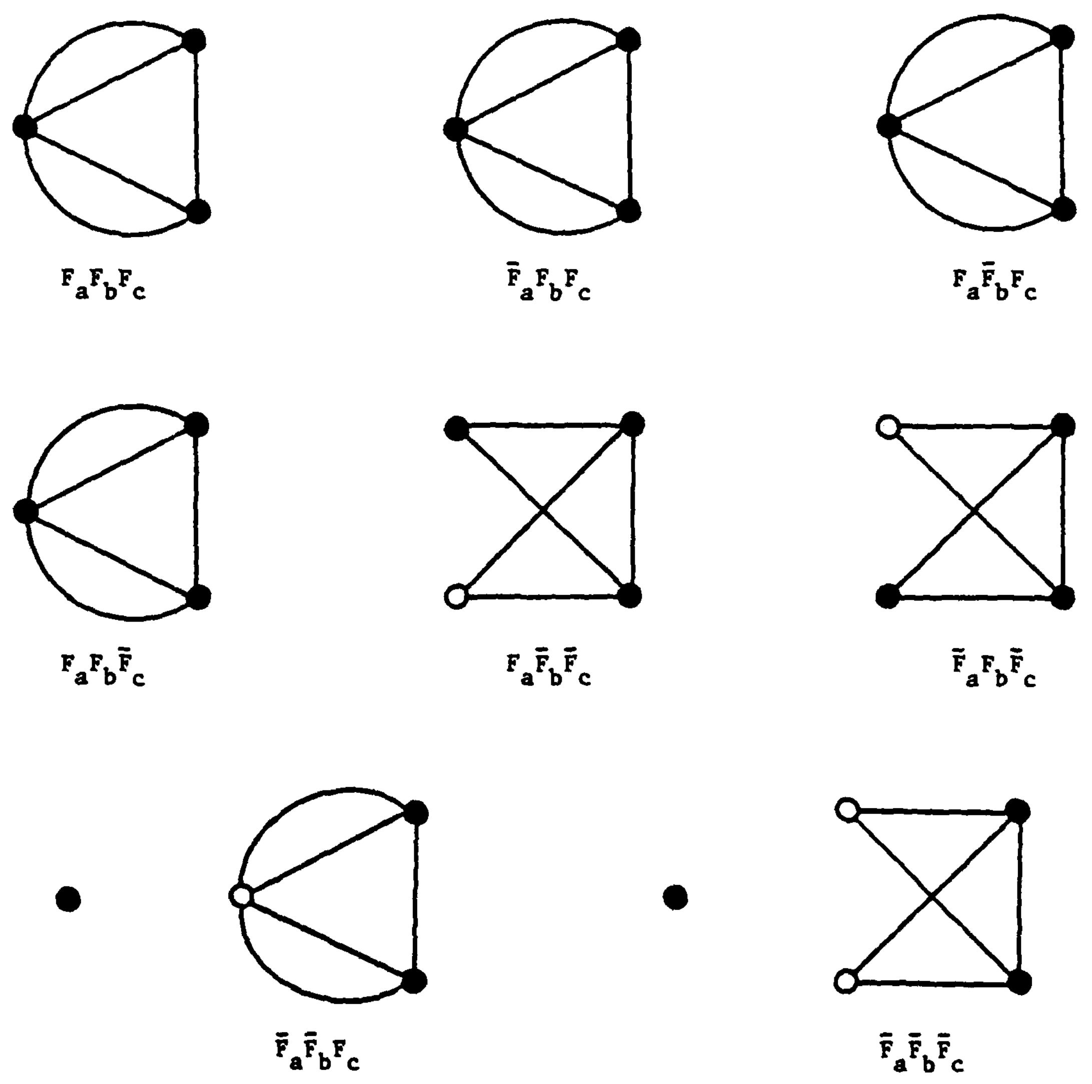


Induced Graphs Obtained by Factoring on e_a

FIGURE 2



(a) Graph G_K with a Type 1 Polygon



(b) Graphs Induced by Factoring on the Edges of the Polygon

FIGURE 3

$$R(G_K) = \sum_{\underline{F} \in \mathbf{F}} p_a^{z_a} q_a^{1-z_a} p_b^{z_b} q_b^{1-z_b} p_c^{z_c} q_c^{1-z_c} R(G_K|\underline{F})$$

Figure 3b shows the eight graphs induced by the terms of the above equation. Note that four of the induced graphs are identical and that two others have zero reliability since $w \in K$ is disconnected in these graphs. With the above introduction, we are now ready to show that a reliability-preserving reduction exists for each of the polygon-to-chain transformations given in Table 1.

