AD-A267 488

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

2. REPORT DATE
1993

3. REPORT TYPE AND DATES COVERED
THESIS/DISSERTATION

4. TITLE AND SUBTITLE
Full and Partial Multicommodity Cuts

5. FUNDING NUMBERS

DTIC ELECTE AUG 6 1993 S C D

6. AUTHOR(S)

Roger Chapman Burk

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

AFIT Student Attending: Univ of North Carolina

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/CI/CIA- 93-13D

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
DEPARTMENT OF THE AIR FORCE
AFIT/CI
2950 P STREET
WRIGHT-PATTERSON AFB OH 45433-7765

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for Public Release IAW 190-1
Distribution Unlimited
MICHAEL M. BRICKER, SMSgt, USAF
Chief Administration

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

93 8 05 139

93-18086

14. SUBJECT TERMS

15. NUMBER OF PAGES
153

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT

18. SECURITY CLASSIFICATION OF THIS PAGE

19. SECURITY CLASSIFICATION OF ABSTRACT

20. LIMITATION OF ABSTRACT

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)

FULL AND PARTIAL MULTICOMMODITY CUTS

by

Roger Chapman Burk

A dissertation submitted to the faculty of
The University of North Carolina at Chapel Hill
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in the Department of Operations Research

Chapel Hill, 1993

Approved by:

Advisor: J. Scott Provan

Reader: Thomas H. Brylawski

Reader: Mark E. Hartmann

Reader: David S. Rubin

Reader: Jon W. Tolle

FULL AND PARTIAL MULTICOMMODITY CUTS

Roger Chapman Burk

(under the direction of Dr. J. Scott Provan)

ABSTRACT

The problems of finding multicommodity cuts and partial multicommodity cuts in graphs are investigated. A multicommodity flow graph is a graph with k vertex pairs identified as the terminals (source and sink) for k commodities. A (full) multicommodity cut is a set of elements (edges or vertices) whose removal from such a graph cuts all source-to-sink paths for all commodities. A partial multicommodity cut is defined as a set of elements whose removal prevents more than a given number r of commodities from being connected by disjoint paths. For the full multicommodity cut problem, polynomial algorithms are found for any fixed k in a T-planar graph (one with all terminals on the boundary) and for $k = 3$ in a general planar graph. The T-planar problem is shown to be NP-complete for varying k unless the terminals are in non-crossing order; a polynomial algorithm is developed for that case. For partial multicommodity cuts, polynomial algorithms are developed for $r = k - 1$, $r = k - 2$, and $r = 1$ in T-planar non-crossing graphs. In the special case in which there is a common source for every commodity, the partial multicommodity cut problem is shown to be polynomial as long as r or $k - r$ is bounded, even in general graphs.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

D–E	a problem concerning arc-disjoint paths in a directed graph
D–V	a problem concerning node-disjoint paths in a directed graph
D–V2	a problem concerning node-disjoint paths in a directed graph having two commodities
D–VPB	a problem concerning node-disjoint paths in a planar directed graph having all terminals on the boundary
De–	a problem concerning removing arcs from a directed graph
DeE	a problem concerning removing arcs from a directed graph and leaving arc-disjoint paths
DeV	a problem concerning removing arcs from a directed graph and leaving node-disjoint paths
DJP	the disjoint paths problem (p. 21)
DvE	a problem concerning removing nodes from a directed graph and leaving arc-disjoint paths
DvV	a problem concerning removing nodes from a directed graph and leaving node-disjoint paths
FMCC	full multicommodity cut (problem or set of graph elements; p. 1)
GCC	general cut condition (p. 23)
IC	intersection criterion (p. 25)
IMCF	integer multicommodity flow problem (p. 23)
MCC	multicommodity cut (problem or set of graph elements; p. 1)
MCF	multicommodity flow problem (p. 16)
MTC	multiterminal cut (problem or set of graph elements; p. 18)

PMCC	partial multicommodity cut (problem or set of graph elements; p. 1)
U--P*BN	a problem concerning a planar undirected graph having all terminals on the boundary in non-crossing order
U--P*Eu*	a problem concerning an undirected graph that is planar and Eulerian when augmented with the demand edges
U--PBEu*	a problem concerning a planar undirected graph that has all terminals on the boundary and that is Eulerian when augmented with the demand edges
U-E	a problem concerning edge-disjoint paths in an undirected graph
U-E2	a problem concerning edge-disjoint paths in an undirected graph having two commodities
U-EEu*	a problem concerning edge-disjoint paths in an undirected graph that is Eulerian when augmented with the demand edges
U-EP	a problem concerning edge-disjoint paths in a planar undirected graph
U-EP*	a problem concerning edge-disjoint paths in an undirected graph that is planar when augmented with the demand edges
U-EP*BN	a problem concerning edge-disjoint paths in a planar undirected graph having all terminals on the boundary in non-crossing order
U-EP*Eu*	a problem concerning edge-disjoint paths in an undirected graph that is planar and Eulerian when augmented with the demand edges
U-EPB	a problem concerning edge-disjoint paths in a planar undirected graph having all terminals on the boundary
U-EPBEu*	a problem concerning edge-disjoint paths in a planar undirected graph that has all terminals on the boundary and that is Eulerian when augmented with the demand edges
U-EPEu*	a problem concerning edge-disjoint paths in a planar undirected graph that is Eulerian when augmented with the demand edges

U–V	a problem concerning vertex-disjoint paths in an undirected graph
U–V2	a problem concerning vertex-disjoint paths in an undirected graph having two commodities
U–VP	a problem concerning vertex-disjoint paths in a planar undirected graph
U–VPB	a problem concerning vertex-disjoint paths in a planar undirected graph having all terminals on the boundary
U–VP*k*	a problem concerning vertex-disjoint paths in a planar undirected graph having *k* commodities
Ue–P	a problem concerning edge-cuts in a planar undirected graph
Ue–P*BN	a problem concerning edge-cuts in a planar undirected graph having all terminals on the boundary in non-crossing order
Ue–P3	a problem concerning edge-cuts in a planar undirected graph having three commodities
Ue–PB	a problem concerning edge-cuts in a planar undirected graph having all terminals on the boundary
Ue–PB*k*	a problem concerning edge-cuts in a planar undirected graph that has all terminals on the boundary and has *k* commodities
Ue–P*k*	a problem concerning edge-cuts in a planar undirected graph having *k* commodities
UeE	a problem concerning removing edges from an undirected graph and leaving edge-disjoint paths
UeEP*BN	a problem concerning removing edges from a planar undirected graph that has all terminals on the boundary in non-crossing order and leaving edge-disjoint paths
UeV	a problem concerning removing edges from an undirected graph and leaving vertex-disjoint paths
Uv–PB	a problem concerning vertex-cuts in a planar undirected graph having all terminals on the boundary

UvE	a problem concerning removing vertices from an undirected graph and leaving edge-disjoint paths
UvV	a problem concerning removing vertices from an undirected graph and leaving vertex-disjoint paths
UvVPB	a problem concerning removing vertices from a planar undirected graph having all terminals on the boundary and leaving vertex-disjoint paths
VLSI	very large scale integrated (circuit)

1 INTRODUCTION

The multicommodity cut problem can be stated as follows:

(MCC) Given a graph $\mathcal{G}$, k pairs of points (s_i, t_i) that are vertices in $\mathcal{G}$, and an integer $r < k$, what is the minimum-cardinality set C such that in the graph $\mathcal{G} - C$ one can find no more than r disjoint paths, each of which joins the members of a distinct (s_i, t_i) pair?

If $r = 0$, a complete separation of every s_i from its corresponding t_i is required; this is a *full multicommodity cut* (FMCC) problem. (Some researchers have called this the *multicommodity disconnecting set* problem.) The $r > 0$ case is a *partial multicommodity cut* (PMCC) problem. The problem's name comes from the interpretation that there are k *commodities*, each with a *source* s_i and a *sink* t_i, and these commodities are constrained to flow along the edges of the graph. It is desired to know how the flow of all or some of the commodities may be cut. The purpose of this research was to investigate multicommodity cuts, to discover when such problems are tractable and when they are not, and where possible to develop efficient algorithms to solve them.

There are three ambiguities in the above MCC problem statement, each of which can be resolved in two ways. This results in eight basic cases for the problem. The three ambiguities are shown on the following page.

D/U The graph may be Directed or Undirected. (If directed, the s_i–t_i paths must be from s_i to t_i.)

v/e The set C may be a set of Vertices or a set of Edges. (In directed graphs it is more common to use the terms *nodes* and *arcs*; for the sake of generality the terms *vertex* and *edge* will be used when either type of graph is meant.)

V/E If $r \neq 0$, the paths may be required to be disjoint in their Vertices or only in their Edges.

It will be convenient to use a three-letter code as a shorthand way of referring to the eight basic cases of MCC, as well to corresponding cases of related problems that will be discussed later. The letters will be stated in the above order, with a hyphen substituted if the distinction is irrelevant for the problem under consideration (as for instance when $r = 0$ and the V/E distinction is empty). Thus, DvV means the problem in a Directed graph of removing Vertices so as to leave Vertex-disjoint paths, while U–E refers to a problem in an Undirected graph of finding Edge-disjoint paths, without reference to a set of elements being removed

An important variation on the MCC problem is the weighted problem. This is the situation in which a real-valued "weight" is assigned to each vertex or edge and it is required to find the minimum-weight set C, rather than the minimum-cardinality set. (If all weights are the same, the weighted problem reduces to the unweighted.) The aim of this research was to investigate the minimum-cardinality problem. However, it often turned out that the results were just as applicable to the weighted version. For this reason, many of the propositions and algorithms in this work are given for the more general weighted problem.

The Motivation for the Research

One of the motivations for this line of research was technical interest in the interplay between combinatorial methods and minimum-cut methods rooted in linear programming. Another was the hope that results from graph theory could be used to make efficient algorithms to solve practical problems. However, the original focus of interest was on an application in biochemistry, in theoretical modeling of metabolic systems. This is just one of many applications of multicommodity cuts. Besides the original problem in biochemistry, the most obvious applications are in military planning and in communication network analysis.

The biochemical application comes from Kohn and Lemieux [1991], who proposed a method of modeling the interacting biochemical reactions within a cell as a flow network in which every reaction is represented as a node. The crucial self-regulating properties of such a metabolic system are found in the feedback paths, *i.e.* the paths that connect the product of an enzyme-catalyzed reaction to a reactant of the same reaction. A key step in analyzing the regulatory properties of the network is finding the smallest possible set of reactions such that at least one reaction is on every feedback path. This can be formulated as an FMCC problem. The s_i–t_i pairs are the product-reactant pairs in each enzymatic reaction, and the set of reactions covering every feedback path is a cut-set.

Military applications were the motivation for some of the first research on MCCs [Bellmore, Greenberg, and Jarvis, 1970]. The supply network for an army in the field can be modeled appropriately as a multicommodity flow network. The multiple commodities can arise because different units take their support from different points, or because different types of supplies (*e.g.* ammunition, food, fuel) have different origins and destinations. The opposing force will want to attack this supply network in the most economical manner possible in order to stop the flow of all or some supplies. This can be formulated as an MCC problem.

The third important application of MCCs is in the reliability analysis of communications networks. In this case, the s_i–t_i pairs represent parties that need to be connected. The problem is to find the smallest set of communication nodes or links whose failure will prevent a separate line being assigned either to each pair or to more than a given number of pairs. This is perhaps the most obvious and important application for partial multicommodity cuts.

Outline of the Work

This work is in nine chapters. The next chapter (Chapter 2) gives formal definitions for the terms used, establishes the notation and standard symbols to be used. and where appropriate mentions restrictions on the types of problems considered. Chapter 3 surveys current research on multicommodity cuts and on three related problems: multicommodity flows, multiterminal cuts, and disjoint paths. Chapter 4 gives a few basic results on multicommodity cuts in order to set the stage for the following chapters.

Chapter 5 is the central chapter in this work. The subject is full multicommodity edge-cuts in planar graphs. After an initial section on the corresponding multiterminal cut problem, a polynomially bounded algorithm is developed for a fixed number of commodities when all the sources and sinks (collectively *terminals*) are on the boundary of the graph. If the terminals are not restricted to the boundary the problem is more difficult, but a polynomial algorithm is developed for the three-commodity case. If the number of commodities is not fixed then the planar multicommodity cut problem is shown to be NP-complete, even if the graph is a rectangular grid with the terminals on the boundary. A final section in Chapter 5 shows that the problem is polynomial if the terminals are on the boundary in what is called "non-crossing order" (see p. 13), and develops an algorithm for that case.

Chapter 6 extends the discussion of MCCs in such graphs to include partial multicommodity cuts. This chapter develops algorithms for the k-commodity PMCC problem in planar graphs with terminals on the boundary in non-crossing order when the parameter r (the number of disjoint s_i–t_i paths allowed to remain) is equal to $k-1$, $k-2$, and 1.

Chapter 7 extends some of the major results of chapters 5 and 6 to vertex-cuts. The general problem of multicommodity vertex-cuts in planar graphs with terminals on the boundary is shown to be NP-complete if the number of commodities k varies, but polynomial algorithms are presented for fixed k and for varying k with non-crossing terminals. The partial multicommodity vertex-cut problem is solved for $r = k - 1$.

Chapter 8 considers the special case in which all commodities have their sources (or equivalently, their sinks) at the same vertex. This restriction makes the problem easier in many ways. Even in general graphs full multicommodity cuts become easy, and the partial multicommodity cut problem is shown to be polynomially bounded as long as either r or $k - r$ is bounded. These results hold in directed graphs as well as in undirected, and they hold for both edge- and vertex-cuts. In addition, the cases DeV and DvE, in which nodes are removed from a graph to limit arc-disjoint paths or *vice versa*, are investigated. Finally, in the special case of an undirected planar graph with all sinks on the boundary, an algorithm is developed that is polynomially bounded for arbitrary k and r.

Chapter 9 concludes this work by summarizing the results and describing some likely directions for future research on multicommodity cuts.

