

Reliability of Partial k -tree Networks

Erick Mata-Montero

CIS-TR-90-14

June 8, 1990

Abstract

Recent developments in graph theory have shown the importance of the class of partial k -trees. This large class of graphs admits several algorithm design methodologies that render efficient solutions for a large number of problems inherently difficult for general graphs. In this thesis we develop such algorithms to solve a variety of reliability problems on partial k -tree networks with node and edge failures. We also investigate the problem of designing uniformly optimal 2-trees with respect to the 2-terminal reliability measure.

We model a communication network as a graph in which nodes represent communication sites and edges represent bidirectional communication lines. Each component (node or edge) has an associated probability of operation. Components of the network are in either operational or failed state and their failures are statistically independent. Under this model, the reliability of a network G is defined as the probability that a given connectivity condition holds. The l -terminal reliability of G , $Rel_l(G)$, is the probability that any two of a given set of l nodes of G can communicate. Robustness of a network to withstand failures can be expressed through network resilience, $Res(G)$, which is the expected number of distinct pairs of nodes that can communicate. Computing these and other similarly defined measures is #P-hard for general networks.

We use a dynamic programming paradigm to design linear time algorithms that compute $Rel_l(G)$, $Res(G)$, and some other reliability and resilience measures of a partial k -tree network given with an embedding in a k -tree (for a fixed k).

Reliability problems on directed networks are also inherently difficult. We present efficient algorithms for directed versions of typical reliability and resilience problems restricted to partial k -tree networks without node failures. Then we reduce to those reliability problems allowing both node and edge failures.

Finally, we study 2-terminal reliability aspects of 2-trees. We characterize uniformly optimal 2-trees, 2-paths, and 2-caterpillars with respect to Rel_2 and identify local graph operations that improve the 2-terminal reliability of 2-tree networks.

VITA

NAME OF THE AUTHOR: Erick Mata-Montero

PLACE OF BIRTH: San José, Costa Rica

DATE OF BIRTH: November 6, 1958

GRADUATE AND UNDERGRADUATE SCHOOLS ATTENDED:

University of Oregon
Universidad de Costa Rica

DEGREES AWARDED:

Doctor of Philosophy, 1990, University of Oregon
Master of Science, 1986, University of Oregon
Licentiate, 1981, Universidad de Costa Rica
Bachelor, 1980, Universidad de Costa Rica

AREAS OF SPECIAL INTEREST:

Algorithmic Graph Theory
Analysis of Algorithms
Object-Oriented Systems
Programming Languages

PROFESSIONAL EXPERIENCE:

Graduate Teaching Fellow, Department of Computer and Information Science,
University of Oregon, Eugene, 1989-90

Graduate Research Assistant, Department of Computer and Information Science,
University of Oregon, Eugene, 1988-89

Graduate Teaching Fellow, Department of Computer and Information Science,
University of Oregon, Eugene, 1988

Graduate Researcher, Department of Computer and Information Science,
University of Oregon, Eugene, 1987

Instructor, Department of Computer Science, Instituto Tecnológico de Costa Rica,
Cartago, 1980-84

Instructor, Department of Computer Science, Universidad de Costa Rica, San José,
1981-82

Research Assistant, School of Biology, Universidad de Costa Rica, San José, 1980

Teaching Assistant, School of Mathematics, Universidad de Costa Rica, San José,
1977-79

AWARDS AND HONORS:

Gross Scholarship, University of Oregon, 1988-89

Tektronix Fellowship, 1987

USAID/Instituto Tecnológico de Costa Rica Scholarship, 1984-86

Licenciate Thesis, approved with honors, Universidad de Costa Rica, 1980

Honors Scholarship, Universidad de Costa Rica, 1977

PUBLICATIONS:

MATA-MONTERO, E. Resilience of Partial k -tree Networks. In *Congressus
Numerantium* (1990), vol. 74, pp. 107-123.

MATA-MONTERO, E. Resilience of Partial k -trees with edge and node Failures.
Tech. Rep. CIS-TR-89-15, University of Oregon, 1989.

ACKNOWLEDGEMENTS

I wish to express my gratitude to my adviser, Andrzej Proskurowski. Andrzej not only introduced me to the study of partial k -trees and combinatorics of network reliability, but also provided many ideas, challenges, insight, and support throughout this research.

Special thanks are due to Charles Colbourn for his many suggestions and for his encouragement. I am also indebted to the Computer and Information Science Department and to the Office of Naval Research for their partial support to this research.

DEDICATION

To my wife, Alejandra

TABLE OF CONTENTS

Chapter	Page
I.	INTRODUCTION 1
	Network Reliability 1
	Terminology 3
	Thesis Overview 4
II.	PARTIAL K -TREES 6
	Definitions 7
	Background 8
	Paths and Caterpillars 12
	An Algorithm Design Methodology for Partial k -trees 13
	Complexity Issues 18
III.	EFFICIENT EXACT ALGORITHMS FOR PARTIAL K -TREE NETWORKS 22
	Introduction 22
	Definitions 22
	Efficient Exact Algorithms 26
	l -Terminal Reliability 27
	Resilience 38
	Broadcast Resilience 57
	l -Broadcast Resilience 58
	Network Broadcast Facility Location Problem 60
	Concluding Remarks 60
IV.	EFFICIENT EXACT ALGORITHMS FOR DIRECTED PARTIAL K -TREE NETWORKS 63
	Introduction 63
	Terminology 64
	Efficient Exact Algorithms 65
	s, L -Connectedness 66
	Reachability 70
	Directed Broadcast Resilience 72

<i>l</i> -Directed Broadcast Resilience	73
Directed Resilience	73
Node Failures	73
V. UNIFORMLY OPTIMAL 2-TREES WITH RESPECT TO REL_2	81
Terminology	82
Uniformly optimal 2-trees with Adjacent Distinguished Nodes	83
Uniformly Optimal 2-trees	90
Uniformly Optimal 2-paths	106
Uniformly Optimal 2-caterpillars	107
Alternative Optimality Criteria	108
Summary and Final Remarks	115
VI. CONCLUSIONS AND FUTURE RESEARCH	120
BIBLIOGRAPHY	123

CHAPTER I

INTRODUCTION

Network Reliability

The reliability of a network is a measure of the network's ability to perform a given communication task in the presence of failures. The effect of a failure on the overall performance of a network may vary from unnoticeable (e.g., failure of a communication site that is not used in the communication task), to serious (e.g., long delays induced by re-routing of messages), to catastrophic (e.g., failures of all the nodes in a cutset). Combinatorial methods are a powerful tool in the analysis of network reliability problems and in the design of reliable networks.

Research in combinatorics of network reliability focuses on three main areas: the formulation of realistic reliability criteria, the development of analytical tools to compare networks according to these criteria, and the synthesis of networks that are optimal with respect to these criteria. Our research is concerned with the latter two objectives. We are interested in the development of efficient algorithms to compute network reliability measures and in characterizing optimal networks with respect to a variety of reliability measures.

Network failures may occur at the protocol and at the topological level. The typical cause of failures at the protocol level is a bad routing algorithm. A routing algorithm may fail to meet delay requirements, it may cause traffic bottlenecks that flood the network, or it may fail to detect an operational path even though one exists.

Failures at the topological level can be classified as deterministic or probabilistic. We can view deterministic failures as the result of adversary attacks that follow a deterministic pattern. For example, the adversary may follow the rule of destroying at most k communication lines at a time. Under this assumption we may use edge connectivity as

a reliability criterion and investigate the corresponding analysis and synthesis problems, namely, defining efficient algorithms to compute edge connectivity of a network and finding optimal networks with respect to edge connectivity. Deterministic reliability problems have been well studied. We can, for example, use classical work on network flow algorithms to solve problems concerning edge connectivity. A thorough account of the early work in deterministic network reliability can be found in [31]; more recent work is surveyed in [19].

Probabilistic failures can be viewed as the result of random adversary attacks or random component wearout. We assume that each component (node and edge) of the network has an associated probability of operation and that the operation of a component does not affect the operation of any other component. Under this model, the reliability of a network G with respect to a given communication task (e.g., broadcast) is the probability that the communication task can be carried out in G . In spite of the assumption of statistical independence of edge failures, it is generally agreed that probabilistic reliability measures are, in general, more realistic and useful than deterministic reliability measures. However, it is also well known that most of the analysis problems in probabilistic models are $\#P$ -complete. For instance, computing the 2-terminal reliability of a network is $\#P$ -complete, even when the network is planar [55].

Synthesis problems in probabilistic models also seem to be extremely difficult. For instance, a famous conjecture by Boesch, concerning the existence of uniformly optimal (n, m) graphs with respect to all-terminal reliability, only recently has been proved wrong [47]. Colbourn presents an excellent survey of the work on probabilistic network reliability in [23].

The apparent intractability of the probabilistic reliability problems has led to the development of efficient approximation algorithms and of exact algorithms for restricted classes of graphs (see [23] for a comprehensive discussion). The first part of our research is concerned with the development of exact algorithms to compute probabilistic reliability measures on partial k -tree networks.

We have three main reasons for restricting the study of exact algorithms to partial k -tree networks. First of all, the class of partial k -trees, even for small values of k , is

considerably general. For instance, the class of partial 4-trees contains the class of series-parallel, Halin, Δ -Y, and outerplanar graphs [15, 28].

Secondly, Arnborg and Proskurowski [10] devised an algorithm design methodology for partial k -trees that leads to the development of efficient, often linear time, algorithms for a variety of #P-hard problems restricted to partial k -trees. We have employed this technique to solve a number of #P-complete probabilistic reliability problems in linear time, when the network is a partial k -tree, k is fixed, and an embedding in a k -tree is given.

Thirdly, exact algorithms for partial k -trees may be useful in approximating the probabilistic reliability of general networks. Edge packing is a well known heuristic technique that gives an approximation of the probabilistic reliability of a network. We can approximate the reliability of G by finding an edge-packing of G (using some heuristic method), computing the exact reliability of each subgraph in the collection, and then combining these exact reliabilities. Colbourn [23] has suggested edge-packing by partial 2-trees as a promising alternative to obtain better lower bounds for some reliability problems. We believe that better, and still efficient, approximation algorithms may result if we use edge packings by partial k -trees ($k = 3$ or $k = 4$) and the exact algorithms presented in Chapter III and Chapter IV.

The second part of our research is concerned with synthesis problems. We identify uniformly optimal 2-tree networks with respect to the 2-terminal reliability measure. Our interest is in characterizing the best (most reliable) and worst (least reliable) k -trees with respect to some reliability measure and in better understanding the relationships between the k -tree topology and the reliability of the underlying network. Neufeld and Colbourn [50] characterized the best 2-trees with respect to all-terminal reliability. Clark et al. [21] characterized the best 2-paths with respect to resilience.

Terminology

Except for a few explicitly defined concepts, we use the basic graph theoretic terminology as defined in [37]. Our terminology concerning network reliability follows the

terminology in [23]. In particular, we want to emphasize that we are concerned with the study of probabilistic models of reliability only. We present terminology specific to partial k -trees and network reliability in Chapter II and Chapter III, respectively.

Thesis Overview

This thesis is organized as follows. Chapter II discusses the algorithm design methodology of Arnborg and Proskurowski [10]. We open the section with some historical background, a brief overview of related work, and some fundamental definitions and results concerning partial k -trees. We conclude this section with a discussion of complexity issues in the design of algorithms for partial k -trees.

Chapter III presents our results concerning exact algorithms for probabilistic reliability measures on undirected partial k -tree networks. A brief presentation of fundamental definitions and complexity issues precedes the description of our algorithms. Then we use the algorithm design methodology presented in Chapter II to construct linear time algorithms computing the l -terminal reliability, l -broadcast resilience, and resilience of a network, as well as a polynomial time algorithm solving network broadcast facility problems [49].

Reliability problems on directed networks are extremely difficult. Even though there is a simple technique to reduce some undirected reliability problems to their directed counterparts. There is no known technique to reduce directed reliability problems to their undirected counterparts maintaining the input network a partial k -tree. Therefore, the algorithms in Chapter III are not useful to solve directed network reliability problems. Chapter IV discusses techniques to obtain polynomial time algorithms for the directed versions of the reliability problems solved in Chapter III. The network model is initially simplified by assuming that nodes are fail-safe. Then we discuss how to generalize the algorithms to deal with node failures.

Chapter V presents our results concerning the synthesis of uniformly optimal partial k -trees ($k = 2$). The results in Chapter V characterize the best and worst 2-trees, 2-paths, and 2-caterpillars with respect to the 2-terminal reliability. Similar to the results of [50],

the results in this chapter can also be used to improve the reliability of a network by performing simple local operations on the topology of the network.

Chapter VI presents a summary of our contributions and future research directions.

CHAPTER II

PARTIAL k -TREES

The success of a model depends on a delicate balance between the modeling power and the analytical power of the model. The modeling power of a model measures how general the domain of application of the model is. For example, we could use general graphs or trees to model computer communication networks. Evidently, the modeling power of general graphs is larger than the modeling power of trees. However, trees are a very attractive model because many difficult problems for general graphs are easy for trees.

The analytical power of a model measures how good the model is as an analytical tool. The computational complexity of questions that the model is expected to help solve determines the analytical power of the model. Not surprisingly, the modeling power and the analytical power of a model are typically in conflict. Increasing the modeling power of a model without substantially degrading its analytical power is an important accomplishment.

The class of partial k -trees is an attractive subject of study because both its modeling and its analytical power are “very good.” The modeling power of the class of partial k -trees is illustrated by the number of important families of graphs that are contained in the class of partial k -trees (e.g., series-parallel, Halin, and bounded cutwidth graphs) [15]. Their analytical power is suggested by the variety of graph theoretic NP-complete problems that have been solved in polynomial and even linear time when restricted to partial k -trees (for a fixed value of k) [10, 16, 43]. From a graph theoretical point of view, the interesting structural properties of partial k -trees make them a research subject in their own right.

Definitions

We follow the graph theoretic terminology in [37]. Unless otherwise specified, a graph $G = (V, E)$ is loopless, undirected, and simple. We also denote the set of nodes of G as $V(G)$ and the set of edges of G as $E(G)$. A clique of G is a (not necessarily maximal) complete subgraph of G . A k -clique is a clique that has exactly k nodes. A graph H is a partial graph of G if H is a spanning subgraph of G . We use $H \subseteq G$ to denote that H is a partial graph of G and $H \leq G$ to denote that H is a subgraph of G . For any set W of nodes of G , the subgraph induced by W is the maximal subgraph of G with node set W . For any given node v in $V(G)$, the attachment of v in G is the subgraph of G induced by the nodes adjacent to v .

Let G and H be two graphs, the union of G and H is the graph $G \cup H = (V(G) \cup V(H), E(G) \cup E(H))$. The complement \overline{G} of graph G has $V(G)$ as its node set, but two nodes in \overline{G} are adjacent if and only if they are not adjacent in G . We use $G - v$ to denote the subgraph of G induced by $V(G) \setminus \{v\}$. Similarly, for any subset of nodes $W \subseteq V(G)$, $G - W$ denotes the subgraph of G induced by $V(G) \setminus W$. The subset of nodes W is a graph separator of G if $G - W$ has more connected components than G . Given a pair of nodes u, v in $V(G)$ and a subset of nodes $W \subseteq V(G)$, W is a u - v separator if u and v are connected in G but disconnected in $G - W$.

A graph is chordal if every cycle with four or more nodes has a chord, i.e., an edge between two nodes that are not consecutive in the cycle. ¹ A perfect elimination ordering (peo) of a graph G is a one-to-one numbering v_1, \dots, v_n of the nodes of G such that for each i ($i = 1, \dots, n$), the higher-numbered neighbors of v_i form a clique. A graph is chordal if and only if it has a perfect elimination ordering [32].

A graph G is k -decomposable if it meets either of the following conditions [9]:

- (a) G has $k + 1$ or fewer nodes;

¹Chordal graphs have been re-discovered and renamed several times. Some of the names given to chordal graphs are: perfect elimination, triangulated, and circuit rigid graphs.

- (b) There is a subgraph S of G , with at most k nodes, such that $G - V(S)$ is disconnected and has connected components C_1, \dots, C_l , and each of the subgraphs of G induced by $V(S) \cup V(C_i)$, augmented by the edges in $E(\bar{S})$ is k -decomposable.

For the following definitions let us consider graphs that contain both multiple edges and loops. A contraction of an edge is the replacement of two adjacent nodes and the edge between them with a single node, with all edges previously incident to either of the removed nodes now being incident to the new node. A graph G contains another graph H as a minor if a subgraph of G can be transformed into H via a sequence of contractions. The MC_H problem (Minor Containment with respect to a fixed graph H) consists of determining if a fixed graph H is a minor of the input graph.

Background

Beineke and Pippert [14] initiated the study of k -trees by generalizing a recursive, generative definition of trees (1-trees) as follows. Let k be a fixed positive integer. A graph is a (full) k -tree if it satisfies either of the following conditions:

- (a) It is the complete graph on k nodes, K_k .
- (b) It has a node v of degree k with completely connected neighbors, and the graph obtained by removing v and its incident edges is a k -tree.

A graph is a partial k -tree if it is a partial graph of a k -tree. Figure 1 shows some examples of k -trees and partial k -trees. Notice that, because we can always find a perfect elimination ordering for any k -tree, k -trees are chordal graphs. In fact, it is easy to verify that for any k -tree on n nodes the higher numbered neighbors of the first $n - k$ nodes of a perfect elimination ordering induce a k -clique. It is also easy to verify that a graph is a partial k -tree if and only if it is k -decomposable. A node v is a k -leaf if its neighbors induce a k -clique K .

Three years after the paper by Beineke and Pippert was published, an independent paper by Rose [58] introduced k -trees and established many of their essential properties. By that time, chordal graphs had already been the object of algorithmic study. For

example, Rose [57] applied chordal graphs to Gaussian elimination of sparse symmetric systems of linear equations, and Gavril [33] developed polynomial time algorithms for a variety of NP-complete problems restricted to chordal graphs. Research on chordal graphs dates back to the early sixties [26].

Neither [14] nor [58] mention the class of partial k -trees. At the time researchers were not aware of the modeling and analytical power of partial k -trees. However, forests, series-parallel graphs, and other restricted cases of partial k -trees had already been studied (e.g., [27, 36]). Subsequent findings of algorithmic properties of chordal graphs (e.g. [34, 59]) increased the interest in the class of k -trees during the seventies and early eighties. At the same time, new classes of graphs, which were later found to be special cases of partial k -trees, were defined and studied as independent classes (e.g., outerplanar [37], bounded bandwidth [46], and Δ -Y graphs [51]).

The last decade has been a period of synthesis of the multiple, scattered results accumulated during the seventies. Robertson and Seymour's work on graph minors paved the way for the realization of the modeling and analytical power of partial k -trees [56]. Using Robertson and Seymour's terminology, we can define the class of partial k -trees as the class of graphs with tree width $\leq k$. This alternative definition has facilitated the proof that several important classes of graphs are partial k -trees for some value of k [15] and has opened new avenues of research [38].

Robertson and Seymour define the class of graphs of tree-width $\leq k$ as follows. Let $G = (V, E)$ be a graph. A tree decomposition of G is a pair $(\{X_i \mid i \in I\}, T = (I, F))$,

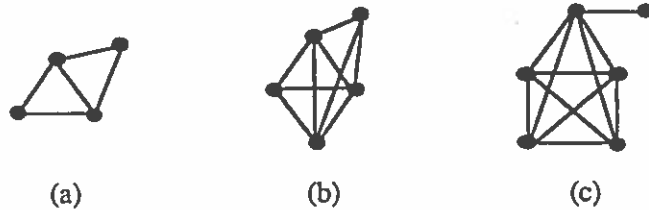


Figure 1: (a) A 2-tree (b) A 3-tree (c) A partial 4-tree.

such that the set $\{X_i \mid i \in I\}$ is a family of subsets of V and T is a tree with the following properties:

- (a) $\bigcup_{i \in I} X_i = V$.
- (b) For each edge $e = \{v, w\}$ in E , there is a subset X_i , $i \in I$, such that $v \in X_i$ and $w \in X_i$.
- (c) For all i, j, k in I , if j lies on the path in T from i to k , then $X_i \cap X_k \subseteq X_j$.

The tree-width of a tree decomposition $(\{X_i \mid i \in I\}, T = (I, F))$ is $\max_{i \in I} |X_i| - 1$. The tree-width of G is the minimum tree-width of a tree decomposition of G , taken over all possible tree decompositions of G .

For several years, Robertson and Seymour have been working on what they call the “Graph Minors” project. This research has turned into a series of fifteen reports with eight more planned. The original goal of this research was to prove Wagner’s conjecture that for every infinite set of graphs one graph is a minor of another (technically, this can be stated by saying that the class of all graphs, ordered by minors, is well quasi-ordered [61]). However, the theory developed to prove Wagner’s conjecture has turned out to have other applications. We will not attempt to survey Robertson and Seymour’s research here. Instead, we will present some highlights to illustrate its consequences (see [38] for an excellent overview).

Robertson and Seymour proved Wagner’s conjecture with the following theorem.

Theorem 2.1 (Robertson and Seymour) A minor-closed family \mathcal{F} of graphs is describable as the set of all graphs that do not contain, as a minor, any of the graphs in a certain fixed, finite set of graphs \mathcal{L} .

The set \mathcal{L} in Theorem 2.1 is called the set of minimal forbidden minors of the family \mathcal{F} . This theorem provides a powerful generalization of Kuratowski’s theorem characterizing planar graphs by their forbidden substructures.

Another easy but very important corollary of Theorem 2.1 is that there is a fixed, finite set $\mathcal{L} = \mathcal{L}(k)$ of forbidden minors for the class of partial k -trees. Therefore, we

can test if a graph G is a partial k -tree by testing if an element of \mathcal{L} is a minor of G . This approach is useful if we know not only the set \mathcal{L} but also an efficient algorithm to solve MC_H . Unfortunately, Robertson and Seymour's results do not tell us how to find the set of forbidden minors \mathcal{L} ; they do not even indicate what the cardinality of the set of forbidden minors \mathcal{L} is.² Also, even though in another breakthrough Robertson and Seymour solve MC_H in polynomial time (quadratic time if H is planar, cubic time otherwise), the enormous constants involved make the result of theoretical interest only. One of the constants involved is a tower of 2's whose number of levels of exponentiation is worse than exponential in the number of nodes of H . David Johnson commented on this aspect in [38]: "Unfortunately, for any instance $G = (V, E)$ than one can fit into the known universe, one would easily prefer $|V|^{70}$ to even constant time, if that constant had to be one of Robertson and Seymour."³

The consequences of the research by Robertson and Seymour are far reaching. They not only resolved a large number of fundamental open problems but also illustrated, in a rather dramatic form, how the concept of polynomial time algorithm does not always capture our intuition of "efficient" algorithm. Also, some of their results are non-constructive. So, the belief that, a proof of $NP = P$ would transform the proofs of NP-completeness of many problems into polynomial time algorithms, is not necessarily valid. On the positive side, the possibility of using these powerful results to prove, albeit non-constructively, that some problems are in P may help researchers to focus their attention on finding the algorithms that they know must exist for those problems. Stimulated by those results, a good number of problems believed to be hard have been proved, non-constructively, to be in P (see [29] for a survey).

The development of reduction methodologies that render efficient algorithms for many NP-complete problems restricted to partial k -trees [10, 16] is another fundamental result achieved in the eighties. This algorithm design methodology has made evident the

²Arnborg et al. [11] discovered the set $\mathcal{L}(3)$, which has cardinality four. The class of graphs with path width 2 has been characterized in [30] by 110 minimal forbidden minors.

³D. Johnson, "The NP-completeness column: an ongoing guide" (1987), p. 289.

analytical power of partial k -trees and has provided a uniform perspective for the analysis and design of algorithms for partial k -trees. We discuss this algorithm design methodology later in this chapter.

Today, chordal graphs, partial k -trees, and other classes of tree-like graphs are the object of very active research. ⁴ Applications to the analysis and design of reliable computer networks [24, 43, 60], and current research on parallel algorithms [18, 20, 40, 48] have further added to the interest in these classes.

Paths and Caterpillars

In this section we define some subclasses of k -trees that play an important role in the study of uniformly most reliable k -trees. A k -clique K of a k -tree is either peripheral, separator, or exterior. K is a separator if $V(K)$ is a graph separator, it is peripheral if one of its nodes is a k -leaf, and it is exterior if neither of the conditions is fulfilled. A k -path is a k -tree with less than $k+2$ nodes or with exactly two k -leaves. A k -path is a k -fan if it has less than $k+2$ nodes or if its k -leaves share exactly $k-1$ neighbors. A k -path is a k -line if there is a perfect elimination ordering v_1, \dots, v_n such that the higher numbered neighbors of v_i ($1 \leq i \leq n-k$) are v_{i+1}, \dots, v_{i+k} . Figure 2 presents some examples of 2-paths. The following theorem, a generalization of a property of trees, is used in Chapter V.

Theorem 2.2 (Proskurowski [52]) In a k -tree G , the vertices of minimal subgraphs separating two non-adjacent nodes induce a k -path.

A k -caterpillar consists of a body and zero or more hairs. The body of a k -caterpillar is a k -path and each hair is a k -leaf whose attachment is a k -clique separator of the body. A k -book is a k -tree with $n-k$ k -leaves, i.e., a k -caterpillar with a k node body. The following theorem establishes an interesting relationship between k -trees and $(k-1)$ -caterpillars.

⁴Several other classes of tree-like graphs have been defined; for example, hook-up graphs [39], classes generated by context-free hyperedge replacement grammars [35], and k -terminal graphs [63, 62]. These are all classes of graphs with bounded tree-width. We refer the reader to [54] for a concise overview of tree-like classes of graphs.

Theorem 2.3 (Proskurowski [52]) Given a k -tree ($k \geq 2$) and two non-adjacent nodes u and v , the union of k -cliques separating u and v is a $(k - 1)$ -caterpillar.

Robertson and Seymour defined path decomposition and path-width of a graph similarly to tree decomposition and tree-width; the only difference is that in the decomposition $(\{X_i \mid i \in I\}, T = (I, F))$, T is a path. The following result establishes that graphs of bounded path-width and k -caterpillars are closely related.

Theorem 2.4 (Proskurowski [53]) A graph has path-width $\leq k$ if and only if it is a partial k -caterpillar.

An Algorithm Design Methodology for Partial k -trees

Arnborg and Proskurowski [10] have defined an algorithm design methodology, a reduction paradigm, for partial k -trees that leads to the development of efficient reduction algorithms for a variety of NP-hard problems restricted to partial k -trees. The reduction paradigm assumes that k is a fixed positive integer and that the input partial k -tree is given with an embedding in a k -tree. To simplify our presentation, we will discuss this reduction paradigm assuming that the input graph is a k -tree rather than a partial k -tree given with an embedding in a k -tree. In the last section of this chapter we discuss complexity issues that arise when the input graph is not a full k -tree.

The k -decomposability of k -trees suggests a top-down, recursive approach to solve a given problem X on a k -tree G . If the k -tree G has exactly k nodes (i.e., G is a k -clique)

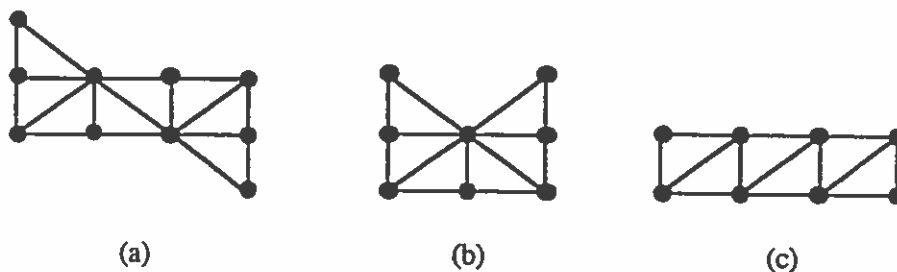


Figure 2: (a) A 2-path (b) A 2-fan (c) A 2-line.

we solve problem X on G directly. Otherwise we identify a separator S (a k -clique) and decompose G into l subgraphs G_1, \dots, G_l such that each G_i is the subgraph induced by $V(S) \cup V(C_i)$, where C_1, \dots, C_l are the connected components of $G - S$. Then we recursively solve problem X (or a slight variant of it) on each k -tree G_i ($i = 1, \dots, l$) and, finally, we combine the solutions.