Polygon-to-chain reductions:

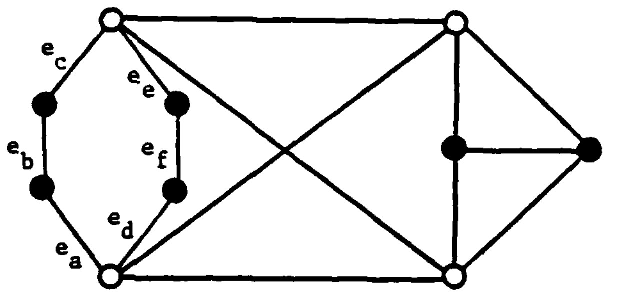
Theorem 1: Suppose G_K contains a type j polygon. Let $G'_{K'}$ denote the graph obtained from G_K by replacing the polygon Δ_j with the chain χ_j having appropriately defined edge probabilities, and let Ω_j be the corresponding multiplication factor, all as in Table 1. Then, $R(G_K) = \Omega_j R(G'_{K'})$.

We prove the exactness of reduction 7 only, since reductions 1-6 may be shown in a similar fashion. Figure 4 illustrates the use of the theorem on a general graph containing a type 7 polygon and Figures 5 and 6 are used to illustrate the proof of the theorem. To improve readability in the proof, we have dropped the subscript "7" on α, β, δ, γ, and Ω even though, strictly speaking, these are all functions of the type of reduction.

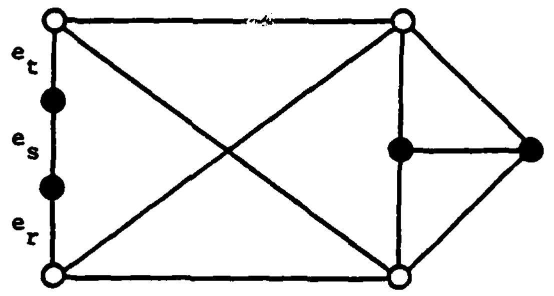
Proof of Theorem 1: Let F_i be the event that edge e_i in the polygon is working and let $\bar{F}_i$ be the event that edge e_i has failed. $\underline{F}$ denotes a compound event or state such as $F_a F_b \bar{F}_c F_d \bar{F}_e F_f$ and $\mathbf{F}$ denotes the set of all 2^6 such states. Also, $z_i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

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

Only sixteen of the possible sixty-four states are non-failed states where $R(G_K|\underline{F}) \neq 0$. Each non-failed state will induce a new graph with a corresponding set of K-vertices of which there are only four different possibilities. Figure 5 gives these four graphs G_{i,K_i}, $i = 1,2,3,4$, plus the summed state probabilities in each case, α, β, δ, and γ. Thus, by grouping and eliminating terms, Equation 2 is reduced to



G_K



$G'_{K'}$

$$\alpha = q_a p_b p_c q_d p_e p_f$$

$$\beta = p_a q_b p_c (q_d p_e p_f + p_d q_e p_f + p_d p_e q_f) + p_a p_b q_c p_f (p_d q_e + q_d p_e) + q_a p_b p_c p_d (q_e p_f + p_e q_f)$$

$$\delta = p_a p_b q_c p_d p_e q_f$$

$$\gamma = p_a p_b p_c p_d p_e p_f \left(1 + \frac{q_a}{p_a} + \frac{q_b}{p_b} + \frac{q_c}{p_c} + \frac{q_d}{p_d} + \frac{q_e}{p_e} + \frac{q_f}{p_f}\right)$$

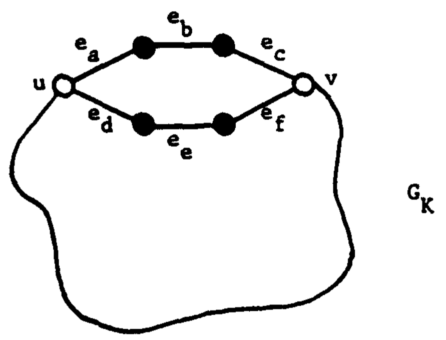
$$p_r = \frac{\gamma}{\alpha + \gamma} \qquad p_s = \frac{\gamma}{\beta + \gamma} \qquad p_t = \frac{\gamma}{\delta + \gamma}$$