2 NOTATION AND DEFINITIONS

This chapter lists definitions for terms used throughout this work. As far as possible these are the standard definitions used in the relevant field of research. It also gives some notation and symbology that will be constant for the rest of the work. The areas covered are set theory, graph theory, multiterminal and multicommodity flow graphs, and algorithmic complexity.

2.1 Set Theory

The empty set is designated ϕ. An *i-partition* of a set S is a collection of i subsets S_j of S such that $\bigcup_{j=1}^{i} S_j = S$ and $S_a \cap S_b = \phi$ for any $a \neq b$. A *partition* is any i-partition. The *cardinality* or *size* of a set is the number of its members; a set is *larger* or *smaller* than another according as its cardinality is larger or smaller. A subset S_j is said to be *minimal* with regard to some property if no proper subset of it has that same property. It is *minimum* with regard to a property if no smaller set at all has that property; thus a minimum set is also minimal but not *vice versa*. *Maximal* and *maximum* are defined similarly.

The members of a set may have positive real-valued numbers called *weights* associated with them. In this case the *weight of a set* is the sum of the weights of its members. In this work zero and negative weights will not be considered, since the interest is in minimum-weight sets and any zero- or negative-weight elements can automatically be included in the solution. In the context of an algorithm, the term *infinite* will be used loosely to describe a weight assigned to an element that is large enough to ensure the element's selection or nonselection, as appropriate.

A *multiset* is a collection of elements like a set, but unlike a set some of its elements may occur more than once.

2.2 Graph Theory

The notation and terminology for graphs is generally that of Wilson [1985]. A few modifications have been made in this work work to emphasize certain parallels between multicommodity edge-cuts and multicommodity vertex-cuts.

- Definitions referring to graphs in general
  - A *graph*, denoted $\mathcal{G}$, is a pair (V, E), where V is a nonempty finite set of elements called *vertices* and E is a finite multiset of elements called *edges*, each of which is a pair of elements of V. Sometimes E will loosely be called an edge set, even though it may have repeated elements. In this work loops (elements of E of the form (v, v)) will be disregarded as irrelevant to multicommodity cut problems.
  - The elements of E can be denoted (u, v) or uv.
  - A graph is *directed* or *undirected* according as the elements of E are ordered or unordered pairs. A directed graph is also called a *digraph*. The vertices and edges of a digraph are also called *nodes* and *arcs*. In this work graphs are undirected unless otherwise stated.
  - For $e = (u, v) \in E$, u and v are called the *endvertices* of e. The edge e and its endvertices are said to be *on* each other.
  - Two edges are *parallel* if they have the same endvertices.

- The *degree* of $v \in V$ is the cardinality of $\{e \in E : e = (v, u)$ or $e = (u, v)\}$ (disregarding loops). The *in-degree* of a node v in a digraph is the cardinality of $\{e \in E : e = (u, v)\}$ and its *out-degree* is the cardinality of $\{e \in E : e = (v, u)\}$.
- A graph is *Eulerian* if the degree of every node is even.
- A *subgraph* of $\mathcal{G}$ is a graph $\mathcal{G}' = (V', E')$ such that $V' \subseteq V$ and $E' \subseteq E$.
- For any $F \subseteq E$, the *subgraph induced by* F is the graph whose edge set is F and vertex set is all the vertices on members of F.
- For any $F \subseteq E$, $\mathcal{G} - F$ is the graph $(V, E - F)$.
- For any $U \subset V$, $\mathcal{G} - U$ is the graph whose vertex set is $V - U$ and whose edge set is all members of E that are not on a member of U.
- For any $x \in E$ or $x \in V$, $\quad \mathcal{G} - x \equiv \mathcal{G} - \{x\}$.

- Definitions concerning planar graphs
  - Every graph can be represented (not uniquely) by a set of points corresponding to its vertices and a set of lines or curves connecting the points and representing its edges. A graph is *planar* if it has a representation that is embedded in a plane and in which no two edge lines intersect except at the vertex points. In this work a planar graph will be considered together with a fixed representation in a plane; this combination is called a *plane graph.*
  - A *grid* is a plane graph whose representation is a subset of a rectangular lattice.
  - The edges of a planar graph $\mathcal{G}$ divide the plane into a number of regions, exactly one of them infinite in extent. These are the *faces* of $\mathcal{G}$.
  - The *boundary* of a planar graph is the boundary of its infinite face.
  - The *geometric-dual* $\mathcal{G}^D = (V^D, E^D)$ of a planar graph $\mathcal{G}$ has exactly one vertex for every face in $\mathcal{G}$ and exactly one edge for every edge. An edge in E^D connects two vertices in V^D if and only if the corresponding edge in E is in the common boundary of their corresponding faces in $\mathcal{G}$.

- Terms having to do with paths and connectivity in graphs
  - A *path* in a graph is a sequence of distinct edges of the form (v_0, v_1), (v_1, v_2), ..., $(v_{\ell-1}, v_\ell)$ such that $v_a \neq v_b$ if $a \neq b$, except that possibly $v_0 = v_\ell$. A path in a digraph can be *directed* in the obvious way.

- A *circuit* is a path in which $v_0 = v_\ell$.
- A *triangle* is a circuit of three edges.
- A path is *covered by* the edges in it and by the endvertices of those edges.
- The *union* of two graphs $\mathcal{G}_1 = (V_1, E_1)$ and $\mathcal{G}_2 = (V_2, E_2)$ is the graph $(V_1 \cup V_2, E_1 \cup E_2)$.
- A graph is *connected* if it cannot be represented as the union of two graphs with disjoint vertex sets; otherwise it is *disconnected.* All graphs considered in this work are connected unless stated otherwise.
- A *component* of a graph is a maximal connected subgraph. Thus a connected graph has exactly one component.
- A *disconnecting set* is any $X \subseteq E$ or $X \subset V$ such that $\mathcal{G} - X$ is disconnected or empty. In this work a disconnecting set will be an edge set unless otherwise stated.
- A graph is *i-edge-connected* if the size of its smallest disconnecting edge set is i or more. It is *i-vertex-connected* if the size of its smallest disconnecting vertex set is i or more.

• Definitions regarding edge- and vertex-cuts in graphs

- A minimal disconnecting set is a *cut-set* or a *simple cut.* It can also be called an *edge-cut* or *vertex-cut,* as appropriate. A cut-set will be an edge-cut in this work unless otherwise stated.
- An *i-way cut* or *i-cut* is a minimal set $X \subseteq E$ or $X \subset V$ such that $\mathcal{G} - X$ has exactly i components. Thus a two-way cut is a cut-set. When the context is clear, an i-way cut can also be called an edge-cut or vertex-cut, as appropriate.
- A *compound cut* is an i-way cut with $i > 2$.
- For any $U \subset V$, $\Gamma(U)$ denotes the set of edges with one endvertex in U and the other in $V - U$. Thus $\Gamma(U)$ is a simple cut or can be partitioned into simple cuts.
- Suppose in a digraph there is a directed path from node v to node u. Then a *v–u directed cut* is a minimal set of arcs or nodes X such that $\mathcal{G} - X$ contains no such path.
- An edge-cut of size 1 is also called a *bridge.*
- A vertex-cut of size 1 is also called an *articulation vertex.*

- Terminology concerning trees and forests
  - A *forest* is a graph that contains no circuits.
  - A *tree* is a connected forest.
  - For any $U \subseteq V$, the *Steiner tree on* U is a connected subgraph of $\mathcal{G}$ whose vertex set includes U and whose edge set has minimum cardinality or minimum weight.

A flow problem can be defined on a directed or undirected graph by designating one vertex as a source, one vertex as a sink, and assigning a capacity to each edge. Among the well-known consequences of the work of Ford and Fulkerson [1956] are efficient algorithms to find a maximum feasible flow in such a problem and a minimum capacity edge-cut between the source and the sink; the value of a maximum flow and the value of a minimum capacity cut are the same. By reformulating the problem the minimum weighted vertex-cut can also be found. Also, if all the capacities are integer-valued, then there exists a maximum-flow solution such that every flow along every edge (and consequently the total flow) is integer-valued.

2.3 Multiterminal and Multicommodity Flow Graphs

The multicommodity cut problem has already been defined (p. 1). This section gives the standard conventions and notation that will be used in discussing such problems and the graphs in which they occur. Some definitions will also be given for the closely-related class of multiterminal graphs.

A *multiterminal graph* is an undirected graph $\mathcal{G} = (V, E)$ together with a set of terminals $T \subseteq V$. The cardinality of T will always be represented by k, and $T =$

$\{t_1, t_2, \ldots, t_k\}$. An instance of a multiterminal cut problem is given by a multiterminal graph $\mathcal{G}$, which is assumed to include a designation of T.

A *multicommodity flow graph* is a directed or undirected graph $\mathcal{G} = (V, E)$ together with a set of terminals $T = \{s_1, t_1, s_2, t_2, \ldots, s_k, t_k\} \subseteq V$. The following terms will have constant meaning in this work when discussing multicommodity flow graphs.

- $\mathcal{G}$ The connected graph (V,E) along which the commodities flow; also called the *supply graph*; $|V| = n$ and $|E| = m$.
- $\mathcal{H}$ The graph with the same vertices as $\mathcal{G}$ and edges (s_i, t_i), $i = 1, \ldots, k$; also called the *demand graph*
- $\mathcal{G}^*$ $\mathcal{G}$ with the edges of $\mathcal{H}$ added; also called the *augmented supply graph*
- k The number of commodities
- T The set of vertices that are terminals; $2 \leq |T| \leq 2k$, since some of the terminals may be co-located
- r The allowed number of disjoint paths in a partial multicommodity cut problem
- C A minimal solution to a full multicommodity cut problem, sometimes called a *full cut* or *full multicommodity cut (FMCC)* when no ambiguity results
- J A minimal solution to a partial multicommodity cut problem, called an *interjacent set*; also called a *partial cut* or *partial multicommodity cut (PMCC)* when no ambiguity results (a minimal interjacent set need not be a cut-set or even a disconnecting set)
- $\mathcal{R}$ The remaining graph $\mathcal{G} - C$ or $\mathcal{G} - J$

The terminals s_j and t_j are *mates* of each other and are required to be distinct; they constitute a *terminal pair* or *pair of terminals*. A solution will be said to *separate* a commodity j if there is no s_j–t_j path in $\mathcal{R}$. Two components of $\mathcal{R}$ will be said to *share* a commodity j if one of them contains s_j and the other t_j. An instance of a multicommodity cut problem is given by a multicommodity flow graph $\mathcal{G}$ (which is assumed to include T) and r.

In vertex-cut problems in multiterminal and multicommodity flow graphs, the terminals themselves are always taken to be ineligible to be in the solution.

A multiterminal graph or multicommodity flow graph is *T-planar* if it is planar and has a representation with all terminals on the boundary. In a T-planar graph, two terminals are *adjacent* if there is a path from one to the other that runs around the boundary of $\mathcal{G}$ and contains no other terminals. The members of a sequence of terminals each of which is adjacent to the next is a set of *consecutive* terminals. In a T-planar multicommodity flow graph, two commodities i and j are *interleaved* or *in interleaved order* if their terminals occur in order s_i, s_j, t_i, t_j either clockwise or counterclockwise around the boundary of $\mathcal{G}$; otherwise they are in *non-crossing order*. A graph is *non-crossing* if all the commodities in it are. The order of terminals in a multiterminal graph is without significance.

A multicommodity cut problem is *unweighted* if the solution of smallest cardinality is sought. If the relevant elements have a weight defined on them and the minimum weight solution is sought, then the problem is *weighted.* Thus an unweighted problem is equivalent to a weighted one in which all weights are the same. All problems in this work are unweighted unless otherwise stated.

The descriptive codes introduced in Chapter 1 (DvV, U-E, *etc.*) will be used as a shorthand to designate particular cases of multicommodity cut and related problems. In addition, it will be convenient to add letter codes to designate particular graph configurations that are frequently studied:

P	The graph $\mathcal{G}$ is planar
P*	The graph $\mathcal{G}^*$ is planar
Eu	The graph $\mathcal{G}$ is Eulerian (all vertices of even degree)
Eu*	The graph $\mathcal{G}^*$ is Eulerian
B	In a planar graph, all terminals are on the boundary
N	In a planar graph with all terminals on the boundary, the terminals are in non-crossing order
2,3,...	The number of pairs of terminals (parameter k) is 2, 3, . . .

Thus, MCC UvVP*B2 would designate a multicommodity vertex-cut problem in an undirected graph in which the remaining paths are to be vertex-disjoint, $\mathcal{G}$ augmented by all edges (s_i, t_i) is planar, all terminals are on the boundary of one face of $\mathcal{G}$, and there are two commodities. A general T-planar multicommodity cut problem is Ue–PB and a non-crossing one is Ue–P*BN.

Unless otherwise stated, in this work a multicommodity cut problem will be a full multicommodity edge-cut problem in an undirected graph (Ue–).

2.4 Complexity and Intractability

The reader is assumed to be generally familiar with the theory of intractability as given (for instance) in Garey and Johnson [1979], from which the following definitions are taken. A function $f(x)$ is $O(g(x))$ if there is a constant c such that $|f(x)| \leq c\,|g(x)|$ for all sufficiently large non-negative values of x. An algorithm is said to be *polynomial time*, *polynomially bounded*, or simply *polynomial* if the number of operations it performs is $O(p(y))$, where y is an encoding of the algorithm's input and $p(y)$ is a polynomial in y. The algorithm and the work in it are also said to be $O(p(y))$. A problem is said to be polynomial or polynomially bounded if it can be solved with a polynomial algorithm. The term *NP-complete* describes a certain large and well-studied class of decision problems, all of which can be reduced to the others in polynomial time and for none of which is a polynomial algorithm known. (We refer the reader to Garey and Johnson for a specific definition of this class.) A problem is *NP-hard* if a polynomial algorithm for it would imply a polynomial solution to an NP-complete problem.

A problem that will arise frequently in the algorithms in this work is that of finding the shortest paths between all pairs of vertices in a planar graph. This problem is known to be $O(n^2)$, where n is the number of vertices in the graph [Frederickson. 1987]. We will use $O(\alpha)$ to mean the complexity of the planar all-pairs shortest paths problem.