The reduction paradigm in [10] proceeds bottom-up using a dynamic programming approach. First, we associate a state with each k -clique in the graph. The state of each k -clique contains some local information that will be combined with the information in other states to solve problem X . Once each k -clique has been assigned an initial state, we proceed to eliminate $n - k$ nodes of G following a perfect elimination ordering. Each time we eliminate one node v we destroy a number of k -cliques whose states contain valuable information. So, before removing v we combine the states of these k -cliques and save the result as the state of the k -clique induced by the neighbors of v . When the $n - k$ nodes have been removed from G we are left with a root R of G . R is a k -clique whose state contains enough information to solve problem X on G . We need to make some observations before we formalize these ideas.

Algorithm 1 formalizes the reduction paradigm. Let us suppose that we want to solve problem X on a k -tree G . The first step of the algorithm, the initialization step, finds the first $n - k$ nodes of a perfect elimination ordering and initializes the state of each k -clique in the graph G . The initial state of each k -clique K is computed by a function $e(K)$. Each reduction step removes one of the $n - k$ nodes in the queue PEO . Upon removal of a node v , the algorithm performs two sub-steps. First it “combines” the states of $k+1$ k -cliques. We use f to denote the function that computes such a combination of states. The result of applying f to the states of the k k -cliques that will be destroyed and to the state of the neighborhood of v is called the “state” of $K^+(v)$.⁵ The second

⁵The “state” of $K^+(v)$ is ephemeral; we compute it once and immediately use it to update the state of $K(v)$. Once the state of $K(v)$ has been updated, we destroy $K^+(v)$ by removing the node v . So, $state(K^+(v))$ is simply an intermediate value that we calculate to update the state of $K(v)$. We believe that the metaphor of having a state for $K^+(v)$ is

sub-step combines the effect of the edges that connect v to its neighborhood ($K(v)$) and the state of $K^+(v)$. Algorithm 1 represents this second combination of information as the computation of $g(\text{state}(K^+(v)), S(v))$. The termination step extracts the solution to problem X from the state of the root R and the effect of the edges in R .

Algorithm 1 (reduction paradigm)

Input: $G = (V, E)$, a k -tree (for a fixed k)

1. Initialization step

$PEO \leftarrow$ empty queue

Do $n - k$ times:

 Let v be a k -leaf of $G - PEO$

 Let $K(v)$ be the (k -clique) attachment of v in $G - PEO$

 Let $K^+(v)$ be the $(k+1)$ -clique induced by $V(K(v)) \cup \{v\}$

 For all nodes u in $V(K(v))$ do:

 Let $K^u(v)$ be the k -clique induced by $V(K^+(v)) \setminus \{u\}$

$\text{state}(K^u(v)) \leftarrow e(K^u(v))$

 Append v to PEO

$\text{state}(R) \leftarrow e(R)$

2. Reduction steps

For each node v in PEO , in order, do:

$\text{state}(K^+(v)) \leftarrow f(\{\text{state}(K^u) \mid u \in V(K^+(v))\})$

 Let $S(v)$ be the graph induced by the edges incident to v

$\text{state}(K(v)) \leftarrow g(\text{state}(K^+(v)), S(v))$

 Remove v from G

3. Termination step

$Solution \leftarrow h(\text{state}(R), \text{edges in } R)$

The state $\text{state}(K)$ of each k -clique K describes solutions to a problem (usually a generalization of the original problem) restricted to the subgraph induced by the union of

useful in understanding and devising the functions f and g for specific problems.

the nodes in the k -clique and those removed nodes that the k -clique separates from all non-removed nodes with the exclusion of all edges between nodes in the k -clique. Typically, the state is indexed by elements of an index set whose cardinality is constant with respect to the size of G . The size of $state(K)$ is the cardinality of its index set. The specification of an algorithm that uses the reduction paradigm described above consists of five main parts: the definition of $state(K)$ and algorithms to compute the functions e , f , g , and h in Algorithm 1.

We need to formalize some concepts that are crucial in definitions of reduction algorithms. If K is a k -clique, $v \notin V(K)$ is a descendant of K in a perfect elimination ordering if and only if each higher numbered neighbor of v is either a member of $V(K)$ or a descendant of K . The connected components of the subgraph induced by all descendants of K are branches on K . Figure 3 (a) depicts a 3-tree in which K is the 3-clique induced by the nodes v_6 , v_7 , and v_8 . Figure 3 (b) presents the branches on K .

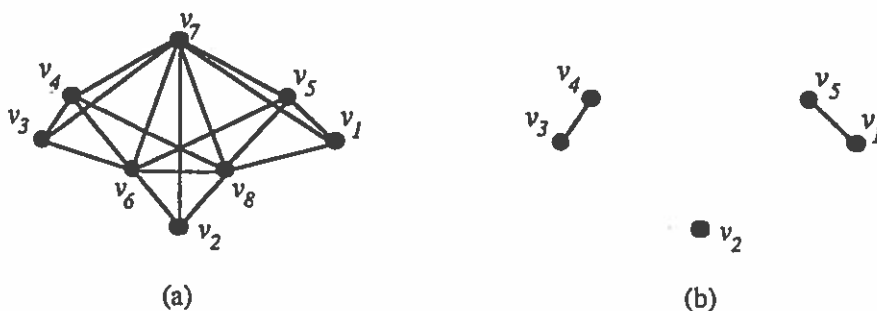


Figure 3: (a) A 3-tree (b) Branches on K .

Suppose that we have a perfect elimination ordering defining a reduction process. We associate two subgraphs, $B(K)$ and $B'(K)$, with each k -clique K . These two subgraphs change as the reduction progresses. We use $B(K)$ to denote the subgraph induced by the removed nodes that the k -clique K separates from all non-removed nodes. We call $B(K)$ the removed branches subgraph on K . $B'(K)$ denotes the subgraph induced by the nodes in $K \cup B(K)$ without the edges between nodes in K . We call $B'(K)$ the shell of K . The state of a k -clique K describes solutions to problems restricted to the shell $B'(K)$. Figure 4 illustrates these concepts. After v_1 , v_2 , v_3 , and v_4 have been removed (but not v_5 yet)

from the graph in Figure 3, $B(K)$ becomes the graph in Figure 4 (a), and $B'(K)$ becomes the graph in Figure 4 (b).

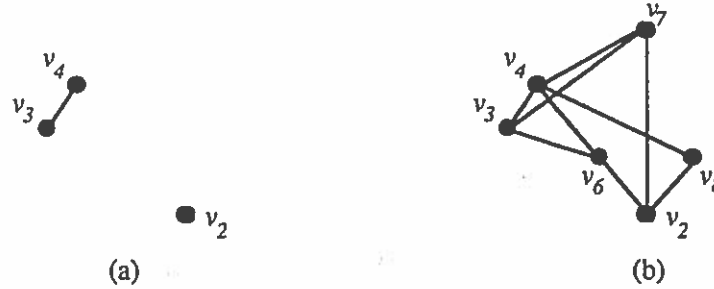


Figure 4: $B(K)$ and $B'(K)$ after removing v_1 , v_2 , v_3 and v_4 from the graph in Figure 3.

The following equations explicitly state how $B(K)$ and $B'(K)$ change during the execution of Algorithm 1. Notice that these equations also define $B(K^+)$ and $B'(K^+)$.

Initialization step:

$$B(K) = \text{the empty graph}(\emptyset, \emptyset) \quad (\text{II.1})$$

$$B'(K) = (V(K), \emptyset) \quad (\text{II.2})$$

Reduction steps:

Let $K = K(v)$, $K^+ = K^+(v)$, $K^u = K^u(v)$, and $S = S(v)$

$$B(K^+) = \bigcup_{u \in V(K^+)} B(K^u) \quad (\text{II.3})$$

$$B'(K^+) = \bigcup_{u \in V(K^+)} B'(K^u) \quad (\text{II.4})$$

$$B(K) = \text{subgraph induced by } V(B(K^+)) \cup \{v\} \quad (\text{II.5})$$

$$B'(K) = B'(K^+) \cup S \quad (\text{II.6})$$

Termination step:

$$B(R) = G - R \quad (\text{II.7})$$

$$B'(R) = G \text{ without the edges in } E(R) \quad (\text{II.8})$$

Complexity Issues

Typically, the reduction paradigm renders algorithms that run in time linear in the number of nodes of the k -tree. The following lemma states that if the initialization of each k -clique takes $\mathcal{O}(1)$ time, each reduction step takes $\mathcal{O}(1)$, and the termination step takes $\mathcal{O}(n)$ time, the resulting algorithm runs in $\mathcal{O}(n)$ time.

Lemma 2.5 Let k be a positive integer number, $G = (V, E)$ be a k -tree on n nodes, and A be a reduction algorithm to solve problem X on G . If the initialization of each k -clique takes $\mathcal{O}(1)$ time, each reduction step takes $\mathcal{O}(1)$, and the termination step takes $\mathcal{O}(n)$ time, algorithm A runs in $\mathcal{O}(n)$ time.

Proof: Algorithm 2 is an implementation of Algorithm 1 that makes explicit the amount of computation involved in the reduction paradigm. Let us consider the initialization step. The amount of time spent in this step is proportional to the amount of time spent traversing adjacency lists. It is easy to prove that a k -tree has exactly $k \times (k - 1)/2 + (n - k) \times k$ edges. In addition, the adjacency list of each node is traversed only two times. So, the initialization step takes $\mathcal{O}(n)$ time. Clearly each reduction step takes constant time, and the termination step takes $\mathcal{O}(n)$ time. ■

Algorithm 2 (detailed description of reduction paradigm)

Input: $G = (V, E)$, a k -tree (for a fixed k)

1. Initialization

$PEO \leftarrow$ empty list

For all v in V do:

Traverse $\text{AdjacencyList}(v)$ to determine and record $\text{degree}(v)$

If $\text{degree}(v) = k$ then push v onto stack L

Do $n - k$ times:

$v \leftarrow \text{Pop}(L)$

Traverse $\text{AdjacencyList}(v)$ to find $K(v)$, the attachment of v in G

$K^+(v) \leftarrow (k+1)$ -clique induced by $V(K(v)) \cup \{v\}$

$S(v) \leftarrow$ graph induced by the edges $\{v, u\}, \forall u \in V(K(v))$

For all nodes u in $K(v)$ do:

$K^u(v) \leftarrow k$ -clique induced by $V(K^+(v)) \setminus \{u\}$

$\text{state}(K^u(v)) \leftarrow e(K^u(v))$

$\text{degree}(u) \leftarrow \text{degree}(u) - 1$

If $\text{degree}(u) = k$ then push u onto L

Append v to PEO

$\text{degree}(v) \leftarrow 0$

$\text{state}(R) \leftarrow e(R)$

2. Reduction

For each v in PEO do:

$\text{state}(K^+(v)) \leftarrow f(\{\text{state}(K^u(v)) \mid u \in K^+(v)\})$

$\text{state}(K(v)) \leftarrow g(\text{state}(K^+(v)), S(v))$

3. Termination

$\text{Solution} \leftarrow h(\text{state}(R), \text{edges in } R)$

The reduction paradigm is fundamentally the same when the input graph G is not a full k -tree but a partial k -tree given with an embedding in a k -tree G' . We use the k -tree G' to guide the reduction process but compute the functions e , f , g , and h using the partial k -tree. If the partial k -tree is not given with an embedding in a k -tree, we have to deal with an additional problem, namely, finding an embedding of G in a k -tree. Typically, the running time of reduction algorithms on partial k -trees is dominated by the time needed to find an embedding in a k -tree. Arnborg, Corneil and Proskurowski [6] proved that finding the smallest k for which a given graph is a partial k -tree is NP-hard. However, they also

found an $\mathcal{O}(n^{k+2})$ time algorithm to test if a given graph is a partial k -tree, for a fixed value k . This algorithm also finds the embedding of the partial k -tree. For small values of k ($k \leq 3$) the embedding can be found in linear time using terminating sets of reduction rules ([60] [44]). Unfortunately, this reduction technique does not work for values of k greater than 3 [41]. Bodlaender [17] presents a self-reduction algorithm that finds and embedding of a partial k -tree in a k -tree in $\mathcal{O}(n^2)$ time. In a recent paper [7], Arnborg et al. present a technique to test if a graph is a partial k -tree in linear time. They describe an algorithm that will produce, from a formula in monadic second order logic (MSOL) and an integer k such that the class defined by the formula is of tree-width $\leq k$, a set of graph rewriting rules that reduces any member of the class to one of a finite set of graphs. Any algorithm that tests if a graph is a partial k -tree can be used to find an embedding in a k -tree as follows: Add an edge and test if the resulting graph is a partial k -tree; if it is, leave the edge in, otherwise try adding a different edge; continue until the graph becomes a (full) k -tree.

All NP-hard problems that have been solved using the reduction paradigm run in time exponential in k . This is not surprising as any n node graph is a partial n -tree. Thus, unless one believes that $P = NP$ there is not much hope of having such reduction algorithms run in time polynomial in k .

Recent work by Courcelle [25] and Arnborg et al. [8] establish relationships between logic formalisms and the complexity of recognizing certain graph properties. In particular, they prove, constructively, that if a graph property can be expressed in monadic second order logic, one can decide, in linear time, whether a given partial k -tree has this property (assuming that the value of k is fixed and that the partial k -tree is given with an embedding in a k -tree). This is a very important result since many NP-complete graph theoretic problems can be expressed in MSOL (e.g. Hamiltonian Circuit).

Arnborg et al. also extend the concept of MS properties (graph properties definable in MSOL) to EMS problems (problems definable in Extended Monadic Second Order Logic). With this generalization one can define a large number of counting and extremum problems (e.g., Minimum Dominating Set). The importance of this generalization is that

an EMSOL formula can be transformed into a polynomial time algorithm that computes such formula. Although we suspect it possible, we have not been able to express the reliability problems addressed in the next section as EMS problems.

A natural question to ask about this reduction paradigm is whether it can be used to design NC algorithms. In particular, it seems that one could naturally extend the tree-contraction techniques in [45] and [1] to prune the partial k -tree (perform reduction steps) in parallel. Bodlaender [18] uses this approach to prove that the embedding problem can be solved in parallel poly-log time. In addition, he proves that a linear time sequential reduction algorithm on partial k -trees can be transformed into an NC algorithm if there are NC algorithms to compute the functions e , f , g , and h in Algorithm 1. All the linear time algorithms presented in Chapter III and Chapter IV satisfy this condition, and therefore can be transformed into NC algorithms. However, because of the large constants involved, Bodlaender's result is mainly of theoretical interest.

CHAPTER III

EFFICIENT EXACT ALGORITHMS FOR PARTIAL K -TREE NETWORKS

Introduction

In the previous chapter we presented a reduction paradigm that has been used to solve efficiently a large number of NP-hard problems restricted to partial k -trees. Arnborg and Proskurowski [10] employed this technique to solve two important reliability problems on partial k -tree networks without node failures, namely, the all-terminal reliability and the l -terminal reliability. Other reliability measures such as the resilience have been solved in polynomial time only when the network has a very restricted topology (e.g., partial 2-trees [24, 5]). In this chapter we present applications of the reduction paradigm to the exact solution of a variety of network reliability problems restricted to partial k -trees. All the reliability problems solved in this chapter are #P-complete on general graphs.

We begin our presentation with the definition of some fundamental network reliability concepts and problems. Then we present our technique to extend the solutions in [10] to a network model in which both nodes and edges may fail. Specifically, we give linear time reduction algorithms to solve the following network reliability problems on partial k -tree networks: all-terminal reliability, l -terminal reliability, broadcast resilience, l -broadcast resilience, and resilience. We also present a quadratic time solution to the network broadcast facility location problem restricted to partial k -tree networks.

Definitions

A probabilistic graph $G = (V, E)$ is a graph in which each component (node or edge) c has an associated fixed precision real number p_c , such that $0 \leq p_c \leq 1$. We model computer communication networks as probabilistic graphs $G = (V, E)$ wherein each node v in V represents a communication site and each edge e in E represents a bidirectional

communication line between two sites. Furthermore, for each component c of the graph we interpret p_c as the probability of operation of component c . Components of the network are in either operational or failed state. Component failures are assumed to be statistically independent. A component c is fail-safe if its probability of operation is 1. Hereafter all graphs are assumed to be probabilistic.

Traditionally, the reliability of a network G is defined as the probability that a given communication task T can be performed in G . For example, if the task T consists of exchanging information between l distinguished nodes of G , the reliability of G (l -terminal reliability) is defined as the probability that the graph contains paths between each pair of distinguished nodes. The n -terminal (all-terminal) and the 2-terminal reliability are two of the most widely used measures. In the former case we are interested in computing the probability that the network contains a spanning tree, in the latter we are concerned with the probability that there is a path connecting two distinguished nodes in G . Naturally, we can define other reasonable reliability measures by selecting communication tasks other than the ones mentioned above. We need a few more definitions before presenting a general formula to compute the reliability of G with respect to an arbitrary communication task T .

A state S of a network $G = (V, E)$ is a set of nodes and edges of G ; we interpret S as the set of nodes and edges that are operational in G . The probability that G is in state S is

$$\prod_{v \in S \cap V} p_v \prod_{v \in V \setminus (S \cap V)} (1 - p_v) \prod_{e \in S \cap E} p_e \prod_{e \in E \setminus (S \cap E)} (1 - p_e),$$

where p_v and p_e denote the probability that node v is up and edge e is up respectively.

We use subgraphs of G to represent states of the network. Notice however that, unless G has no edges, there are more states than subgraphs of G . So, each subgraph $H = (V_H, E_H)$ of G represents a class of states of G , namely those states of G in which nodes in V_H are up, nodes in $V \setminus V_H$ are down, edges in E_H are up, and edges in $E'_H \setminus E_H$ are down, where E'_H is the set of edges of the subgraph of G induced by V_H (see Figure 5). The operational subgraph of G is the subgraph of G defined by the operational nodes

and the operational edges that are incident on two operational nodes. $P_G[H]$ denotes the probability that H is the operational subgraph of G (equivalently, it denotes the probability that the state of G is any of the states represented by H). Thus,

$$P_G[H] = \prod_{v \in V_H} p_v \prod_{v \in V \setminus V_H} (1 - p_v) \prod_{e \in E_H} p_e \prod_{e \in E'_H \setminus E_H} (1 - p_e).$$

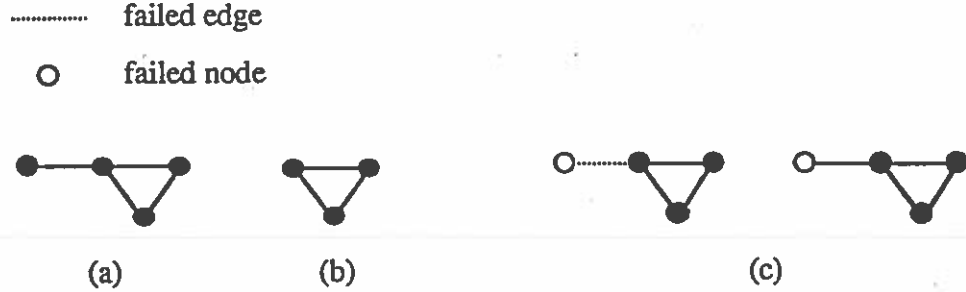


Figure 5: (a) Graph G (b) Subgraph H (c) States represented by H .

We extend the definition of P_G to the domain of sets of subgraphs of G in the natural way. Given a set A of subgraphs of G , $P_G[A]$ denotes the probability that the operational subgraph of G is in A . Therefore $P_G[A] = \sum_{H \in A} P_G[H]$.

A communication task T can be characterized by the set of subgraphs of G in which T can be carried out. Following [23], we call such a set of subgraphs $OP_T(G)$. A pathset of G with respect to T is an element of $OP_T(G)$. We can now formulate the reliability of a network G with respect to a communication task T as

$$Rel_T(G) = P_G[OP_T(G)] = \sum_{H \in OP_T(G)} P_G[H].$$

Let H be a subgraph of G and u, v be two nodes in $V(G)$. We say that u is connected to v via H (denoted $u \stackrel{H}{\sim} v$) iff u and v are elements of $V(H)$ and there is a path, consisting of zero or more edges in $E(H)$, that connects node v to node u ; when

$H = G$ we prefer the notation $u \sim v$ over $u \stackrel{H}{\sim} v$. A node v is connected to a set of nodes W via a graph H (denoted $u \stackrel{H}{\sim} W$) if $u \stackrel{H}{\sim} v$ for some node v in W . For a subset $W \subseteq V(G)$ we define the binary function $r(W, H)$ to be 1 if for all u, v in W $u \stackrel{H}{\sim} v$, otherwise it is 0. Similarly, for a subset $W \subseteq V(G)$ we define the connectivity function $r(u, W, H)$ to be 1 if $u \stackrel{H}{\sim} W$, otherwise it is 0. We can, for example, define $Rel_l(G, L)$ as

$$Rel_l(G, L) = \sum_{H \leq G} P_G[H] r(L, H).$$

Computing the all-terminal reliability of a network is a #P-complete problem, even for planar networks (Vertigan, in [22]). Furthermore, computing the 2-terminal reliability of a network is also a #P-complete problem, even when the network is planar, acyclic, with bounded degree nodes, with fail-safe nodes, and with all the edges having identical probability of operation [55].

Reliability measures defined in terms of probability that a given communication task can be performed are often too rough an estimate of the robustness of a network with respect to the communication task. For instance, two n node networks G_1 and G_2 , with the same all-terminal reliability, may not be equally convenient for broadcasts. In particular, the expected number of pairs of nodes that can communicate in G_1 may be considerably larger than the expected number of nodes that can communicate in G_2 . Also, the all-terminal reliability of a network does not give us any information about the best source(s) for the broadcast operation. The reliability measures described next provide more detailed information that is valuable in comparing and designing reliable networks.

The broadcast resilience of a network with respect to a specific node v is the expected number of nodes that can be reached from v [49].¹ This measure is useful in selecting a node as a source for broadcasts. The l -broadcast resilience of G with respect to a set L of l nodes is the expected number of nodes that can be reached from at least one node in L . Typically, we are interested in finding the set of nodes L' that maximizes the l -broadcast

¹Alternatively, we can interpret this measure as the average size of the connected component of G that contains the node v .

resilience of a given network G over all subsets of l nodes of G . Nel [49] calls this the Network Broadcast Facility Location (NBFL) problem and proves that it is #P-complete even when $l = 1$.

The resilience of a network is the expected number of pairs of distinct nodes that can communicate. Although this measure provides important additional, fine grain information about the reliability of a network, it is not computed often because it apparently involves more work than computing traditional measures such as the 2-terminal reliability. For example, we do not know of any algorithm for Res on general graphs that runs in time better than $\min(\mathcal{O}(n^2 f_1(n)), \mathcal{O}(n f_2(n)))$, where the running time of the fastest algorithm to compute the 2-terminal reliability is $\mathcal{O}(f_1(n))$ and the running time of the fastest algorithm for Res_v is $\mathcal{O}(f_2(n))$.

Table 1 summarizes our terminology for reliability measures and their corresponding problems. We define the directed versions of these problems and their reduction algorithms in Chapter IV.

Table 1: Some Reliability Measures and Their Associated Problems

Measure	Notation	Problem
All-terminal reliability	$Rel_A(G)$	Rel_A
Two-terminal reliability	$Rel_2(G, s, t)$	Rel_2
l -terminal reliability	$Rel_l(G, L)$	Rel_l
Broadcast resilience	$Res_v(G)$	Res_v
l -broadcast Resilience	$Res_l(G, L)$	Res_l
Resilience	$Res(G)$	Res

Efficient Exact Algorithms

Natural restrictions of the network model include assuming that only nodes fail, that only edges fail, that the network is (un)directed, or that all components operate with the same probability. No combination of these restrictions seems to affect the inherent complexity of reliability problems for general graphs. Research on efficient exact algorithms has therefore concentrated on networks with restricted topologies and, possibly,

with some of the restrictions mentioned above.

Trees, wheels, and other classes of graphs have very good analytical power with respect to reliability problems. However, their modeling power is very limited. The rest of this chapter presents our algorithms for the reliability problems in Table 1, restricted to partial k -trees networks. We assume that partial k -trees are always given with an embedding in a k -tree and that k is a fixed positive integer. Furthermore, without loss of generality we assume that the input graph has been transformed into an embedding k -tree by adding edges e with $p_e = 0$.

l -Terminal Reliability

Given a network $G = (V, E)$ and a subset L of nodes in V , we are interested in computing the probability that a subgraph of G has a connected component that includes all the nodes in L . Arnborg and Proskurowski [10] solved Rel_A in linear time on partial k -tree networks with fail-safe nodes. The assumption that nodes are fail-safe is not really restrictive since the all-terminal reliability with node failures is the probability that all nodes are operational times the all-terminal reliability without node failures. The same argument does not apply, though, to the other reliability measures in Table 1.

Arnborg et al. [10] also solved Rel_l in linear time on partial k -tree networks with fail-safe nodes. In this section we show how to generalize their reduction algorithm to partial k -tree networks with both node and edge failures. We will be particularly detailed in the presentation of this algorithm because we use it to illustrate the general technique and because most of the definitions employed, or slight variants of them, will also be used in other reduction algorithms in this chapter and in Chapter IV.

In Chapter II we mentioned that the description of a reduction algorithm consists of five main parts. First we define the state of each k -clique. Then we specify how to compute e (initialization step), f , g (reduction steps), and h (termination step) in Algorithm II.

The state of each k -clique K consists of statistical information about the shell $B'(K)$. We want to define the state in such a way that we can achieve the following goals:

- (a) The size of $state(K)$ should be constant with respect to the size of the input graph; otherwise the reduction algorithm would not run in $\mathcal{O}(n)$ time.
- (b) The states should be “combinable”, i.e., we must be able to update the state of K by “combining” the states of the k k -cliques that are destroyed in a reduction step.
- (c) The amount of information in $state(K)$ should be sufficient to solve Rel_A on the graph $B'(K) \cup K$. This guarantees that the termination step gives us Rel_A .

The key idea to accomplish (a), (b), and (c) above is to define an equivalence relation of constant index between subgraphs of the shell $B'(K)$ and to maintain information about each equivalence class of subgraphs of $B'(K)$ in $state(K)$. We need to introduce some notation to formalize this idea.

The set of connected components of a graph defines a partition of the nodes of the graph in a natural way: each block of the partition corresponds to the nodes of a connected component (if the graph is empty, i.e., (\emptyset, \emptyset) , the partition associated is $\{\emptyset\}$). A subpartition of a set A is a partition of a subset of A . We use subpartitions of V to represent the connected components of operational subgraphs of $G = (V, E)$. Given a subpartition π of nodes, $V(\pi)$ denotes the set of nodes in π . We use $P_G[\pi]$ to denote the probability that a subgraph of G has the connected components denoted by π . A connected component C is isolated from a set of nodes W if no node in W is also in C .

Let L be a set of distinguished nodes in G . We are interested in identifying two kinds of connected components in subgraphs of G , namely, those connected components that have a distinguished node and those that do not. We call the former gray connected components and the latter white connected components. We use pairs of partitions to represent colored connected components. Formally, a colored partition of V is a pair $\alpha = (\pi, \sigma)$ such that π is a partition of V and σ is a non-empty subset of π .² A colored subpartition of V is a colored partition of a subset of V and denotes the connected components of a subgraph of G and the color of each connected component; blocks in σ represent the gray components of a subgraph of G . We use $P_G[\alpha]$ to represent the probability that

²Notice that σ may be the partition $\{\emptyset\}$.

a subgraph of G has precisely the connected components denoted by α . The set $V(\alpha)$ denotes the set of nodes in the colored partition α .

The contraction $Cont(G, W)$ of a graph $G = (V, E)$ with respect to a set of nodes W is the subpartition of W obtained by intersecting each connected component of G with W (if $V \cap W = \emptyset$ the result is $\{\emptyset\}$). We choose the name “contraction” because $Cont(G, W)$ can also be obtained as follows:

For each edge $e = \{v, w\}$ do:

If $v \notin W$ or $w \notin W$ then

If $v \notin W$ then

Contract e and call the new node w

else

Contract e and call the new node v

Remove all nodes $v \notin W$ from the contracted graph $G' = (V', E')$

$Cont(G, W) \leftarrow$ partition of V' defined by connected components of G'

We extend the definition of $Cont$ to deal with colored connected components. For the set of distinguished nodes L , the L -contraction $Cont_L(G, W)$ of G with respect to a set of nodes W is the colored subpartition (π, σ) of W where π is the contraction of G with respect to W and σ is the contraction with respect to W of the graph induced by the gray connected components in G .