$$\Omega = \frac{(\alpha + \gamma)(\beta + \gamma)(\delta + \gamma)}{\gamma^2}$$

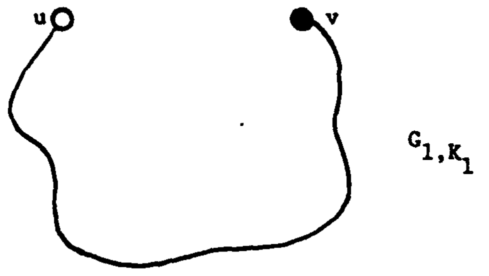
$$R(G_K) = \Omega R(G'_{K'})$$

Type 7 Polygon-to-Chain Reduction

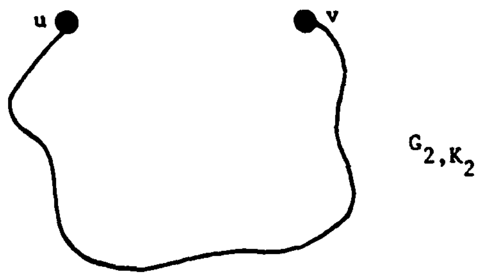
FIGURE 4



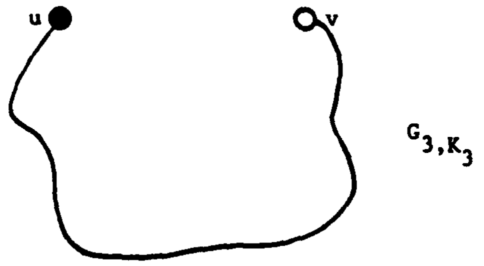
(a) Schematic of a Graph with a Type 7 Polygon



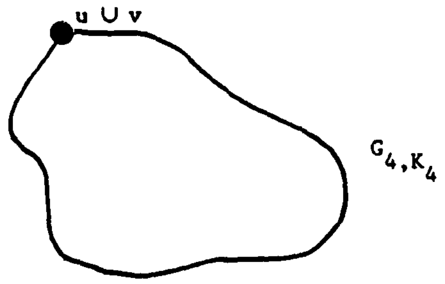
$\alpha = q_a p_b p_c q_d p_e p_f$



$$\begin{aligned}
\beta &= p_a q_b p_c q_d p_e p_f + p_a q_b p_c p_d q_e p_f \\
&+ p_a p_b q_c q_d p_e p_f + p_a p_b q_c p_d q_e p_f \\
&+ p_a p_b q_c p_d q_e p_f + q_a p_b p_c p_d q_e p_f \\
&+ q_a p_b p_c p_d p_e q_f \\
&= p_a q_b p_c (q_d p_e p_f + p_d q_e p_f + p_d p_e q_f) \\
&+ p_a p_b q_c p_f (p_d q_e + q_d p_e) \\
&+ q_a p_b p_c p_d (q_e p_f + p_e q_f)
\end{aligned}$$



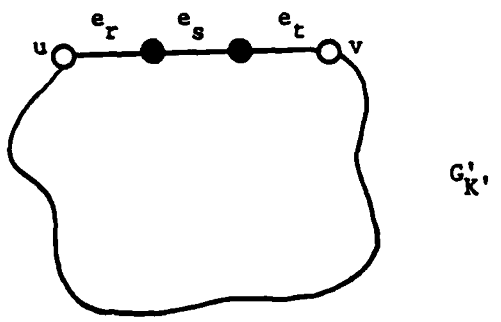
$\delta = p_a p_b q_c p_d p_e q_f$



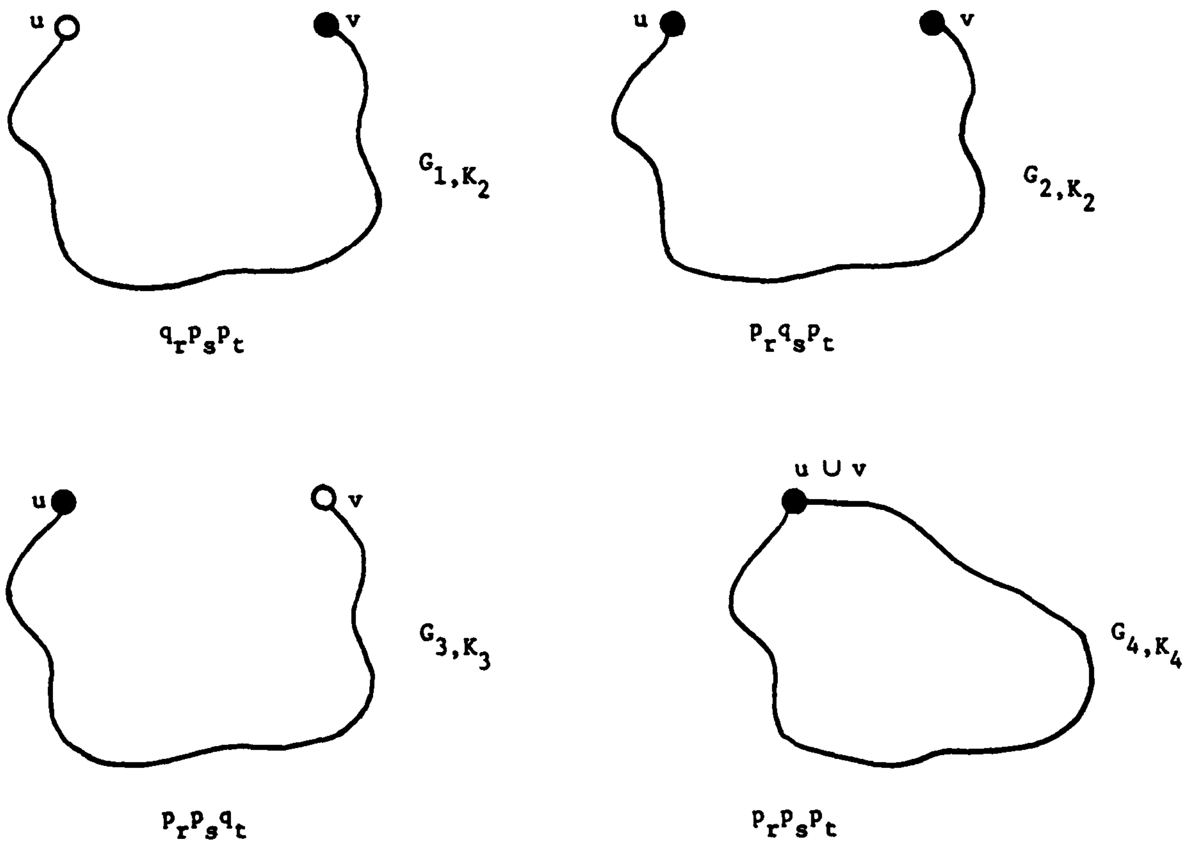
$$\begin{aligned}
\gamma &= p_a p_b p_c p_d p_e p_f + q_a p_b p_c p_d p_e p_f \\
&+ p_a q_b p_c p_d p_e p_f + p_a p_b q_c p_d p_e p_f \\
&+ p_a p_b p_c q_d p_e p_f + p_a p_b p_c p_d q_e p_f \\
&+ p_a p_b p_c p_d p_e q_f \\
&= p_a p_b p_c p_d p_e p_f \left(1 + \frac{q_a}{p_a} + \frac{q_b}{p_b} + \frac{q_c}{p_c} + \frac{q_d}{p_d} + \frac{q_e}{p_e} + \frac{q_f}{p_f}\right)
\end{aligned}$$