3 SURVEY OF RELATED RESEARCH

Research that is relevant to the problem of full and partial multicommodity cuts covers four topics: multicommodity flows, multiterminal cuts, (full) multicommodity cuts themselves, and disjoint paths. A section of this chapter is devoted to each of these. There is no published research specifically on partial multicommodity cuts, but it is clearly related to the problem of finding disjoint s_i–t_i paths in a multicommodity flow graph. This is a large problem in its own right. Separate subsections in Section 3.4 will address the taxonomy of the many variations of the disjoint paths problem, its computational complexity, its generalization in the integer multicommodity flow problem, and known results for some special cases.

3.1 Multicommodity Flows

A multicommodity flow problem concerns finding feasible flows in a given multicommodity flow graph $\mathcal{G}$. In its decision-problem form it can be defined like this (all values are assumed to be real and non-negative):

(MCF) Given a multicommodity flow graph with maximum capacities defined on each edge, and with demands or total flows f_i required between each pair of terminals (s_i, t_i) for $1 \leq i \leq k$, are there flows that are simultaneously feasible and will satisfy every demand?

Other variations include the the construction problem (finding the feasible flows), the maximum sum-of-flows problem, and the minimum cost problem (in which a cost per unit flow for each edge is defined). MCF is a linear program in any of these forms, and so in a sense it is a well-solved problem. Assad [1978] and Kennington [1978] review and compare practical solution algorithms.

Other researchers have concentrated on what can be deduced about multicommodity flows by graph-theoretical methods. They have been particularly interested in finding when the maximum feasible flow is equal to the smallest-capacity cut between all s_i and all t_i vertices. This has long been known to be true for the case of one commodity [Ford and Fulkerson, 1956]. In 1963 Hu [22] proved it for two commodities in an undirected graph, and in 1980 Seymour [49] for up to six commodities if there are no more than four distinct terminals. Other results have been limited to special types of planar graphs. Okamura and Seymour [1981] showed that the maximum flow equals minimum cut if $\mathcal{G}$ is T-planar, Seymour [1981b] if $\mathcal{G}^*$ is planar, Okamura [1983] if $\mathcal{G}$ is planar and has two faces such that each terminal pair is on one of them, and Okamura [1983] again if $\mathcal{G}$ is planar and every terminal pair is either on a given face or shares one particular vertex on that face. Schrijver [1983] has summarized results of this sort (except the last two). These specialized results have led to efficient algorithms for finding feasible multicommodity flows on such graphs or for deciding if such a feasible flow exists. These have been offered by Okamura and Seymour [1981b], Hassin [1984], and Matsumoto, Nishizeki, and Saito [1985].

In these cases in which maximum flow equals minimum cut, the researchers also showed that if the arc capacities are integer-valued, then there is a maximum flow that is half-integer-valued for each commodity (*i.e.* the flows are integer multiples of $\frac{1}{2}$). This becomes significant when the problem of integer multicommodity flows is *considered* (p. 23).

3.2 Multiterminal Cuts

The multiterminal cut problem can be defined as follows:

(MTC) Given a multiterminal graph $\mathcal{G}$, what is the minimum cardinality set C such that in the graph $\mathcal{G} - C$ there is no path from one terminal to any other?

(Some researchers call this the *multiway* or *k-way cut* problem.) The solution C is also itself called a multiterminal cut (MTC). The problem can require an edge-cut or a vertex-cut, and it has a weighted as well as an unweighted version. Research for $k > 2$ has concentrated on the weighted edge-cut case.

A k-terminal cut problem is equivalent to a full multicommodity cut problem with $\frac{1}{2}k(k-1)$ commodities, their terminals being distributed among k vertices in such a way that every pair of vertices shares a commodity. In particular, a two-terminal cut problem is the same as a one-commodity cut problem, *i.e.* a conventional minimum-cut problem. The answer to this problem is readily available from the max-flow min-cut theorem [Ford and Fulkerson, 1956]. Thus the $k = 2$ MTC problem is essentially solved.

The general multiterminal cut problem was investigated by Dahlhaus *et al.* [1992]. They showed that the weighted problem is polynomially bounded if $\mathcal{G}$ is planar and k is fixed (case Ue–Pk), that the unweighted planar problem is NP-hard if k is part of the input (case Ue–P), and that the unweighted problem in general graphs is NP-hard for all fixed $k \geq 3$ (case Ue–k).

3.3 Multicommodity Cuts

Full multicommodity cuts have been investigated from an algorithmic and from a graph-theoretic point of view. However, the number of papers with significant results is small (three algorithmic and one theoretic) and they are fifteen to thirty years old. Also, there seems to have been little interplay between the algorithmic and theoretical approaches. This section will first cover the papers on FMCC algorithms (which were actually second in order of appearance), and then describe the theoretical results that have been published.

Algorithms for multicommodity cuts were first developed by Bellmore, Greenberg, and Jarvis in 1970 [4]. They consider the De– problem with capacitated arcs and offer two algorithms to solve it. The first is an implicit enumeration of all possible cuts. The second uses constraint generation to find a set-covering problem whose solution will yield a minimum MCC. The algorithm they give is as follows. Let a multicommodity flow graph $\mathcal{G} = (V, E)$ be given, and let x be an incidence vector on E. Let $\mathbf{A}$ be the path-arc incidence matrix for some set of s_i–t_i paths in $\mathcal{G}$, *i.e.* $a_{bc} = 1$ if arc c is on path b which goes from some s_i to some t_i. Let $\mathbf{B}^\ell$ be the set of $\ell \times 1$ binary matrices, and let $\mathbf{1}$ be a column vector of 1's.

Algorithm 1 (General Weighted De– FMCC [4])

input: $\mathcal{G}$ for a weighted De– FMCC problem

output: Optimal solution arc set

begin

(1) Find the shortest s_i–t_i paths in $\mathcal{G}$ for each i. Use the resulting k paths to construct $\mathbf{A}$.

(2) Solve the following set-covering problem to get $x^* \in \mathbf{B}^{|E|}$:

$$\begin{aligned} &\text{minimize } \sum_j x_j \\ &\text{subject to } \mathbf{A}x \geq \mathbf{1} \\ &\qquad\qquad x_j \in \{0,1\}\,,\ 1 \leq j \leq |E| \end{aligned}$$

(3) Let D be the edge set $\{j : x_j^* = 1\}$. For $i = 1$ to k, find the shortest s_i–t_i path in $\mathcal{G} - D$. If no such paths exist, then x^* is a feasible cover for all s_i–t_i paths as well as an optimal cover for a subset of them; therefore it is optimal for all of them. D is the solution to the MCC problem. Return D and stop.

(4) Otherwise, add a constraint row to $\mathbf{A}$ for every path found in Step *3*. Go to Step *2*.

end

Jarvis and Tindall [1972] refined Algorithm 1 by showing how constraints may be dropped from $\mathbf{A}$ when it grows too large; it need never exceed $|E|+1$ rows. Aneja and Vemuganti [1977] concentrated on improving the technique used to solve the covering problem in Step 2. After this paper, it seems that interest in the multicommodity cut problem died away.

These three papers on the FMCC problem shared a common approach. They all were concerned with the De– case (though, since their algorithms work for a capacitated version of the problem, it seems likely that their results will generalize). All three were interested in computational methods, all developed variations of the same algorithm, and all gave computational experience on test cases. However, none provided a complexity analysis of the algorithm. None made use of graph theory.

Two basic propositions on multicommodity cuts were published as lemmas in a paper by Hu on two-commodity flows in 1963 [22]. They are these:

Proposition 1 [Hu, 1963] *In a weighted two-commodity multicommodity cut problem, let C_1 be a minimum single-commodity cut separating $\{s_1, s_2\}$ from $\{t_1, t_2\}$, and let C_2 be such a cut separating $\{s_1, t_2\}$ from $\{t_1, s_2\}$. Then the smaller of C_1, C_2 is a minimum two-commodity cut.* ■

Proposition 2 [Hu, 1963] *If C is a minimal solution to a full multicommodity cut problem, then $\mathcal{R} = \mathcal{G} - C$ has at most $k+1$ components.* ■

The bulk of Hu's paper was on finding feasible flows. It does not seem that the graph-theoretic approach to multicommodity cuts was pursued beyond this beginning.

3.4 Disjoint Paths

The much-studied disjoint paths problem can be defined as the following decision problem:

(DJP) Given a multicommodity flow graph $\mathcal{G}$, are there k disjoint s_i–t_i paths in it, one for each commodity?

DJP can also be formulated as a construction problem (find the paths if they exist). It has edge- and vertex-disjoint cases, and these can be in directed or undirected graphs. It has received much attention both because of its theoretical interest and because of its important applications in the design of very large scale integrated (VLSI) circuits.

Frank [1990b] has recently published a survey of the state of knowledge for DJP and some related problems. DJP can also be formulated as a special case of finding integer multicommodity flows (IMCF). The following subsections give a taxonomy of the DJP cases and their relationship, review complexity results for DJP, and discuss the state of knowledge for IMCF, the four basic cases of DJP, and a few of its specialized versions.

The DJP Cases and their Relationship

The basic distinctions between the DJP problems are made according to whether the graph is directed or undirected and whether the paths are vertex- or only edge-disjoint. Using the D/U and V/E codes defined on page 14, the four resulting basic cases can be called D–V, D–E, U–V, and U–E. Furthermore, there are easy ways to reformulate a U–E as a U–V and a U–V as a D–V problem [Schrijver, n.d.], and D–V and D–E problems can be reformulated as each other. In this sense the U–E case is the easiest and the D–V and D–E cases the hardest. The other shorthand codes defined on p. 14 can be used to designate restricted versions of the problem.

The Intractability of the DJP Cases

This subsection will summarize the versions of the DJP problem that are known to be NP-complete. This was recently reviewed by Middendorf and Pfeiffer [1990]. All four basic cases are NP-complete in general, even for planar graphs, even for grids [Frank, 1990b]. In addition, the U–E case remains NP-complete if the graph is Eulerian [Middendorf and Pfeiffer, 1990], U–V remains NP-complete if $\mathcal{G}$ is planar and has maximum vertex degree of 3 [*ibid.*], and D–E remains NP-complete even if the number of commodities is restricted to two [Fortune, Hopcroft, and Wyllie, 1980].

The Integer Multicommodity Flow Problem

The integer multicommodity flow problem, phrased as a decision problem, is this:

(IMCF) Given a multicommodity flow graph with integer-valued maximum capacities defined on each edge, and with integer-valued total flows f_i required for each commodity, are there integer-valued flows that are simultaneously feasible and will sat' .f· every demand?

There is also a correspoı. ng construction problem and a maximum sum-of-integer-flows problem. The graph can be directed or undirected, so these two basic cases can be called D– – and U– –. If demands and capacities are all 1, then IMCF becomes the D–E or U–E DJP problem. Other special versions of IMCF correspond to D–V and U–V DJP. Even, Itai, and Shamir [1976] showed that both basic cases of IMCF are NP-complete, even for just two commodities and even if all capacities are 1. (This immediately implies that the U–E and D–E DJP problems are NP-complete even when $\mathcal{H}$ is restricted to two sets of parallel edges.) Evans and Jarvis [1979] found sufficient conditions for the existence of IMCFs, but they are computationally laborious to evaluate, requiring the comparison of every pair of cycles in the graph. Most other research on the IMCF problem has concentrated on finding tractable special cases of undirected planar graphs, which turn out to be cases in which the maximum flow can be shown to be half-integral. Before discussing these results it will be convenient to discuss the *general cut condition* (GCC), which is important for both the IMCF and DJP problems, as well as for partial multicommodity cuts.

The general cut condition is a statement of the obviously necessary condition that for any edge-cut in a graph, the total flow required between terminals on opposite sides of the cut must not exceed the capacity of the edges in the cut. In a directed problem, it is the capacity of the arcs oriented in the correct direction that counts. The *surplus* of a cut is its capacity minus the required flow across it (some reasearchers call this the *free capacity* or *excess capacity*). These concepts apply to a DJP problem exactly as to IMCF with all demands and capacities equal to 1. For both DJP and IMCF, many results are of the form of finding cases in which GCC is sufficient as well as necessary for the existence of a feasible solution. In IMCF, the GCC has been shown to be necessary and sufficient for U--PBEu* [Okamura and Seymour, 1981] and U--P*Eu* [Seymour, 1981b]. Polynomial algorithms also result immediately from these results.

The directed case of IMCF has not received the attention that the undirected case has. Nagamochi and Ibaraki [1989] showed that if a certain kind of class of directed networks has the max-flow min-cut property, then that class also has the integral max flow property. These classes are those such that if a network $\mathcal{N}$ is in the class (a network being a directed graph, a set of ordered pairs of terminals, and integer-valued capacities and demands), then the same network with either a capacity or a demand decreased by one is also in the class.

Disjoint Paths Case U–E

This is perhaps the most widely studied of the DJP cases. Published results fall into four broad categories: special cases in which the general cut condition is sufficient as well as necessary; special cases in which GCC plus an additional condition is necessary and sufficient; special cases in which sufficient conditions have been found: and polynomial algorithms for the decision or construction problems. A common thread is the consideration of graphs that are planar, or T-planar, or planar with the terminals on two faces, or Eulerian when augmented with the (s_i, t_i) demand edges, or limited to a fixed number of commodities. Consideration of these configurations cuts across the four categories mentioned earlier, but some configurations find more application in one than in another.