Let us now consider H , a non-empty subgraph of G . We use $\Pi(H)$ to denote the set of subpartitions of $V(H)$ and $\mathcal{C}(H)$ to denote the set of colored subpartitions of $V(H)$. We are interested in characterizing those subgraphs of G in which all the distinguished nodes in $V(G)$ are operational and no gray connected component is isolated from $V(H)$. Let us call such subgraphs of G the set of favorable subgraphs of G with respect to H . For each colored subpartition α in $\mathcal{C}(H)$, $SG(G, H, \alpha)$ denotes the set of subgraphs G' of G such that $Cont_L(G', V(H)) = \alpha$, all the distinguished nodes in G are elements of $V(G')$, and no gray connected components is isolated from $V(H)$ in G' . In other words, $SG(G, H, \alpha)$ is the set of favorable subgraphs of G whose L -contraction with respect to $V(H)$ is α .

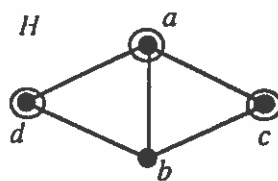
It is easy to verify that the set of favorable subgraphs of G with respect to H can be partitioned into equivalence classes, each of which is $SG(G, H, \alpha)$, where α is a colored subpartition of the set of nodes $V(H)$. Let us consider, in particular, a partially reduced k -tree and one of its k -cliques K . We can partition the set of favorable subgraphs of the shell $B'(K)$ into equivalence classes corresponding to colored subpartitions of $V(K)$. Figure 6 (a) depicts a 2-tree G where K is the 2-clique induced by the set of nodes $\{a, b\}$, the set of distinguished nodes is $L = \{a, c, d\}$, and node c has been removed. There are eleven different colored subpartitions of the set $\{a, b\}$ and, therefore, eleven equivalence classes in the shell $B'(K)$. Figure 6 (b) shows the four non-empty classes of favorable subgraphs of $B'(K)$ with respect to K (recall that the edge $\{a, b\}$ is not part of the shell $B'(K)$).

We can now define the index set of the state of each k -clique. Let K be a k -clique of a partially reduced k -tree G . Let $\alpha_1, \dots, \alpha_q$ be an enumeration of all the colored subpartitions of the nodes in K .³ We partition the set of favorable subgraphs of the shell $B'(K)$ into the equivalence classes $SG(B'(K), K, \alpha_1), \dots, SG(B'(K), K, \alpha_q)$. The state of K consists of statistical information about each equivalence class of subgraphs of the shell $B'(K)$. Specifically, the state of each k or $k + 1$ -clique K consists of the values $s(\alpha_1, K), \dots, s(\alpha_q, K)$ such that for each colored subpartition $\alpha = (\pi, \sigma)$ in $\mathcal{C}(K)$

$$\begin{aligned} s(\alpha, K) &= P_{B'(K)}[SG(B'(K), K, \alpha) \parallel up(V(\alpha), K)] \\ &= \sum_{H \in SG(B'(K), K, \alpha)} P_{B'(K)}[H \parallel up(V(\alpha), K)], \end{aligned} \quad (\text{III.9})$$

where $P[A \parallel B]$ denotes $P[A \mid B]$ if $P[B] > 0$, otherwise it is 0. In addition, we use $up(W, K)$ to denote the condition $up(W) \wedge dn(V(K) \setminus W)$, i.e., the nodes in W are up and the nodes in $V(K) \setminus W$ are down. Thus, $s(\alpha, K)$ is the probability that a subgraph of the shell $B'(K)$ belongs to the class $SG(B'(K), K, \alpha)$, given that the nodes in $V(\alpha)$ are

³Notice that, for a fixed value of k , q is constant (although exponential in k). Also notice that we need to consider only colored subpartitions (π, σ) such that $|\sigma| \leq l$ and such that the nodes w in $V(K) \cap L$ are in $V(\sigma)$.



(a)

Colored subp. α	Subgraphs in $SG(B'(K), K, \alpha)$
$(\{\{a,b\}\}, \{\{a,b\}\})$	
$(\{\{a\}, \{b\}\}, \{\{a\}, \{b\}\})$	
$(\{\{a\}, \{b\}\}, \{\{a\}\})$	
$(\{\{a\}\}, \{\{a\}\})$	

(b)

Figure 6: (a) A 2-tree (b) Non-empty equivalence classes induced by subpartitions of $V(K)$.

operational and the nodes in $V(K) \setminus V(\alpha)$ are down. If the probability that the nodes in $V(K) \setminus V(\alpha)$ are down is zero, $s(\pi, K)$ is defined as zero.⁴

The next four lemmata define the initialization, reduction, and termination steps of our algorithm for Rel_l .

Lemma 3.1 (initialization) Let G be a k -tree network and L be a subset of nodes of G . Then for all K and α such that K is a k -clique of G , and $\alpha = (\pi, \sigma)$ is a colored subpartition

⁴For the sake of simplicity we assume that the probability of operation of each node v in G is positive. If some p_v is zero we can either modify the formulae in this section or remove v and apply the algorithm to the resulting partial k -tree.

in $\mathcal{C}(K)$,

$$s(\alpha, K) = \begin{cases} 1 & \text{if } \alpha \text{ consists of zero or more singletons, } \prod_{v \in V(K) \setminus V(\alpha)} (1 - p_v) \neq 0, \\ & V(\sigma) \subseteq L, \text{ and } (V(K) \setminus V(\sigma)) \cap L = \emptyset \\ 0 & \text{otherwise.} \end{cases}$$

Proof: Immediate from the definition of $B'(K)$ (Equation II.2) and the definition of $s(\alpha, K)$ (Equation III.9) ■

Let us now consider the reduction step. Let G be a (partially reduced) k -tree, and v be a k -leaf of G with neighborhood K . Let K^+ be the graph induced by $V(K) \cup \{v\}$, and u_1, \dots, u_{k+1} be the nodes in K^+ . Also, for all $i = 1, \dots, k+1$, let K^i denote the k -clique induced by the nodes in $V(K^+) \setminus \{u_i\}$. The reduction step consists of two parts. First we compute the state of K^+ by combining the information in the states of K^i for all $i = 1, \dots, k+1$ (Lemma 3.4). Then we update the state of K by considering the state of K^+ and the effect of the edges that connect v to K (Lemma 3.6).

We need some additional notation. Let π be a partition. Following [10], we use π/u to denote the partition obtained by removing u from its block in π and then removing the block if it became empty. Furthermore, the join of two partitions π_1 and π_2 , denoted $\pi_1 \vee \pi_2$, is the partition obtained by taking the union of intersecting blocks until a partition of the union remains (e.g., $\{\{a, b\}, \{c\}, \{d\}\} \vee \{\{a, d\}, \{b, c, e\}, \{f\}\} = \{\{a, b, c, d, e\}, \{f\}\}$). The join of two colored partitions $\alpha_1 = (\pi_1, \sigma_1)$, denoted $\alpha_1 \vee \alpha_2$, is the colored partition $(\pi_1 \vee \pi_2, \sigma_3)$ where σ_3 is obtained by removing from $\pi_1 \vee \pi_2$ all the blocks that are disjoint from $V(\sigma_1 \vee \sigma_2)$.

To compute the state of K^+ we consider all possible ways of obtaining α^+ , a colored subpartition in $\mathcal{C}(K^+)$, as the join of $\alpha_1, \dots, \alpha_{k+1}$, where α_i is a colored partition of $V(\alpha^+) \setminus \{u_i\}$ ($i = 1, \dots, k+1$). We use $T(\alpha^+, K^+)$ to denote the set of $(k+1)$ -tuples of colored subpartitions of nodes in K^+ such that their join is α^+ and the i -th entry is a

colored partition of $V(\alpha^+) \setminus \{u_i\}$ ($i = 1, \dots, k+1$). Formally,

$$T(\alpha^+, K^+) = \{(\alpha_1, \dots, \alpha_{k+1}) \mid \bigvee_{i=1}^{k+1} \alpha_i = \alpha^+ \wedge \forall i = 1, \dots, k+1, \alpha_i \in \mathcal{C}(V(\alpha^+) \setminus \{u_i\})\}.$$

We use $\vec{\alpha}$ to denote a $(k+1)$ -tuple in $T(\alpha^+, K^+)$ and $\vec{\alpha}_i$ to denote the i -th entry of $\vec{\alpha}$.

By definition of $B'(K^+)$ (Equation II.4), a subgraph H of the shell $B'(K^+)$ is the (graph) union of $k+1$ graphs H_1, \dots, H_{k+1} such that each H_i is a subgraph of $B'(K^i)$ ($i = 1, \dots, k+1$). Furthermore, a subgraph H is in $SG(B(K^+), K^+, \alpha^+)$ if and only if each H_i is in $SG(B(K^i), K^i, \alpha_i)$ for colored subpartitions α_i such that $(\alpha_1, \dots, \alpha_{k+1})$ is an element of $T(\pi^+, K^+)$. Formally, we make the following observations.

Observation 3.2 There is a bijection ϕ from $SG(B'(K^+), K^+, \alpha^+)$ to

$$\bigcup_{\vec{\alpha} \in T(\alpha^+, K^+)} SG(B'(K^1), K^1, \vec{\alpha}_1) \times \dots \times SG(B'(K^{k+1}), K^{k+1}, \vec{\alpha}_{k+1})$$

such that $\phi(H) = (H_1, \dots, H_{k+1})$ iff $\bigcup_{i=1}^{k+1} H_i = H$. In particular,

$$\sum_{\substack{H \text{ in} \\ SG(B'(K^+), K^+, \alpha^+)}} P_{B'(K^+)}[H \parallel Y] = \sum_{\vec{\alpha} \in T(\alpha^+, K^+)} \sum_{\substack{(H_1, \dots, H_{k+1}) \text{ in} \\ X_1 \times \dots \times X_{k+1}}} P_{B'(K^+)}[H_1 \cup \dots \cup H_{k+1} \parallel Y],$$

where Y denotes the condition $up(V(\alpha^+), K^+)$ and $X_i = SG(B'(K^i), K^i, \vec{\alpha}_i)$, $1 \leq i \leq k+1$. Moreover, for each $(k+1)$ -tuple (H_1, \dots, H_{k+1}) in $X_1 \times \dots \times X_{k+1}$,

$$P_{B'(K^+)}[H_1 \cup \dots \cup H_{k+1} \parallel Y] = \prod_{i=1}^{k+1} P_{B'(K^i)}[H_i \parallel up(V(\vec{\alpha}_i), K^i)].$$

The following observation is useful in proving Lemma 3.4.

Observation 3.3 Given m finite sets X_1, \dots, X_m and m real functions f_1, \dots, f_m with domains X_1, \dots, X_m respectively,

$$\prod_{i=1}^m \sum_{x \in X_i} f_i(x) = \sum_{\substack{(x_1, \dots, x_m) \text{ in} \\ X_1 \times \dots \times X_m}} \prod_{i=1}^m f_i(x_i).$$

We can now prove the following lemma.

Lemma 3.4 Let G be a k -tree network that has been partially reduced using some perfect elimination ordering and the general reduction paradigm. Let v be the next k -leaf to be removed. Let K be the neighborhood of v and K^+ be the subgraph of G induced by $V(K) \cup \{v\}$. Then for any colored subpartition $\alpha^+ \in \mathcal{C}(K^+)$,

$$s(\alpha^+, K^+) = \sum_{\bar{\alpha} \in T(\alpha^+, K^+)} \prod_{i=1}^{k+1} s(\bar{\alpha}_i, K^i).$$

Proof: Using the definition of $s(\alpha^+, K^+)$ (Equation III.9) and Observation 3.2 we get

$$\begin{aligned} s(\alpha^+, K^+) &= \sum_{\substack{H \text{ in} \\ s\alpha(B'(K^+), K^+, \alpha^+)}} P_{B'(K^+)}[H \parallel up(V(\alpha^+), K^+)] \\ &= \sum_{\bar{\alpha} \in T(\alpha^+, K^+)} \sum_{\substack{(H_1, \dots, H_{k+1}) \text{ in} \\ X_1 \times \dots \times X_{k+1}}} P_{B'(K^+)}[H_1 \cup \dots \cup H_{k+1} \parallel up(V(\alpha^+), K^+)] \\ &= \sum_{\bar{\alpha} \in T(\alpha^+, K^+)} \sum_{\substack{(H_1, \dots, H_{k+1}) \text{ in} \\ X_1 \times \dots \times X_{k+1}}} \prod_{i=1}^{k+1} P_{B'(K^i)}[H_i \parallel up(V(\bar{\alpha}_i), K^i)]. \end{aligned}$$

The result follows by Observation 3.3 ■

We now show how to update the state of K given the state of K^+ . Let S be the star network induced by the k edges that are incident on v . Let $\mathcal{C}'(S)$ denote the set of colored subpartitions α_S of nodes in S that consist of only singletons possibly with the exception of the set containing node v . Additionally, α_S is such that if v is in L then $v \in V(\alpha_S)$. The set $\mathcal{C}'(S)$ models the set of operational subgraphs of S in which v is operational if it is distinguished. Edges and nodes of S that are operational may cause two or more connected components of the operational subgraph of $B'(K)$ to become connected. So we update the state of K by considering all possible ways of obtaining each colored subpartition α in $\mathcal{C}(K)$ as the join of pairs (α_1, α_2) of subpartitions in $\mathcal{C}(K^+)$ and $\mathcal{C}'(S)$, respectively. The following set defines formally a superset of the pairs of colored subpartitions that we want

to consider:

$$A = \{(\alpha^+, \alpha_S) \mid \alpha^+ \in \mathcal{C}(K^+), \alpha_S \in \mathcal{C}'(S), V(\alpha^+) = V(\alpha_S), (\alpha^+ \vee \alpha_S)/v = \alpha\}.$$

Some of the pairs in A should not be considered as they correspond to subgraphs of the shell $B'(K)$ in which a gray connected component that contains v is isolated from the k -clique K . So we define

$$PS(\alpha, K) = \{(\alpha^+, \alpha_S) \mid (\alpha^+, \alpha_S) \in A \text{ and } \{v\} \notin \sigma'\},$$

where $(\pi', \sigma') = \alpha^+ \vee \alpha_S$. The following observation is useful in proving Lemma 3.6.

Observation 3.5 There is a bijection ψ such that

$$\psi : SG(B'(K), K, \alpha) \mapsto \bigcup_{\substack{(\alpha^+, \alpha_S) \text{ in} \\ PS(\alpha, K)}} SG(B'(K^+), K^+, \alpha^+) \times SG(S, S, \alpha_S)$$

and $\psi(H) = (H_1, H_2)$ iff $H = H_1 \cup H_2$.⁵

We can now establish how to update the state of K from the state of K^+ and the star graph S .

Lemma 3.6 Let G be a k -tree network that has been partially reduced using some perfect elimination ordering and the general reduction paradigm. Let $L \subseteq V(G)$ and v be the next k -leaf to be removed. Let K be the neighborhood of v , and K^+ be the subgraph of G induced by $V(K) \cup \{v\}$. In addition, let S be the star graph consisting of the k edges that link v to K . Then for all colored subpartitions α in $\mathcal{C}(K)$,

$$s(\alpha, K) = \sum_{\substack{(\alpha^+, \alpha_S) \text{ in} \\ PS(\alpha, K)}} s(\alpha^+, K^+) P_S[\alpha_S \parallel up(V(\alpha), K)].$$

⁵Recall that, at this point, $B(K)$ denotes the set of removed branches of K after v has been removed, i.e., it includes v ; K^+ and $B'(K^+)$ were computed before v was removed.

Proof: The proof follows by applying Observation 3.5 to the definition of $s(\alpha, K)$ and performing basic algebraic manipulations. Let us use Y to denote the condition $up(V(\alpha), K)$. In addition, let Y^+ denote the condition $up(V(\alpha^+), K^+)$. By definition (Equation III.9) and Observation 3.5

$$\begin{aligned} s(\alpha, K) &= \sum_{H \in SG(B'(K), K, \alpha)} P_{B'(K)}[H \parallel Y] \\ &= \sum_{\substack{(\alpha^+, \alpha_S) \text{ in} \\ PS(\pi, K)}} \sum_{\substack{H^+ \text{ in} \\ SG(B'(K^+), K^+, \alpha^+)}} \sum_{\substack{H_S \text{ in} \\ SG(S, S, \alpha_S)}} P_{B'(K)}[H^+ \cup H_S \parallel Y]. \end{aligned}$$

But by statistical independence of component failures

$$P_{B'(K)}[H^+ \cup H_S \parallel Y] = P_{B'(K^+)}[H^+ \parallel Y^+] P_S[H_S \parallel Y].$$

Therefore

$$s(\alpha, K) = \sum_{\substack{(\alpha^+, \alpha_S) \text{ in} \\ PS(\pi, K)}} s(\alpha^+, K^+) \sum_{\substack{H_S \text{ in} \\ SG(S, S, \alpha_S)}} P_S[H_S \parallel Y],$$

which is the desired result. ⁶ ■

We can use lemmata 3.1, 3.4, and 3.6 to reduce any k -tree G to a k -clique R . But we want the root R to be a k -clique that contains at least one distinguished node. In this way, we know that each subgraph of G in $OP_T(G)$ is the (graph) union of a subgraph in $SG(B'(R), R, \alpha)$ and a subgraph of the root R , where α is some colored subpartition of $V(R)$. The condition $V(R) \cap L \neq \emptyset$ is not restrictive as a k -tree with more than k nodes has at least two k -leaves; so, in finding the first $n - k$ elements of a peo for G (cf. Algorithm II) we can always avoid removing a specific k -leaf that is an element of L (if any).

The following set defines formally the pairs of colored subpartitions of $V(R)$ that

⁶Recall that $P_S[\alpha_S]$ is the probability that the connected components of S are those defined by α_S .

we want to consider in the termination step. Let $\alpha = (\pi, \sigma)$ such that $|\sigma| = 1$ and

$$Q(R, \alpha) = \{(\alpha_1, \alpha_2) \mid \alpha_1, \alpha_2 \in \mathcal{C}(R), V(\alpha_1) = V(\alpha_2), \alpha_1 \vee \alpha_2 = \alpha\}.$$

Clearly, there is a unique decomposition of any subgraph H in $SG(G, R, \alpha)$ as the (graph) union of a pair of subgraphs (H_1, H_2) in $SG(B'(R), R, \alpha_1) \times SG(R, R, \alpha_2)$. The following lemma states how to compute $Rel_l(G)$ from the information in the state of the k -clique R and R itself.

Lemma 3.7 (termination) Let $G = (V, E)$ be a k -tree network and $L \subseteq V$. Let R be a root of G obtained by applying the reduction paradigm and lemmata 3.1, 3.4, and 3.6 to G , and such that $V(R) \cap L \neq \emptyset$. Then

$$Rel_l(G) = \sum_{\alpha \in \mathcal{F}(R)} \sum_{\substack{(\alpha_1, \alpha_2) \text{ in} \\ Q(R, \alpha)}} s(\alpha_1, R) P_R[\alpha_2],$$

where $\mathcal{F}(R) = \{\alpha \mid \alpha \in \mathcal{C}(R), \alpha = (\pi, \sigma), |\sigma| = 1\}$.

Proof: We know

$$Rel_l(G) = \sum_{H \leq G} P_G[H] r(L, H) = \sum_{\alpha \in \mathcal{F}(R)} \sum_{H \in SG(G, R, \alpha)} P_G[H].$$

Expressing each subgraph H in $S(G, R, \alpha)$ as the union of pairs (H_1, H_2) in $SG(B'(R), R, \alpha_1) \times SG(R, R, \alpha_2)$,

$$Rel_l(G) = \sum_{\alpha \in \mathcal{F}(R)} \sum_{\substack{(\alpha_1, \alpha_2) \text{ in} \\ Q(R, \alpha)}} \sum_{\substack{H_1 \text{ in} \\ SG(B'(R), R, \alpha_1)}} \sum_{\substack{H_2 \text{ in} \\ SG(R, R, \alpha_2)}} P_G[H_1 \cup H_2].$$

But

$$P_G[H_1 \cup H_2] = P_{B'(R)}[H_1 \parallel up(V(\alpha_1), R)] P_R[H_2].$$

So

$$Rel_l(G) = \sum_{\alpha \in \mathcal{C}(R)} \sum_{\substack{(\alpha_1, \alpha_2) \text{ in} \\ Q(R, \alpha)}} s(\alpha_1, R) P_R[\alpha_2].$$

■ From lemmata 2.5, 3.1, 3.4, 3.6, and 3.7 follows our result.

Theorem 3.8 The l -terminal reliability of a partial k -tree network given with an embedding in a k -tree can be computed in $\mathcal{O}(n)$ time.

Resilience

Reliability measures defined in terms of expected values rather than probabilities of an event seem more difficult to calculate. Most of the work on these measures is based on network models restricted not only in their topology (partial 2-trees, typically), but also in the kinds of failures that may occur (e.g., only edge failures). Furthermore, to our knowledge, all the results assume that the network is undirected.

Table 2 summarizes the different network models that have been studied. Rows of the table classify networks according to the type of probability associated with node failures. Nodes may be fail-safe, fail with the same probability, or fail with arbitrary probability. Similarly, columns classify networks according to the type of probability associated with edge failures. The lower right corner of the table represents the most general network model. We present the results concerning resilience using this tabular format.

Table 2: Classification of Models According to Type of Component Failures

		Edge failures		
		$\forall e, p_e = 1$	$\forall e, p_e = c_1$	$\forall e, 0 \leq p_e \leq 1$
Node failures	$\forall v, p_v = 1$	-		
	$\forall v, p_v = c_2$			
	$\forall v, 0 \leq p_v \leq 1$			

The resilience of a network is the expected number of pairs of distinct nodes that can communicate. *Res* is a #P-complete problem even when the network is planar and

the nodes are fail-safe [24]. The only linear time algorithms for Res that we are aware of assume that the network is a partial 2-tree [5, 42, 4]. Table 3 and Table 4 describe the complexity of the of the asymptotically fastest polynomial time algorithms for Res on partial k -trees. The linear time algorithm presented in this section applies to all the networks described in Table 3 and Table 4.

Table 3: Linear and Quadratic Time Algorithms for Res on Partial 2-tree Networks

Res		Edge failures		
		$\forall e, p_e = 1$	$\forall e, p_e = c_1$	$\forall e, 0 \leq p_e \leq 1$
Node	$\forall v, p_v = 1$	-	$\mathcal{O}(n)$ [5]	$\mathcal{O}(n^2)$ [24], $\mathcal{O}(n)$ [42]
failures	$\forall v, p_v = c_2$	$\mathcal{O}(n)$ [4]		
	$\forall v, 0 \leq p_v \leq 1$			$\mathcal{O}(n)$ [43]

Table 4: Linear and Quadratic Time Algorithms for Res on Partial k -tree Networks

Res		Edge failures		
		$\forall e, p_e = 1$	$\forall e, p_e = c_1$	$\forall e, 0 \leq p_e \leq 1$
Node	$\forall v, p_v = 1$	-		
failures	$\forall v, p_v = c_2$			
	$\forall v, 0 \leq p_v \leq 1$			$\mathcal{O}(n)$ [43]

We can formulate $Res(G)$ as

$$Res(G) = \sum_{H \leq G} P_G[H] Pairs(H),$$

where $Pairs(H)$ is the number of pairs $\{u, v\}$ of nodes in V such that $u \stackrel{H}{\sim} v$ and $u \neq v$.

Alternatively, we can formulate $Res(G)$ as

$$Res(G) = \frac{1}{2} \sum_{u \in V} \sum_{\substack{v \in V \\ v \neq u}} Rel_2(G, u, v).$$

Therefore, by Theorem 3.8 we can compute the resilience of a partial k -tree network in $\mathcal{O}(n^3)$ time. We obtain a faster algorithm if we express $Res(G)$ in terms of 1-broadcast resilience:

$$Res(G) = \frac{1}{2} \left(\sum_{u \in V} Res_u(G) - p_u \right).$$

Thus, a linear time algorithm for Res_u (like the one discussed later, see Theorem 3.17) gives a quadratic time algorithm for Res . However, we are interested in developing a linear time algorithm for Res . The following lemma presents another characterization of Res that we will exploit in our linear time reduction algorithm for Res .

Lemma 3.9 Let $G = (V, E)$ be a network. Then

$$Res(G) = \frac{1}{2} \left(\sum_{H \leq G} P_G[H] \sum_{C \in Comp(H)} |V(C)|^2 - \sum_{v \in V} p_v \right),$$

where $Comp(H)$ denotes the set of connected components of H .

Proof: By definition of resilience

$$\begin{aligned} Res(G) &= \sum_{H \leq G} P_G[H] \sum_{C \in Comp(H)} \frac{|V(C)|^2 - |V(C)|}{2} \\ &= \frac{1}{2} \sum_{H \leq G} P_G[H] \sum_{C \in Comp(H)} |V(C)|^2 - \\ &\quad \frac{1}{2} \sum_{H \leq G} P_G[H] |V(C)|. \end{aligned}$$

So all we need to prove is

$$\sum_{H \leq G} P_G[H] |V(C)| = \sum_{v \in V} p_v.$$

But this follows because

$$\begin{aligned}
\sum_{H \leq G} P_G[H] |V(C)| &= \sum_{H \leq G} P_G[H] \sum_{v \in V} IsOp(H, v) \\
&= \sum_{v \in V} \sum_{H \leq G} P_G[H] IsOp(H, v) \\
&= \sum_{v \in V} p_v,
\end{aligned}$$

$$\text{where } IsOp(H, v) = \begin{cases} 1 & \text{if } v \in V(H), \\ 0 & \text{otherwise} \end{cases}$$

■

We now define the reduction algorithm for *Res*. We follow the terminology used in the algorithm for *Res_l*. However, unlike the algorithm for *Rel_l*, we do not use colored subpartitions but subpartitions of sets of nodes of a network. In addition, given a subgraph H of G , we define $SG(G, H, \pi)$ as the set of subgraphs G' of G such that $Cont(G', V(H)) = \pi$, for each subpartition π in the set of subpartitions $\Pi(V(H))$. The following four collections of values define the state of a k -clique or $(k+1)$ -clique K .

For each subpartition π in $\Pi(K)$, the state of K includes the value $s(\pi, K)$. We define $s(\pi, K)$ as the probability that a subgraph of the shell $B'(K)$ belongs to the class $SG(B'(K), K, \pi)$, given that the nodes in $V(\pi)$ are up and the nodes in $V(K) \setminus V(\pi)$ are down. If the probability that the nodes in $V(K) \setminus V(\pi)$ are down is zero, $s(\pi, K)$ is defined as zero.⁷ So

$$\begin{aligned}
s(\pi, K) &= P_{B'(K)}[SG(B'(K), K, \pi) \parallel up(V(\pi), K)] \\
&= \sum_{H \in SG(B'(K), K, \pi)} P_{B'(K)}[H \parallel up(V(\pi), K)], \tag{III.10}
\end{aligned}$$

where $P[A \parallel B]$ denotes $P[A | B]$ if $P[B] > 0$, otherwise it is 0.

⁷For the sake of simplicity we assume that the probability of operation of each node v in G is positive. If some p_v is zero we can either modify the formulae in this section or remove v and apply the algorithm to the resulting partial k -tree.

Also, for all non-empty subpartitions π in $\Pi(K)$ and each block C in π , the state of K includes $E(\pi, K, C)$. We define $E(\pi, K, C)$ as follows:

$$E(\pi, K, C) = \sum_{\substack{H \text{ in} \\ SG(B'(K), K, \pi)}} P_{B'(K)}[H \parallel up(V(\pi), K)] BN(K, H, C), \quad (\text{III.11})$$

where $BN(K, H, C)$ is the number of branch nodes (nodes in $B(K)$) connected to C via H . It is easy to verify that if $s(\pi, K) > 0$, $E(\pi, K, C)/s(\pi, K)$ is a conditional expected value, namely the expected number of nodes in $B(K)$ that are connected to C via H , given that H is a member of the class $SG(B'(K), K, \pi)$.