(b) Non-failed Induced Graphs

FIGURE 5



(a) Graph of Fig. 5 with Polygon Replaced by Chain



(b) Non-failed Induced Graphs

FIGURE 6

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

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 Figure 6. Using conditional probabilities again,

$$\begin{aligned} R(G'_{K'}) = p_r q_s p_t R(G'_{K'} | (F_r \bar{F}_s F_t)) + q_r p_s p_t R(G'_{K'} | (\bar{F}_r F_s F_t)) \\ + p_r p_s q_t R(G'_{K'} | (F_r F_s \bar{F}_t)) + p_r p_s q_t R(G'_{K'} | (F_r F_s F_t)) \end{aligned} \tag{4}$$

where only the non-failed states have been written.

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

$$\begin{aligned} \Omega\ R(G'_{K'}) = \Omega p_r q_s p_t R(G_{1,K_1}) + \Omega q_r p_s p_t R(G_{2,K_2}) \\ + \Omega p_r p_s q_t R(G_{3,K_3}) + \Omega p_r p_s p_t R(G_{4,K_4}) \end{aligned} \tag{5}$$

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

$$\begin{aligned} \alpha &= \Omega q_r p_s p_t = \Omega(1-p_r) p_s p_t \\ \beta &= \Omega p_r q_s p_t = \Omega p_r (1-p_s) p_t \\ \delta &= \Omega p_r p_s q_t = \Omega p_r p_s (1-p_t) \\ \gamma &= \Omega p_r p_s p_t \end{aligned}$$

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 Equation 4, we obtain

$$\begin{aligned} \Omega\ R(G'_{K'}) &= \alpha R(G_{1,K_1}) + \beta R(G_{2,K_2}) + \delta R(G_{3,K_3}) + \gamma R(G_{4,K_4}) \\ &= R(G_K)\ \Box \end{aligned}$$

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 Figure 5 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 *s-p* complex graphs

In this section we set down some properties of series-parallel and, in particular, *s-p* complex graphs. We prove that *s-p* complex graphs must admit a polygon-to-chain reduction if all simple reductions have first been performed. 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.□

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 series-parallel. □

An important implication of Property 4 is that, if G_K is *s-p* complex, then application of a simple reduction to G_K results in a graph which again is *s-p* complex. On the other hand, a polygon-to-chain reduction on G_K results in a graph which is either *s-p* complex or *s-p* reducible. By next proving that every *s-p* complex 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 an *s-p* complex 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 *s-p* complex, 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 done, and letting $M \leftarrow M(1-q_a q_b)$ whenever a degree-2 reduction is done 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 reliability of an s-p complex graph

The objective here is to develop an efficient, linear-time algorithm for computing the reliability of an s-p complex graph. This algorithm should also compute the reliability of an s-p reducible graph as a special case and tell us if the graph is not series-parallel. 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. If Δ is a polygon formed by two chains $x_1(u,v)$ and $x_2(u,v)$, then we use the notation $\Delta(u,v)=\chi_1(u,v) \bigcup \chi_2(u,v)$. The algorithm is presented next, followed by a short discussion and then a proof of its validity and linear complexity.