GCC has been shown to be necessary and sufficient for the existence of disjoint paths for a variety of conditions when the augmented graph $\mathcal{G}^*$ is Eulerian. These include U–EP*Eu* and U–EPBEu* [Seymour, 1981b]; whenever no subgraph can be contracted to the complete graph on five vertices (K_5) [Seymour, 1981a]; U–EPEu* when $\mathcal{G}$ can be drawn with all s_i on one finite face and all t_i on another in the same cyclical order and opposite direction [Schrijver, 1989]; and U–EEu* when the demand graph $\mathcal{H}$ has a minimum cover of one or two vertices or is a K_4 or C_5 graph (circuit on 5 vertices) when parallel edges are replaced by a single edge [Frank, 1990b; Schrijver, 1991]. (These conditions on $\mathcal{H}$ can be stated as this: $\mathcal{H}$ has neither of the configurations of Figure 1 as a subgraph [Schrijver, 1991].) Frank has described two situations in which GCC plus another condition is necessary and sufficient for the existence of disjoint paths. The first involves an *intersection criterion* (IC), which turns out to be always necessary for a feasible solution. Let a *tight set* be any set $X \subset V$ such that the cut-set $\Gamma(X)$ has zero surplus. The IC states that for any

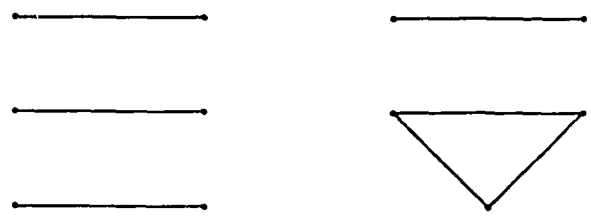


Figure 1: Forbidden Configurations in $\mathcal{H}$ for Sufficiency of GCC in DJP U–EEu*

two tight sets X and Y, $\Gamma(X \cap Y)$ must have an even surplus. The necessity of this condition comes from this: If the surplus of $\Gamma(X \cap Y)$ is odd, any feasible solution of the DJP problem would leave at least one edge in it unused. This edge would also be part of $\Gamma(X)$, $\Gamma(Y)$, or both. But no feasible solution can leave any member of such tight cuts unused. Frank [1990c] shows that IC and GCC together are sufficient as well as necessary for the case U–EP* when all terminals are on one or two faces of $\mathcal{G}$. In another paper, Frank [1985] gives a more complicated condition that is necessary and sufficient for the case U–EPB when all interior nodes are even. The condition is this: Take any family of edge-cuts each of which divides $\mathcal{G}$ into two connected parts. Let $\mathcal{G}'$ be $\mathcal{G}$ with with the union of these cuts removed. Let q be the number of components of $\mathcal{G}'$ with vertex set X such that Γ(X) with respect to $\mathcal{G}^*$ is odd. Then for any such family, the surplus of every cut in it must equal or exceed $q/2$.

The third category of U–E DJP results involves conditions that can simply be shown to be sufficient. One such condition follows directly from Frank's result above for U–EP* when $\mathcal{H}$ is on one or two faces of $\mathcal{G}$ [1990c]: if all surpluses are strictly positive, the problem is feasible. Similarly, in U–EPB when all interior nodes are even, if all surpluses are positive then the problem is feasible [Frank, 1990b]. Two more results for planar graphs when all surpluses are nonnegative and even are due to Okamura [1983]: if there are at most two faces such that the source and sink for each commodity are on one of them, the problem is feasible; if one set of terminal pairs is on one face and the rest of the pairs share a common vertex on that same face, the problem is also feasible. Some other sufficiency results are even more specialized. Frank [1990b] surveys a series of results of the following form: for some integer p and k, if a graph is p-edge-connected, then any k-commodity DJP U–E problem defined on it is feasible regardless of which vertices are the terminals.

The last major category of results is that of decision or construction algorithms. Sometimes the proof of a sufficiency result can be used directly in a polynomial algorithm. This is the case for Frank's special cut condition for the U–EPB case with all interior nodes even [1985]. Seymour [1980a] developed a polynomial algorithm for the U–E2 problem; it can also be used as the basis for a construction algorithm. Based on the work of Frank [1985, 1990b], Becker and Mehlhorn [1986] have offered both decision and construction algorithms for U–EPB with all vertices even, and also for U–EPBEu*. Middendorf and Pfeiffer [1990] have shown that U–EP in general is polynomial if the terminals are on a bounded number of faces of $\mathcal{G}$.

Besides these direct results for variations on the U–E problem, others could be derived from results for the U–V, D–V, or D–E cases. All that required is that any special requirements (*e.g.* planarity) persist after transforming the problem.

Disjoint Paths Case U–V

The problem of finding vertex-disjoint paths in an undirected graph has not been investigated in as many variations as the corresponding edge-disjoint problem. The common restrictions are to planar graphs, perhaps with the terminals on one or two faces, and to two-commodity problems. U–VP results have recently been surveyed by Sebő [1990]. The general cut condition (GCC) applies as a necessary condition to U–V in a modified form: no subset of ℓ vertices can separate more than ℓ terminal pairs. Necessary and sufficient conditions (including GCC) have been found for two cases. These will be discussed in this subsection, followed by some sufficient conditions that have been found and some results on algorithms.

The first necessary and sufficient condition was found for the U–VPB case by Robertson and Seymour [1986]. The condition is GCC plus the supply graph $\mathcal{G}$ being cross-free, *i.e.* having the terminals for no two commodities in interleaved order s_i, s_j, t_i, t_j. Since in a planar graph vertex-disjoint paths can never cross, this is obviously a necessary condition. The same investigators also found necessary and sufficient conditions for the general U–VP problem [Robertson and Seymour, 1988], but the conditions do not seem to lend themselves to succinct statement. They correspond to the graph being "big enough" and to terminals of different pairs being "far enough apart."

Research on special sufficient conditions for the existence of disjoint U–V paths has been limited to finding the level of connectivity in a graph at which any given pairs can be connected with disjoint paths. Some results of this sort have been given by Watkins [1968], Larman and Mani [1970], and Shiloach [1980].

The last category of results concerns the development of practical or polynomially bounded algorithms for the U–V DJP problem. This has followed a fairly straightforward development. First Perl and Shiloach [1978] gave construction algorithms for two commodities in planar graphs. Shiloach [1980] and Seymour [1980a] then independently showed that the general U–V2 problem is polynomial. Robertson and Seymour [1985] proved that U–VPk is polynomial for any fixed number k of commodities, but the proof did not yield a practical algorithm (the degree of the polynomial exceeds 10^{100} for $k = 4$). Shortly afterwards, the same writers showed that U–VP is polynomial for arbitrary k if all terminals are on one or two faces [1986]; this yielded an easy greedy algorithm for the U–VPB case. Recently, Schrijver [1990] has proved the generalization that U–VP is polynomial if the terminals are on any bounded number of faces of $\mathcal{G}$. Also, Middendorf and Pfeiffer [1990] have shown that a restricted U–V problem is polynomial if and only if *some planar graph is excluded from appearing* as a subgraph in the restricted class.

Disjoint Paths Cases D–E and D–V

Published results for disjoint paths in directed graphs are so few that it is convenient to discuss the edge-disjoint and vertex-disjoint cases together. In any case, either general problem can be reformulated as the other by splitting nodes, as already mentioned. There are no known situations in which the general cut condition is sufficient by itself. There is one situation in which a necessary and sufficient condition is known. A few sufficiency results and a couple of polynomial algorithms have been published.

Very recently, Ding, Schrijver, and Seymour [1992] proved that a D–VPB problem has a solution if and only if cross-freeness and a slightly extended GCC hold. Cross-freeness is obviously necessary, just as it was for U–VPB. GCC is extended to state that any arc-cut that separates $\mathcal{G}$ into two components must contain arcs in the proper direction in the same order as the terminal pairs it separates. A valid situation is shown in Figure 2.

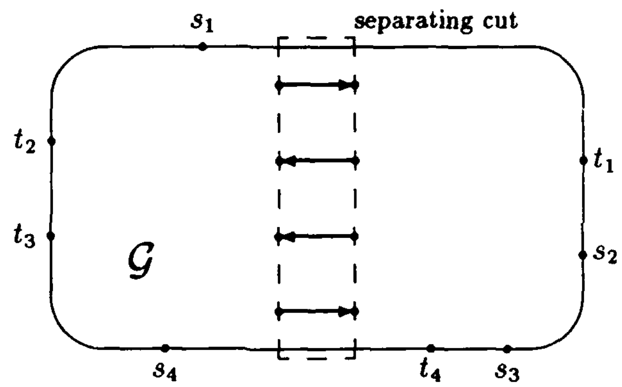


Figure 2: Extended GCC for D–VPB Problem (Example)

Thomassen [1985] has shown that for D–V2 with all nodes (except possibly the terminals) having both in-degree and out-degree at least 2, a solution will exist if the graph is nonplanar, or if it is planar and the terminals are not on one face, or if it is planar and they are on one face but in cross-free order. Besides this somewhat specialized result, known sufficiency conditions are limited to some graphs of very high connectivity in which any vertices can feasibly be taken as terminals [Frank, 1990b].

Algorithmic results are limited to two cases. Ding, Schrijver, and Seymour [1992] provide an algorithm for their D–VPB case based on their necessary and sufficient conditions for it. The other case is for graphs without directed circuits: Perl and Shiloach [1978] give a polynomial construction algorithm for D–V2, and Fortune, Hopcroft, and Wyllie [1980] show that the D–Vk decision problem in such a graph is polynomial for any fixed integer k.

Specialized Versions of the Disjoint Paths Problem

There are a few frequently-studied special variations on the disjoint paths problem. These are more distantly related to the multicommodity cut problem, so they do not need a detailed treatment here. First is finding not just one path between each terminal pair but as many as possible; this is reviewed by Frank [1990b]. Second is homotopic routing, *i.e.* finding exact paths once one is given a general routing around certain "holes" in the space in which the graph in embedded. This is an important practical problem in VLSI circuit design, and has been reviewed by Frank [1990a], Frank and Schrijver [1990], and Schrijver [1990]. The third variation is also of special application to VLSI circuit design. This is the problem of disjoint paths in a grid: circuit chips often have a grid-based design structure. The terminals in a grid are often taken to be on the boundary. Mehlhorn and Preparata [1986], Lai and Sprague [1987], and Suzuki, Ishiguro, and Nishizeki [1990] are examples of this line of research.

4 GENERAL RESULTS

This chapter contains some simple results of a general nature that will be used in subsequent chapters. It consists of three sections. The first concerns the relationship between full multicommodity cuts and multiterminal cuts. The second gives two lemmas on partial multicommodity cuts. The third describes some multicommodity cut problems that can immediately be seen to be intractable based on others' work. We assume throughout the paper that $\mathcal{G}$ is a connected graph.

4.1 Full Multicommodity Cuts and Multiterminal Cuts

The multiterminal cut problem was defined in Section 3.2 as the problem of finding the smallest set of edges in a multiterminal graph that separates each terminal from all of the others. Clearly a minimal multiterminal cut for k terminals will separate $\mathcal{G}$ into exactly k components: there must be one for each terminal, and an edge connecting a component with no terminal to any other component would be superfluous in the cut. Define an *i-way multiterminal-separation problem* as this: given a multiterminal graph $\mathcal{G}$ and an i-partition of its terminal set T, find the smallest set of edges that will separate each element of the partition from all of the others (without necessarily leaving each set in one component). Obviously, a multiterminal-separation problem can be reduced to a multiterminal cut problem by introducing an extra terminal for every element of the partition and connecting it to each of the corresponding members by a large number of parallel edges. There is an important relationship between a full multicommodity cut problem and certain multiterminal-separation problems that it induces. This relationship will be discussed next.

Any minimal FMCC will separate $\mathcal{G}$ into components, which in turn induce in the terminal set T a partition with the property that s_i and t_i are in different sets for all i. Call any partition of a multicommodity terminal set a *feasible partition* if it has this property. Clearly an optimal FMCC is also the minimum multiterminal-separation cut for the partition of T that it induces. This leads to the following lemma.

Lemma 1 *In a full multicommodity cut problem, any minimum-cardinality i-way multiterminal-separation cut over all feasible i-partitions,* $2 \leq i \leq k+1$, *is an optimal solution.*

Proof: Such a cut is clearly a feasible solution. Suppose there is a feasible FMCC C that has smaller cardinality than any of these multiterminal-separation cuts. It cannot induce a partition of T into fewer than $k+2$ sets. By Proposition 2, it cannot be optimal. ■

Define a *supreme cut* in an undirected full multicommodity cut problem to be any multiterminal-separation cut using a feasible 2-partition. A supreme cut is always feasible but not necessarily optimal. Proposition 1 in Chapter 3 states that for $k = 2$, the smallest supreme cut is the optimal FMCC. This is not true for $k \geq 3$, as can be seen in the graph in Figure 3. Every 2-partition of the terminals yields a supreme cut of at least four edges, but the optimal FMCC consists of the three edges a. Note that

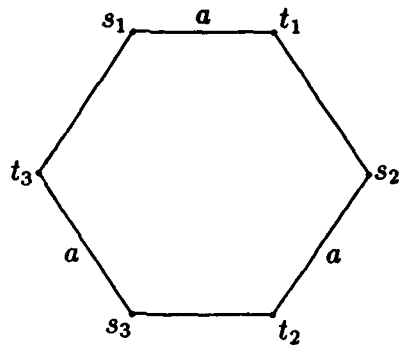


Figure 3: The Optimal FMCC Is Not a Supreme Cut

for this graph, the remaining graph $\mathcal{R}$ always contains at least three components. One might conjecture that the minimum supreme cut is the optimal FMCC if it only leaves two components in $\mathcal{R}$. However, this is also not true, as Figure 4 shows. The

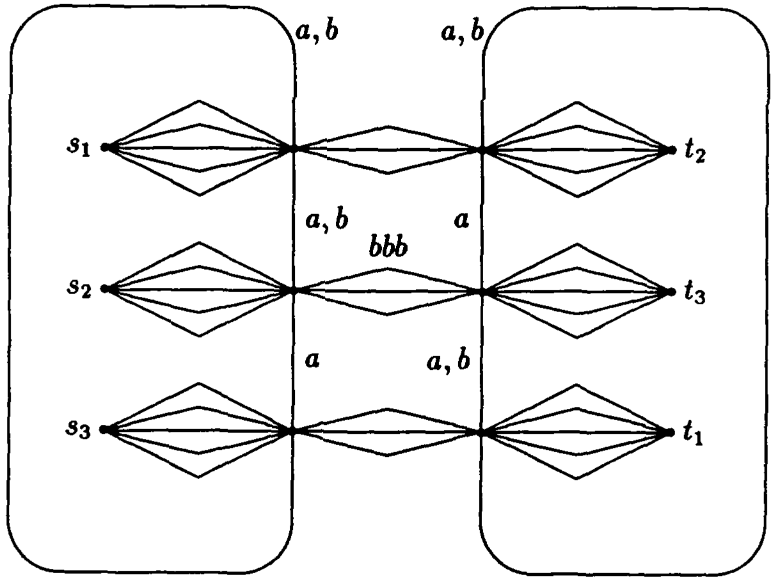


Figure 4: Minimum Supreme Cut Leaving Two Components in $\mathcal{R}$

minimum FMCC in this case is the six edges labeled a. A minimum supreme cut (*e.g.* $\{t_1, s_2, s_3\}$ *vs.* $\{s_1, t_2, t_3\}$) contains seven edges (labeled b), and leaves $\mathcal{R}$ in two components.