In addition, for all non-empty subpartitions π in $\Pi(K)$, and C_1, C_2 blocks of π , the state of K includes $EP(\pi, K, C_1, C_2)$, where

$$EP(\pi, K, C_1, C_2) = \sum_{\substack{H \text{ in} \\ SG(B'(K), K, \pi)}} P_{B'(K)}[H \parallel up(V(\pi), K)] BN(K, H, C_1) BN(K, H, C_2). \quad (\text{III.12})$$

Again, it is easy to verify that if $s(\pi, K) \neq 0$, $EP(\pi, K, C_1, C_2)/s(\pi, K)$ is a conditional expected value, namely the expected number of pairs (s, t) of nodes in $B(K)$ such that $s \stackrel{H}{\sim} C_1$, and $t \stackrel{H}{\sim} C_2$, given that H , a subgraph of $B'(K)$, is a member of $SG(B'(K), K, \pi)$.

Finally the state of K includes the value $EIP(K)$. We use $EIP(K)$ to denote the expected number of pairs of nodes in $B(K)$ that can communicate but are separated (isolated) from K . Formally, we define

$$EIP(K) = \sum_{H \leq B'(K)} P_{B'(K)}[H] \sum_{C \in I(H, K)} |V(C)|^2, \quad (\text{III.13})$$

where $I(H, K)$ is the set of connected components C of H that are isolated from K , i.e., $V(C) \cap V(K) = \emptyset$.

The next four lemmata define the initialization, reduction, and termination steps

of our algorithm for the resilience problem. The initialization lemma follows from the definitions of $B(K)$, $B'(K)$, $s(\pi, K)$, $E(\pi, K, C)$, $EP(\pi, K, C_1, C_2)$, and $EIP(K)$ (equations II.1, II.2, III.10, III.11, III.12, and III.13, respectively).

Lemma 3.10 (initialization) Let G be a k -tree network. Then

(a) For all K and π such that K is a k -clique of G and π is a subpartition in $\Pi(K)$,

$$s(\pi, K) = \begin{cases} 1 & \text{if } \pi \text{ consists of zero or more singletons and } \prod_{v \in V(K) \setminus V(\pi)} (1 - p_v) \neq 0, \\ 0 & \text{otherwise.} \end{cases}$$

(b) For all K , π , and C such that K is a k -clique of G , π is a non-empty subpartition in $\Pi(K)$, and C is a block in π ,

$$E(\pi, K, C) = 0.$$

(c) For all K , π , C_1 , and C_2 such that K is a k -clique of G , π is a non-empty subpartition in $\Pi(K)$, and C_1 and C_2 are sets of nodes in π ,

$$EP(\pi, K, C_1, C_2) = 0.$$

(d) For all K such that K is a k -clique of G ,

$$EIP(K) = 0.$$

Let us now consider the reduction step. Let G be a (partially reduced) k -tree, and v be a k -leaf of G with neighborhood K . Let K^+ be the graph induced by $V(K) \cup \{v\}$, and u_1, \dots, u_{k+1} be the nodes in K^+ . Also, for all $i = 1, \dots, k+1$, let K^i denote the k -clique induced by the nodes in $V(K^+) \setminus \{u_i\}$.

To compute the state of K^+ we consider all possible ways of obtaining π^+ , a subpartition in $\Pi(K^+)$, as the join of π_1, \dots, π_{k+1} , where π_i is a partition of $V(K^+) \setminus \{u_i\}$

($i = 1, \dots, k+1$). We use $T(\pi^+, K^+)$ to denote the set of $(k+1)$ -tuples of subpartitions of nodes in K^+ such that their join is π^+ and the i -th subpartition is a partition of $V(\pi^+) \setminus \{u_i\}$ ($i = 1, \dots, k+1$). Formally,

$$T(\pi^+, K^+) = \{(\pi_1, \dots, \pi_{k+1}) \mid \bigvee_{i=1}^{k+1} \pi_i = \pi^+ \wedge \forall i = 1, \dots, k+1, \pi_i \text{ is a partition of } V(\pi^+) \setminus \{u_i\}\}.$$

We use $\vec{\pi}$ to denote a $(k+1)$ -tuple in $T(\pi^+, K^+)$ and $\vec{\pi}_i$ to denote the i -th entry of $\vec{\pi}$.

By definition of $B'(K^+)$, a subgraph H of the shell $B'(K^+)$ is the (graph) union of $k+1$ graphs H_1, \dots, H_{k+1} such that each H_i is a subgraph of $B'(K^i)$ and $i = 1, \dots, k+1$ (cf. Equation II.4). Furthermore, a subgraph H is in $SG(B(K^+), K^+, \pi^+)$ if and only if each H_i is in $SG(B(K^i), K^i, \pi_i)$ for subpartitions π_i such that $(\pi_1, \dots, \pi_{k+1})$ is an element of $T(\pi^+, K^+)$. Formally, we make the following observations.

Observation 3.11 There is a bijection ϕ from $SG(B'(K^+), K^+, \pi^+)$ to

$$\bigcup_{\substack{(\pi_1, \dots, \pi_{k+1}) \\ \text{in } T(\pi^+, K^+)}} SG(B'(K^1), K^1, \pi_1) \times \dots \times SG(B'(K^{k+1}), K^{k+1}, \pi_{k+1})$$

such that $\phi(H) = (H_1, \dots, H_{k+1})$ iff $\bigcup_{i=1}^{k+1} H_i = H$. In particular,

$$\sum_{\substack{H \text{ in} \\ SG(B'(K^+), K^+, \pi^+)}} P_{B'(K^+)}[H \parallel Y] = \sum_{\vec{\pi} \in T(\pi^+, K^+)} \sum_{\substack{(H_1, \dots, H_{k+1}) \text{ in} \\ X_1 \times \dots \times X_{k+1}}} P_{B'(K^+)}[\bigcup_{i=1}^{k+1} H_i \parallel Y],$$

where Y denotes the condition $up(V(\pi^+), K^+)$ and $X_i = SG(B'(K^i), K^i, \vec{\pi}_i)$, $1 \leq i \leq k+1$.

Moreover, for each $(k+1)$ -tuple (H_1, \dots, H_{k+1}) in $X_1 \times \dots \times X_{k+1}$,

$$P_{B'(K^+)}[\bigcup_{i=1}^{k+1} H_i \parallel Y] = \prod_{i=1}^{k+1} P_{B'(K^i)}[H_i \parallel up(V(\vec{\pi}_i), K^i)].$$

We can now prove the following lemma.

Lemma 3.12 Let G be a k -tree network that has been partially reduced using some perfect elimination ordering and the general reduction paradigm. Let v be the next k -leaf to be removed. Let K be the neighborhood of v , and K^+ be the subgraph of G induced by $V(K) \cup \{v\}$. Then

(a) For all π^+ such that π^+ is a subpartition in $\Pi(K^+)$,

$$s(\pi^+, K^+) = \sum_{\bar{\pi} \in T(\pi^+, K^+)} \prod_{i=1}^{k+1} s(\bar{\pi}_i, K^i).$$

(b) For all π^+ and C such that π^+ is a non-empty subpartition in $\Pi(K^+)$ and C is a block in π^+ ,

$$E(\pi^+, K^+, C) = \sum_{\bar{\pi} \in T(\pi^+, K^+)} \sum_{i=1}^{k+1} \prod_{\substack{j=1 \\ j \neq i}}^{k+1} s(\bar{\pi}_j, K^j) \sum_{\substack{D \in \mathcal{R}_i \\ D \subseteq C}} E(\bar{\pi}_i, K^i, D).$$

(c) For all π^+ , C_1 , and C_2 such that π^+ is a non-empty subpartition in $\Pi(K^+)$ and $C_1, C_2 \in \pi^+$,

$$EP(\pi^+, K^+, C_1, C_2) = \sum_{\bar{\pi} \in T(\pi^+, K^+)} \sum_{i=1}^{k+1} \sum_{j=1}^{k+1} \sum_{\substack{D_1 \in \mathcal{R}_i \\ D_1 \subseteq C_1}} \sum_{\substack{D_2 \in \mathcal{R}_j \\ D_2 \subseteq C_2}} F(i, j, D_1, D_2),$$

where

$$F(i, j, D_1, D_2) = \begin{cases} \prod_{\substack{l=1 \\ l \neq i}}^{k+1} s(\bar{\pi}_l, K^l) EP(\bar{\pi}_i, K^i, D_1, D_2) & \text{if } i = j, \\ \prod_{\substack{l=1 \\ l \neq i \wedge l \neq j}}^{k+1} s(\bar{\pi}_l, K^l) E(\bar{\pi}_i, K^i, D_1) E(\bar{\pi}_j, K^j, D_2) & \text{otherwise.} \end{cases}$$

(d) $EIP(K^+) = \sum_{i=1}^{k+1} EIP(K^i)$.

Proof: Let us use Y to denote the condition $up(V(\pi^+), K^+)$ and use Y_i to denote the condition $up(V(\bar{\pi}_i), K^i)$.

(a) Using the definition of $s(\pi^+, K^+)$ (Equation III.10) and Observation 3.11 we get

$$\begin{aligned}
s(\pi^+, K^+) &= \sum_{\substack{H \text{ in} \\ SG(B'(K^+), K^+, \pi^+)}} P_{B'(K^+)}[H \parallel Y] \\
&= \sum_{\pi \in T(\pi^+, K^+)} \sum_{\substack{(H_1, \dots, H_{k+1}) \text{ in} \\ X_1 \times \dots \times X_{k+1}}} P_{B'(K^+)}[\bigcup_{i=1}^{k+1} H_i \parallel Y] \\
&= \sum_{\pi \in T(\pi^+, K^+)} \sum_{\substack{(H_1, \dots, H_{k+1}) \text{ in} \\ X_1 \times \dots \times X_{k+1}}} \prod_{i=1}^{k+1} P_{B'(K^i)}[H_i \parallel Y_i].
\end{aligned}$$

The result follows by Observation 3.3.

(b) Analogously, we use the definition of $E(\pi^+, K^+, C)$ (Equation III.11), and Observation 3.11 to obtain

$$E(\pi^+, K^+, C) = \sum_{\pi \in T(\pi^+, K^+)} \sum_{\substack{(H_1, \dots, H_{k+1}) \text{ in} \\ X_1 \times \dots \times X_{k+1}}} \prod_{i=1}^{k+1} P_{B'(K^i)}[H_i \parallel Y_i] BN(K^+, \bigcup_{j=1}^{k+1} H_j, C).$$

But

$$BN(K^+, \bigcup_{j=1}^{k+1} H_j, C) = \sum_{j=1}^{k+1} \sum_{\substack{D \subseteq C \\ D \in \pi_j}} BN(K^j, H_j, D). \quad (\text{III.14})$$

So, simple algebraic manipulation and Observation 3.3 yield the desired result.

(c) Similarly, we use the definition of $EP(\pi^+, K^+, C_1, C_2)$ (Equation III.12), and Observation 3.11 to get that $EP(\pi^+, K^+, C_1, C_2)$ is

$$\sum_{\pi \text{ in} T(\pi^+, K^+)} \sum_{\substack{(H_1, \dots, H_{k+1}) \text{ in} \\ X_1 \times \dots \times X_{k+1}}} \prod_{i=1}^{k+1} P_{B'(K^i)}[H_i \parallel Y_i] BN(K^+, \bigcup_{j=1}^{k+1} H_j, C_1) BN(K^+, \bigcup_{j=1}^{k+1} H_j, C_2).$$

Equation III.14 and additional algebraic manipulation suffice to prove (c).

(d) By definition of $EIP(K^+)$ (Equation III.13),

$$EIP(K^+) = \sum_{H \subseteq B'(K^+)} P_{B'(K^+)}[H] \sum_{C \in I(H, K^+)} |V(C)|^2.$$

We can compute the sum above by enumerating the subgraphs H of the shell $B'(K^+)$ in some convenient order. For each subset W of nodes in K^+ we define $Z_i(W)$ as the set of subgraphs of the shell $B'(K^i)$ such that $V(H) \cap V(K^i) = W \setminus \{u_i\}$. Thus,

$$\sum_{\substack{H \subseteq B'(K^+) \\ V(H) \cap V(K^+) = W}} P_{B'(K^+)}[H] = \sum_{\substack{(H_1, \dots, H_{k+1}) \text{ in} \\ Z_1(W) \times \dots \times Z_{k+1}(W)}} P_{B'(K^+)}[\bigcup_{j=1}^{k+1} H_j]$$

and

$$EIP(K^+) = \sum_{W \subseteq V(K^+)} \sum_{\substack{(H_1, \dots, H_{k+1}) \text{ in} \\ Z_1(W) \times \dots \times Z_{k+1}(W)}} P_{B'(K^+)}[\bigcup_{j=1}^{k+1} H_j] \sum_{C \in I(\bigcup_{j=1}^{k+1} H_j, K^+)} |V(C)|^2.$$

But

$$\sum_{C \in I(\bigcup_{j=1}^{k+1} H_j, K^+)} |V(C)|^2 = \sum_{i=1}^{k+1} \sum_{C \in I(H_i, K^i)} |V(C)|^2.$$

So

$$EIP(K^+) = \sum_{i=1}^{k+1} \sum_{W \subseteq V(K^+)} \sum_{\substack{(H_1, \dots, H_{k+1}) \text{ in} \\ Z_1(W) \times \dots \times Z_{k+1}(W)}} P_{B'(K^+)}[\bigcup_{j=1}^{k+1} H_j] f(H_i), \quad (\text{III.15})$$

where $f(H_i) = \sum_{C \in I(H_i, K^i)} |V(C)|^2$.

Define

$$X = \sum_{\substack{(H_1, \dots, H_{k+1}) \text{ in} \\ Z_1(W) \times \dots \times Z_{k+1}(W)}} P_{B'(K^+)}[\bigcup_{j=1}^{k+1} H_j] f(H_i). \quad (\text{III.16})$$

Clearly,

$$X = \sum_{\substack{(H_1, \dots, H_{k+1}) \text{ in} \\ Z_1(W) \times \dots \times Z_{k+1}(W)}} P_{B'(K^+)} \left[\bigcup_{j=1}^{k+1} H_j \parallel up(W, K^+) \right] g(W, K^+) f(H_i),$$

where $g(W, K^+)$ is the probability that the nodes in W are the operational nodes in K^+ , i.e.,

$$g(W, K^+) = \prod_{w \in W} p_w \prod_{u \in V(K^+) \setminus W} (1 - p_u).$$

By statistical independence of node and edge failures and by Observation 3.3,

$$\begin{aligned} X &= g(W, K^+) \sum_{\substack{(H_1, \dots, H_{k+1}) \text{ in} \\ Z_1(W) \times \dots \times Z_{k+1}(W)}} \prod_{j=1}^{k+1} P_{B'(K^j)}[H_j \parallel up(W, K^+)] f(H_i) \\ &= g(W, K^+) \prod_{\substack{j=1 \\ j \neq i}}^{k+1} \sum_{\substack{H \in Z_j(W) \\ j \neq i}} P_{B'(K^j)}[H \parallel up(W, K^+)] \\ &\quad \sum_{H \in Z_i(W)} P_{B'(K^i)}[H \parallel up(W, K^+)] f(H). \end{aligned} \tag{III.17}$$

Clearly,

$$\sum_{H \in Z_j(W)} P_{B'(K^j)}[H \parallel up(W, K^+)] = 1.$$

So

$$\begin{aligned} X &= g(W, K^+) \sum_{H \in Z_i(W)} P_{B'(K^i)}[H \parallel up(W, K^+)] f(H) \\ &= q_i \sum_{H \in Z_i(W)} P_{B'(K^i)}[H] f(H), \end{aligned} \tag{III.18}$$

where q_i is p_{u_i} if node u_i is in W , otherwise $q_i = (1 - p_{u_i})$.

Combining equations III.15, III.16, and III.18 we obtain

$$EIP(K^+) = \sum_{i=1}^{k+1} \sum_{W \subseteq V(K^+)} q_i \sum_{H \in \mathcal{Z}_i(W)} P_{B'(K^i)}[H] f(H),$$

and the result follows by observing that for each set of nodes W such that $u_i \in W$ the set $W \setminus \{u_i\}$ is also a subset of $V(K^+)$. Thus

$$\begin{aligned} EIP(K^+) &= \sum_{i=1}^{k+1} \sum_{W \subseteq V(K^+)} \sum_{H \in \mathcal{Z}_i(W)} P_{B'(K^i)}[H] f(H) \\ &= \sum_{i=1}^{k+1} \sum_{H \subseteq B'(K^i)} P_{B'(K^i)}[H] f(H). \end{aligned}$$

■

We now show how to update the state of K given the state of K^+ . Let S be the star network induced by the k edges that link v to K . Let $\Pi'(S)$ denote the set of subpartitions of nodes in S that consist of singletons only possibly with exception of the set containing node v . The set $\Pi'(S)$ models the set of operational subgraphs of S . We update the state of K by considering all possible ways of obtaining each subpartition π in $\Pi(K)$ as the join of pairs (π_1, π_2) of subpartitions in $\Pi(K^+)$ and $\Pi'(S)$, respectively. The following set defines formally the pairs of subpartitions that we want to consider:

$$PS(\pi, K) = \{(\pi_1, \pi_2) \mid \pi_1 \in \Pi(K^+), \pi_2 \in \Pi'(S), V(\pi_1) = V(\pi_2), \text{ and } (\pi_1 \vee \pi_2)/v = \pi\}.$$

The following Observation is useful in proving Lemma 3.14.

Observation 3.13 There is a bijection ψ such that

$$\psi : SG(B'(K), K, \pi) \mapsto \bigcup_{\substack{(\pi_1, \pi_2) \\ \text{in } PS(\pi, K)}} SG(B'(K^+), K^+, \pi_1) \times SG(S, S, \pi_2)$$

and $\psi(H) = (H_1, H_2)$ iff $H = H_1 \cup H_2$.

We can now establish how to update the state of K from the state of K^+ and the star graph S .

Lemma 3.14 Let G be a k -tree network that has been partially reduced using some perfect elimination ordering and the general reduction paradigm. Let v be the next k -leaf to be removed. Let K be the neighborhood of v , and K^+ be the subgraph of G induced by $V(K) \cup \{v\}$. In addition, let S be the star graph consisting of the k edges that link v to K . Then

- (a) For all π such that π is a subpartition in $\Pi(K)$,

$$s(\pi, K) = \sum_{\substack{(\pi_1, \pi_2) \\ \text{in } PS(\pi, K)}} s(\pi_1, K^+) P_S[\pi_2 \parallel up(V(\pi), K)].$$

- (b) For all π, C , such that π is a non-empty subpartition in $\Pi(K)$ and C is a block in π ,

$$E(\pi, K, C) = \sum_{\substack{(\pi_1, \pi_2) \\ \text{in } PS(\pi, K)}} P_S[\pi_2 \parallel up(V(\pi), K)] \left(\sum_{\substack{D \in \pi_1 \\ D \subseteq C'}} E(\pi_1, K^+, D) + z(v, C') \right),$$

where C' is the block of $\pi_1 \vee \pi_2$ such that $C \subseteq C'$, and

$$z(v, C') = \begin{cases} s(\pi_1, K^+) & \text{if } v \in C', \\ 0 & \text{otherwise.} \end{cases}$$

- (c) For all π, C_1 , and C_2 such that π is a non-empty subpartition in $\Pi(K)$, and C_1 and C_2 are blocks of the subpartition π ,

$$EP(K, \pi, C_1, C_2) = \sum_{\substack{(\pi_1, \pi_2) \\ \text{in } PS(\pi, K)}} P_S[\pi_2 \parallel up(V(\pi), K)]$$

$$\sum_{\substack{D_1 \in \pi_1 \\ D_1 \subseteq C_1'}} \sum_{\substack{D_2 \in \pi_2 \\ D_2 \subseteq C_2'}} (EP(\pi_1, K^+, D_1, D_2) + R(v, D_1, D_2)),$$

where

$$R(v, D_1, D_2) = \begin{cases} 2 E(\pi_1, K^+, D_1) + s(\pi_1, K^+) & \text{if } v \in D_1 \wedge v \in D_2, \\ E(\pi_1, K^+, D_1) & \text{if } v \notin D_1 \wedge v \in D_2, \\ E(\pi_1, K^+, D_2) & \text{if } v \in D_1 \wedge v \notin D_2, \\ 0 & \text{otherwise.} \end{cases}$$

- (d) Let $\Pi(v, K^+, S)$ be the set of pairs (π_1, π_2) of subpartitions in $\Pi(K^+)$ such that $V(\pi_1) = V(\pi_2)$, $\pi_2 \in \Pi'(S)$, $\{v\} \in \pi_1$, and $\{v\} \in \pi_2$. Then

$$EIP(K) = EIP(K^+) +$$

$$\sum_{\substack{(\pi_1, \pi_2) \\ \text{in } \Pi(v, K^+, S)}} (EP(\pi_1, K^+, \{v\}, \{v\}) + 2 E(\pi_1, K^+, \{v\}) + s(\pi_1, K^+)) P_S[\pi_2 \parallel up(V(\pi), K)].$$

Proof: The proof follows by applying Observation 3.13 to the definition of $s(\pi, K)$, $E(\pi, K, C)$, $EP(\pi, K, C_1, C_2)$, and $EIP(K)$, and then performing basic algebraic manipulations. Let us use Y to denote the condition $up(V(\pi)) \wedge dn(V(K) \setminus V(\pi))$. In addition, let Y_1 denote the condition $up(V(\pi_1)) \wedge dn(V(K^+) \setminus V(\pi_1))$.

- (a) By definition (Equation III.10) and Observation 3.13,

$$\begin{aligned} s(\pi, K) &= \sum_{H \in SG(B'(K), K, \pi)} P_{B'(K)}[H \parallel Y] \\ &= \sum_{\substack{(\pi_1, \pi_2) \\ \text{in } PS(\pi, K)}} \sum_{H_1 \text{ in } SG(B'(K^+), K^+, \pi_1)} \sum_{H_2 \text{ in } SG(S, S, \pi_2)} P_{B'(K)}[H_1 \cup H_2 \parallel Y]. \end{aligned}$$

But

$$P_{B'(K)}[H_1 \cup H_2 \parallel Y] = P_{B'(K^+)}[H_1 \parallel Y_1] P_S[H_2 \parallel Y]. \quad (\text{III.19})$$

Therefore

$$s(\pi, K) = \sum_{\substack{(\pi_1, \pi_2) \\ \text{in } PS(\pi, K)}} s(\pi_1, K^+) \sum_{\substack{H_2 \text{ in} \\ SG(S, S, \pi_2)}} P_S[H_2 \parallel Y],$$

which is the desired result. ⁸

- (b) Analogously, we use the definition of $E(\pi, K, C)$ (Equation III.11) and Observation 3.13 to obtain

$$\begin{aligned} E(\pi, K, C) &= \\ E(\pi, K, C) &= \\ &\sum_{\substack{(\pi_1, \pi_2) \\ \text{in } PS(\pi, K)}} \sum_{H_1 \in X_1} \sum_{H_2 \in X_2} P_{B'(K)}[H_1 \cup H_2 \parallel Y] BN(K, H_1 \cup H_2, C), \end{aligned} \quad (\text{III.20})$$

where $X_1 = SG(B'(K^+), K^+, \pi_1)$ and $X_2 = SG(S, S, \pi_2)$.

Let C' be the block of $\pi_1 \vee \pi_2$ such that $C \subseteq C'$. Notice that a block C in $(\pi_1 \vee \pi_2) \setminus \{v\}$ is obtained by taking $\bigcup_{\substack{D \in \pi_1 \\ D \subseteq C'}} D \setminus \{v\}$. Thus,

$$BN(K, H_1 \cup H_2, C) = \sum_{\substack{D \in \pi_1 \\ D \subseteq C'}} BN(K^+, H_1, D) + \delta(v, C'), \quad (\text{III.21})$$

$$\text{where } \delta(v, C') = \begin{cases} 1 & \text{if } v \in C', \\ 0 & \text{otherwise.} \end{cases}$$

Combining equations III.19, III.20, and III.21,

$$\begin{aligned} E(\pi, K, C) &= \\ &\sum_{\substack{(\pi_1, \pi_2) \\ \text{in } PS(\pi, K)}} \left(\sum_{H_1 \in X_1} P_{B'(K^+)}[H_1 \parallel Y_1] \left(\sum_{\substack{D \in \pi_1 \\ D \subseteq C'}} BN(K^+, H_1, D) + \delta(v, C') \right) \right) \sum_{H_2 \in X_2} P_S[H_2 \parallel Y]. \end{aligned}$$

⁸Recall from section 2 that $P_S[\pi_2]$ is the probability that the connected components of S are those defined by π_2 .

Simple algebraic manipulation completes the proof.

(c) By definition (Equation III.12) and Observation 3.13,

$$\begin{aligned}
 EP(\pi, K, C_1, C_2) = & \\
 & \sum_{\substack{(\pi_1, \pi_2) \\ \text{in } PS(\pi, K)}} \sum_{H_1 \in X_1} \sum_{H_2 \in X_2} P_{B'(K)}[H_1 \cup H_2 \parallel Y] BN(K, H_1 \cup H_2, C_1) BN(K, H_1 \cup H_2, C_2),
 \end{aligned} \tag{III.22}$$

where X_1 and X_2 are defined as in (b) above.

Besides, by Equation III.21, the product $BN(K, H_1 \cup H_2, C_1) BN(K, H_1 \cup H_2, C_2)$ is one of the following values:

$$\left(\sum_{\substack{D_1 \in \pi_1 \\ D_1 \subseteq C'_1}} BN(K^+, H_1, D_1) + 1 \right) \left(\sum_{\substack{D_2 \in \pi_1 \\ D_2 \subseteq C'_2}} BN(K^+, H_1, D_2) + 1 \right)$$

if $\delta(v, C'_1)\delta(v, C'_2) = 1$, or

$$\left(\sum_{\substack{D_1 \in \pi_1 \\ D_1 \subseteq C'_1}} BN(K^+, H_1, D_1) + 1 \right) \sum_{\substack{D_2 \in \pi_1 \\ D_2 \subseteq C'_2}} BN(K^+, H_1, D_2)$$

if $\delta(v, C'_1) = 1$ but $\delta(v, C'_2) = 0$, or

$$\sum_{\substack{D_1 \in \pi_1 \\ D_1 \subseteq C'_1}} BN(K^+, H_1, D_1) \left(\sum_{\substack{D_2 \in \pi_1 \\ D_2 \subseteq C'_2}} BN(K^+, H_1, D_2) + 1 \right)$$

if $\delta(v, C'_1) = 0$ but $\delta(v, C'_2) = 1$, or

$$\sum_{\substack{D_1 \in \pi_1 \\ D_1 \subseteq C'_1}} BN(K^+, H_1, D_1) \sum_{\substack{D_2 \in \pi_1 \\ D_2 \subseteq C'_2}} BN(K^+, H_1, D_2)$$

otherwise.

The result follows by considering each of the four cases above and simplifying Equation III.21 accordingly.