Algorithm

MAIN

Input: A nonseparable graph G with vertex set V, $|V|\geqslant 2$, edge set E, $|E|\geqslant 2$, and set $K\subsetneq V$, $|K|\geqslant 2$. 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.

Variables: G_K and all vertex "marks" are represented by global data structures and variables. All other variables are local.

```
(1)  (Initialize) $M \leftarrow 1$.
(2)  (Initialize stack) Construct stack, $T \leftarrow \{v \mid v \in V$ and $\deg(v) > 2\}$ marking all such $v$
     "onstack".
(3)  (Perform all series and degree-2 reductions)
     (a)  For each vertex $v \in V$ such that $\deg(v) = 2$ and $v \notin K$, SERIESREDUCE($v$).
     (b)  For each vertex $v \in V$ such that $\deg(v) = 2$ and $v \in K$, and while $|E| > 2$,
          DEGREE2REDUCE($v, M$).
(4)  If $T$ is empty then
     (a)  ($G$ may be reduced to two parallel edges) If $|E| = 2$ then Print("$R(G_K)=$"
          $M(1-q_a q_b)$ ) and STOP.
     (b)  (Otherwise $G$ has not been completely reduced) Print("$G$ is not series-
          parallel") and STOP.
(5)  ($T$ is not empty) Remove $v$ from $T$ and mark $v$ "offstack".
(6)  ($v$ may have been involved in a reduction since it was put on the stack) If
     $\deg(v) = 2$ or $v$ is marked "deleted" then go to (4).
(7)  (Begin search or continue search for a polygon with one endpoint at $v$) Search
     chains emanating from $v$ until one of two cases occurs:
     (a)  ($v$ is found to be chain-adjacent to 3 distinct vertices) 3 chains
          $\chi(v,u_1)$, $\chi(v,u_2)$, $\chi(v,u_3)$ are found such that $u_1 \neq u_2 \neq u_3 \neq u_1$. In this case go
          to (4).

          or
     (b)  A polygon $\Delta(v,w)$ is found.
(8)  (A polygon has been found, make the polygon-to-chain reduction)
     POLYREDUCE($\Delta(v,w), \chi(v,w), M$).
(9)  If $|E| = |V|$ then ($G$ has been reduced to a single cycle)
     (a)  $T \leftarrow \emptyset$.
     (b)  Go to (3).
(10) ($G$ has not been reduced to a single cycle. Four cases can arise depending on the
     new degrees of $v$ and $w$.)
     (a)  If $\deg(v) > 2$ and $\deg(w) > 2$ then go to (7).
     (b)  If $\deg(v) > 2$ and $\deg(w) = 2$ then
          (i)   CHAINREDUCE($\chi(v,w), \chi(v,y), M$).
          (ii)  If $y$ is "offstack" then mark $y$ "onstack" and put $y$ on $T$.
          (iii) Go to (7).
```

```
        (c) If $\deg(v)=2$ and $\deg(w)>2$ then
            (i) CHAINREDUCE($\chi(v,w),\chi(x,w),M$).
            (ii) If $x$ is "offstack" then mark $x$ "onstack" and put $x$ on $T$.
            (iii) If $w$ is "offstack" then mark $w$ "onstack" and put $w$ on $T$.
            (iv) (Since $T$ cannot be empty) Go to (5).
        (d) (Otherwise $\deg(v)=2$ and $\deg(w)=2$)
            (i) CHAINREDUCE($\chi(v,w),\chi(x,y),M$).
            (ii) If $x$ is "offstack" then mark $x$ "onstack" and put $x$ on $T$.
            (iii) If $y$ is "offstack" then mark $y$ "onstack" and put $y$ on $T$.
            (iv) (Since $T$ cannot be empty) Go to (5).
End of MAIN.
```

SERIESREDUCE(v)

Input: A vertex v such that $v \notin K$ and $\deg(v)=2$.

(This routine reduces G_K by making a series reduction on the two edges incident on v.

```
    (1) Let $x$ and $y$ be the vertices adjacent to $v$ and let $e_a=(v,x)$ and $e_b=(v,y)$.
    (2) (Carry out the reduction)
        (a) Delete edges $e_a$ and $e_b$ from $G_K$.
        (b) Mark v "deleted".
        (c) Add new edge $e_c=(x,y)$ to $G_K$.
        (d) $p_c \leftarrow p_a p_b$
    (3) Return.
End of SERIESREDUCE
```

DEGREE2REDUCE(v,M)

Input: A vertex v such that $v \in K$ and $\deg(v)=2$. Multiplier M.

Output: Revised value of multiplier M.