Lemma 1 is strengthened by the following two additional results. Two subsets of T will be said to *share a commodity* if one contains the source and the other the sink for that commodity.

Lemma 2 *In any full multicommodity cut problem, there exists a feasible i-partition of the terminals such that each set in the partition shares at least one commodity with each of the others, and the minimum-cardinality i-way multiterminal-separation cut is an optimal solution to the problem.*

Proof: Suppose two subsets T_1, T_2 in the minimum-i i-partition of T yielding the optimal FMCC C_i share no commodity. Then the $(i-1)$-partition with T_1 and T_2 combined is also feasible, and the corresponding minimum $(i-1)$-way multiterminal-separation cut is a valid FMCC C_{i-1}. C_i is optimal, so $|C_{i-1}| \geq |C_i|$. But C_i is also a valid $(i-1)$-way multiterminal-separation cut, so $|C_{i-1}| \leq |C_i|$. So $|C_i| = |C_{i-1}|$, and C_i or an FMCC of the same size can be found via a partition into fewer sets, contradicting the assumption. ∎

Corollary 1 *The optimal solution for a full multicommodity cut problem is the smallest of all minimum i-way multiterminal-separation cuts using feasible i-partitions with*

$$2 \leq i \leq \left\lfloor \sqrt{2k + \tfrac{1}{4}} + \tfrac{1}{2} \right\rfloor \tag{1}$$

Proof: By Lemma 2, only feasible partitions in which each set shares a commodity with every other need be considered. If there are i sets each sharing at least one of k pairs of terminals, we must have $\frac{1}{2}i(i-1) \leq k \Rightarrow i \leq \left\lfloor \sqrt{2k + \frac{1}{4}} + \frac{1}{2} \right\rfloor$. ∎

It is important to distinguish between the number of components in $\mathcal{R}$ and the value of i in the i-way multiterminal-separation cut used to find $\mathcal{R}$. For the graph in Figure 5, the unique optimum FMCC is found by the $\{s_1, \ldots, s_k\}$ *vs.* $\{t_1, \ldots, t_k\}$

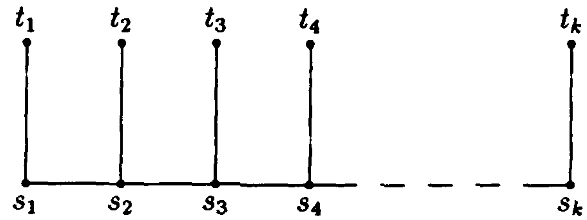


Figure 5: A Graph with at Least $k+1$ Components in $\mathcal{R}$

two-way cut, but it leaves $k+1$ components in $\mathcal{R}$. The supreme cut $\{s_1, t_2, s_3, t_4, \ldots\}$ *vs.* $\{t_1, s_2, t_3, s_4, \ldots\}$ results in the maximum FMCC and leaves $2k$ components in $\mathcal{R}$. The only necessary relationship is that $\mathcal{R}$ will have at least as many components as the order of the cut.

Unfortunately, the reduction of the FMCC problem to a set of multiterminal-separation cut problems is of limited practical use. The multiterminal cut problem and hence the multiterminal-separation cut problem is NP-hard, even for $k = 3$ [Dahlhaus *et al.*, 1992].

4.2 Partial Multicommodity Cuts

The following results will be useful when considering partial multicommodity cuts. They apply to both edge-cut, edge-disjoint and to vertex-cut, vertex-disjoint problems (UeE and UvV).

Lemma 3 *Let* **P** *be an instance of a UeE, UvV, DeE, or DvV partial multicommodity* *cut problem, and let J_r and J_{r+1} be minimum interjacent sets leaving r and $r+1$ disjoint paths, respectively. Then $|J_r| \geq |J_{r+1}| + 1$.*

Proof: In $\mathcal{R}_r = \mathcal{G} - J_r$ no more than r disjoint s_i–t_i paths can be found. For any $x \in J_r$, consider $\mathcal{R}' = \mathcal{G} - (J_r - \{x\})$. No more than $r+1$ disjoint paths can be found in $\mathcal{R}'$, since at most one additional disjoint path can be drawn through the single added element x, which has in effect been removed from J_r and added to $\mathcal{R}_r$. So $J_r - \{x\}$ is a feasible interjacent set for leaving $r+1$ disjoint paths, and $|J_r| - 1$ is an upper bound on $|J_{r+1}|$. ■

Lemma 4 *Let* **P** *be an instance of a UeE, UvV, DeE, or DvV partial multicommodity cut problem, and let J be a minimal interjacent set leaving at most r disjoint paths. Then exactly r disjoint s_i–t_i paths can be drawn in $\mathcal{R} = \mathcal{G} - J$.*

Proof: If J is feasible then no more than r paths can be drawn in $\mathcal{R}$. Suppose fewer than r paths can be drawn. If any element of J is deleted from the interjacent set, and so added to $\mathcal{R}$, then at most one additional disjoint path is made possible. So J without this element is still feasible, contradicting its minimality. ■

4.3 Two Propositions on Intractability

This section concludes the chapter by giving two propositions on the intractability of large classes of multicommodity cut problems. The first is on full edge-cuts in undirected graphs, while the second covers a variety of partial cut problems.

The first proposition makes it unlikely that Proposition 1 can be extended to cover more than two commodities.

Proposition 3 *The full multicommodity edge-cut problem in general graphs is NP-hard for $k \geq 3$, even for k fixed.*

Proof: A three-terminal cut problem can be reduced to a multicommodity cut with $k = 3$ by identifying one of the three terminals as s_1 and t_2, another as s_2 and t_3, and the third as t_1 and s_3. The multiterminal cut problem is NP-hard *even* for k fixed at 3 [Dahlhaus *et al.*, 1992]. ■

The second proposition is based on the fact that a partial multicommodity cut problem cannot be easier than the corresponding disjoint paths problem. A disjoint paths decision problem with k_1 commodities can be reduced to a PMCC problem with $k_1 + 1$ commodities and $r = k_1$ by attaching to the graph a simple structure with a new source and sink that can be disconnected by the removal of one structure element. The solution to such a PMCC cannot be of cardinality more than 1. If it is empty, then the answer to the DJP decision problem is "no," otherwise the answer is "yes." Since many versions of the DJP problem are NP-hard, the corresponding versions of the PMCC problem are also intractable.

Proposition 4 *All cases of the partial multicommodity cut problem are NP-hard, and they remain NP-hard even under the following restrictions:*

- ◇ UeE, UvV, UeV, and UvE on an induced subgraph of a grid with $r = k - 1$
- ◇ UvV and UeV on a planar graph with maximum vertex-degree of 3 and $r = k - 1$
- ◇ DvV and DeV on a grid with $r = k - 1$
- ◇ DeE and DvE with k fixed and $r = 2$

Proof: NP-hard restricted versions of the disjoint paths problem (see Section 3.4) can be reduced to these problems, as outlined in the paragraph above. ■

5 FULL MULTICOMMODITY CUTS IN PLANAR GRAPHS

This chapter discusses various cases of the Ue–P full multicommodity cut problem: the supply graph $\mathcal{G} = (V, E)$ is taken to be undirected and planar, and a minimum set of edges $C \subseteq E$ is sought such that $\mathcal{G} - C$ will contain no s_i–t_i paths. In Section 5.2 an algorithm for the T-planar case Ue–PB is presented; the algorithm is polynomially bounded if the number of commodities k is fixed. If the terminals are not restricted to the boundary of $\mathcal{G}$ then the problem becomes more complicated. In Section 5.3 a polynomial algorithm for the Ue–Pk case with $k = 3$ is presented. Sections 5.4 and 5.5 consider the problem when k is part of the input. If k is not fixed then the T-planar FMCC problem becomes NP-complete even under some strong restrictions on $\mathcal{G}$; the proofs for this are the main content of Section 5.4. However, if the terminals are in non-crossing order (Ue–P*BN) then a polynomial algorithm is again possible, and such an algorithm is developed in Section 5.5.

The algorithms in this chapter mostly depend on a correspondence between edge-cuts in $\mathcal{G}$ and trees in a modification of the geometric-dual of $\mathcal{G}$. This modified dual will be called $\mathcal{G}^{Dm}$; the next section describes how $\mathcal{G}^{Dm}$ is constructed and proves the necessary relationships. The next section also applies the method to the Ue–PB multiterminal cut (MTC) problem. This has three purposes. It illustrates the use of the modified dual in a problem that is a bit simpler than the FMCC problem. It shows that an important special case of the planar MTC problem is polynomially bounded, even though the general problem is NP-complete [Dahlhaus *et al.*, 1992]. Finally, since an MTC problem can be reformulated as an FMCC problem with co-located terminals, it provides a polynomial algorithm for a special case of the Ue–PB FMCC problem.

5.1 The Modified Dual $\mathcal{G}^{Dm}$ and the Ue–PB MTC Problem

The modified dual graph $\mathcal{G}^{Dm}$ is formed in the following way. Let $\mathcal{G}$ be a T-planar graph with terminals $t_1, t_2, \ldots, t_k$ numbered in order clockwise around the boundary. Form the geometric-dual $\mathcal{G}^D$, but split the dual vertex corresponding to the infinite face into k vertices, one for each of the k paths into which the boundary of $\mathcal{G}$ is partitioned by the terminals. Call these k vertices the ***separation-vertices*** of $\mathcal{G}^{Dm}$, and label them $\sigma_1, \sigma_2, \ldots, \sigma_k$ in clockwise order around the boundary of $\mathcal{G}^{Dm}$ in such a way that σ_i corresponds to the path from t_{i-1} clockwise to t_i, and σ_1 corresponds to the path from t_k clockwise to t_1 in $\mathcal{G}$ (see Figure 6). This is the modified dual graph $\mathcal{G}^{Dm} = (V^{Dm}, E^{Dm})$. Note that it is planar, and that there is a one-to-one correspondence between the edges of $\mathcal{G}$ and those of $\mathcal{G}^{Dm}$.

We will use certain conventions when discussing $\mathcal{G}$ and $\mathcal{G}^{Dm}$. The notation $[t_i, t_j]$ means the set of terminals in $\mathcal{G}$ from t_i clockwise through t_j; the set $[\sigma_i, \sigma_j]$ is similarly defined in $\mathcal{G}^{Dm}$. If i is not in $\{1, \ldots, k\}$ then $\sigma_i \equiv \sigma_{i \pm k}$, and similarly for t_i. If C is a set of edges in $\mathcal{G}$ then C^{Dm} is the set of corresponding edges in $\mathcal{G}^{Dm}$ (not the

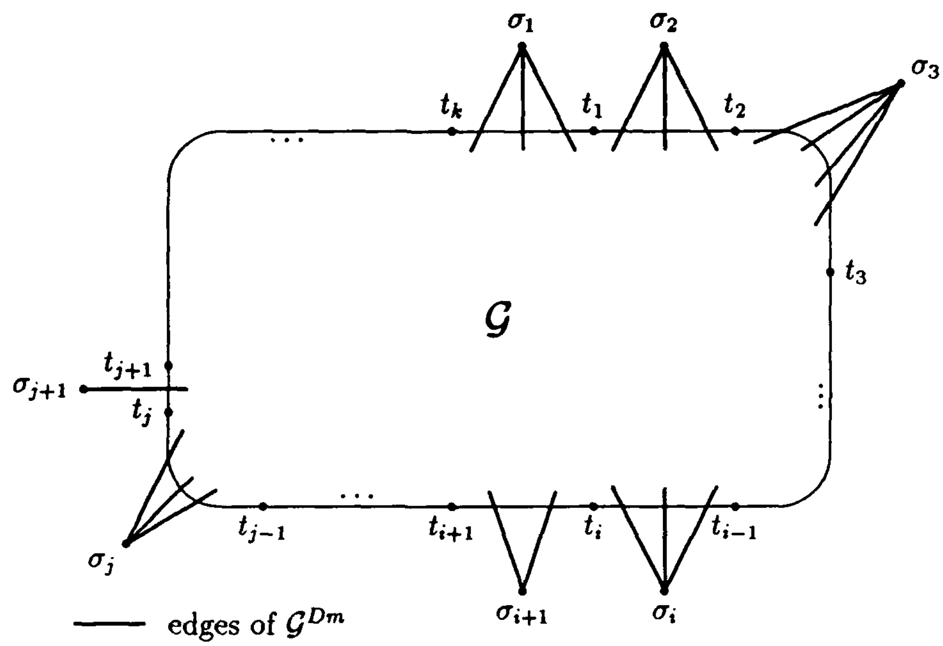


Figure 6: Construction of $\mathcal{G}^{Dm}$

modified dual of the subgraph induced by C). It should cause no confusion if C^{Dm} is also used to refer to the subgraph induced in $\mathcal{G}^{Dm}$ by that set of edges.

Because of the planarity of $\mathcal{G}$, an edge-cut C that separates t_i from t_j corresponds to a set of edges C^{Dm} containing a path from one of $[\sigma_{i+1}, \sigma_j]$ to one of the vertices in $[\sigma_{j+1}, \sigma_i]$. A minimum cut corresponds to a shortest path. Similarly, a path in $\mathcal{G}^{Dm}$ from σ_a to σ_b corresponds to a cut-set in $\mathcal{G}$ separating $[t_a, t_{b-1}]$ from $[t_b, t_{a-1}]$; a shortest path between σ_a and σ_b corresponds to a minimum edge-cut between those two sets of terminals. The following proposition extends the correspondence between paths and cuts to the case of multiterminal cuts.