(d) By definition (Equation III.13),

$$EIP(K) = \sum_{H \leq B'(K)} P_{B'(K)}[H] \sum_{C \in I(H,K)} |V(C)|^2.$$

We can enumerate the subgraphs of $B'(K)$ in an more convenient order:

$$EIP(K) = \sum_{H_1 \leq B'(K^+)} \sum_{\substack{H_2 \leq S \\ V(H_2) \cap V(K^+) = V(H_1) \cap V(K^+)}} P_{B'(K)}[H_1 \cup H_2] \sum_{C \in I(H_1 \cup H_2, K)} |V(C)|^2.$$

For each subgraph H of $B'(K)$, a connected component C of H is isolated from the k -clique K if and only if C is isolated from K^+ or $V(C) \cap V(K^+) = \{v\}$ and there are no edges in H connecting node v to a node in K . Let $\Pi(v, K^+, S)$ be the set of pairs (π_1, π_2) of subpartitions in $\Pi(K^+)$ such that $V(\pi_1) = V(\pi_2)$, $\pi_2 \in \Pi'(S)$, $\{v\} \in \pi_1$, and $\{v\} \in \pi_2$. Then

$$EIP(K) = A + B,$$

where

$$A = \sum_{H_1 \leq B'(K^+)} \sum_{\substack{H_2 \leq S \\ V(H_2) \cap V(K^+) = V(H_1) \cap V(K^+)}} P_{B'(K)}[H_1 \cup H_2] \sum_{C_1 \in I(H_1, K^+)} |V(C_1)|^2$$

and

$$B = \sum_{\substack{(\pi_1, \pi_2) \text{ in} \\ \Pi(v, K^+, S)}} \sum_{H_1 \in \mathcal{X}_1} \sum_{H_2 \in \mathcal{X}_2} P_{B'(K)}[H_1 \cup H_2] (BN(K^+, H_1, \{v\}) + 1)^2.$$

Clearly

$$P_{B'(K)}[H_1 \cup H_2] = P_{B'(K^+)}[H_1] P_S[H_2 \parallel up(V(H_2) \cap V(K^+)) \wedge dn(V(K^+) \setminus V(H_2))]$$

and

$$\sum_{\substack{H_2 \leq S \\ V(H_2) \cap V(K^+) = V(H_1) \cap V(K^+)}} P_{B'(K)}[H_2 \parallel up(V(H_2) \cap V(K^+)) \wedge dn(V(K^+) \setminus V(H_2))] = 1.$$

Thus

$$A = \sum_{H_1 \leq B'(K^+)} P_{B'(K^+)}[H_1] \sum_{C_1 \in I(H_1, K^+)} |V(C_1)|^2.$$

Similarly,

$$B = \sum_{\substack{(\pi_1, \pi_2) \text{ in } \\ \Pi(v, K^+, S)}} \sum_{H_1 \in X_1} P_{B'(K)}[H_1 \parallel Y_1] (BN(K^+, H, \{v\}) + 1)^2 \sum_{H_2 \in SG(S, S, \pi_2)} P_S[H_2]$$

and simple algebraic manipulation gives

$$B = \sum_{\substack{(\pi_1, \pi_2) \text{ in } \\ \Pi(v, K^+, S)}} (EP(\pi_1, K^+, \{v\}) + 2E(\pi_1, K^+, \{v\}) + s(\pi_1, K^+)) P_S[\pi_2].$$

■

We can use lemmata 3.10, 3.12, and 3.14 to reduce any k -tree G to a k -clique R . We compute $Res(G)$ by combining the information in the state of the k -clique R with the effect of the edges between nodes in R . Before computing $Res(G)$ we extend the values in the state of R (statistics about $B'(R)$) to values about the graph G itself. Some additional notation is in order. Let $Res'(G)$ denote the expected number of *ordered* pairs of nodes in G that can communicate (including pairs of the form (u, u)). So

$$Res'(G) = \sum_{H \leq G} P_G[H] \sum_{C \in Comp(H)} |V(C)|^2.$$

Notice that by Lemma 3.9

$$Res(G) = \frac{1}{2} (Res'(G) - \sum_{v \in V} p_v).$$

Therefore we only need to prove that $Res'(G)$ can be computed from the state of the root R and the effect of the nodes and edges in R .

To account for the effect of the edges between nodes in R we define the following functions. Letting π be any non-empty subpartition of the nodes in the root R and $C \in \pi$, define

$$EP'(\pi, R, C) = \sum_{H \in SG(G, R, \pi)} P_G[H] N(H, C)^2,$$

where $N(H, C) = |\{y \in G \mid y \stackrel{H}{\sim} C\}|$. Finally, let $EIP'(R)$ denote the expected number of ordered pairs (u, v) of nodes in G that can communicate such that $u \neq R$ and $v \neq R$. So

$$EIP'(R) = \sum_{H \leq G} P_G[H] \sum_{C \in I(H, R)} |V(C)|^2.$$

The following lemma states how to compute $Res'(G)$ from the state of the root R .

Lemma 3.15 (termination) Let $G = (V, E)$ be a k -tree network and R be a root of G obtained by applying the reduction paradigm and lemmata 3.10, 3.12, and 3.14 to G . Then

- (a) For all π, C , such that π is a non-empty subpartition in $\Pi(R)$, and C is a block of π ,

$$EP'(\pi, R, C) = \sum_{\substack{(\pi_1, \pi_2) \\ \pi_1 \wedge \pi_2 \text{ part. of } V(\pi) \\ \pi_1 \vee \pi_2 = \pi}} P_R[\pi_2] (s(\pi_1, R) |C|^2 + \sum_{\substack{D \in \pi_1 \\ D \subseteq C}} (2 |C| E(\pi_1, R, D) + EP(\pi_1, R, D, D))).$$

- (b) $EIP'(R) = EIP(R)$.

- (c) $Res'(G) = EIP'(R) + \sum_{\pi \in \Pi(R)} \sum_{C \in \pi} EP'(\pi, R, C)$.

Proof: The proofs follow easily by algebraic manipulation of the definitions of $EP'(\pi, R, C)$, $EIP'(R)$, and $Res'(G)$. We present some details of the proof for (a) only.

Let Y denote the condition $up(V(\pi_1)) \wedge dn(V(R) \setminus V(\pi_1))$. Clearly,

$$\begin{aligned} EP'(\pi, R, C) &= \sum_{H \in SG(G, R, \pi)} P_G[H] N(H, C) \\ &= \sum_{\substack{(\pi_1, \pi_2) \\ \pi_1 \wedge \pi_2 \text{ part. of } V(\pi) \\ \pi_1 \vee \pi_2 = \pi}} \sum_{\substack{H_1 \text{ in} \\ SG(B'(R), R, \pi_1)}} \sum_{\substack{H_2 \text{ in} \\ SG(R, R, \pi_2)}} P_{B'(R)}[H_1 \mid Y] P_R[H_2] N(H_1 \cup H_2, C)^2 \end{aligned}$$

and the result follows because

$$N(H_1 \cup H_2, C)^2 = |C|^2 + 2|C| \sum_{\substack{D \in \pi_1 \\ D \subseteq C}} BN(H_1, D) + \left(\sum_{\substack{D \in \pi_1 \\ D \subseteq C}} BN(H_1, D) \right)^2.$$

■

Therefore, we have the following theorem.

Theorem 3.16 The resilience of a partial k -tree network given with an embedding in a k -tree can be computed in $\mathcal{O}(n)$ time.

Proof: Correctness follows from lemmata 3.10, 3.12, 3.14, and 3.15. Timing follows from lemmata 3.10, 3.12, 3.14, 3.15 and 2.5. ■

Broadcast Resilience

The broadcast resilience $Res_v(G)$ of a network G with respect to a node v is the expected number of nodes that can be reached from v . Thus

$$Res_v(G) = \sum_{H \subseteq G} P_G[H] \sum_{u \in V} \tau(v, \{u\}, H).$$

We can compute $Res_v(G)$ with a straightforward simplification of the reduction algorithm for Res . First we make sure that the root R includes node v . Secondly, we maintain in $state(K)$ the values $s(\pi, K)$ and $E(\pi, K, C)$ only (for all subpartitions π in $\Pi(K)$ and $C \in \pi$). This is possible because the algorithm for Res updates the values $s(\pi, K)$ and

$E(\pi, K, C)$ by combining values $s(\pi', K')$ and $E(\pi', K', C')$ only. Finally, we modify the termination step to compute

$$Res_v = \sum_{\pi \in \Pi'(R)} E(\pi, R, C_v),$$

where $\Pi'(R)$ is the set of subpartitions of $V(R)$ such that $v \in V(\pi)$, C_v is the block of π that contains v , and $E'(\pi, K, C)$ is the expected number of nodes in the connected component that contains the nodes $V(C)$. It is easy to verify that

$$E'(\pi, K, C) = \sum_{\substack{(\pi_1, \pi_2) \\ \pi_1 \wedge \pi_2 \text{ part. of } V(\pi) \\ \pi_1 \vee \pi_2 = \pi}} P_R[\pi_2] \left(s(\pi_1, R) |C| + \sum_{\substack{D \in \pi_1 \\ D \subseteq C}} E(\pi_1, R, D) \right). \quad (\text{III.23})$$

Therefore we obtain the following theorem.

Theorem 3.17 The broadcast resilience of a partial k -tree network given with an embedding in a k -tree can be computed in $\mathcal{O}(n)$ time.

l -Broadcast Resilience

The l -broadcast resilience of a network G is defined as

$$Res_l(G, L) = \sum_{H \leq G} P_G[H] \sum_{v \in V} \tau(v, L, H), \quad (\text{III.24})$$

i.e., it is the expected number of nodes that receive a message broadcasted from each node in L . This measure is useful in determining the best sources for broadcasts in a network. We are interested in solving Res_l for $l > 1$.

It would be convenient if we could solve Res_l by using the reduction algorithm for Res_v . For instance, we may consider computing $\sum_{v \in L} Res_v$. However, this approach would only give us an upper bound on Res_l (it would count more than once the nodes connected to L). Another alternative is to try to solve Res_l from the information in $state(R)$. This approach works when L is a subset of the nodes in R .

Theorem 3.18 Let $G = (V, E)$ be a partial k -tree network and R be a root of G obtained by applying the reduction algorithm for Res_v to G . Let $L \subseteq V(R)$. Then

$$Res_l(G, L) = \sum_{\pi \in \Pi(R)} \sum_{\substack{C \in \pi \\ V(C) \cap L \neq \emptyset}} E'(\pi, K, C) \quad (\text{III.25})$$

and $Res_l(G, L)$ can be computed in $\mathcal{O}(n)$ time.

Proof: The identity follows by simple algebraic manipulation of Equation III.23 and Equation III.24. Timing is a corollary of Theorem 3.17. \blacksquare

A linear time solution for Res_l with respect to an arbitrary set L can be obtained by defining a reduction algorithm explicitly for this problem. We will present neither a detailed description of this reduction algorithm nor a formal proof of correctness because it is a combination of the ideas already presented in the algorithms for Rel_l and Res_v . Naturally, as there is a set of distinguished nodes L , we borrow the notion of colored subpartitions from the algorithm for Rel_l . In contrast with the approach for Rel_l , any subgraph of G is favorable. In particular, distinguished nodes may be elements of connected components that are isolated from a k -clique K . From the algorithm for Res we borrow the idea of keeping track of values similar to $s(\pi, K)$, $E(\pi, K, C)$, and $EIP(K)$. The following definitions formalize these ideas.

Let H be a subgraph of G . For each colored subpartition α in $\mathcal{C}(H)$, $SG(G, H, \alpha)$ is the set of subgraphs of G whose L -contraction with respect to $V(H)$ is α . The state of a k -clique is defined by the following three identities.

For each subpartition $\alpha \in \mathcal{C}(K)$,

$$s(\alpha, K) = \sum_{H \in SG(B'(K), K, \alpha)} P_{B'(K)}[H \parallel up(V(\alpha), K)].$$

For each subpartition $\alpha = (\pi, \sigma) \in \mathcal{C}(K)$ and each block C in π ,

$$E(\alpha, K, C) = \sum_{H \in SG(B'(K), K, \pi)} P_{B'(K)}[H \parallel up(V(\pi), K)] BN(K, H, C).$$

We also keep track of the expected number of nodes in gray connected components that are isolated from K :

$$EIN(K) = \sum_{H \leq G} P_{B'(K)}[H] \sum_{\substack{C \in I(H,K) \\ V(C) \cap L \neq \emptyset}} |V(C)|.$$

The initialization, reduction, and termination steps are defined by lemmata very similar to the lemmata for Res .

Theorem 3.19 The l -broadcast resilience of a partial k -tree network given with an embedding in a k -tree can be computed in $\mathcal{O}(n)$ time.

Network Broadcast Facility Location Problem

Nel [49] defines the Network Broadcast Facility Location problem (with parameter l) as finding a set of l nodes of G that maximizes the l -broadcast resilience of G , i.e.,

$$NBFL_l(G) = \max_{\substack{L \subseteq V(G) \\ |L|=l}} \{Res_l(G, L)\}.$$

Therefore, a trivial corollary of Theorem 3.19 is

Corollary 3.20 The $NBFL_l$ problem on partial k -tree networks given with an embedding in a k -tree can be computed in $\mathcal{O}(n^{l+1})$ time.

Concluding Remarks

The reduction paradigm introduced in [10] is a powerful tool to solve network reliability problems restricted to partial k -tree networks. In this chapter we presented exact linear time algorithms to solve Rel_l , Res , Res_l , and a polynomial time algorithm for $NBFL_l$. The technique employed can be applied to solve other network reliability problems. For example, given two nodes that maximize $Res_v(G)$ we may want to discriminate among them by computing the standard deviation of X , where X is a random variable denoting the size of the connected component containing the selected node. We believe

that the basic technique presented here can be generalized to compute this and other reliability measures.

Res is a particularly difficult problem. To our knowledge it had been solved in polynomial time only when the network is a partial 2-tree and either edges or nodes fail, but not both. The reduction algorithm for *Res* that we presented is very useful not only because it runs in linear time, but also because it is the basis for linear and polynomial time algorithms for *Res_v*, *Res_l*, and *NBFL_l*.

Even though this research is not concerned with the development of efficient approximation algorithms for general networks, we must point out that the availability of efficient exact algorithms for restricted classes of graphs may result in better approximation algorithms for general graphs. In particular, edge-packing techniques may benefit from the existence of efficient exact algorithms for partial k -trees. Colbourn [23] indicates that edge-packing by partial 2-trees is a promising option to obtain better lower bounds for some reliability problems.

The reduction algorithms presented in this chapter run in time linear in the size of the graph but exponential in k . Naive implementations of these algorithms make them practical for small values of k only ($k \leq 4$). However, careful implementations may result in more generally applicable algorithms. Table 5 presents the size of the state of each k -clique in the algorithms developed in this chapter. The expression $\left\{ \begin{smallmatrix} n \\ l \end{smallmatrix} \right\}$ stands for the number of ways to partition a set of n elements into l nonempty subsets (a Stirling number of the second kind).

Table 5: Size of $state(K)$ in Reduction Algorithms for Reliability Problems

Problem	Edge failures	Node and edge failures
Rel_A	$\sum_{i=1}^k \binom{k}{i}$	$\sum_{i=1}^{k+1} \binom{k+1}{i}$
Rel_l	$\sum_{i=1}^k \binom{k}{i} \sum_{j=0}^l \binom{i}{j}$	$\sum_{i=1}^{k+1} \binom{k+1}{i} \sum_{j=0}^l \binom{i-1}{j}$
Res_v	$\sum_{i=1}^k \binom{k}{i} (i+1)$	$\sum_{i=1}^{k+1} \binom{k+1}{i} i$
Res	$1 + \sum_{i=1}^k \binom{k}{i} (i+1 + i(i-1)/2)$	$1 + \sum_{i=1}^{k+1} \binom{k+1}{i} (i + (i-1)(i-2)/2)$
Res_l	$1 + \sum_{i=1}^k \binom{k}{i} \sum_{j=0}^l \binom{i}{j} (i+1)$	$1 + \sum_{i=1}^{k+1} \binom{k-1}{i} \sum_{j=0}^l \binom{i-1}{j} i$

CHAPTER IV
EFFICIENT EXACT ALGORITHMS FOR DIRECTED PARTIAL K -TREE
NETWORKS

Introduction

In the previous chapter we presented efficient exact algorithms for several reliability problems restricted to undirected partial k -tree networks (given with an embedding in a k -tree). In this chapter, we present polynomial time solutions to the directed counterparts of these problems. The study of efficient algorithms for directed reliability problems is motivated by the fact that there is no known technique to reduce a general directed problem to its undirected counterpart.¹ Thus the algorithms given in Chapter III do not seem to be applicable for directed reliability problems. In fact, even if a linear time reduction existed, the tree width of the undirected graph may become impractically large.

We must also point out that the results in this chapter do not supersede those in Chapter III. On the one hand, although it is well known that Rel_i can be reduced to its directed counterpart via an $\mathcal{O}(m)$ time transformation, such a reduction increases the tree width of the undirected graph from k to $2k + 1$ (if nodes fail, Lemma 4.9). As all of these reduction algorithms run in time exponential in k , a small increase in the value of k may turn an algorithm impractical. On the other hand, there is no known technique to reduce the undirected versions of resilience to its directed counterpart (if nodes fail). Besides, the reduction algorithms for the directed versions of Res_v and Res do not run in linear time but in $\mathcal{O}(n^2)$ and $\mathcal{O}(n^3)$ time, respectively.

This chapter is organized as follows. In the next section we define some basic ter-

¹Directed and undirected versions of the problems solved in Chapter IV are #P-complete in general. Therefore, there are polynomial time reductions between them. However, we are only interested in linear time reductions between these problems.

minology and redefine some concepts that we previously presented in terms of undirected graphs. We then discuss how to modify the algorithms given in the previous chapters to deal with directed networks. We initially employ a simplified network model in which nodes are fail-safe. In a closing subsection, we discuss how to cope with node failures.

Terminology

Unless otherwise stated, we follow the graph theoretic terminology in [37]. We model directed networks with probabilistic digraphs $G = (V, E)$, where V is a set of nodes and $E \subseteq V \times V$ is the set of arcs. Nodes represent communication sites and arcs (u, v) in E represent a communication line from node u (the origin) to node v (the destination). A node v is reachable from node u if there is a (directed) path from u to v , i.e., $u = v$ or v is reachable from a node w such that (u, w) is an arc in E . Each arc e has an associated probability of operation p_e such that $0 \leq p_e \leq 1$. Table 6 summarizes our terminology for directed reliability problems. Let G be a directed network and s, t , be nodes in $V(G)$. The reachability of G with respect to s , $Conn_A(G, s)$, is the probability that all nodes in $V(G)$ can be reached from s , i.e., it is the probability that a broadcast operation from node s is successful. The s, t -connectedness of G , $Conn_2(G, s, t)$, is the probability that there is a (directed) path from s to t . Given a subset L of l nodes in $V(G)$, the s, L -connectedness of G , $Conn_l(G, s, L)$, is the probability that L is reachable from s , i.e., the probability that there is a path from node s to each element in L . The directed broadcast resilience of G with respect to a node s , $DRes_s(G)$, is the expected number of nodes that can be reached from s (including s). The l -directed broadcast resilience of G with respect to a set of l nodes L , $DRes_l(G, L)$, is the expected number of nodes that can be reached from nodes in L (including nodes in L). The directed resilience of G , $DRes(G)$, is the expected number of pairs of nodes that can communicate.

The following definitions are natural extensions of concepts defined previously in terms of undirected graphs. Let $G = (V, E)$ be a digraph, a digraph $G' = (V', E')$ is a subgraph of G if the set of nodes V' is a subset of V and the set of arcs E' is a subset of E ; G' is a partial graph of G if it is a subgraph of G and $V' = V$. As nodes are fail-safe

Table 6: Directed Versions of Some Reliability Measures and Their Associated Problems

Measure	Notation	Problem
Reachability	$Conn_A(G, s)$	$Conn_A$
s, t -connectedness	$Conn_2(G, s, t)$	$Conn_2$
s, L -connectedness	$Conn_l(G, s, L)$	$Conn_l$
Directed broadcast resilience	$DRes_s(G)$	$DRes_s$
l -Directed broadcast resilience	$DRes_l(G, L)$	$DRes_l$
Directed resilience	$DRes(G)$	$DRes$

in our directed network model, we use partial graphs to model the state of the network. Given two nodes u, v in $V(G)$, we say that u is connected to v via H , a partial graph of G , and denote it $u \overset{H}{\sim} v$, if v is reachable from u in H . Node u is connected to a set of nodes L via H , $u \overset{H}{\sim} L$ if u is connected to each element of L via H . We define the connectivity function $\tau(u, L, H)$ to be 1 if $u \overset{H}{\sim} L$, otherwise it is 0. For instance, we can formulate $Conn_l(G, s, L)$ as

$$Conn_l(G, s, L) = \sum_{H \subseteq G} P_G[H] \tau(s, L, H),$$

where $H \subseteq G$ denotes that H is partial graph of G .

A digraph G is a directed k -tree if it is obtained from a k -tree G' by replacing each edge $\{u, v\}$ in $E(G')$ with arcs (u, v) and (v, u) . A directed partial k -tree is a partial graph of a directed k -tree. Let G be a digraph, the underlying multigraph of G is the multigraph G' obtained by replacing each arc (u, v) with an edge $\{u, v\}$. A digraph K is a k -clique if there is an arc connecting each pair of nodes in K . Other concepts such as removed branches on K , shell of K , k -leaf, etc., are defined as expected.

Efficient Exact Algorithms

All the problems in Table 6 are #P-complete for general networks. Agrawal and Satyanarayana [2] developed a linear time algorithm for $Conn_l$ restricted to partial 2-trees. We are not aware of the existence of any polynomial time algorithms for directed

partial k -trees when $k > 2$. Ball and Provan [13] proved that $Conn_A$ can be computed in $\mathcal{O}(m)$ time for directed acyclic graphs. This is an interesting result because $Conn_I$ is #P-complete for directed acyclic graphs [23]. This is the only case in which $Conn_I$ and $Conn_A$ have been proved to differ in complexity [23].

The rest of this chapter presents algorithms for the reliability problems in Table 6 restricted to directed partial k -tree networks. We assume that partial k -trees are always given with an embedding in a k -tree and that k is a fixed positive integer. Furthermore, to simplify our presentation, we assume that nodes are fail-safe. Without loss of generality, we also assume that the input graph has been transformed into an embedding k -tree by adding arcs e with $p_e = 0$.

The reduction algorithms for undirected reliability problems keep track of certain statistical information about classes of subgraphs of the shell of each k -clique. For each k -clique K and each subpartition π of the nodes in $V(K)$, we maintain information about the subgraphs of the shell $B'(K)$ whose “contraction” with respect to K is π .² The fact that edges represent bidirectional communication lines allowed us to represent connectivity between nodes in $V(K)$ via subpartitions of the set of nodes $V(K)$. Directed graphs require a different approach. The general idea that we use is a natural extension of the ideas in the previous chapter. Instead of using subpartitions of the set $V(K)$ to partition the set of subgraphs of the shell $B'(K)$ into equivalence classes, we use digraphs on the set of nodes $V(K)$ to partition the set of partial graphs (nodes are assumed fail-safe) of the shell $B'(K)$. Most of the concepts and definitions in Chapter III have natural counterparts for the directed version of the reliability problems.

s, L -Connectedness

Let G be a directed k -tree network and let L be a set of l nodes in $V(G)$. We are interested in computing the probability that there is a path from a specific node s , the

²The definition of “contraction” varies from problem to problem.

source of G , to each node in L , i.e.,

$$Conn_l(G, s, L) = \sum_{H \subseteq G} P_G[H] r(s, L, H).$$

We need a few definitions before presenting the reduction algorithm for $Conn_l$.

Let v be a node of G with incoming arcs $(x_1, v), \dots, (x_i, v)$ and outgoing arcs $(v, y_1), \dots, (v, y_j)$, for some non negative integer values i and j . The substitution of node v in G , $subs(G, v)$, is the operation on G that removes node v from $V(G)$ and adds one edge from each node x_1, \dots, x_i to each node y_1, \dots, y_j . Let $G = (V, E)$ be a directed graph and W be a set of nodes such that $W \cap V \neq \emptyset$. The contraction of G with respect to W , $Cont(G, W)$, is the graph obtained by substituting all nodes in $V \setminus W$. Clearly, for any two nodes u, v in W , $u \overset{G}{\sim} v$ if and only if $u \overset{Cont(G, W)}{\sim} v$.

Let G be a directed graph, H be a subgraph of G , and L be a set of nodes (not necessarily in $V(G)$). Let $D = (V(H) \cup L', E')$ be a digraph such that $L' = L \cap V(G)$. We are interested in those partial graphs of G such that all the nodes in L' can be reached from $V(H)$. Let us call such partial graphs of G the set of favorable subgraphs of G with respect to H . We use $PG(G, H, L, D)$ to represent the set of favorable subgraphs of G such that $Cont(G, V(H) \cup L) = D$. As in the undirected case, it is easy to verify that the set of favorable subgraphs of G with respect to H is partitioned into equivalence classes, each of which is $PG(G, H, L, D)$, where the set of nodes of D is $V(H) \cup (L \cap V(G))$ and all nodes in $L \cap V(H)$ are reachable from $V(H)$. In particular, let us consider G , a partially reduced directed k -tree network and one of its k -cliques K . Let L be the set of distinguished nodes that we want to reach from the source s and let us assume that s is not in the shell $B'(K)$. Let $\gamma(K)$ be the set of directed graphs D with node set $V(K) \cup L'$, for each L' subset of nodes of L , and such that all nodes in L' are reachable from $V(K)$ in D . We partition the set of favorable subgraphs of the shell $B'(K)$ with respect to K into the equivalence classes $PG(B'(K), K, L, D_1), \dots, PG(B'(K), K, L, D_q)$, where D_1, \dots, D_q is an enumeration of the elements in $\gamma(K)$. Thus, the state of K consists of the values

$s(D_1, K), \dots, s(D_q, K)$,³ where for each digraph D in $\gamma(K)$,

$$s(D, K) = \sum_{H \in PG(B'(K), K, L, D)} P_{B'(K)}[H]. \quad (\text{IV.26})$$

The next four lemmata define the initialization, reduction, and termination steps of the reduction algorithm for $Conn_l$.

Lemma 4.1 Let G be a directed k -tree network and L be a subset of nodes of G . Then for all K and D such that K is a k -clique of G and D is a digraph in $\gamma(K)$

$$s(D, K) = \begin{cases} 1 & \text{if } D = (V(K), \emptyset) \\ 0 & \text{otherwise.} \end{cases}$$

Proof: Immediate by definition of $B'(K)$ (Equation II.2) and the definition of $s(D, K)$ (Equation IV.26). ■

The reduction algorithm for $Conn_l$ proceeds in the usual way. The only special consideration is that we reduce G to a root R that includes the source s . Let us consider the reduction step. Let v be a k -leaf of G such that the neighborhood of v in the underlying multigraph is K . As in the undirected case, we consider two steps. First we combine the states of K^1, \dots, K^{k+1} to obtain the state of the $(k+1)$ -clique K^+ . Then we update the state of K to reflect the fact that v becomes one of the nodes in the removed branches $B(K)$. Unlike the undirected case, the reduction step does not perform joins of partitions but a digraph union operation on digraphs. Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two digraphs, we define the digraph union of G_1 and G_2 as $G_1 \cup G_2 = (V_1 \cup V_2, E_1 \cup E_2)$ (notice that this operation does not introduce multiple arcs with the same origin and destination). To compute the state of K^+ we consider all possible ways of obtaining D^+ , a digraph in

³We have defined $\gamma(K)$ in a rather naive way. We could define it in other ways that use less space. For example, we could further constrain $\gamma(K)$ to include only transitive digraphs. We prefer our naive definition because it simplifies the description of the reduction algorithm.

$\gamma(K^+)$, as the digraph union of D_1, \dots, D_{k+1} , where D_i is a digraph in $\gamma(K^i)$. Let

$$T(D^+, K^+) = \{(D_1, \dots, D_{k+1}) \mid \forall i = 1, \dots, k+1, D_i \in \gamma(K^i) \text{ and } \bigcup_{i=1}^{k+1} D_i = D^+\}$$

Lemma 4.2 Let G be a directed k -tree network that has been partially reduced using some perfect elimination ordering and the general reduction paradigm. Let v be the next k -leaf to be removed. Let K be the neighborhood of v in the underlying multigraph and K^+ be the subgraph of G induced by $V(K) \cup \{v\}$. Then for all digraphs D^+ in $\gamma(K^+)$

$$s(D^+, K^+) = \sum_{\vec{D} \in T(D^+, K^+)} \prod_{i=1}^{k+1} s(\vec{D}_i, K^i)$$

Lemma 4.3 Let G be a directed k -tree network that has been partially reduced using some perfect elimination ordering and the general reduction paradigm. Let $L \subseteq V(G)$ and v be the next k -leaf to be removed. Let K be the neighborhood of v in the underlying multigraph and K^+ be the subgraph of G induced by $V(K) \cup \{v\}$. In addition, let S be the star digraph induced by the k arcs that connect v to each node in $V(K)$ and the k arcs that connect each node in $V(K)$ to v . Then, for each directed graph D in $\gamma(K)$,

$$s(D, K) = \sum_{\substack{(D^+, D_S) \\ \text{in } PP(D, K)}} s(D^+, K^+) P_S(D_S),$$

where $PP(D, K)$ is a set of pairs of partial graphs defined as follows:

$$PP(D, K) = \{(D^+, D_S) \mid D^+ \in \gamma(K^+), D_S \in \beta(S), \text{Cont}(D^+ \cup D_S, V(K) \cup L) = D\},$$

where $\beta(S)$ is the set of partial graphs of S .

The following lemma defines the termination step of the reduction algorithm for $Conn_t$.