(This routine reduces G_K by performing a degree-2 reduction at v if v is adjacent to two K-vertices.)

```
    (1) Let $x$ and $y$ be the vertices adjacent to $v$ and let $e_a=(v,x)$ and $e_b=(v,y)$.
    (2) If $x,y \in K$ then
        (a) Delete edges $e_a$ and $e_b$ from $G_K$.
        (b) Mark v "deleted".
        (c) Add new edge $e_c=(x,y)$ to $G_K$.
        (d) $p_c \leftarrow (p_a p_b)/(1-q_a q_b)$
        (e) $M \leftarrow M(1-q_a q_b)$
    (3) Return.
End of DEGREE2REDUCE
```

CHAINREDUCE($\chi(v,w)$,$\chi(s,t)$,M)

Input: A subchain $\chi(v,w)$ obtained in a polygon-to-chain reduction of a polygon $\Delta(v,w)$. A multiplier M.

Output: A completely reduced chain $\chi(s,t)$ obtained from the chain containing the subchain $\chi(v,w)$. New multiplier M.

(This routine finds the chain containing the subchain $\chi(v,w)$ and makes any series and degree-2 reductions to this chain in G_K.)

```
(1) If $deg(v)>2$ and $deg(w)=2$ then
    (a) $s \leftarrow v$
    (b) Search from $w$ away from $v$ to find the first vertex $t$ such that deg$(t)>2$.
(2) If $deg(v)=2$ and $deg(w)>2$ then
    (a) $t \leftarrow w$
    (b) Search from $v$ away from $w$ to find vertex $s$ such that deg$(s)>2$.
(3) If $deg(v)=2$ and $deg(w)=2$ then
    (a) Search from $w$ away from $v$ to find the first vertex $t$ such that deg$(t)>2$.
    (b) Search from $v$ away from $w$ to find the first vertex $s$ such that deg$(s)>2$.
(4) Define the new chain $\chi(s,t)$ which is a superset of the subchain $\chi(v,w)$.
(5) (Make any possible series and degree-2 reductions on $\chi(s,t)$ in $G_K$)
    (a) For each vertex $u \in \chi(s,t)$ such that deg$(u)=2$ and $u \notin$K,
        SERIESREDUCE($v$).
    (b) For each vertex $u \in \chi(s,t)$ such that deg$(u)=2$ and $u \in$K,
        DEGREE2REDUCE($u,M$).
(6) Return.
```

End of CHAINREDUCE

POLYREDUCE($\Delta(v,w)$,$\chi(v,w)$,M)

Input: Polygon $\Delta(v,w)$ and multiplier M.

Output: Chain or subchain $\chi(v,w)$ resulting from polygon-to-chain reduction of $\Delta(v,w)$ and new multiplier M.

(This routine reduces G_K by making the polygon-to-chain reduction on $\Delta(v,w)$ in G_K. It returns the new multiplier M and the chain or subchain $\chi(v,w)$ resulting from the reduction.)

```
(1) Determine which of the 8 types of polygons $\Delta(v,w)$ is, say type $j$.
(2) If $j=8$ then ($\Delta(v,w)$ is two edges in parallel)
    (a) Let $e_a$ and $e_b$ be the two edges forming $\Delta(v,w)$.
    (b) $p_a \leftarrow 1-q_a q_b$
    (c) Delete $e_b$ from $G_K$.
    (d) Let $\chi(v,w)$ be $e_a$.
(2) If $j \leqslant 7$ then (Apply Theorem 1)
```

```
        (a) Update $G_K$ by replacing $\Delta(v,w)$ with appropriate chain $\chi(v,w)$ as given in
            Table 1.
        (b) Compute edge probabilities for edges in $\chi(v,w)$ using the appropriate formulas
            in Table 1.
        (c) Compute $\Omega_j$ from Table 1.
        (d) $M \leftarrow M\Omega_j$
    (4) Return.
End of POLYREDUCE
```

Simplicity and clarity dictate that the algorithm be presented in a form which is not completely structured. The primary departure from a structured program occurs at Step (7) of MAIN where chains emanating from a vertex v are searched. Here the algorithm searches until it finds that v is chain-adjacent to three distinct vertices or until it finds a polygon. If three chain-adjacent vertices are found, the next vertex from the stack T is checked as long as T is not empty. If a polygon is found, then it is reduced by a call to POLYREDUCE. If $\deg(v)$ remains greater than 2 after the reduction, the algorithm returns to Step (7) and continues the search; no chains searched previously from v need be searched again.

Another expediency has been the reduction of G to two edges in parallel instead of to a single edge. This device simplifies the program and allows us to avoid performing a polygon-to-chain reduction on something which is not strictly a polygon by our definition, because both end vertices are of degree 2. One final comment is that the stack T could be any sort of simple linked list, since the order in which the vertices are inserted and removed is unimportant. A stack is just a convenient implementation of a linked list.

The correctness of the algorithm is not hard to show. Arguments similar to those presented here may be found in [13] where the problem is the recognition of two-terminal series-parallel directed graphs. Suppose firstly that G consists of a single cycle. The series and degree-2 reductions at Step (3) (all steps are in MAIN) will reduce G_K to two edges in parallel and T will be empty. The algorithm therefore gives $R(G_K)$ at Step (4.a).