Proposition 5 *An edge-set C in a T-planar undirected graph $\mathcal{G}$ is a multiterminal cut if and only if C^{Dm} has a component that includes all the separation-vertices $\sigma_1, \ldots, \sigma_k$.*

Proof: $\Rightarrow$ Let C be a MTC in $\mathcal{G}$, and let σ_a and σ_b be in different components of C^{Dm}. (These separation-vertices must be endvertices for some edges in C^{Dm}, or else t_{a-1} and t_{b-1} are not separated from t_a and t_b, respectively.) Let $\sigma_{a'}$ and $\sigma_{a''}$ be the separation-vertices in the same component as σ_a that are closest in the boundary of $\mathcal{G}$ to some separation-vertex in the component that includes σ_b; $\sigma_{a'}$ and $\sigma_{a''}$ are in the clockwise and counterclockwise directions from a vertex in that component, respectively (see Figure 7). Planarity ensures that $\sigma_{a'}$ and $\sigma_{a''}$ will be well-defined.

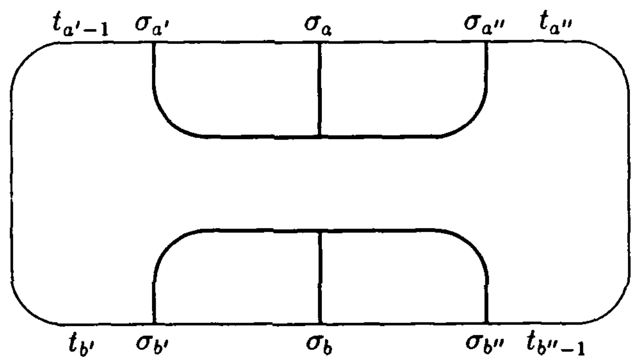


Figure 7: Proof of Proposition 5

Let $\sigma_{b'}$ and $\sigma_{b''}$ be the separation-vertices in σ_b's component and closest to $\sigma_{a'}$ and $\sigma_{a''}$, respectively. There can be no path in C^{Dm} corresponding to a cut in $\mathcal{G}$ between $t_{a'-1}$ and $t_{a''}$, for no path in C^{Dm} connects one of $[\sigma_{a'}, \sigma_{a''}]$ to a member of $[\sigma_{a''+1}, \sigma_{a'-1}]$. Nor can we have $t_{a'-1}$ identical to $t_{a''}$, for the component containing σ_b comes between. So the cut C does not separate $t_{a'-1}$ and $t_{a''}$, a contradiction.

$\Leftarrow$ If a component contains every separation-vertex in $\mathcal{G}^{Dm}$, then it contains a path from σ_a to σ_b for any $a \neq b$. Therefore C must contain a cut between any two terminals. ∎

The minimum subgraph of $\mathcal{G}^{Dm}$ that includes all the separation-vertices is by definition the Steiner tree in $\mathcal{G}^{Dm}$ on those vertices. Hence we get the following corollary:

Corollary 2 *An edge-set C is a minimum MTC in a T-planar graph $\mathcal{G}$ if and only if C^{Dm} is a Steiner tree on the separation-vertices of $\mathcal{G}^{Dm}$.*

The separation-vertices in $\mathcal{G}^{Dm}$ are on its boundary, and the problem of finding the minimum-edge-weight Steiner tree in a planar graph on a subset of its boundary vertices is known to be polynomial [Provan, 1988; Erickson, Monma, and Veinott, 1987; Provan, 1983]. Since a weighted version of $\mathcal{G}^{Dm}$ can be obtained from $\mathcal{G}$ in polynomial time, we also get the following:

Corollary 3 *The weighted multiterminal cut problem in T-planar undirected graphs is polynomially bounded.*

As already noted, the planar MTC problem is NP-complete if the terminals are allowed to be anywhere in $\mathcal{G}$ [Dahlhaus *et al.*, 1992].

These results suggest an algorithm for solving a weighted Ue–PB MTC problem. The method is to form $\mathcal{G}^{Dm}$, then use Provan's algorithm [1988] to find a Steiner tree on it. The major work would be finding the tree, which would be $O(\alpha + f^2k^2)$, where α is the work required to find the distance between each pair of vertices in $\mathcal{G}^{Dm}$ ($\alpha = f^2$ [Frederickson, 1987]) and f is the number of vertices in that graph (*i.e.* the number faces in $\mathcal{G}$).

5.2 The Ue–PBk FMCC Problem

Next consider the problem of finding a weighted full multicommodity cut in a

T-planar undirected graph. An algorithm is developed that is polynomially bounded for any fixed number of commodities k. (It will be shown later that the problem is NP-hard if k is part of the input; see Section 5.4.) The algorithm is an adaptation of Provan's method for finding Steiner trees [1988]; it operates on the modified dual graph $\mathcal{G}^{Dm}$.

The modified dual $\mathcal{G}^{Dm}$ is defined for a T-planar multicommodity flow graph just as it was for a multiterminal graph in the previous section. It is the normal geometric-dual with the vertex corresponding to the infinite face of $\mathcal{G}$ split into one separation-vertex for each path between consecutive terminals in the boundary of $\mathcal{G}$. The only difference between the FMCC and MTC cases is in the labeling of the separation-vertices, and comes from the fact that in $\mathcal{G}$ the terminal set T is the union of a set of sources $\{s_1, \ldots, s_k\}$ and a set of sinks $\{t_1, \ldots, t_k\}$. These sets may occur in any order around the boundary of $\mathcal{G}$, and two or more terminals can be co-located (except the source and sink of one commodity). Label the distinct terminals in $\mathcal{G}$ from a_1 to a_z in clockwise order around the boundary from an arbitrary start. Note that $z = 2k$ if none of the terminals coincide; otherwise $z < 2k$. Then $\mathcal{G}^{Dm}$ as already defined will have exactly z separation-vertices; label them likewise from σ_1 clockwise to σ_z, with σ_1 falling between a_z and a_1 (see example in Figure 8). Each edge in $\mathcal{G}^{Dm}$ has the same weight as the corresponding edge in $\mathcal{G}$.

(It will turn out to be awkward if a terminal is encountered twice in tracing the boundary of $\mathcal{G}$. For this reason, if a terminal is an articulation vertex, it should be displaced to the end of an infinite-weight edge before constructing $\mathcal{G}^{Dm}$. This has

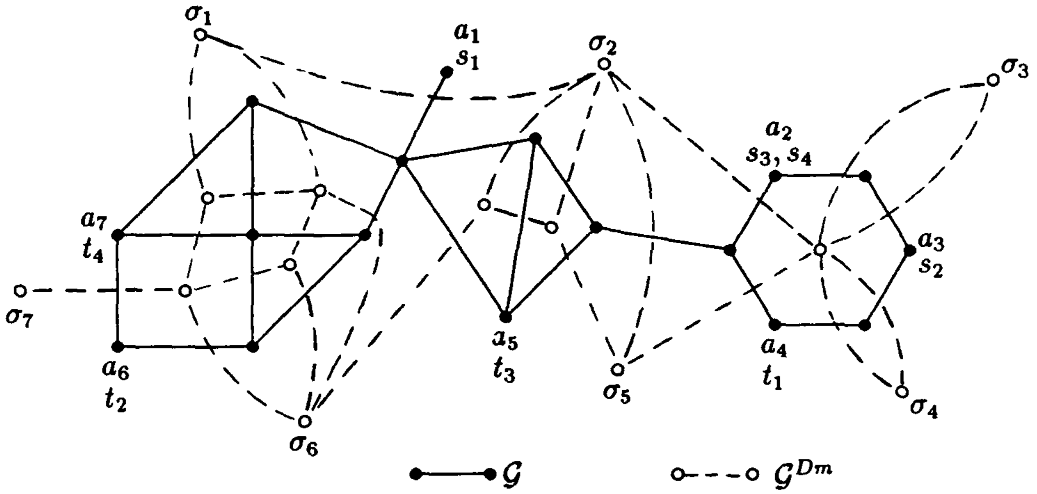


Figure 8: Example of $\mathcal{G}^{Dm}$ for an MCF Graph

been done for $s_1 \equiv a_1$ in Figure 8.)

The following definitions are needed to deal with the distribution of terminals around the boundary of $\mathcal{G}$:

$$[a_i, a_j] \equiv \text{the set of distinct terminal vertices in } \mathcal{G} \text{ from } a_i \text{ clockwise through } a_j \tag{2}$$

$$K_i \equiv \text{the set of commodities with a terminal at } a_i \tag{3}$$

In Figure 8, $K_2 = \{3, 4\}$. (An index i of a_i that is incremented or decremented out of the range $\{1, \ldots, z\}$ is to be taken modulo z.) Now we can extend the relationship between cuts in $\mathcal{G}$ and trees in $\mathcal{G}^{Dm}$ to the case of multicommodity flow graphs.

Proposition 6 *In a weighted T-planar multicommodity flow graph $\mathcal{G}$, a minimum full multicommodity edge-cut C will correspond to a forest in $\mathcal{G}^{Dm}$, and each tree in that forest will be a Steiner tree on some subset of the separation-vertices.*

Proof: Note that s_i is separated from t_i in $\mathcal{G}$ by a set of edges C if and only if C^{Dm} includes a path from σ_a to σ_b, where σ_a lies between s_i and t_i (reckoned clockwise) and

σ_b lies similarly between t_i and s_i. Now if C is an FMCC, then C^{Dm} must contain at least two separation-vertices; otherwise the boundary of $\mathcal{G}$ will connect all the terminals in $\mathcal{R}$. If C^{Dm} is not a forest, then it contains a circuit. But every circuit in $\mathcal{G}^{Dm}$ encloses only nonterminals in $\mathcal{G}$, so one of the edges corresponding to the circuit could be deleted from C without connecting any terminals in $\mathcal{R}$, and the cut would not be minimum-weight. Furthermore, C^{Dm} can have no degree-1 vertices except separation vertices, for the edge on such a vertex would correspond to a superfluous edge in C. So C^{Dm} is a forest and all its degree-1 vertices are separation-vertices. It remains only to show that every tree in it is in fact a minimum-weight Steiner tree on the separation-vertices it contains.

Let one of these trees contain the set of separation-vertices $\{\sigma_b, \sigma_c, \ldots, \sigma_x, \sigma_y\}$, occuring in the given order clockwise around $\mathcal{G}^{Dm}$. Then the corresponding edge set in $\mathcal{G}$ (which is a subset of the FMCC C) serves to separate the terminal vertex sets $[a_y, a_{b-1}]$, $[a_b, a_{c-1}]$, ..., $[a_x, a_{y-1}]$, each from the others. But the minimum-weight Steiner tree on those separation-vertices will accomplish exactly the same thing, and at minimum cost. Therefore the tree must be a minimum-weight Steiner tree on its separation-vertices, as required. ■

(This proposition can also be proved directly from Lemma 1 on p. 33 and Corollary 2 on p. 43.) From this point a tree in $\mathcal{G}^{Dm}$ will be referred to as *separating* a commodity when the tree's edge-set C^{Dm} corresponds to a set of edges C in $\mathcal{G}$ that separates that commodity's source from its sink.

As a consequence of Proposition 6 we have the following:

Proposition 7 *The weighted full multicommodity edge-cut problem in a T-planar graph is polynomially bounded for any fixed number of commodities k.*

Proof: Proposition 6 showed that the optimal FMCC in $\mathcal{G}$ corresponds to Steiner trees on subsets of the separation-vertices in $\mathcal{G}^{Dm}$. If k is fixed then the number of separation-vertices is bounded at $2k$, and the number of possible subsets of them is also bounded. So all that is required to solve the problem is to evaluate a bounded number of minimum-weight Steiner tree problems and pick the smallest-weight tree or combination of trees that separates all commodities. Since the Steiner tree problem in a weighted T-planar graph is polynomial, and checking whether a given set of trees separates all commodities is polynomial, this restricted FMCC problem is also polynomial. ■

(This proposition can also be proved directly from Lemma 1 on p. 33 and Corollary 3 on p. 43.) Of course, it can be expected that there are algorithms that are much better than examining all possible Steiner trees. We now develop one that runs in time $O(f^2)$, where f is the number of faces in $\mathcal{G}$ or vertices in $\mathcal{G}^{Dm}$, if k is fixed.

The method used in the algorithm for the Ue–PB FMCC problem is adapted from Provan's method of finding a Steiner tree in a T-planar graph [1988]. The approach is to find the minimum-weight forest of Steiner trees in $\mathcal{G}^{Dm}$ that separates all commodities. It relies on the fact that if any Steiner tree in $\mathcal{G}^{Dm}$ is partitioned into two pieces, the separation-vertices in the two pieces cannot be in interleaved order around the boundary of $\mathcal{G}^{Dm}$. Also, the separation-vertices on any two separate trees in $\mathcal{G}^{Dm}$ cannot be interleaved.

Some definitions will be needed for the algorithm.

$$\text{term}[\sigma_i, \sigma_j] \equiv \text{the set of commodities with at least one terminal falling between } \sigma_i \text{ and } \sigma_j, \text{ i.e. in } [a_i, a_{j-1}] \ (\text{term}[\sigma_i, \sigma_i] \equiv \phi) \quad (4)$$

$$\text{comm}[\sigma_i, \sigma_j] \equiv \text{the set of commodities with both terminals falling between } \sigma_i \text{ and } \sigma_j \ (\text{comm}[\sigma_i, \sigma_i] \equiv \phi) \quad (5)$$

Thus term$[\sigma_i, \sigma_j]$ is the union of K_i clockwise through K_{j-1}; comm$[\sigma_i, \sigma_j]$ is the set of commodities represented twice in K_i through K_{j-1}. In Figure 8, term$[\sigma_2, \sigma_6] = \{1, 2, 3, 4\}$ and comm$[\sigma_2, \sigma_6] = \{3\}$. A set of edges in $\mathcal{G}^{Dm}$ (or the subgraph induced by them) will be called a $[\sigma_i, \sigma_j]$*-feasible subforest* if it corresponds to a subset of a minimal FMCC in $\mathcal{G}$, separates all commodities in comm$[\sigma_i, \sigma_j]$, and contains no separation-vertices that are not in $[\sigma_i, \sigma_j]$.