Lemma 4.4 Let G be a directed k -tree network and $L \subseteq V(G)$. Let R be a root of G obtained by applying the reduction paradigm and lemmata 4.1, 4.2, and 4.3 to G , and

such that the source s is an element of $V(R)$. Then

$$Conn_l(G, s, L) = \sum_{(D_1, D_2) \in Q(R, s, L)} s(D_1, R) P_R[D_2],$$

where $Q(R, s, L) = \{(D_1, D_2) \mid D_1 \in \gamma(R), D_2 \in \beta(R), s^{D_1 \cup D_2} L\}$.

The reduction algorithm for $Conn_l$ runs in linear time because l is constant and because the initialization, reduction and termination steps can be performed in constant time.

Theorem 4.5 Let G be a directed partial k -tree network with fail-safe nodes and given with an embedding in a k -tree. Let s be a node in $V(G)$ and L be a subset of l nodes of $V(G)$. The s, L -connectedness of G , $Conn_l(G, s, L)$, can be computed in $\mathcal{O}(n)$ time.

If $l = n$ the size of the state of each k -clique K will not be constant but exponential in n . Therefore, we need to modify the algorithm for $Conn_l$ to solve $Conn_A$, the reachability problem.

Reachability

The reachability of G with respect to a source s is the probability that all nodes in $V(G)$ can be reached from s . In this section we show how to modify the linear time algorithm for $Conn_l$ to solve $Conn_A$.

Let K be a k -clique of G and D be a partial graph of K . We define $PG(G, K, D)$ to be the set of partial graphs of G such that $Cont(G, V(K)) = D$ and all nodes in $V(G)$ can be reached from $V(K)$. We use $\gamma(G)$ to denote the set of partial graphs of G . The index set of the state of K is $\gamma(K)$, the set of partial graphs of K . Thus, the state of each k -clique K consists of the values

$$s(D, K) = P_{B'(K)}[PG(B'(K), K, D)] = \sum_{H \in PG(B'(K), K, D)} P_{B'(K)}[H],$$

for each digraph D in $\gamma(K)$.

The following equations define the initialization, reduction, and termination steps of the reduction algorithm for $Conn_A$. The initialization step is the same as for $Conn_l$, i.e., for each digraph D in $\gamma(K)$,

$$s(D, K) = \begin{cases} 1 & \text{if } D = (V(K), \emptyset) \\ 0 & \text{otherwise.} \end{cases}$$

The first part of the reduction step also remains the same. For each digraph D^+ in $\gamma(K^+)$,

$$s(D^+, K^+) = \sum_{\vec{D} \in T(D^+, K^+)} \prod_{i=1}^{k+1} s(\vec{D}_i, K^i),$$

where

$$T(D^+, K^+) = \{(D_1, \dots, D_{k+1}) \mid \forall i = 1, \dots, k+1, D_i \in \gamma(K^i) \text{ and } \bigcup_{i=1}^{k+1} D_i = D^+\}.$$

The second part of the reduction step changes slightly. We update the state of K as follows: for each digraph D in $\gamma(K)$,

$$s(D, K) = \sum_{\substack{(D^+, D_S) \\ \text{in } PP(D, K)}} s(D^+, K^+) P_S(D_S),$$

where $PP(D, K)$ is a set of pairs of graphs (D^+, D_S) such that $D^+ \in \gamma(K^+)$, $D_S \in \gamma(S)$, $Cont(D^+ \cup D_S, V(K)) = D$, and v is reachable from $V(K)$ in $D^+ \cup D_S$.

The formulation of the termination step is identical to the formulation of the termination step for $Conn_l$.

$$Conn_A(G, s, L) = \sum_{(D_1, D_2) \in Q(R, s, L)} s(D_1, R) P_R[D_2],$$

where

$$Q(R, s, L) = \{(D_1, D_2) \mid D_1 \in \gamma(R), D_2 \in \gamma(R), s^{D_1 \cup D_2} L\}$$

Theorem 4.6 Let G be a directed partial k -tree network with fail-safe nodes and given with an embedding in a k -tree. Let s be a node in $V(G)$. The reachability of G with respect to s , $Conn_A(G, s)$, can be computed in $\mathcal{O}(n)$ time.

Although the algorithm for $Conn_l$ has to be modified to solve $Conn_A$, the changes are very straightforward. Similar modifications of the algorithm for $Conn_l$ lead to a linear time solution for $Conn_{n-l}$, where l is constant.

Directed Broadcast Resilience

Similarly to the undirected case, the directed broadcast resilience $DRes_s(G)$ of a digraph can be computed by calculating the s, t -connectedness n times:

$$\begin{aligned} DRes_s(G) &= \sum_{H \subseteq G} P_G[H] \sum_{t \in V(G)} r(s, \{t\}, H) \\ &= \sum_{t \in V(G)} Conn_2(G, s, t). \end{aligned} \tag{IV.27}$$

Theorem 4.7 Let G be a directed partial k -tree network with fail-safe nodes and given with an embedding in a k -tree. The broadcast resilience $DRes_s(G)$ G with respect to any node s in $V(G)$ can be computed in $\mathcal{O}(n^2)$ time.

Proof: Immediate by Equation IV.27 and the linear time reduction algorithm for $Conn_2$. ■

A linear time reduction algorithm for $DRes_s$ does not seem to follow obviously from the linear time reduction for Res_v . A naive approach would be to maintain two sets of values in the state of each k -clique K , namely, $s(D, K)$, the probability that the contraction of $B'(K)$ with respect to the set of nodes $V(K)$ is the digraph D , and $E(D, K, v)$ for each subgraph D of K and each node v in $V(K)$. The value $E(D, K, v)$ would be similar to $E(\pi, K, C)$ in the reduction algorithm for Res_v , i.e., when $s(D, K) \neq 0$, $E(D, K, v)/s(D, K)$ would be the expected number of nodes w in the removed branches $B(K)$ such that w is reachable from v in a partial graph of the shell $B'(K)$. This approach does not work because it does not guarantee that we count reachable nodes exactly once.

We conjecture, however, that a linear time reduction algorithm for $DRes$, exists. We leave this problem for future research.

l -Directed Broadcast Resilience

As for the undirected case, there is no known formula expressing l -directed broadcast resilience in terms of other reliability measures for which we have a reduction algorithm. We do not have a reduction algorithm for $DRes_l$. However, we believe that a linear time reduction algorithm for $DRes_v$, would indicate how to solve $DRes_l$.

Directed Resilience

The directed resilience of a digraph can be formulated as follows:

$$DRes(G) = \sum_{u \in V(G)} \sum_{v \in V(G)} Conn_2(G, u, v) \quad (IV.28)$$

$$= \sum_{u \in V(G)} DRes_u(G). \quad (IV.29)$$

Thus the linear time reduction algorithm for $Conn_2$ and Equation IV.28 give us the following theorem.

Theorem 4.8 Let G be a directed partial k -tree network with fail-safe nodes and given with an embedding in a k -tree. The resilience of G can be computed in $\mathcal{O}(n^3)$ time.

Notice that by Equation IV.29 a linear time algorithm for $DRes_u$, would give us a quadratic time algorithm for $DRes$.

Node Failures

s, L -Connectedness

So far we have assumed that the nodes of a network do not fail. This assumption is not always restrictive. For example, $Conn_A(G, s)$ with node failures is the product of $Conn_A(G, s)$ without node failures and $\prod_{v \in V(G)} p_v$. Furthermore, Ball has proved that $Conn_l$ with node failures can be reduced to $Conn_l$ without node failures [12]. Ball's

reduction proceeds as follows. Let $G = (V, E)$ be a directed network with edge and node failures. Let s be a node in V and L be a subset of l nodes of V , $l \geq 1$. We can transform G into a directed network $\varphi(G) = (V', E')$ such that nodes in V' are fail-safe and there is a node s' in V' and a subset L' of l nodes in V' such that

$$\text{Conn}_l(G, s, L) = \text{Conn}_l(\varphi(G), s', L').$$

The transformation φ consists of splitting each node v in V into two nodes v^i and v^o , adding an arc (v^i, v^o) with probability of operation p_v , and replacing each incoming arc (u, v) with an arc (u, v^i) and each outgoing arc (v, w) with an arc (v^o, w) . If v is the source s , we make $s' = v^i$; if v is an element of the set L , node v^o becomes one of the elements of L' . Clearly, this transformation can be implemented in $\mathcal{O}(m)$ time. Thus, if G is a directed partial k -tree network, the transformation φ can be implemented in $\mathcal{O}(n)$ time. It is also easy to verify that for any pair of nodes u, w in V ,

$$P_{\varphi(G)}\{u^i \sim w^o\} = P_G\{u \sim w\}, \quad (\text{IV.30})$$

$$P_{\varphi(G)}\{u^i \sim w^i\} = P_G\{u \sim w\}/p_w, \quad (\text{IV.31})$$

and

$$P_{\varphi(G)}\{u^o \sim w^i\} = P_G\{u \sim w\}/(p_u p_w), \quad (\text{IV.32})$$

where $p_u > 0$, and $p_w > 0$.

The following lemma states that if G has bounded tree-width, so does $\varphi(G)$.

Lemma 4.9 Let G be a directed partial k -tree network. The digraph $\varphi(G)$ is a directed partial $(2k + 1)$ -tree network.

Proof: We prove that if G has tree-width $\leq k$ then $\varphi(G)$ has tree-width $\leq 2k + 1$. Let $(\{X_i \mid i \in I\}, T = (I, F))$ be a tree decomposition of G such that $\max_{i \in I} |X_i| - 1 = k$. The following tree decomposition of $\varphi(G)$ has tree-width $2k + 1$: $(\{Y_i \mid i \in I\}, T = (I, F))$,

where

$$Y_i = \{v^i, v^o \mid v \in X_i\}.$$

Let us first prove that this is a tree decomposition of $\varphi(G)$. Clearly,

$$\bigcup_{i \in I} Y_i = V(\varphi(G)).$$

Consider an arc e in $E(\varphi(G))$. Arc e is either of the form (v^i, v^o) or (v^o, w^i) . In the former case, both v^i and v^o are elements of Y_l for each index l such that $v \in X_l$. In the latter case, by construction, arc (v, w) is an element of $E(G)$ and thus there exists an index i in I such that $v \in X_i$ and $w \in X_i$. By definition of Y_i , v^o and w^i are elements of Y_i .

Now, let $i, j, l \in I$ such that j lies in a path from i to l in T . We have to prove that $Y_i \cap Y_l \subseteq Y_j$. Let x be an element of $Y_i \cap Y_l$. If $x = v^i$ then, by definition of Y_i and Y_l , v is an element of $X_i \cap X_l$ and therefore, $v \in X_j$. But then, by definition of Y_j , v^i is an element of Y_j . If $x = v^o$ we proceed analogously. Finally, it is clear that the tree-width of this tree decomposition is $2k + 1$. ■

Therefore, we obtain the following theorem.

Theorem 4.10 Let G be a directed partial k -tree network with node and edge failures. Let s be a node in $V(G)$ and L be a subset of nodes $L \subseteq V(G)$. The s, L -connectedness of G can be computed in $O(n)$ time

Proof: Immediate by Theorem 4.5 and Lemma 4.9. ■

Alternatively, we can design an $O(n)$ time algorithm for $Conn_l$ with node and edge failures by modifying the algorithm for $Conn_l$ with fail-safe nodes. The main change consist of defining the index set $\gamma(K)$ of each k -clique K as the set of subgraphs of K (instead of the set of partial graphs of K).

Directed Broadcast Resilience

We do not know of any algorithm to transform a network with node and edge failures into a network without node failures that has the same directed broadcast resilience. The

transformation φ defined in the previous section does not preserve the directed broadcast resilience. However, we can establish the following relationships between $DRes_s(G)$ and $DRes_s(\varphi(G))$.

Lemma 4.11 Let $G = (V, E)$ be a directed network with node and edge failures such that for all v in V , $p_v > 0$. Then

$$DRes_s(G) = DRes_{s,i}(\varphi(G)) - \sum_{w^i \in V'} Conn_2(\varphi(G), s^i, w^i).$$

If all nodes in V fail with the same probability p ,

$$DRes_s(G) = DRes_{s,i}(\varphi(G))p/(1+p).$$

Proof: Let $G' = (V', E')$ be $\varphi(G)$. By definition of $DRes_s$,

$$\begin{aligned} DRes_{s,i}(G') &= \sum_{w \in V'} P_{G'}[s^i \sim w] \\ &= \sum_{w^o \in V'} P_{G'}[s^i \sim w^o] + \sum_{w^i \in V'} P_{G'}[s^i \sim w^i]. \end{aligned} \quad (IV.33)$$

By Equation IV.30, Equation IV.33 becomes

$$DRes_{s,i}(G') = DRes_s(G) + \sum_{w^i \in V'} Conn_2(G', s^i, w^i), \quad (IV.34)$$

which proves the first part of the lemma. To prove the second part of the lemma let us assume that nodes operate with probability p . By Equation IV.31, Equation IV.34 is

$$\begin{aligned} DRes_{s,i}(G') &= DRes_s(G) + \sum_{w \in V} Conn_2(G, s, w)/p \\ &= DRes_s(G) + DRes_s(G)/p. \end{aligned}$$

So

$$DRes_s(G) = DRes_{s,i}(G')p/(1+p).$$

Theorem 4.12 Let G be a directed partial k -tree network with node and edge failures. Let s be a node in $V(G)$. The directed broadcast resilience of G with respect to s can be computed in $O(n^2)$ time. ■

Proof: Compute $\varphi(G)$ in linear time and then compute $DRes_{s^i}(\varphi(G))$ and $Conn_2(\varphi(G), s^i, w^i)$ for all nodes $w^i \in V'$. The result follows by Theorem 4.5, Theorem 4.7, and Lemma 4.11. ■

Lemma 4.11 may have another important consequence. If our conjecture that $DRes_s$ without node failures can be solved in linear time is true, $DRes_s$ with node and edge failures can also be solved in linear time if nodes fail with the same probability.

Directed Resilience

The transformation φ can also be used to reduce the directed resilience problem with node and edge failures to a collection of reliability problems without node failures.

Lemma 4.13 Let $G = (V, E)$ be a directed network with node and edge failures such that for all v in V $p_v > 0$. Let $G' = (V', E')$ be $\varphi(G)$. Then

$$\begin{aligned}
 DRes(G') &= DRes(G) + \sum_{v \in V} p_v + \sum_{v^i \in V'} \sum_{\substack{w^o \in V' \\ w^o \neq v^i}} P_{G'}[v^i \sim w^o] / p_v + \\
 &\quad \sum_{v^i \in V'} \sum_{\substack{w^o \in V' \\ w^o \neq v^i}} P_{G'}[v^i \sim w^o] / p_w + \\
 &\quad \sum_{v^i \in V'} \sum_{\substack{w^o \in V' \\ w^o \neq v^i}} P_{G'}[v^i \sim w^o] / (p_v p_w) + \sum_{v^o \in V'} Conn_2(G', v^o, v^i).
 \end{aligned}$$

If all nodes in V fail with the same probability p ,

$$DRes(G) = \frac{p^2}{(p+1)^2} (DRes(G') - \sum_{v^o \in V'} Conn_2(G', v^o, v^i) - np).$$

Proof: Let $G' = (V', E')$ be $\varphi(G)$. By definition of $DRes$,

$$DRes(G') = \sum_{v \in V'} \sum_{\substack{w \in V' \\ w \neq v}} P_{G'}[v \sim w].$$

We can break down the sum above as

$$\begin{aligned} DRes(G') &= \sum_{v^i \in V'} \sum_{w^o \in V'} P_{G'}[v^i \sim w^o] + \\ &\quad \sum_{v^i \in V'} \sum_{\substack{w^i \in V' \\ w^i \neq v^i}} P_{G'}[v^i \sim w^i] + \\ &\quad \sum_{v^o \in V'} \sum_{\substack{w^o \in V' \\ w^o \neq v^o}} P_{G'}[v^o \sim w^o] + \\ &\quad \sum_{v^o \in V'} \sum_{\substack{w^i \in V' \\ w^o \neq v^i}} P_{G'}[v^o \sim w^i]. \end{aligned} \tag{IV.35}$$

The first term of the sum above is

$$\begin{aligned} \sum_{v^i \in V'} \sum_{w^o \in V'} P_{G'}[v^i \sim w^o] &= \sum_{v^i \in V'} \sum_{\substack{w^o \in V' \\ w^o \neq v^o}} P_{G'}[v^i \sim w^o] + \\ &= \sum_{v^i \in V'} P_{G'}[v^i \sim v^o]. \end{aligned} \tag{IV.36}$$

By Equation IV.30, $P_{G'}[v^i \sim w^o] = P_G[v \sim w]$. Besides, $P_{G'}[v^i \sim v^o] = p_v$. Thus, Equation IV.36 becomes

$$\begin{aligned} \sum_{v^i \in V'} \sum_{w^o \in V'} P_{G'}[v^i \sim w^o] &= \sum_{v \in V} \sum_{\substack{w \in V \\ w \neq v}} P_G[v \sim w] + \sum_{v \in V} p_v \\ &= DRes(G) + \sum_{v \in V} p_v. \end{aligned} \tag{IV.37}$$

Similarly, the second term of the right hand side of Equation IV.35 is

$$\sum_{v^i \in V'} \sum_{\substack{w^i \in V' \\ w^i \neq v^i}} P_{G'}[v^i \sim w^i] = \sum_{v^i \in V'} \sum_{\substack{w^o \in V' \\ w^o \neq v^o}} P_{G'}[v^i \sim w^o] / p_w$$

$$= \sum_{v \in V} \sum_{\substack{w \in V \\ w \neq v}} P_G[v \sim w]/p_w. \quad (\text{IV.38})$$

The third term of the right hand side of Equation IV.35 is

$$\begin{aligned} \sum_{v^o \in V'} \sum_{\substack{w^o \in V' \\ w^o \neq v^o}} P_{G'}[v^o \sim w^o] &= \sum_{v^i \in V'} \sum_{\substack{w^o \in V' \\ w^o \neq v^i}} P_{G'}[v^i \sim w^o]/p_{v^i} \\ &= \sum_{v \in V} \sum_{\substack{w \in V \\ w \neq v}} P_G[v \sim w]/p_v. \end{aligned} \quad (\text{IV.39})$$

The fourth term of the right hand side of Equation IV.35 is

$$\begin{aligned} \sum_{v^o \in V'} \sum_{w^i \in V'} P_{G'}[v^o \sim w^i] &= \sum_{v^o \in V'} \sum_{\substack{w^i \in V' \\ w^i \neq v^i}} P_{G'}[v^o \sim w^i] + \sum_{v^o \in V'} P_{G'}[v^o \sim v^i] \\ &= \sum_{v \in V} \sum_{\substack{w \in V \\ w \neq v}} \frac{P_G[v \sim w]}{p_v p_w} + \sum_{v^o \in V'} \text{Conn}_2(G', v^o, v^i). \end{aligned} \quad (\text{IV.40})$$

Thus, the first part of the lemma follows by Equation IV.35, IV.37, IV.38, IV.39, and IV.40. Now, if all nodes in V fail with the same probability p , Equation IV.37 becomes

$$\sum_{v^i \in V'} \sum_{w^o \in V'} P_{G'}[v^i \sim w^o] = D\text{Res}(G) + np, \quad (\text{IV.41})$$

Equation IV.38 becomes

$$\sum_{v^i \in V'} \sum_{\substack{w^i \in V' \\ w^i \neq v^i}} P_{G'}[v^i \sim w^i] = D\text{Res}(G)/p, \quad (\text{IV.42})$$

Equation IV.39 becomes

$$\sum_{v^o \in V'} \sum_{\substack{w^o \in V' \\ w^o \neq v^o}} P_{G'}[v^o \sim w^o] = D\text{Res}(G)/p, \quad (\text{IV.43})$$

and Equation IV.40 becomes

$$\sum_{v^o \in V'} \sum_{w^i \in V'} P_{G'}[v^o \sim w^i] = DRes(G)/p^2 + \sum_{v^o \in V'} Conn_2(G', v^o, v^i). \quad (IV.44)$$

Combining Equations IV.35, IV.41, IV.42, IV.43, and IV.44 we obtain

$$DRes(G) = \frac{p^2}{(p+1)^2} (DRes(G') - \sum_{v^o \in V'} Conn_2(G', v^o, v^i) - np).$$

■

Theorem 4.14 Let G be a directed partial k -tree network with node and edge failures. The directed resilience of G can be computed in $\mathcal{O}(n^3)$ time.

Proof: Compute $\varphi(G)$ (linear time) and apply Theorem 4.5, Theorem 4.7, and Lemma 4.13 to $\varphi(G)$. ■

We know that if $DRes$, without node failures can be solved in linear time, $DRes$ without node failures can be solved in quadratic time. Therefore, if our conjecture that $DRes$, without node failures can be solved in linear time is true, $DRes$ with node and edge failures can be solved in quadratic time when nodes fail with the same probability (using Lemma 4.13).

CHAPTER V

UNIFORMLY OPTIMAL 2-TREES WITH RESPECT TO REL_2

In the previous two chapters we presented efficient algorithms to solve a variety of network reliability problems. In this chapter we are interested in a design problem. We want to characterize the “best” and “worst” 2-trees with respect to Rel_2 . In order to address this problem, we simplify the network model by assuming that only edges fail and that they all fail with the same probability p ($0 < p < 1$). Given a reliability measure Rel we use $Rel(G, p)$ to denote the reliability of the network modeled by the probabilistic graph G , in which each edge fails with probability p . It is important to observe that the reliability of two arbitrary graphs on n nodes, G_1 and G_2 , may “cross”, i.e., $Rel(G_1, p) < Rel(G_2, p)$ for some values of p but $Rel(G_1, p) > Rel(G_2, p)$ for other values of p [3]. This means that topology alone may not be sufficient to compare two specific networks. However, it may still be possible that there are some networks that are uniformly optimal with respect to Rel . An network G on n nodes and m edges is uniformly optimal with respect to Rel if $Rel(G, p) \geq Rel(H)$ for all p , $0 < p < 1$, and all graphs H on n nodes and m edges.¹

Boesch conjectured that uniformly optimal networks exist for any pair (n, m) of nodes and edges. However, Myrvold [47] recently disproved the conjecture. Nevertheless, for restricted classes of networks there may still exist uniformly optimal networks for any pair (n, m) .

In this chapter we characterize the best and worst 2-trees, 2-caterpillars, and 2-paths with respect to Rel_2 . Specifically, we address the following question: Given a class C of

¹Unless ambiguity arises, we will simply say that G is a best, most reliable or optimal network. When the inequality holds in the other direction we will say that G is a worst or least reliable network.

graphs (where \mathcal{C} is the class of 2-trees, 2-caterpillars, or 2-paths) and an integer n , $n \geq 2$, what is the n node graph G in \mathcal{C} , with two distinguished nodes x , y , that maximizes (minimizes) $Rel_2(H, x, y, p)$ over all n node graphs in \mathcal{C} ?

Neufeld and Colbourn [50] found that the best 2-trees with respect to Rel_A are the 2-paths. This result is somewhat surprising as 2-paths with apparently very different characteristics such as the 2-line on n nodes (a 2-path of maximum diameter) and the 2-fan on n nodes (a 2-path of minimum diameter) are both optimal. Clark et al. [21] proved the non-existence of uniformly optimal 2-trees with respect to Res . They proved that for values of p close to 0 the most resilient 2-tree is the 2-book, but for values of p close to 1, any 2-tree that maximizes Res is a 2-path. In addition, the most resilient 2-path is the 2-fan.

Terminology

We assume that the probability of operation of each edge of any graph is always a fixed value p ($0 < p < 1$). Thus we use $Rel_2(G, x, y)$ instead of $Rel_2(G, x, y, p)$. Given a graph G and nodes x , y , z in $V(G)$, we define the following functions:

$$\begin{aligned} s(G, [xy]) &= Rel_2(G, x, \{y\}), \\ s(G, x[y]) &= 1 - Rel_2(G, x, \{y\}), \\ s(G, x[yz]) &= P_G[y \sim z \wedge x \not\sim y], \\ s(G, [xyz]) &= P_G[x \sim y \wedge y \sim z], \\ s(G, xy[z]) &= P_G[x \not\sim z \wedge y \not\sim z]. \end{aligned}$$

Each function denotes the probability that a spanning subgraph of G meets certain connectivity conditions (suggested by the arrangement of the parenthesis in the second argument). The following observations are trivial implications of the definitions above.

Observation 5.1 Let G be a graph and let x , y , z be nodes in $V(G)$. Then

$$(a) \quad s(G, x[y]) = s(G, x[yz]) + s(G, xz[y]).$$

$$(b) s(G, x[yz]) + s(G, y[xz]) + s(G, xy[z]) + s(G, [xyz]) = 1.$$

$$(c) \text{ If } n > 2 \text{ and } \{x, y\} \in E(G), \text{ then } s(G, [xy]) > p \text{ and } s(G, x[y]) < 1 - p.$$

An edge of a 2-tree is either a separator (the removal of its end points separates the graph into more than one connected component), a peripheral edge (one of its end points is a 2-leaf), or an exterior edge (if it is neither a separator nor peripheral). A head of a 2-line is a peripheral edge e such that adding a new node and making it adjacent to the end points of e results in a 2-line.

Uniformly Optimal 2-trees with Adjacent Distinguished Nodes

Intuitively, it seems that the distinguished nodes of an optimal 2-tree should be adjacent. We prove that this is in fact true in the next section. Meanwhile, we assume that distinguished nodes are always adjacent and proceed to identify the best and worst 2-trees on n nodes. We call the edge induced by the distinguished nodes the distinguished edge. We need to state some properties of 2-books first. Let $cb(n)$ denote the probability that the end points of the spine of a 2-book on n nodes are connected. Similarly, let $db(n)$ denote the probability that the end points of the spine of a 2-book on n nodes are disconnected. The following lemma gives closed forms for $cb(n)$ and $db(n)$.

Lemma 5.2 Let G be a 2-book on n nodes. Let p be the probability of operation of each edge in $E(G)$. Then $db(n) = (1 - p)(1 - p^2)^{n-2}$ and $cb(n) = 1 - (1 - p)(1 - p^2)^{n-2}$.

Proof: As $cb(n) + db(n) = 1$ we only need to prove one of the identities of the theorem. Let G be a 2-book on n nodes and let v be a 2-leaf of G . Let $\{x, y\}$ be the spine of G . The probability that x and y are disconnected is the probability that they are disconnected in $G - v$ times the probability that at least one of the two edges incident on v is failed, i.e., $db(n) = db(n - 1)(1 - p^2)$. Besides, $db(2) = (1 - p)$. This is a simple recurrence relation with solution $db(n) = (1 - p)(1 - p^2)^{n-2}$. ■

We also need closed forms for similar measures defined on 2-paths. It is easy to verify that for any pair of 2-paths P_1 and P_2 and any pair of peripheral edges $\{x_1, y_1\} \in E(P_1)$

and $\{x_2, y_2\} \in E(P_2)$, the probability that x_1 is connected to y_1 in P_1 is the same as the probability that x_2 is connected to y_2 in P_2 . Let $cp(n)$ denote the probability that the end points of a peripheral edge of a 2-path on n nodes are connected and let $dp(n)$ denote the probability that the end points of a peripheral edge of a 2-path on n nodes are disconnected.

Lemma 5.3 Let P be a 2-path on n nodes. Let p be the probability of operation of each edge in $E(P)$. Then

$$dp(n) = \frac{(1-p)^2 + (1-p)^{n-1}p^n}{1-p(p-1)}$$

and

$$cp(n) = \frac{p - (1-p)^{n-1}p^n}{1-p(p-1)}.$$

Proof: Let $\{x, y\}$ be a peripheral edge and x be a 2-leaf of P . If $n = 2$ the theorem is trivially true. Let $n > 2$ and let z be the other node adjacent to x . The graph $P - x$ is also a 2-path and the edge $\{y, z\}$ is peripheral. The probability that x and y are disconnected in P is the probability that the two edges incident on x fail plus the probability that $\{x, y\}$ fails, $\{x, z\}$ is operational and y is not connected to z in $P - x$. So $dp(n) = (1-p)^2 + dp(n-1)(1-p)p$. Induction on n and the observation that $dp(n) + cp(n) = 1$ completes the proof. ■

The following technical lemmata simplify the proofs of Theorem 5.6 and Theorem 5.7.