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 at Step (7). Since all initial series and degree-2 reductions were performed at Step (3), by Property 5, any polygon found must be one of the eight specified types. If a polygon is found and then reduced at Step (8), 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 containing the subchain. These reductions are made at Step (10.b), (10.c), or (10.d). Thus, every time Step (7) is entered, the graph admits no series or degree-2 reductions, and any polygon found will be one of the eight given types.

Vertices are continually removed from the stack T and replaced, at most two at a time, only when reductions are made. Since only a finite number of reductions can be made, T must eventually become empty. If $|E|=2$ at that point, then $R(G_K)$ is correctly given at Step (4.a) since only reliability-preserving series, degree-2, and polygon-to-chain reductions are ever performed. Property 4 proves that the original graph must have been series-parallel. Suppose instead that $|E|>2$ when T becomes empty. 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 on the stack and its chain-adjacent vertices checked at Step (7), and (ii) whenever the chain-adjacency of a vertex or vertices is altered (this can occur to at most two vertices at a time) at Step (8), then this vertex or vertices are returned to the stack 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 series-parallel if and only if G' is 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 must provide more details of its implementation. We use a multi-linked adjacency list form to represent the given graph 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. However, since we assume our input graph to be nonseparable and to contain at least two edges, this implies that $|V| \leqslant |E|$, and the complexity is simply $O(|E|)$. For the same reason, the complexity of the whole algorithm will be $O(|E|)$.

Using the above representation, it is obvious that any series or degree-2 reduction can be carried out in constant time. Since POLYREDUCE only needs to check for eight different types of polygons, all of limited size, any polygon-to-chain reduction can also 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, Steps (2) and (3) in MAIN can be performed in $O(|V|)$ time; therefore, we need only consider the central portion of MAIN, Steps (4) through (10). Each time chains emanating from the current vertex v are checked at Step (4), the maximum amount of work which can be performed is some constant amount, i.e., the amount needed to find three chains with distinct end vertices u_1, u_2, and u_3, plus some amount of work proportional to the number of polygon-to-chain reductions made from v. Initially, at most all the vertices can be on the stack, and after every polygon-to-chain reduction, at most two vertices can be returned to the stack. An upper bound on the number of vertices which can ever be checked is therefore $|V|+2(|E|-|V|)=2|E|-|V|$, since at most $|E|-|V|$ polygon-to-chain reductions can ever be performed by POLYREDUCE. For some constant C_1, the total amount of work required until

T becomes empty will thus be bounded by $C_1(2|E|-|V|)$ plus the amount of work required to make all polygon-to-chain reductions using POLYREDUCE and the subchain reductions using CHAINREDUCE.

We have already shown that the amount of work required by a polygon-to-chain reduction in POLYREDUCE is bounded by a constant. However, after a call to POLYREDUCE which reduces $\Delta(v,w)$ to $\chi(v,w)$, a call to CHAINREDUCE is necessary if $\deg(v)=2$ or $\deg(w)=2$. The chain $\chi(s,t)$ containing the subchain $\chi(v,w)$ must be identified and then reduced, if possible, with series and degree-2 reductions. This can be accomplished in constant time also, since the length of any chain $\chi(s,t)$ is at most 9. This worst case could occur if $\deg(v)=\deg(w)=2$ after the polygon-to-chain reduction of $\Delta(v,w)$ to $\chi(v,w)$, and the subchains $\chi(s,v)$, $\chi(w,t)$, and $\chi(v,w)$, which were proper chains before the reduction, are at their maximum possible lengths of 3. It therefore follows that the amount of work associated with a polygon-to-chain reduction, a call to POLYREDUCE and a possible call to CHAINREDUCE, is bounded by some constant, say C_2, and that the amount of work associated with all polygon-to-chain reductions is bounded by $C_2(|E|-|V|)$. We can now see that the amount of work required until T is empty is bounded by $C_1(2|E|-|V|)+C_2(|E|-|V|)$. The only work which is unaccounted for comes from the final series and degree-2 reductions which may be necessary if G is reduced to a single cycle, but again, this is an $O(|V|)$ operation. Thus, we have proved 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 algorithm

The algorithm of section 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 section 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, which can easily be done by searching all chains emanating from all vertices v with $\deg(v)>2$.