Two functions on $\mathcal{G}^{Dm}$ will also be needed. The first is:

$$\mathbf{T}[\sigma_a, \sigma_b] \equiv \text{weight of a minimum-weight } [\sigma_a, \sigma_b]\text{-feasible subforest with } \sigma_a \text{ and } \sigma_b \text{ in the same component} \quad (6)$$

Thus $\mathbf{T}[\sigma_a, \sigma_b]$ is the weight of an optimal tree that is on σ_a, σ_b, and possibly some of $[\sigma_a, \sigma_b] - \{\sigma_a, \sigma_b\}$, plus the weight of all other trees on subsets of $[\sigma_a, \sigma_b] - \{\sigma_a, \sigma_b\}$. For the second function, suppose we know that some optimal forest has $[\sigma_j, \sigma_i]$ and $[\sigma_i, \sigma_j] - \{\sigma_i, \sigma_j\}$ in different components, and let K be the subset of commodities in term$[\sigma_j, \sigma_i]\backslash$comm$[\sigma_j, \sigma_i]$ that are not separated by the components on $[\sigma_j, \sigma_i]$. For instance, if the only trees on members of $[\sigma_j, \sigma_i]$ are one that contains σ_j and σ_c and another that contains σ_d and σ_i ($\sigma_c, \sigma_d \in [\sigma_j, \sigma_i]$), then the commodities in term$[\sigma_c, \sigma_d]\backslash$comm$[\sigma_j, \sigma_i]$ are not separated by any tree on $[\sigma_j, \sigma_i]$, and term$[\sigma_c, \sigma_d] \cap$

term$[\sigma_i, \sigma_j]$ is such a set K. These commodities have only one terminal between σ_i and σ_j (reckoned clockwise) but nevertheless need to be separated by a tree on a subset of $[\sigma_i, \sigma_j] - \{\sigma_i, \sigma_j\}$. Then the second function can be defined as follows:

$$\mathbf{S}([\sigma_i, \sigma_j], K) \equiv \text{weight of a minimum-weight } [\sigma_i, \sigma_j]\text{-feasible and (unless } \sigma_{i+1} \equiv \sigma_j) \; [\sigma_{i+1}, \sigma_{j-1}]\text{-feasible subforest that also separates all commodities in } K \tag{7}$$

Thus $\mathbf{S}([\sigma_i, \sigma_j], K)$ is the weight of the optimal trees that are on subsets of $[\sigma_{i+1}, \sigma_{j-1}]$ when the commodities in K have one terminal between σ_i and σ_j (clockwise) and the other between σ_j and σ_i and not separated by any other trees. See Figure 9 for an illustration of the $\mathbf{T}$ and $\mathbf{S}$ functions..

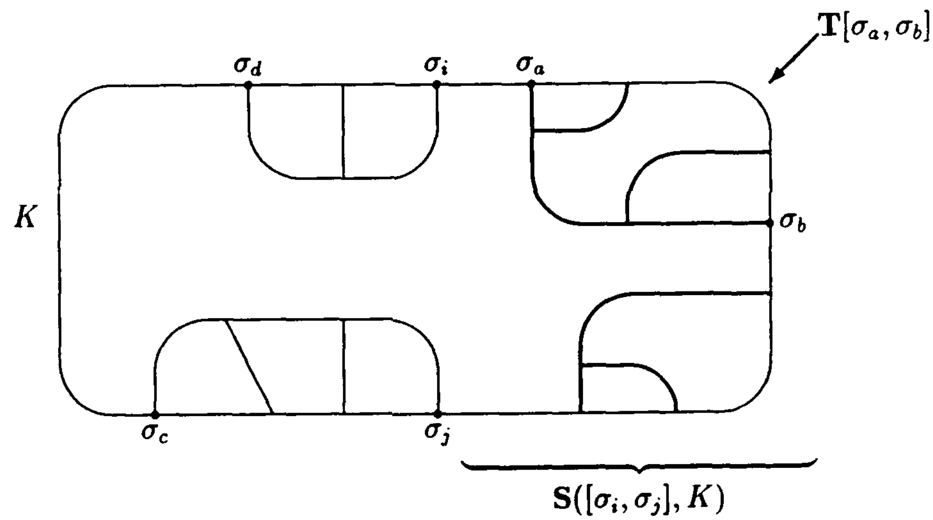


Figure 9: The **T** and **S** Functions (Example)

Now consider any commodity in K_z, and suppose that it is also in K_i ($i \neq z$). This commodity has one terminal at a_z and the other at a_i. To separate this commodity, the optimal forest must have at least one tree with one vertex in $[\sigma_1, \sigma_i]$ and the other in $[\sigma_{i+1}, \sigma_z]$. Of all such trees, one must be closest to a_z (see Figure 10). So if W_{opt}

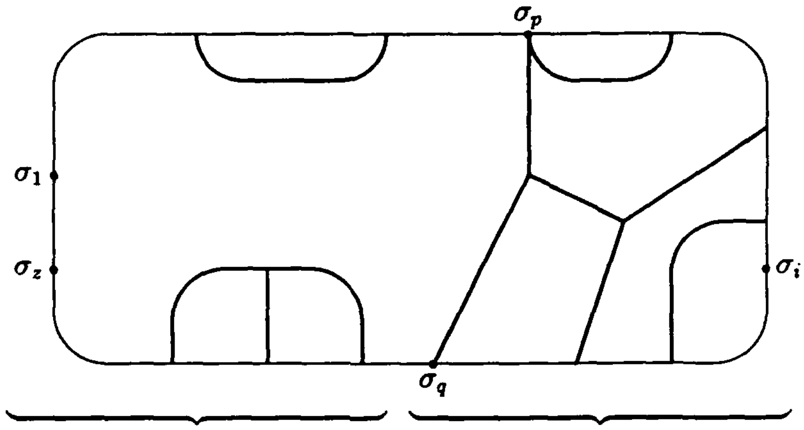


Figure 10: **T** and **S** Functions in an Optimal FMCC (Example)

is the weight of the optimal FMCC, we can write

$$W_{\text{opt}} = \min_{(\sigma_p, \sigma_q) \in Q} \{\mathbf{T}[\sigma_p, \sigma_q] + \mathbf{S}([\sigma_q, \sigma_p], \phi)\} \tag{8}$$

where

$$Q \equiv \{(\sigma_p, \sigma_q) : \sigma_p \in [\sigma_1, \sigma_i], \sigma_q \in [\sigma_{i+1}, \sigma_z]\}$$

T and **S** can be calculated via recursive formulas, which will now be developed.

A recursive formula for $\mathbf{S}([\sigma_i, \sigma_j], K)$. **S** need not be defined if $[\sigma_i, \sigma_j]$ contains only one vertex (*i.e.* $i = j$). Suppose $[\sigma_i, \sigma_j]$ contains only two or three separation-vertices. Then $[\sigma_{i+1}, \sigma_{j-1}]$ contains at most one separation-vertex, and there can be no Steiner tree on it. This gives

$$\mathbf{S}([\sigma_i, \sigma_j], K) = 0 \quad \text{if } |[\sigma_i, \sigma_j]| \leq 3 \text{ and } K \cap \text{term}[\sigma_i, \sigma_j] = \text{comm}[\sigma_i, \sigma_j] = \phi \tag{9}$$

$$\mathbf{S}([\sigma_i, \sigma_j], K) = \infty \quad \text{if } |[\sigma_i, \sigma_j]| \leq 3, \text{ and } K \cap \text{term}[\sigma_i, \sigma_j] \text{ or comm}[\sigma_i, \sigma_j] \text{ is nonempty} \tag{10}$$

Now suppose $[\sigma_i, \sigma_j]$ contains four or more separation-vertices. To begin with,

$$\mathbf{S}([\sigma_i, \sigma_j], K) \quad = \quad 0 \qquad \text{if } K \cap \text{term}[\sigma_i, \sigma_j] = \text{comm}[\sigma_i, \sigma_j] = \phi \tag{11}$$

Otherwise, there must be at least one tree on some subset of $[\sigma_{i+1}, \sigma_{j-1}]$, and of all such trees, one must include a separation-vertex σ_a that is closest to σ_i. This tree's separation-vertices must include a separation-vertex σ_b that is closest to σ_j. However, the terminals between σ_i and σ_a cannot be for commodities in K, nor can both terminals for a commodity be in that interval. This leads to the recursive equation

$$\mathbf{S}([\sigma_i, \sigma_j], K) \; = \min_{(\sigma_a, \sigma_b) \in Q'} \{\mathbf{T}[\sigma_a, \sigma_b] + \mathbf{S}([\sigma_b, \sigma_j], K')\} \tag{12}$$

where

$$\begin{array}{rcll} Q' & \equiv & \{ \quad (\sigma_a, \sigma_b): & \sigma_a \in [\sigma_{i+1}, \sigma_j] - \{\sigma_{j-1}, \sigma_j\} \; , \\ & & & \sigma_b \in [\sigma_{a+1}, \sigma_j] - \{\sigma_j\} \; , \\ & & & K \cap \text{term}[\sigma_i, \sigma_a] = \phi \; , \qquad \text{and} \\ & & & \text{comm}[\sigma_i, \sigma_a] = \phi \qquad\qquad\qquad \} \end{array}$$

and

$$K' = (K \cup \text{term}[\sigma_i, \sigma_a]) \backslash \text{term}[\sigma_a, \sigma_b])$$

(see Figure 9).

A recursive formula for $\mathbf{T}[\sigma_i, \sigma_j]$. Define the following additional functions on the separation vertices of $\mathcal{G}^{Dm}$ and any $\nu \in V^{Dm}$:

$$\mathbf{C}([\sigma_i, \sigma_j], \nu) \equiv \text{weight of a minimum-weight } [\sigma_i, \sigma_j]\text{-feasible subforest with } \nu, \sigma_i, \text{ and } \sigma_j \text{ in the same component} \tag{13}$$

$$\mathbf{B}([\sigma_i, \sigma_j], \nu) \equiv \text{weight of a minimum-weight } [\sigma_i, \sigma_j]\text{-feasible subforest with } \nu, \sigma_i, \text{ and } \sigma_j \text{ in the same component and with } \nu \text{ of degree two or more or identical with } \sigma_i \text{ or } \sigma_j \tag{14}$$

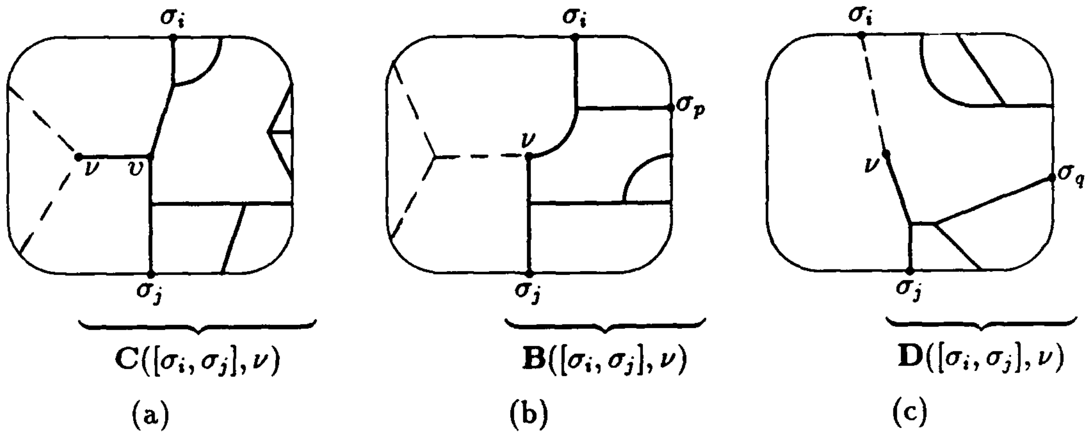


Figure 11: The **C**, **B**, and **D** Functions (Examples)

$\mathbf{D}([\sigma_i, \sigma_j], \nu) \equiv$ weight of a minimum-weight $[\sigma_i, \sigma_j]$-feasible subforest with ν and σ_j in the same component (15)

(See Figure 11.) It is clear that

$$\mathbf{T}[\sigma_i, \sigma_j] = \mathbf{B}([\sigma_i, \sigma_j], \sigma_i) \tag{16}$$

It remains to find recursive formulas for **B**, **C**, and **D**.

A recursive formula for $\mathbf{B}([\sigma_i, \sigma_j], \nu)$. **B** need not be defined unless $[\sigma_i, \sigma_j] \geq 2$. There must be a branch from ν that goes to σ_i and possibly some other separation-vertices, the closest of which to σ_j is some $\sigma_p \in [\sigma_i, \sigma_{j-1}]$ (see Figure 11b). This leads to the formula

$$\mathbf{B}([\sigma_i, \sigma_j], \nu) = \min_{\sigma_p \in [\sigma_i, \sigma_{j-1}]} \{\mathbf{C}([\sigma_i, \sigma_p], \nu) + \mathbf{D}([\sigma_p, \sigma_j], \nu)\} \tag{17}$$

A recursive formula for $\mathbf{C}([\sigma_i, \sigma_j], \nu)$. If $i = j$, then

$$\mathbf{C}([\sigma_i, \sigma_j], \nu) = \mathrm{d}(\nu, \sigma_i) \tag{18}$$

where $\mathrm{d}(\nu, v)$ is the weighted distance in $\mathcal{G}^{Dm}$ between any two vertices ν and v. If $i \neq j$, the subforest corresponding to **C** must include a first branch point at a vertex

v and include the shortest path from ν to v (see Figure 11a; v may coincide with ν, σ_i, or σ_j). Thus

$$\mathbf{C}([\sigma_i, \sigma_j], \nu) = \min_{v \in V^{Dm}} \{\mathrm{d}(\nu, v) + \mathbf{B}([\sigma_i, \sigma_j], v)\} \tag{19}$$

A recursive formula for $\mathbf{D}([\sigma_i, \sigma_j], \nu)$. The tree that includes ν and σ_j must also include a separation-vertex $\sigma_q \in [\sigma_{i+1}, \sigma_j]$ that is farthest from σ_j (see Figure 11c). Thus

$$\mathbf{D}([\sigma_i, \sigma_j], \nu) = \min_{\sigma_q \in [\sigma_{i+1}, \sigma_j]} \{\mathbf{S}([\sigma_i, \sigma_q], \phi) + \mathbf{C}([\sigma_q, \sigma_j], \nu)\} \tag{20}$$

With these functions defined, we can write the following algorithm.