Lemma 5.4 Let G be a 2-book on n nodes, $n > 3$, and let $\{u, v\}$ be a peripheral edge of G . Then $s(G, [uv]) < cb(n)$.

Proof: Let v be a 2-leaf of G . The graph $G - v$ is a 2-book on $n - 1$ nodes. We can therefore express $s(G, [uv])$ and $cb(n)$ in terms of $cb(n - 1)$ by as follows:

$$s(G, [uv]) = cb(n-1)p(1-p) + p$$

$$cb(n) = cb(n-1)(1-p^2) + p^2$$

If e is exterior, G is a 2-tree like the one depicted in Figure 7 (b), where L is a 2-tree with l nodes, R is a 2-tree with r nodes, $l + r = n + 1$, $l \geq 3$, $r \geq 3$, and u, v, w induce a triangle in G . Hence $s(G, [uv]) = p + (1 - p)s(L, [uw])s(R, [vw])$. But by the inductive hypothesis both L and R are 2-books with spine $\{u, w\}$ and $\{v, w\}$, respectively. Thus

$$s(G, [uv]) = p + (1 - p)cb(l)cb(r).$$

However, Lemma 5.5 (a) states that $s(G, [uv]) < cb(n)$. Thus e must be a separator of G (see Figure 7 (c)). So

$$s(G, [uv]) = s(L - e, [uv]) + s(L - e, u[v])s(R, [uv]).$$

By the inductive hypothesis R must be a 2-book and e is its spine. But we also have that

$$s(G, [uv]) = s(R - e, [uv]) + s(R - e, u[v])s(L, [uv]).$$

So L is a 2-book with spine e and so is G . ■

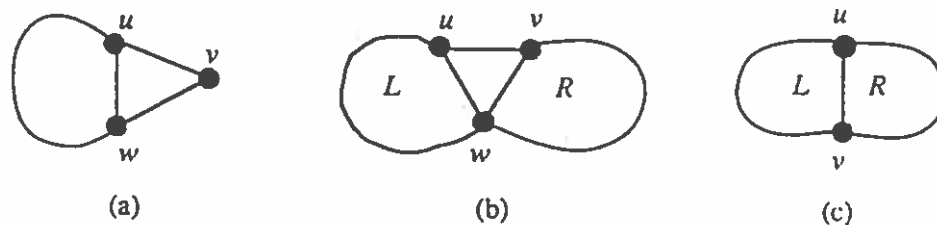


Figure 7: The distinguished edge is peripheral, exterior, or separator.

Notice that the proof of Theorem 5.6 establishes that any non-optimal 2-tree network can be incrementally improved by recursively turning subgraphs into 2-books. Now we characterize the worst 2-trees with respect to Rel_2 (assuming that distinguished nodes are adjacent).

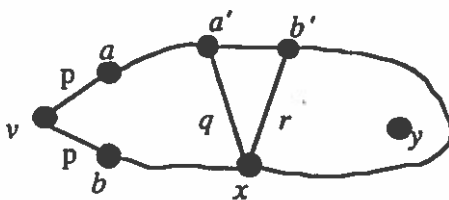


Figure 11: A 2-path.

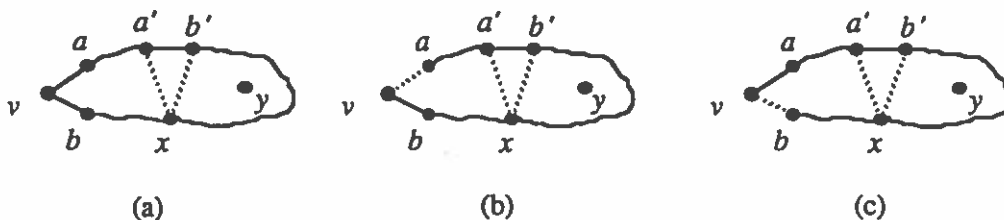


Figure 12: Spanning subgraphs such that $v \overset{H}{\sim} y$ and $v \not\overset{H}{\sim} x$.

The following lemma presents another scenario in which $s(G, x[y])$ can be increased by selecting a 2-leaf v instead of node x .

Lemma 5.17 Let G be a 2-tree network with more than 3 nodes and such that edges may fail with different probability ($0 < p_e < 1$). Let v be a 2-leaf of G and x, z be its neighbors. Then for all nodes $y, y \neq v$, if $p_{\{v,z\}} \leq p_{\{x,z\}}$ then $s(G, [vy]) < s(G, [xy])$.

Proof: As in the proof of Lemma 5.16, we define a one-to-one function ϕ from \mathcal{A} , the set of spanning subgraphs H of G such that $v \overset{H}{\sim} y$ and $v \not\overset{H}{\sim} x$, to \mathcal{A}' , the set of spanning subgraphs H' of G such that $x \overset{H'}{\sim} y$ and $v \not\overset{H'}{\sim} x$. Then we observe that $P_G[H] \leq P_G[\phi(H)]$ for all H in \mathcal{A} and that ϕ is not onto. Let x and z be the two nodes adjacent to node v in G . For each spanning subgraph H in \mathcal{A} , edge $\{v, z\}$ is operational and edge $\{x, z\}$ is failed. Define $\phi(H) = H - \{v, z\} \cup \{x, z\}$. Then ϕ is one-to-one and $P_G[H] = P_G[\phi(H)]$. Notice that edge $\{x, z\}$ is operational in all the spanning subgraphs in $\phi(\mathcal{A})$. As $n > 3$ we know that there are two edge-disjoint paths connecting x to y in $G - v$. Thus there is a spanning

y are disconnected. The number of nodes in the 2-book with spine $\{a, b\}$ and in the 2-book with spine $\{a, c\}$ has to be sufficiently large and the probability of operation of each edge has to be sufficiently small. However, if we rotate G at edge $\{a, x\}$ we obtain $s(\text{rot}(G, \{a, x\}), z[y]) > s(G, x[y])$.

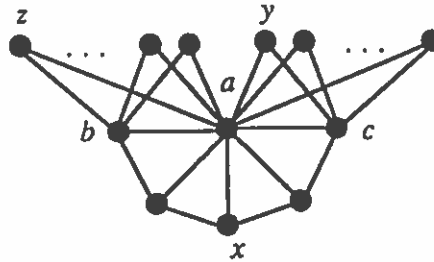


Figure 13: A 2-tree G such that $s(G, x[y]) > s(G, z[y])$.

We now know that the distinguished nodes of a worst 2-tree are 2-leaves. This is a necessary condition but it is not sufficient. The following two technical results allow us to identify the worst 2-trees.

Lemma 5.21 Let G be a 2-tree network, $\{a, b\}$ be an edge in $E(G)$, and y be a node in $V(G)$ such that

$$s(G, ab[y]) = \max_{\substack{H \text{ on } n \text{ nodes} \\ \{c, d\} \in E(H) \\ w \in V(H)}} s(H, cd[w]).$$

Then $\{a, b\}$ is a peripheral edge of G and y is a 2-leaf of G .

Proof: We first prove, by contradiction, that $e = \{a, b\}$ is a peripheral edge. Assume that e is exterior. Figure 14 (a) depicts the general structure of G . Notice that

$$s(G, ab[y]) = s(R, cb[y]) + s(R, b[cy])s(L, a[c]).$$

As e is not peripheral, the number of nodes in L , is greater than 2. If we rotate G with respect to e we obtain (see Figure 14 (b))

$$s(\text{rot}(G, e), uv[y]) = s(R, vw[y]) + s(R, v[wy])(1 - p).$$

But as the number of nodes in L is greater than 2, $s(L, a[c]) < 1 - p$. Therefore, $s(G, ab[y]) < s(\text{rot}(G, e), uv[y])$ and $\{a, b\}$ is not peripheral.

Let us assume that $e = \{a, b\}$ is a separator that decomposes G into two subgraphs L and R such that $|V(L)| = l > 2$, $|V(R)| = l > 2$, and $y \in V(R)$. Let x be a 2-leaf of G that lies in L and such that its two neighbors are u and v . Without loss of generality, let $u \neq b$. Then

$$s(G, xu[y]) \geq s(R, ab[y]) + s(R, a[by])(s(L - e, xu[b])).$$

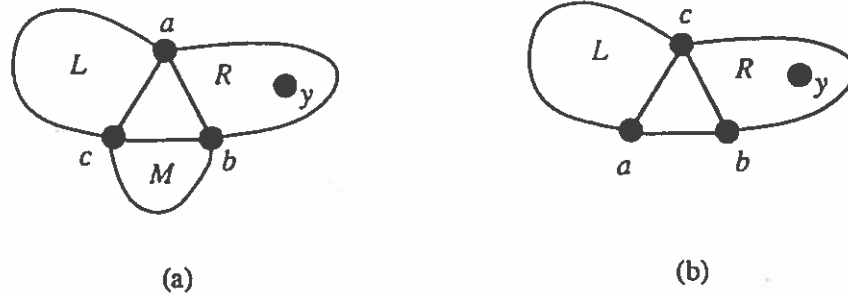
But the first term on the right hand side of the inequality above is $s(G, ab[y])$ and the second term is positive. Thus e is not a separator; it must therefore be peripheral.

Now we prove that node y is a 2-leaf. Assume that y is not a 2-leaf and let $e = \{a, b\}$ be a peripheral edge, a be a 2-leaf, and a, b, c induce a triangle in G . If $y = c$ then $s(G, ab[y]) = (1 - p)^2\gamma$, for some value γ , $0 < \gamma < 1$ (because edges $\{b, y\}$ and $\{a, y\}$ must fail). But for any other 2-leaf v , $s(G, ab[v]) = (1 - p)^2 + \epsilon$, for some value ϵ , $0 < \epsilon < 1$ (because two failed edges suffice to disconnect v from both a and b). So let us suppose $y \neq c$. It is easy to verify that if we collapse nodes a and b into node b , increase the probability of operation of the edge $\{b, c\}$ to $p + (1 - p)p$, and call the resulting graph G' , we obtain

$$s(G, ab[y]) = s(G', b[y]).$$

Also, as $y \neq c$, y is not a 2-leaf of G' . Strictly speaking we cannot use Corollary 5.19 with the graph G' because one edge, $\{c, b\}$, has probability of operation different from the other edges. However, by Observation 5.15, we can find two 2-leaves v, w of G' such that the 2-path $P = \text{red}(G', v, w)$ includes nodes y and b , and $S(G', y[b]) = s(P, y[b])$. Then we use Lemma 5.18 to conclude the proof. ■

Lemma 5.22 Let G be a 2-tree network. Let $\{a, b\}$ be an edge in $E(G)$ and y be a node in $V(G)$ such that $f(G, \{a, b\}, y) = s(G, a[y]) + s(G, b[y])$ is a maximum value of $f(H, \{c, d\}, w) = s(H, c[w]) + s(H, d[w])$ over all 2-trees H on n nodes, all edges $\{c, d\}$ in $E(H)$, and all nodes w in $V(H)$. Then $\{a, b\}$ is peripheral and y is a 2-leaf.

Figure 15: Maximizing $f(G, \{a, b\}, y)$.**Theorem 5.23**

- (a) Let $s(G, x[y])$ be the maximum value of $s(H, u[v])$ over all 2-trees H on n nodes and all pairs of nodes u, v in $V(H)$. Then G is the 2-line and x, y are 2-leaves of G .
- (b) Let $s(G, ab[y])$ be the maximum value of $s(H, uv[z])$ over all 2-trees H on n nodes, all edges $\{u, v\}$ in $E(H)$, and all nodes w in $V(H)$. Then G is the 2-line, $\{a, b\}$ is a head of G , and y is a 2-leaf different from a and from b .
- (c) Let $s(G, a[y]) + s(G, b[y])$ be the maximum value of $s(H, u[w]) + s(H, v[w])$ over all 2-trees H on n nodes, all edges $\{u, v\}$ in $E(H)$, and all nodes w in $V(H)$. Then G is the 2-line, $\{a, b\}$ is a head of G , and y is a 2-leaf different from a and from b .

Proof: We prove the theorem by induction on the number of nodes of the 2-tree G . When $n = 3$ the theorem is trivially true. Let us assume that the theorem is true for 2-trees on n nodes, $n \geq 3$.

- (a) Let G be a 2-tree on n nodes such that $s(G, x[y])$ is maximum. By Corollary 5.20 both x and y are 2-leaves of G . We need to prove that G is a 2-line. Let $\{a, b\}$ be the attachment of node x in G . Subgraphs H in which $x \not\sim^H y$ meet one of the following four conditions: $x \not\sim^H a$ and $x \not\sim^H b$, $x \sim^H a$ and $x \not\sim^H b$, $x \not\sim^H a$ and $x \sim^H b$, or $x \sim^H a$ and $x \sim^H b$.

Let R be the subgraph $G - x$. Then

$$s(G, x[y]) = (1 - p)^2 + p(1 - p)s(R, b[y]) + p(1 - p)s(R, a[y]) + p^2s(R, ab[y]).$$

By the inductive hypothesis R is a 2-line, $\{a, b\}$ is a head of R , and y is a 2-leaf of R . Thus G is a 2-line and both x and y are 2-leaves of G .

- (b) Let us assume now that $s(G, ab[y])$ is maximum. By Lemma 5.21 we know that edge $\{a, b\}$ is peripheral and y is a 2-leaf, $y \neq a$, $y \neq b$. Without loss of generality, let us assume that a is a 2-leaf and let edge $\{b, c\}$ be its attachment in G . Let R be the subgraph $G - a$. Then

$$\begin{aligned} s(G, ab[y]) &= s(R, bc[y]) + s(R, b[cy])(1 - p) \\ &= (1 - p)(s(R, bc[y]) + s(R, b[cy])) + s(R, bc[y])p \\ &= (1 - p)s(R, b[y]) + s(R, bc[y])p. \end{aligned}$$

Thus, by the inductive hypothesis (part (a) and (b)) we are done.

- (c) Now let us assume that $s(G, a[y]) + s(G, b[y])$ is maximum. By Lemma 5.22 edge $\{a, b\}$ is peripheral and node y is a 2-leaf. Without loss of generality, let a be a 2-leaf and $\{b, c\}$ be its attachment in G . Let R be the subgraph $G - a$. Then

$$\begin{aligned} s(G, a[y]) + s(G, b[y]) &= (1 - p)^2 + p(1 - p)s(R, b[y]) + p(1 - p)s(R, c[y]) + \\ &\quad p^2s(R, bc[y]) + (1 - p^2)s(R, b[cy]) + s(R, bc[y]). \end{aligned}$$

Simple algebraic manipulation gives

$$\begin{aligned} s(G, a[y]) + s(G, b[y]) &= (1 - p)^2 + p(1 - p)(s(R, b[y]) + s(R, c[y])) + \\ &\quad (1 - p^2)s(R, b[cy]) + s(R, bc[y]) + p^2s(R, bc[y]) + \\ &\quad p^2s(R, bc[y]) \\ &= (1 - p)^2 + p(s(R, b[y]) + s(R, c[y])) + \\ &\quad (1 - p^2)s(R, b[y]) + 2p^2s(R, bc[y]). \end{aligned}$$

Thus, by the inductive hypothesis (part (a), (b), and (c)) R is a 2-line, b is a 2-leaf, y is a 2-leaf, and edge $\{b, c\}$ is a head of R . Therefore G is a 2-line, y is a 2-leaf in G , and $\{a, b\}$ is a head of G . ■

Uniformly Optimal 2-paths

The study of reliability aspects of 2-paths is important in understanding and solving reliability problems for 2-trees. In the previous section we used results about 2-paths to solve problems on 2-tree networks (e.g., Corollary 5.19). We are also interested in studying reliability aspects of 2-paths because they have been proved to be optimal with respect to other reliability measures. For example, any 2-path is a best 2-tree with respect to all-terminal reliability [50] and a 2-path is optimal with respect to Res when p is close to 1 [21].

In the previous section we identified the best and worst 2-trees with respect to Rel_2 . As the worst 2-tree is the 2-line in which the identified nodes are its two 2-leaves, that is also the worst 2-path. The best 2-tree is not a 2-path but a 2-book. The following theorem characterizes the best 2-path.

Theorem 5.24 A 2-path P is a most 2-terminal reliable 2-path if and only if the distinguished nodes x, y induce a separator in P and edge $\{x, y\}$ separates G into two subgraphs L and R such that $||V(L)| - |V(R)|| \leq 1$.

Proof: Let P be a most reliable 2-path on n nodes ($n \geq 4$) and let its distinguished nodes be x, y . By Lemma 5.10, $\{x, y\}$ is an edge in P . By Theorem 5.7, the distinguished edge is not peripheral. Thus $\{x, y\}$ is either a separator or exterior.

Let us assume that $\{x, y\}$ is exterior. Let x, y, z induce a triangle in P . As $\{x, y\}$ is not peripheral, edge $\{x, z\}$ is a peripheral edge of a 2-path L that has l nodes ($l > 2$). Similarly, $\{y, z\}$ is a peripheral edge of a 2-path R that has r nodes ($r > 2$). Let us express

$s(G, x[y])$ in terms of functions on L and R . It is easy to verify that

$$s(G, x[y]) = (1 - p)(s(L, x[z]) + s(L, [xz])s(R, y[z])).$$

Let us consider the separator $\{x, z\}$. Clearly,

$$s(G, x[z]) = s(L, x[z])((1 - p) + s(R, y[z])p).$$

By our assumption

$$(1 - p)s(L, [xz])s(R, y[z]) \leq s(L, x[z])s(R, y[z])p.$$

Using the closed forms in Lemma 5.3, this becomes

$$(1 - p)(1 - (1 - p)(1 - p^2)^{l-2}) \leq p(1 - p)(1 - p^2)^{r-2}.$$

Without loss of generality, let us assume that $r \geq l$. Then

$$1 \leq (1 - p^2)^{l-2}(p(1 - p^2)^{r-l} + (1 - p)).$$

But $l > 2$. So $(1 - p^2)^{l-2} < 1$. Furthermore, as $r \geq l$, we get $p(1 - p^2)^{r-l} + 1 - p \leq 1$. This leads to a contradiction. Therefore, $\{x, y\}$ is a separator. The theorem follows by Lemma 5.5 (c). ■

Uniformly Optimal 2-caterpillars

The class of 2-caterpillars properly contains the class of 2 paths and is properly contained in the class of 2-trees. The identification of best and worst 2-caterpillars with respect to Rel_2 is trivial as all the extremal 2-trees studied in the first section of this chapter are 2-caterpillars.

Theorem 5.25

- (a) If distinguished nodes must be adjacent, a 2-caterpillar G is best with respect to Rel_2 if and only if G is a 2-book and the distinguished nodes induce the spine of G .
- (b) If distinguished nodes must be adjacent, a 2-caterpillar G is worst with respect to Rel_2 if and only if G is a 2-path and the distinguished nodes induce a peripheral edge of G .
- (c) A 2-caterpillar G is best with respect to Rel_2 if and only if G is a 2-book and the distinguished nodes induce the spine of G .
- (d) A 2-caterpillar G is worst with respect to Rel_2 if and only if G is a 2-line and the distinguished nodes are its two 2-leaves.

Proof: Immediate by Theorem 5.6, Theorem 5.7, Theorem 5.11, and Theorem 5.23. ■

Alternative Optimality Criteria

In the preceding sections we characterized uniformly optimal graphs by identifying triples (G, x, y) such that

$$Rel_2(G, x, y) = \max_{H \in C_n} \max_{\substack{(u,v) \text{ in} \\ V(H) \times V(H)}} Rel_2(H, u, v),$$

where C_n is the set of 2-trees, 2-paths, or 2-caterpillars on n nodes and $0 < p < 1$. We found that 2-books are the best 2-trees with respect to this criterion. However, this result says nothing about the 2-terminal reliability of a 2-book with respect to other pairs of nodes. An average measure of the 2-terminal reliability of a graph seems a better indicator of the suitability of a network with respect to Rel_2 . However, Clark et al. [21] proved that there are no uniformly optimal 2-trees with respect to Res . In this section we define an optimality criterion reflecting the “weakest link” consideration of 2-terminal reliability, namely, we will identify the graph G such that

$$\min_{\substack{(x,y) \text{ in} \\ V(G) \times V(G)}} Rel_2(G, x, y) = \max_{H \in C_n} \min_{\substack{(u,v) \text{ in} \\ V(H) \times V(H)}} Rel_2(H, u, v), \quad (V.57)$$

where C_n is the set of 2-trees, 2-paths, or 2-caterpillars on n nodes and $0 < p < 1$.

The following three lemmata state technical results needed to identify the best 2-trees, i.e., the 2-trees that satisfy Equation V.57. We call a pair of nodes of a graph a worst pair if the probability that they are connected is not better than the probability that any other pair of nodes of the graph are connected. The next lemma identifies the worst pair of nodes of a 2-book.

Lemma 5.26 Let G be a 2-book on n nodes ($n > 3$). $Rel_2(G, x, y)$ is minimum over all pairs of nodes in $V(G) \times V(G)$ if and only if x and y are 2-leaves.

Proof: A pair of nodes in $V(G) \times V(G)$ either induces the spine of G , induces a peripheral edge, or consists of 2-leaves of G . Obviously, if $Rel_2(G, x, y)$ is minimum, x and y do not induce the spine of G . We therefore need to compare only $Rel_2(G, a, b)$ and $Rel_2(G, c, d)$, where a, b induce a peripheral edge of G and c, d are 2-leaves. Let b, f be the end points of the spine of G . Then

$$Rel_2(G, a, b) = p + p(1 - p)Rel_2(G - a, b, f).$$

Now, consider the 2-leaves c, d . Both c and d are adjacent to nodes b and f . If edge $\{c, b\}$ is operational then the probability that c or b are connected to d is less than one (there is at least one non-favorable subgraph, one in which only edge $\{c, b\}$ is operational). If edge $\{c, b\}$ is failed then edge $\{c, f\}$ must be operational and f must be connected to d in $G - c$ for c and d to be connected in G . Thus,

$$Rel_2(G, c, d) = p\gamma + p(1 - p)Rel_2(G - c, b, f),$$

where $\gamma < 1$. Simple algebraic manipulation of the two equations above suffice to prove that $Rel_2(G, a, b) > Rel_2(G, c, d)$. ■

Lemma 5.27 Let G be a 2-book on n nodes ($n > 3$) such that x, y are 2-leaves of G . Let H be a 2-tree on n nodes such that H is not a 2-book and u, v are 2-leaves of H that have the same attachment. Then $Rel_2(G, x, y) > Rel_2(H, u, v)$.

Proof: Consider the 2-book G . Let $\{a, b\}$ be the spine of G and B be the 2-book $G - \{x, y\}$. Then

$$Rel_2(G, x, y) = p^2 + p^2(1 - p^2) + 2p^2(1 - p)^2 Rel_2(B, a, b).$$

Now consider the 2-tree H . Let A be the subgraph $H - \{u, v\}$ and let $\{w, z\}$ be the attachment of u and v . Then

$$Rel_2(H, u, v) = p^2 + p^2(1 - p^2) + 2p^2(1 - p)^2 Rel_2(A, w, z).$$

As H is not a 2-book, A is not a 2-book or it is a 2-book but edge $\{w, z\}$ is not its spine. However, B is a 2-book and $\{a, b\}$ is its spine. Thus $Rel_2(B, a, b) > Rel_2(A, w, z)$ and therefore, $Rel_2(G, x, y) > Rel_2(H, u, v)$. ■

Lemma 5.28 Let G be a 2-tree on n nodes ($n > 3$). Let x and y be two different 2-leaves of G such that edge $\{u, v\}$ is the attachment of x and edge $\{w, z\}$ is the attachment of y . Let $Rel_2(G - \{x, y\}, u, v) \geq Rel_2(G - \{x, y\}, w, z)$. Let H be the 2-tree obtained by replacing edges $\{y, z\}, \{y, w\}$ with edges $\{y, u\}, \{y, v\}$. Then

$$Rel_2(H, x, y) \geq Rel_2(G, x, y)$$

Proof: Let A be the graph $G - \{x, y\}$. We can express the 2-terminal reliability of H with respect to x and y in terms of the 2-terminal reliability of A with respect to nodes u, v as follows:

$$Rel_2(H, x, y) = p^2 + p^2(1 - p^2) + 2p^2(1 - p)^2 Rel_2(A, u, v) \quad (\text{V.58})$$

The lemma is trivially true if edge $\{u, v\}$ is the same as edge $\{w, z\}$. Let us assume that they are different. We consider two cases. First let us assume that edges $\{u, v\}$ and $\{w, z\}$ share one node, say, $v = z$. Then the subgraphs of G in which x and y are connected correspond to one of the following cases: subgraphs in which both edge $\{x, v\}$

and edge $\{y, v\}$ are operational, subgraphs in which both edge $\{x, u\}$ and edge $\{y, w\}$ are operational but edge $\{x, v\}$ and edge $\{y, v\}$ are not both operational, subgraphs in which edge $\{x, v\}$ and edge $\{y, w\}$ are operational but both $\{x, u\}$ and $\{y, v\}$ are failed, and subgraphs in which edge $\{x, u\}$ and edge $\{y, v\}$ are operational but both $\{x, v\}$ and $\{y, w\}$ are failed. Thus

$$Rel_2(G, x, y) = p^2 + p^2(1 - p^2)\gamma + p^2(1 - p)^2 Rel_2(A, v, w) + p^2(1 - p)^2 Rel_2(A, u, v), \quad (V.59)$$

where $\gamma < 1$. As $Rel_2(A, u, v) \geq Rel_2(A, v, w)$, Equation V.58 and Equation V.59 give $Rel_2(H, x, y) > Rel_2(G, x, y)$.

If edges $\{u, v\}$ and $\{w, z\}$ do not share any node then we proceed as in the first case to express $Rel_2(G, x, y)$ as

$$Rel_2(G, x, y) = p^2\gamma_1 + p^2(1 - p^2)\gamma_2 + p^2(1 - p)^2 Rel_2(A, v, w) + p^2(1 - p)^2 Rel_2(A, u, v), \quad (V.60)$$

where $\gamma_1 < 1$ and $\gamma_2 < 1$. Combining Equation V.58 and Equation V.60 we obtain $Rel_2(H, x, y) > Rel_2(G, x, y)$. ■

Theorem 5.29 The 2-book is the best 2-tree on n nodes.

Proof: By Lemma 5.26, x and y are 2-leaves of G . Let H be a 2-tree on n nodes such that H is not a 2-book. Then $n > 3$ and H has at least two 2-leaves w, z . By Lemma 5.27 and Lemma 5.28, $Rel_2(G, x, y) > Rel_2(H, w, z)$, and, of course, the probability that w and z are connected in H is not less than the probability that a pair of worst nodes of H are connected. ■

Corollary 5.30 The 2-book is the best 2-caterpillar on n nodes.

We now characterize the best 2-paths, i.e. the 2-paths that satisfy Equation V.57. First we prove that the worst two nodes of a 2-path are its two 2-leaves. Then we prove that

we can improve the probability that two 2-leaves are connected in a 2-path by performing rotations until the 2-path becomes a 2-fan.

Lemma 5.31 Let G be a 2-path. Let x and z be two distinct 2-leaves of G and v be a node adjacent to x . Then

$$Rel_2(G, x, v) \geq \max(Rel_2(G, z, v), Rel_2(G, z, x))$$

Proof: We first prove that $Rel_2(G, x, v) \geq Rel_2(G, z, x)$. Let edge $\{v, w\}$ be the attachment of node x . As edge $\{v, w\}$ is a z - x separator, Lemma 5.9 gives

$$Rel_2(G, x, v) > Rel_2(G, z, v),$$

or

$$Rel_2(G, x, w) > Rel_2(G, z, v).$$

But by Lemma 5.3, $Rel_2(G, x, v) = Rel_2(G, x, w)$.