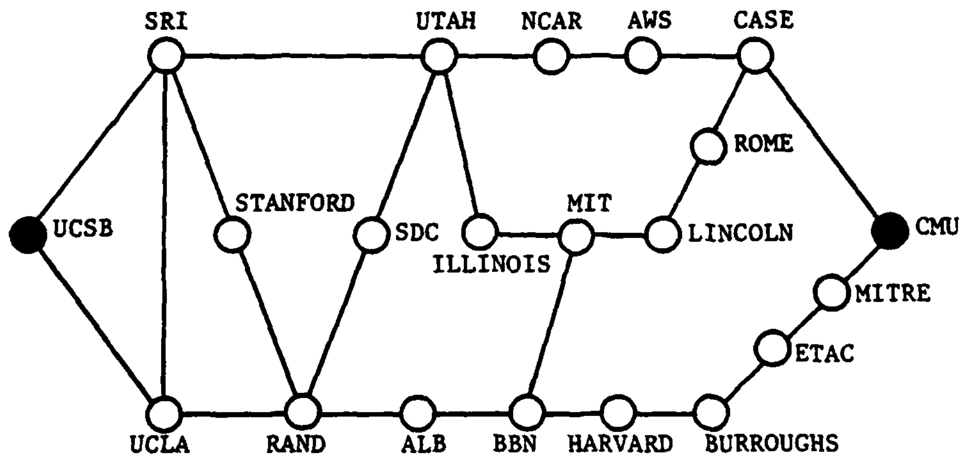
To extend the reduction algorithm as described, Step (4) in MAIN may be replaced by the following, Step(4'):

Algorithm extension, changes to MAIN

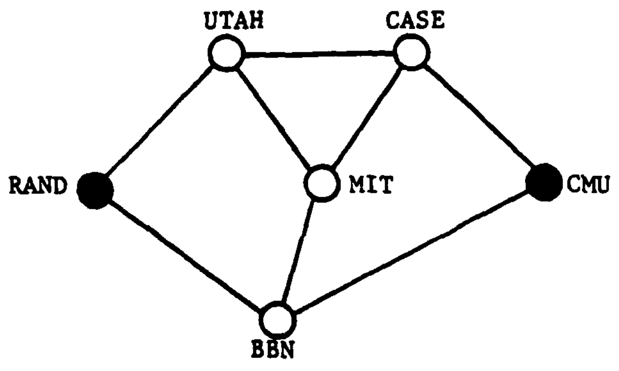
```
(4') If $T$ is empty then
     (a) ($G$ may be reduced to two parallel edges, $e_a$ and $e_b$) If $|E|=2$ then
         (i)  $R(G_K)=M(1-q_a q_b)$.
         (ii) Return to general algorithm with $R(G_K)$.
     (b) (Otherwise $G$ has not been completely reduced) For each $v \in V$ such that
         $\deg(v)>2$, search all chains emanating from $v$ calling
         POLYREDUCE($\Delta(v,w)$, $\chi(v,w)$,M) whenever a polygon
         $\Delta(v,w)=\chi_1(v,w) \bigcup \chi_2(v,w)$ is found.
     (c) Return to general algorithm with reduced $G_K$ and with $M$.
```

In the worst case at Step (4'.b), each chain and thus each edge must be searched twice. 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 Figure 7a [4]. Suppose we are interested in the terminal-pair reliability between UCSB and CMU. Application of the extended algorithm yields a reduced network as shown in Figure 7b 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.



(a) ARPA Computer Network



(b) Reduced Network

FIGURE 7

References

[1] R.E. Barlow and F. Proschan, *Statistical Theory of Reliability*, Holt, Rinehart and Winston, New York, N.Y. (1975).

[2] N. Deo, *Graph Theory with Applications to Engineering and Computer Science*, Prentice-Hall, Englewood Cliffs, N.J., (1974).

[3] R.J. Duffin, "Topology of Series-Parallel Networks," *J. Math. Analysis and Applications*, **10**, 303-318 (1965).

[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**, 203-211 (1973).

[5] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman and Co., San Francisco, California, (1979).

[6] J.N. Hagstrom, "Combinatoric Tools for Computing Network Reliability," Ph.D. Thesis, University of California, Berkeley, (1980).

[7] E. Hansler, G.K. McAulife, R.S. Wilkov, "Exact Calculation of Computer Network Reliability," *Networks*, **4**, 95-112 (1974).

[8] P.M. Lin, B.J. Leon, T.C. Huang, "A New Algorithm for Symbolic System Reliability Analysis," *IEEE Trans. Reliability*, **R-25**, 2-15 (1976).

[9] K.B. Misra, "An Algorithm for the Reliability Evaluation of Redundant Networks," *IEEE Trans. Reliability*, **19**, 146-151 (1970).

[10] F. Moskowitz, "The Analysis of Redundancy Networks," *AIEE Trans. (Commun. Electron.)*, **77**, 627-632 (1958).

[11] A. Satyanarayana and M.K. Chang, "Network Reliability and the Factoring Theorem," *Networks*, to appear. Also, ORC 81-12, Operations Research Center, University of California, Berkeley, (1981).

[12] J. Sharma, "Algorithm for Reliability Evaluation of a Reducible Network," *IEEE Trans. Reliability*, **25**, 337-339 (1976).

[13] J. Valdes, R.E. Tarjan, and E.L. Lawler, "The Recognition of Series Parallel Digraphs," *SIAM J. Computing*, to appear.

DATE
ILME