Now we prove that $Rel_2(G, x, v) \geq Rel_2(G, z, x)$ by induction on n , the number of nodes of G . When $n = 4$, $Rel_2(G, x, v) = Rel_2(G, z, v)$. Let $n > 4$ and let $\{a, b\}$ be the attachment of z in G and $\{v, w\}$ be the attachment of x in G . Without loss of generality, let us assume that a is a 2-leaf of $G - z$ and $v \neq a$. Then

$$Rel_2(G, z, v) = p\gamma + p(1-p)Rel_2(G - z, a, v), \quad (\text{V.61})$$

where $\gamma \leq 1$ ($\gamma = 1$ when $v = b$, otherwise $\gamma < 1$). Similarly,

$$Rel_2(G, x, v) = p + p(1-p)Rel_2(G - x, v, w). \quad (\text{V.62})$$

As edge $\{v, w\}$ is peripheral in $G - x$, and edge $\{x, v\}$ is peripheral in $G - z$,

$$Rel_2(G - x, v, w) = Rel_2(G - z, x, v). \quad (\text{V.63})$$

Combining Equation V.62 and Equation V.63,

$$Rel_2(G, x, v) = p + p(1 - p)Rel_2(G - z, x, v). \quad (V.64)$$

We know that a is a 2-leaf of $G - z$ and $\{x, v\}$ is a peripheral edge of $G - z$. By the inductive hypothesis

$$Rel_2(G - z, x, v) \geq Rel_2(G - z, a, v). \quad (V.65)$$

Combining Equations V.61, V.64, and V.65 we obtain $Rel_2(G, x, v) \geq Rel_2(G, z, v)$. ■

Lemma 5.32 The worst pair of nodes of a 2-path is a pair of 2-leaves.

Proof: Let G be a 2-path on n nodes ($n > 3$) and let x, y be a pair of worst nodes in $V(G)$ such that x is not a 2-leaf. Let $e = \{x, v\}$ be an edge that decomposes G into 2-paths L and R such that y is a node in $V(R)$ and z is a 2-leaf of G in $V(L)$, $z \neq v$, $|V(L)| \geq 3$, and $|V(R)| \geq 3$. We want to prove that $s(G, [xy]) > s(G, [zy])$. Clearly, this is true if and only if $s(G, z[xy]) > s(G, x[yz])$. But, by analysis of cases, it is easy to verify that

$$s(G, z[xy]) > s(L, z[xv])(s(R - e, x[vv]) + s(R - e, [xvv]) + s(R - e, v[xy])),$$

where $s(R - e, [xvv]) > 0$. Also,

$$s(G, x[yz]) = s(L, x[vz])s(R - e, x[vv])$$

Thus we only need to prove that $s(L, z[xv]) \geq s(L, x[vz])$. But this is true if and only if $s(L, [xv]) \geq s(L, [zv])$ (add $s(L, [xvz])$ to both sides of the inequality). Edge $\{x, v\}$ is peripheral in L and node z is a 2-leaf of L , $z \neq v$, $z \neq x$. By Lemma 5.31, $s(L, z[xv]) \geq s(L, x[vz])$. ■

We now prove that 2-fans are the best 2-paths. We follow the approach employed by Clark et al. in [21]. A general graph contains a 2-line on six nodes if there exists a subset of six nodes that induces a 2-line in G (see Figure 16). The following observation

is useful in the proof of Theorem 5.34.

Observation 5.33 Let G be the general graph depicted in Figure 16, where L' is the subgraph $G - V(R)$, R' is the subgraph $G - V(L)$. Let L'' be the subgraph L' with edge $\{c, d\}$ removed and R'' be the subgraph R' with edge $\{c, d\}$ removed.

(a) For all nodes x in $V(L)$, y in $V(R)$,

$$s(L'', d[xc]) > s(L'', c[xd]),$$

and

$$s(R'', c[yd]) > s(L'', d[yc]).$$

(b) A graph is a 2-fan if and only if it does not contain a 2-line on six nodes.

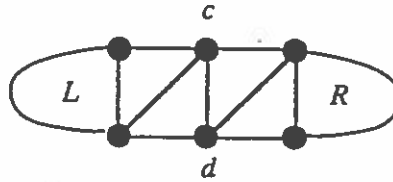


Figure 16: A graph that contains a 2-line.

Theorem 5.34 The 2-fan on n nodes is the best 2-path on n nodes.

Proof: Let G be a best 2-path on n nodes ($n > 3$). Thus G satisfies Equation V.57 and, by Lemma 5.32 the worst two nodes x, y are 2-leaves. If G is not a 2-fan, $n \geq 6$ and, by Observation 5.33, G contains a 2-line. Let G be the graph in Figure 16, where $L, R, L', L'', R',$ and R'' are defined as in Observation 5.33. Let x be a 2-leaf of G in $V(L)$ and y be a 2-leaf of G in $V(R)$. Let e be the edge $e = \{c, d\}$ and G^r be the rotation $G^r = \text{rot}(G, e)$. It suffices to prove that $0 < \text{Rel}_2(G^r, x, y) - \text{Rel}_2(G, x, y)$. Consider a partial graph H of

G^r in which x and y are connected. Nodes c and d are connected in H if and only if c and d are also connected in $rot(H, e)$, a partial graph of G . So we have to consider only partial graphs in which c is not connected to d . It is easy to verify that

$$\begin{aligned} Rel_2(G^r, x, y) - Rel_2(G, x, y) &= s(L'', d[xc])s(R'', c[yd]) + s(L'', c[xd])s(R'', d[yc]) - \\ &\quad s(L'', d[xc])s(R'', d[yc]) - s(L'', c[xd])s(R'', c[yd]) \\ &= (s(L'', d[xc]) - s(L'', c[xd]))(s(R'', c[yd]) - s(R'', d[yc])) \end{aligned}$$

By Observation 5.33, both factors above are positive. ■

Summary and Final Remarks

In this chapter we defined three optimality criteria and identified the extremal 2-trees, 2-paths, and 2-caterpillars according to those criteria. Given a class C_n of graphs, where C_n denotes the class of 2-trees, 2-paths, or 2-caterpillars on n nodes, we defined a best graph of the class C_n as a graph G such that $G \in C_n$ and for some pair of nodes x, y , and for all $p, 0 < p < 1$,

$$Rel_2(G, x, y) = \max_{H \in C_n} \max_{\substack{(u,v) \text{ in} \\ v(H) \times v(H)}} Rel_2(H, u, v). \quad (\text{V.66})$$

A graph $G \in C_n$ is a worst graph of the class C_n if for some pair of nodes x, y , and for all $p, 0 < p < 1$,

$$Rel_2(G, x, y) = \min_{H \in C_n} \min_{\substack{(u,v) \text{ in} \\ v(H) \times v(H)}} Rel_2(H, u, v). \quad (\text{V.67})$$

Finally, we redefined the notion of best graph. A graph $G \in C_n$ is a best graph of the class C_n if for its worst pair of nodes x, y , and for all $p, 0 < p < 1$,

$$Rel_2(G, x, y) = \max_{H \in C_n} \min_{\substack{(u,v) \text{ in} \\ v(H) \times v(H)}} Rel_2(H, u, v). \quad (\text{V.68})$$

A fourth alternative, used by Clark et al. [21], is to define a best graph in \mathcal{C}_n as a graph $G \in \mathcal{C}_n$ such that for all p , $0 < p < 1$,

$$Rel_2(G) = \max_{H \in \mathcal{C}_n} Res(H). \quad (V.69)$$

We can summarize the results given in this chapter using a table with graphical representations of the optimal graphs. Figure 17 illustrates our graphical representations. The graph in Figure 17(a) represents an arbitrary 2-path, the graph in Figure 17(b) represents a 2-fan, the graph in Figure 17(c) represents a 2-book, and the graph in Figure 17(d) represents a 2-line. Distinguished nodes will be depicted as black nodes.

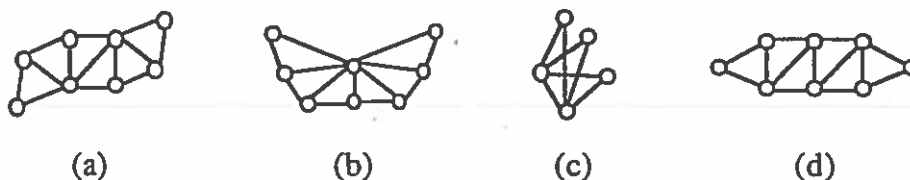










Figure 17: Graphical representation of some graphs.

Table 7 summarizes our results with respect to Equation V.66 and V.67. The first column of Table 7 presents the best and worst 2-paths and 2-trees with the restriction that the two distinguished nodes x and y be adjacent. The second column of Table 7 presents the uniformly optimal (and worst) 2-paths and 2-trees. The best and worst 2-trees are also 2-caterpillars. Thus Table 7 implicitly characterizes the best and worst 2-caterpillars as well.

The best 2-trees and the best 2-paths according to Equation V.68 are the 2-book (which is a 2-caterpillar) and the 2-fan, respectively. The worst pair of nodes is, in both cases, any pair of 2-leaves.

Clark et al. [21] proved that there are no uniformly optimal 2-trees under the “average” criterion (Equation V.69). When p is close to zero the best 2-tree is the 2-book and any 2-path is a worst 2-tree. However, when p is close to 1 the roles are inverted.

Table 7: Best and Worst 2-paths and 2-trees with Respect to Rel_2

		Adjacent x, y	Any pair x, y
2-paths	Best		
	Worst		
2-trees	Best		
	Worst		















For 2-paths, Clark et al. proved that the best 2-path is the 2-fan and the worst 2-path is the 2-line. Table 8 summarizes our results and the results in [21, 50]. In the first column we specify the criterion used to select the extremal graphs. The leftmost maximum or minimum of the criterion is taken over all 2-trees or 2-paths on n nodes. The remaining maximum or minimum (if any) is taken over all pairs of nodes in the graph.

With this information we can do some comparative analysis. Let us consider 2-paths first. If we use Equation V.66 alone, we could say that any 2-path on n nodes is equally reliable with respect to Rel_2 . But this is unrealistic as it is based only on the fact that the 2-terminal reliability of the best pair of nodes of any path (the nodes “in the middle”) is the same. It is more realistic to consider the 2-fan as the best 2-path with respect to Rel_2 . Not only is the 2-fan the best 2-path “on the average” (using Equation V.69), but also it is the best using Equation V.68 and as good as any other 2-path if we use Equation V.66. Similarly, we can claim that, overall, the 2-line is the worst 2-path with respect to Rel_2 . It is the worst 2-path if we use Equation V.67 and it is the worst 2-path if we use Equation V.69.

As there are no uniformly optimal 2-trees with respect to resilience, there is no overall best 2-tree with respect to Rel_2 . A 2-book is the best 2-tree with respect to

Equations V.66, V.68, and (if p is close to zero) V.69. When p is close to one, a 2-book still has the best pair of nodes over all 2-trees (the end points of the spine) and its worst pair of nodes (any pair of 2-leaves) is better than the worst pair of nodes of any other 2-tree. However, when p is close to one, the 2-book is the worst 2-tree on the average (using Equation V.69).

Table 8: Best and Worst 2-paths and 2-trees with Respect to Several Reliability Measures

Criterion	2-trees	2-paths
max max Rel_2		
max min Rel_2		
max Res	 	
min min Rel_2		
min Res	 	
max Rel_A		

In this chapter we also identified local graph operations that can be used to solve other reliability problems related to Rel_2 . For example, given a 2-path, we know which are the best and worst pair of nodes (a pair of adjacent nodes “in the middle” and the two 2-leaves, respectively). Furthermore, given a pair of distinguished nodes x, y in a 2-path P , there is a sequence of nodes that are end points of edges that separate x and y such that for any pair u, v in the sequence, $Rel_2(P, x, y) < Rel_2(P, u, v)$. We can therefore improve the 2-terminal reliability of P , incrementally, by selecting pairs of nodes in the

sequence, until we end up with two adjacent nodes. At this point we may still improve the 2-terminal reliability of P by selecting pairs of adjacent nodes that are located closer to the “middle” of P , until P is marked as in Figure 17(a). If we are not allowed to select other pairs of nodes but to perform rotations only, we can still improve $Rel_2(x, y)$, incrementally, by doing rotations in the 2-path P' induced by x, y and all nodes in minimal x - y separators, until P' becomes a 2-fan. There are also graph operations that can be used to improve the 2-terminal reliability of a 2-tree. We can, for instance, select a pair of adjacent nodes x, y and successively perform graph transformations that turn certain subgraphs into 2-books. The 2-terminal reliability, with respect to x, y , of each of the graphs so obtained, increases, until x, y become the end points of the spine of a book (cf. Theorem 5.6).

It is noteworthy that the results presented in this chapter also characterize the best and worst 2-trees, 2-paths and 2-caterpillars with respect to the following counting problem. Suppose that $f(G, s, t)$ denotes the number of partial graphs of G in which nodes s and t are connected. Then, as all 2-trees on n nodes have $2n - 3$ edges, the 2-terminal reliability of G with respect to s and t is $0.5^{2n-3} f(G, st)$, when $p = 0.5$. Thus f is maximal over all graphs on n nodes and all pairs of nodes when Rel_2 is maximal and $p = 0.5$.

CHAPTER VI

CONCLUSIONS AND FUTURE RESEARCH

In this research we have investigated reliability aspects of partial k -trees. We limited our research to two main areas of combinatorics of network reliability, namely, the design of efficient algorithms to compute reliability measures and the characterization of uniformly optimal networks with respect to a reliability measure.

In Chapter III we considered undirected partial k -tree networks with both node and edge failures. We employed the reduction algorithm by Arnborg and Proskurowski to solve, in linear time, important reliability problems that are $\#P$ -complete for general networks. We assumed that an embedding of the partial k -tree in a k -tree is known and that the value of k is fixed; otherwise, our algorithms run in $\mathcal{O}(n)$ time if $k \leq 3$, and in $\mathcal{O}(n^{k+2})$ if $k > 3$ ¹. We presented linear time algorithms for Rel_l , Res_l , and Res . Res is a particularly hard problem. To our knowledge, it had been solved in polynomial time only when the network is a partial 2-tree and either nodes or edges fail. Our reduction algorithm for Res is useful not only because it computes the resilience of a partial k -tree network efficiently, but also because it is the basis for the linear time algorithms for Res_v and Res_l and for polynomial time solutions for network broadcasting facility problems ($NBFL_l$). We believe that other reliability problems on partial k -tree networks can be solved following our approach for Res and Rel_l .

The availability of efficient algorithms for reliability problems restricted to partial k -tree networks may have important consequences in the design of efficient approximation algorithms for general networks. Edge-packing by partial 2-trees or partial 3-trees is an

¹Even though there exist $\mathcal{O}(n)$ embedding algorithms for higher values of k , they are not known explicitly. They can be obtained, for instance, once once the set of minimal forbidden minors for partial k -trees is known.

interesting research topic that may lead to better, efficiently computable, lower bounds for some reliability problems on general networks.

Chapter IV presents our algorithms for the directed counterparts of the problems solved in Chapter III. Directed network reliability problems seem more difficult than their directed counterparts. We first assumed that nodes do not fail and gave linear time algorithms for $Conn_l$ and $Conn_A$, a quadratic time algorithm for $DRes_s$, and a cubic time algorithm for $DRes$. Then we showed how to reduce those problems to reliability problems allowing both node and edge failures. Although the asymptotic time complexity of the algorithms remains the same, the value k grows to $2k + 1$. Thus it is worthwhile to investigate how to use the reduction paradigm directly on networks with both node and edge failures. Another open problem is to find a linear time algorithm for $DRes_s$. We believe that a more careful analysis will lead to a reduction algorithm that runs in linear time instead of quadratic time. This algorithm would improve the running time of our algorithm for $DRes$ by a factor of n and would probably suggest how to solve $DRes_l$.

Chapter V presents our results concerning uniformly optimal 2-trees with respect to Rel_2 . We defined three optimality criteria and characterized the extremal 2-trees, 2-paths, and 2-caterpillars according to each optimality criterion. Some technical results define local graph operations that improve the 2-terminal reliability of a network with respect to a pair of nodes. These operations can be used to find the best pair of nodes in a given graph, to find a better pair of nodes, and to compare networks. Our results and the results in [21, 50] constitute a useful set of criteria for the design and analysis of reliable 2-tree networks. A natural problem to consider is determining the best 2-trees with respect to Rel_l , for $l > 2$. This problem seems already difficult for $l = 3$. Nevertheless, we believe that our results provide some insight into its solution. For example, it is easy to prove that if the three distinguished nodes induce a triangle, the best 2-tree with respect to Rel_3 (using Equation V.66) consists of the triangle induced by the distinguished nodes and $n - 3$ 2-leaves adjacent to the distinguished nodes.

Another natural extensions to investigate is the characterization of uniformly optimal 2-trees with respect to some reliability measure assuming that not only edges but

also nodes fail. This problem is trivial when the reliability measure is Rel_A . The all-terminal reliability of a network with node and edge failures is the all-terminal reliability of the network without node failures times the probability that all nodes are operational. Therefore, any 2-path is also a uniformly optimal 2-trees with respect to Rel_A when nodes and edges fail. However, the uniformly optimal 2-trees and 2-paths with respect to Rel_l and Res may be different from the uniformly optimal 2-trees and 2-paths without node failures.

Generalizing results for 2-trees to results for k -trees seems very difficult. On the one hand, there is a combinatorial explosion of cases to consider if one uses the same techniques the we employed in this research. On the other hand, naive generalizations of the results for 2-trees to results for k -trees do not seem to work. For example, we know that any 2-path is a uniformly optimal 2-tree with respect to Rel_A . It is reasonable to suspect that any 3-path is also a uniformly optimal 3-tree with respect to Rel_A . However, not all 3-paths have the same all-terminal reliability. We believe that even though the study of the class of 2-trees provides some insight into properties of k -trees in general, it is often insufficient to obtain results for $k > 2$. This is true not only in the study of uniformly optimal graphs but also in the design of efficient algorithms for reliability problems. The class of 3-trees seems more attractive because it more explicitly presents the general properties of k -trees and is still analytically tractable.

In the introduction of this thesis we argued that partial k -trees have very good analytical power and very good modeling power. We believe that this research well illustrates both points. The analytical power of partial k -trees is illustrated by the efficient algorithms developed for reliability problems that are inherently difficult for general networks and by our comprehensive characterization of uniformly optimal 2-trees with respect to Rel_2 . The modeling power of partial k -trees has already been demonstrated by the multitude of families of graphs that are restricted cases of partial k -trees. In this research we further demonstrate this generality by exploring applications to the area of Combinatorics of Network Reliability.

BIBLIOGRAPHY

- [1] ABRAHAMSON, K., DADOUN, N., KIRKPATRICK, D. A., AND PRZYTYCKA, T. A simple parallel tree contraction algorithm. Tech. Rep. Technical Report 87-30, Dept. of Computer Science, The University of British Columbia, 1987.
- [2] AGRAWAL, A., AND SATYANARAYANA, A. Network reliability analysis using 2-connected digraph reductions of a class of directed networks. *Networks* 15 (1985), 239–256.
- [3] AMIN, A. T., SIEGRIST, K. T., AND SLATER, P. J. On the nonexistence of uniformly optimal graphs for pair-connected reliability. Submitted for publication.
- [4] AMIN, A. T., SIEGRIST, K. T., AND SLATER, P. J. Pair-connected reliability of communication networks with vertex failures. Submitted for publication.
- [5] AMIN, A. T., SIEGRIST, K. T., AND SLATER, P. J. On the expected number of pairs of connected vertices: pair connected reliability. In *Proceedings of the Second New Mexico Conf. on Applications of Graph Theory to Computer Networks* (1986).
- [6] ARNBORG, S., CORNEIL, D. G., AND PROSKUROWSKI, A. Complexity of finding embeddings in a k -tree. *SIAM J. Alg. Disc. Meth.* 8 (1987), 277–284.
- [7] ARNBORG, S., COURCELLE, B., PROSKUROWSKI, A., AND SEESE, D. An algebraic theory of graph reduction. Tech. Rep. LABRI 90-02, University of Bordeaux I, 1990.
- [8] ARNBORG, S., LAGERGREN, J., AND SEESE, D. Problems easy for decomposable graphs. In *Proceedings of ICALP 88, Springer-Verlag Lecture Notes in Computer Science* (1988), vol. 317, pp. 38–51.
- [9] ARNBORG, S., AND PROSKUROWSKI, A. Algorithms on graphs with bounded decomposability. *Congressum Numerantium* 53 (1986), 167–170.
- [10] ARNBORG, S., AND PROSKUROWSKI, A. Linear time algorithms for NP-Hard problems restricted to partial k -trees. *Discrete Appl. Math.* 23 (1988), 11–24.
- [11] ARNBORG, S., PROSKUROWSKI, A., AND CORNEIL, D. G. Forbidden minors characterization of partial 3-trees. *Discrete Math.* 80 (1990), 1–19.
- [12] BALL, M. O. Complexity of network reliability computations. *Networks* 10 (1980), 153–165.

- [13] BALL, M. O., AND PROVAN, J. S. Calculating bounds on reachability and connect-
edness in stochastic networks. *Networks* 13 (1983), 253-278.
- [14] BEINEKE, L. W., AND PIPPERT, R. E. Properties and characterizations of k -trees.
Mathematika 18 (1971), 141-151.
- [15] BODLAENDER, H. L. Classes of graphs with bounded tree-width. Tech. Rep. RUU-
CS-86-22, Dept. of Computer Science, University of Utrecht, 1986.
- [16] BODLAENDER, H. L. Dynamic programming algorithms on graphs with bounded
tree-width. To appear, 1987.
- [17] BODLAENDER, H. L. Improved self-reduction algorithms for graphs with bounded
treewidth. Tech. Rep. RUU-CS-88-29, Department of Computer Science, University
of Utrecht, 1988.
- [18] BODLAENDER, H. L. NC-algorithms for graphs with small treewidth. Tech. Rep.
RUU-CS-88-4, Department of Computer Science, University of Utrecht, 1988.
- [19] BOESCH, F. T. Synthesis of reliable networks — a survey. *IEEE Transactions on*
Reliability R-36, 3 (1986), 240-246.
- [20] CHANDRASEKHARAN, N., AND IYENGAR, S. S. NC algorithms for recognizing
chordal graphs and k -trees. Tech. Rep. 86-029, Dept. of Comp. Sci., Louisiana State
University, 1986.
- [21] CLARK, B., NEUFELD, E. M., AND COLBOURN, C. J. Maximizing the mean number
of communicating vertex pairs in series-parallel networks. *IEEE Transactions on*
Reliability R-35 (1986), 247-251.
- [22] COLBOURN, C. J. Personal communication.
- [23] COLBOURN, C. J. *The Combinatorics of Network Reliability*. Oxford University
Press, New York, 1987.
- [24] COLBOURN, C. J. Network resilience. *SIAM J. Alg. Disc. Meth.* 8 (1987), 404-409.
- [25] COURCELLE, B. The monadic second order logic of graphs iii: tree-width, forbidden
minors, and complexity issues. Tech. Rep. I-8852, Université de Bordeaux, 1988.
- [26] DIRAC, G. A. On rigid circuit graphs. *Abh. Math Sem. Univ. Hamburg* 25 (1961),
71-76.
- [27] DUFFIN, R. J. Topology of series-parallel networks. *J. Math. Anal. Appl.* 10 (1965),
303-318.
- [28] EL-MALLAH, E. S. *Decomposition and Embedding Problems for Restricted Networks*.
PhD thesis, University of Waterloo, Waterloo, Ontario, 1987.

- [29] FELLOWS, M. R. The Robertson-Seymour theorems: A survey of applications. preprint, 1988.
- [30] FELLOWS, M. R., KINNERSLEY, N. G., AND LANGSTON, M. A. Finite-basis theorems and a computational-integrated approach to obstruction set isolation. In *Proceedings of Computers and Mathematics Conference* (1989).
- [31] FRANK, H., AND FRISCH, I. *Communication, Transmission and Transportation Networks*. Addison-Wesley, 1971.
- [32] FULKERSON, D., AND GROSS, O. Incidence matrices and interval graphs. *Pa. J. Math.* 15 (1965), 835–855.
- [33] GAVRIL, F. Algorithms for minimum coloring, maximum clique, minimum coloring by cliques, and maximum independent set of a chordal graph. *SIAM J. Comp.* 1 (1972), 180–187.
- [34] GOLUMBIC, M. C. *Algorithmic Graph Theory and Perfect Graphs*. Academic Press, 1980.
- [35] HABEL, A. Graph-theoretic properties compatible with graph derivations. In *Proceedings of WG-88, Springer Verlag Lecture Notes in Computer Science 344* (1988).
- [36] HALIN, R. *Studies on minimally n -connected graphs*. Academic Press, New York, 1971, pp. 129–136.
- [37] HARARY, F. *Graph Theory*. Addison-Wesley, Reading, Mass., 1969.
- [38] JOHNSON, D. S. The many faces of polynomial time, in the NP-completeness column: An ongoing guide. *J. of Algorithms* 8 (1987), 285–303.
- [39] KLAWE, M. M., CORNEIL, D. G., AND PROSKUROWSKI, A. Isomorphism testing in hookup classes. *SIAM J. Alg. Disc. Meth.* 3, 2 (1982), 260–274.
- [40] KLEIN, P. N. Efficient parallel algorithms for chordal graphs. In *29th Annual Symposium on Foundations of Computer Science* (October 24-26 1988), pp. 150–161.
- [41] LAGERGREN, J. The nonexistence of reduction rules giving embedding into a k -tree. Submitted for publication, 1990.
- [42] MATA-MONTERO, E. Resilience of partial k -tree networks. *Congressus Numerantium* 74 (1989), 107–123.
- [43] MATA-MONTERO, E. Resilience of partial k -tree networks with edge and node failures. Tech. Rep. CIS-TR-89-15, Department of Computer and Information Science, University of Oregon, 1989.
- [44] MATOUSEK, J., AND THOMAS, R. Algorithms finding tree-decompositions of graphs. Submitted for publication, 1988.

- [45] MILLER, G. L., AND REIF, J. H. Parallel tree contraction and its applications. In *Proc. 26th Annual IEEE Symp. on Foundations of Comp. Sci.* (1985), pp. 478–489.
- [46] MONIEN, B., AND SUDBOROUGH, I. H. Bandwidth-constrained np-complete problems. In *Proceedings of 13th ACM STOC* (1981), pp. 207–217.
- [47] MYRVOLD, W. Uniformly-most reliable graphs do not always exist. Tech. Rep. DCS-120-IR, Department of Computer Science, University of Victoria, 1989.
- [48] NAOR, J., NAOR, M., AND SCHÄFFER, A. A. Fast parallel algorithms for chordal graphs. In *19th STOC* (1987), pp. 355–364.
- [49] NEL, L. *Network Reliability and Facility Location in Unreliable Networks*. PhD thesis, Computer Science Department, University of Waterloo, 1988.
- [50] NEUFELD, E. M., AND COLBOURN, C. J. The most reliable series-parallel networks. *Networks 15* (1985), 27–32.
- [51] POLITOF, T. *A Characterization and Efficient Reliability Computation of Δ -Y Reducible Networks*. PhD thesis, University of California, Berkeley, 1983.
- [52] PROSKUROWSKI, A. Separating graphs in k -trees: cables and caterpillars. *Discr. Math. 49* (1984), 275–285.
- [53] PROSKUROWSKI, A. Maximal graphs of path-width k or searching a partial k -caterpillar. Tech. Rep. CIS-TR-89-17, Department of Computer and Information Science, University of Oregon, 1989.
- [54] PROSKUROWSKI, A., AND SYSLO, M. M. Efficient computation in tree-like graphs. In press, 1990.
- [55] PROVAN, J. S. The complexity of reliability computations in planar and acyclical graphs. *SIAM Journal on Computing 15* (1986), 694–702.
- [56] ROBERTSON, N., AND SEYMOUR, P. D. Some new results on the well-quasi ordering of graphs. *Annals of Discrete Mathematics 23* (1987), 343–354.
- [57] ROSE, D. Triangulated graphs and the elimination process. *J. Math Anal. Appl.* (1970), 597–609.
- [58] ROSE, D. J. On simple characterizations of k -trees. *Discrete Math. 7* (1974), 317–322.
- [59] ROSE, D. J., TARJAN, R. E., AND LUEKER, G. S. Algorithmic aspects of vertex elimination on graphs. *SIAM J. Comp.* (1976), 266–283.
- [60] WALD, J. A., AND COLBOURN, C. J. Steiner trees, partial 2-trees, and minimal IFI networks. *Networks 13* (1983), 159–167.

- [61] WILF, H. S. THE EDITOR'S CORNER finite list of obstructions. *Journal of Algorithms* (March 1987), 267-271.
- [62] WIMER, T. V. *Linear Time Algorithms on k-terminal graphs*. PhD thesis, Clemson University, South Carolina, 1988.
- [63] WIMER, T. V., HEDETNIEMI, S. T., AND LASKAR, R. A methodology for constructing linear time algorithms. Tech. Rep. TR-85-SEP-11, Clemson University, 1985.