Algorithm 2 (Weighted Ue–PB FMCC)

input: $\mathcal{G}^{Dm}$ for a weighted Ue–PB FMCC problem
output: Weight of the optimal FMCC in $\mathcal{G}$
begin
for every $\nu, v \in V^{Dm}$
  find minimum distance $\mathrm{d}(\nu, v)$
for every $\nu \in V^{Dm}$ and every separation-vertex σ_i
  set $\mathbf{C}([\sigma_i, \sigma_i], \nu) = \mathrm{d}(\sigma_i, \nu)$
  set $\mathbf{T}[\sigma_i, \sigma_i] = 0$
for $i = 1, \ldots, z-1$ (size of separation-vertex set, minus one)
  for $j = 1, \ldots, z$ (start of set)
    set $\ell = (j + i) \bmod z$
    compute $\mathbf{S}([\sigma_j, \sigma_\ell], \phi)$ using (9)–(12)
    if $i \geq 3$ **then**
      for every nonempty subset K of $\mathrm{term}[\sigma_j, \sigma_\ell] - \mathrm{comm}[\sigma_j, \sigma_\ell]$
        compute $\mathbf{S}([\sigma_j, \sigma_\ell], K)$ using (11)–(12)
    for every $\nu \in V^{Dm}$
      compute $\mathbf{D}([\sigma_j, \sigma_\ell], \nu)$ using (20)
      compute $\mathbf{B}([\sigma_j, \sigma_\ell], \nu)$ using (17)
      compute $\mathbf{C}([\sigma_j, \sigma_\ell], \nu)$ using (19)
    set $\mathbf{T}[\sigma_j, \sigma_\ell] = \mathbf{B}([\sigma_j, \sigma_\ell], \sigma_j)$
compute W_{opt} using (8)
return W_{opt}
end

The major work in this algorithm is in four places: (1) finding all the distances $\mathrm{d}(\nu, v)$ ($O(\alpha)$); (2) calculating $\mathbf{S}([\sigma_i, \sigma_j], K)$ for every subset K ($O(2^k k^4)$); (3) calculating $\mathbf{D}$ and $\mathbf{B}$ ($O(fk^3)$); and (4) calculating $\mathbf{C}$ ($O(f^2k^2)$). The total time required is $O(\alpha + 2^k k^4 + fk^3 + f^2k^2)$, where α is the time to calculate every $\mathrm{d}(\nu, v)$ and f is the number of vertices in $\mathcal{G}^{Dm}$. Since $\alpha = f^2$ [Frederickson, 1987], if k is fixed, the algorithm is $O(f^2)$.

5.3 The Non-T-planar Case with $k = 3$

The previous section considered the problem of full multicommodity cuts in planar graphs when all terminals are on the boundary. The problem becomes more difficult if the terminals may be anywhere in the supply graph $\mathcal{G}$. This section considers this Ue–Pk problem for the case of three commodities. The modified dual $\mathcal{G}^{Dm}$ is no longer helpful; instead we must work with the standard geometric-dual $\mathcal{G}^D$.

Let an *isolating cut* $\mathrm{I}(U, W)$ be a minimal set of edges in $\mathcal{G}$ that separates a set of vertices U from another set W (assuming $U \subset V$, $W \subset V$, and $U \cap W = \phi$). There is a duality between circuits in $\mathcal{G}^D$ and isolating cuts in $\mathcal{G}$. Any circuit partitions the faces of $\mathcal{G}$ into two sets. A face is *enclosed* by a circuit if it is in the set that does not also include the infinite face. If $C = \mathrm{I}(U, V - U)$, then C^D is the boundary of a set of faces in $\mathcal{G}^D$ that correspond to either the members of U or those of $V - U$. If $C_{\min} = \mathrm{I}_{\min}(U, W)$ is a minimum isolating cut for two vertex sets U and W, then $C^D_{\min}$ is the boundary of a set of faces in $\mathcal{G}^D$ that includes the faces corresponding either to U or to W. Any minimum isolating cut can be found in polynomial time. The approach in this section will be to find a polynomially-bounded number of minimum isolating cuts that will include an optimal FMCC.

We know from Corollary 1 and Lemma 2 in Chapter 4 that the smallest of four 2-way cuts between sets of three terminals and eight 3-way cuts between sets of two terminals is an optimal FMCC. The 2-way cuts are easily found; each is the minimum isolating cut between the two terminal sets. It remains to find the 3-way cuts. Dahlhaus *et al.* [1992] provide a polynomially-bounded method of finding the minimum 3-way cut separating three single vertices in a planar graph. However, in this FMCC problem the graph will not necessarily be planar after the two terminals in a set are linked. A modification of their method is needed.

From Proposition 2 in Chapter 4, we know that a minimum FMCC will separate $\mathcal{G}$ into no more than four components. In fact, the following proposition will show that we need only consider cuts that separate $\mathcal{G}$ into three or fewer components. Two components of $\mathcal{G} - C = \mathcal{R}$ are *adjacent* if C contains an edge connecting them. Two adjacent components must share a commodity (*i.e.* each contain one of the commodity's terminals) if C is minimal.

Proposition 8 *If an optimal three-commodity edge-cut C separates $\mathcal{G}$ into four components, then C is either (1) an isolating cut, or (2) the union of two isolating cuts separating two of the terminals from the other four.*

Proof: Let C be an optimal Ue–P3 FMCC that separates $\mathcal{G}$ into four components $\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \mathcal{R}_4$. One component must have exactly one terminal; without loss of generality let $\mathcal{R}_1$ contain s_1. $\mathcal{R}_1$ can only be adjacent to one other component, namely the one containing t_1, for otherwise the cut edges between it and any other component would be superfluous. Let $\mathcal{R}_2$ contain t_1. Since $\mathcal{G}$ is connected, $\mathcal{R}_2$ must be adjacent to another component and share a commodity with it; let it be $\mathcal{R}_3$ and suppose $s_2 \in \mathcal{R}_2$ and $t_2 \in \mathcal{R}_3$. At this point there are two possibilities: $\mathcal{R}_4$ can be adjacent to either $\mathcal{R}_2$ or $\mathcal{R}_3$; in either case we can suppose $t_3 \in \mathcal{R}_4$. So the configuration of $\mathcal{R}$ must be like either (a) or (b) in Figure 12. In each case, C is a combination of disjoint cuts, each of which is a minimum cut between two sets of terminals. In

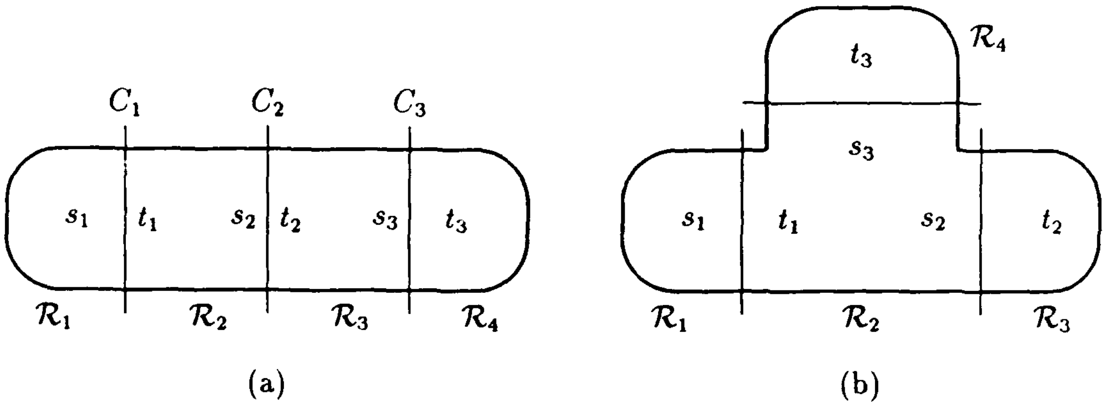


Figure 12: Configurations for Ue–P3 FMCCs with Four Components in $\mathcal{R}$

Figure 12a, $C_1+C_2 = \mathrm{I}(\{t_1, s_2\}, \{s_1, t_2, s_3, t_3\})$ and $C_2+C_3 = \mathrm{I}(\{t_2, s_3\}, \{s_1, t_1, s_2, t_3\})$. In Figure 12b, $C = \mathrm{I}(\{s_1, t_2, t_3\}, \{t_1, s_2, s_3\})$. In either case the proposition holds. ∎

It remains to find the minimal 3-way cuts that divide $\mathcal{G}$ into exactly three components and separate all commodities. If C is such a cut, then C^D will divide $\mathcal{G}^D$ into three regions. One of these will contain the infinite face, so C^D will be made up of two circuits, each of which encloses the faces corresponding to two terminals (and possibly other faces). These two circuits may be edge-disjoint, or they may share a path of one or more edges.

Let us suppose that the optimal FMCC is the minimum 3-way cut for a given partition of T, say $\{s_1, s_2 \,|\, t_2, s_3 \,|\, t_3, t_1\}$. If the dual FMCC C^D is in the form of two disjoint circuits then C can be found in the following way. Find the minimum isolating cut for each set in the partition *vs.* the other four terminals. Find the union of each pair of these three minimum isolating cuts. The smallest of these unions is the FMCC. (This procedure will also find cuts into four components in the form of Figure 12a.)

The only remaining possibility is that the optimal FMCC is a minimum three-way cut for some partition of T and the dual of the cut is two non-disjoint circuits. Call such a cut a *manacle cut* (Figure 13). Note that in $\mathcal{G}^D$ a manacle cut corresponds to

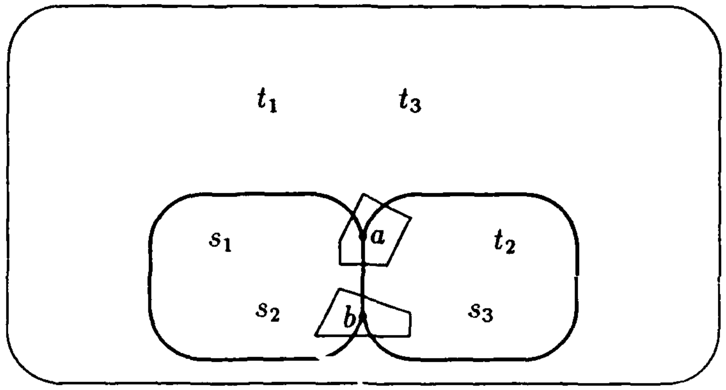


Figure 13: Example of a Manacle Cut

two degree-3 vertices (a and b in Figure 13) and three paths between them.

The shortest path in $\mathcal{G}^D$ between these degree-3 vertices does not necessarily correspond to part of the optimal cut. This is because the shortest path may pass between the two terminals in one partition. In the example in Figure 14, the optimal FMCC

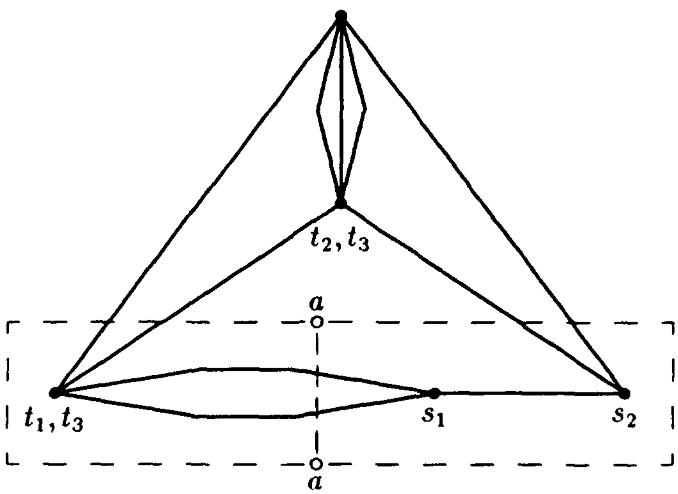


Figure 14: Manacle Cut Not Using Shortest a–b Path

when all edges are equally weighted is the six-edge manacle cut shown. However, the shortest path between vertices a and b is the single edge corresponding to $s_1 s_2$.

Let f be the number of faces in $\mathcal{G}$, *i.e.* the number of vertices in $\mathcal{G}^D$. There are $O(f^2)$ possible choices for a and b, and if we examine all of them we are sure to find the best manacle cut (assuming we are starting with the correct 3-partition of T and selection of which two sets are enclosed in the manacle).

Let T_1 and T_2 be the sets of terminals enclosed in the manacle, let T_3 be the remaining pair, and let a and b be the faces of $\mathcal{G}$ corresponding to the branch points. Note that for a minimal manacle one set of consecutive vertices around the boundary of a must fall outside the manacle, while the rest fall inside (in either of the two interior regions). Let us call these two sets $\mathrm{Out}(a)$ and $\mathrm{In}(a)$, and let $\mathrm{Out}(b)$ and $\mathrm{In}(b)$ be defined the same way. If d is the maximum degree of a vertex in $\mathcal{G}^D$, then there are $O(d^2)$ ways $\mathrm{Out}(a)$ can be chosen and $O(d^4)$ ways all four Out and In sets can be selected.

The boundary of an optimal manacle corresponds to the minimum isolating cut $\mathrm{I}_{\min}(\, T_1 \cup T_2 \cup \mathrm{In}(a) \cup \mathrm{In}(b)\,,\; T_3 \cup \mathrm{Out}(a) \cup \mathrm{Out}(b)\,)$. If that isolating cut does not correspond to a simple circuit in $\mathcal{G}^D$, then the manacle cut is not optimal for this selection of a, b, and their In and Out sets. Once the boundary C_B of the manacle cut is found in this way, then the remaining a–b cut is simply $\mathrm{I}_{\min}(T_1, T_2)$ in $\mathcal{G} - C_B$.

The foregoing arguments prove the correctness of the following algorithm.

Algorithm 3 (Weighted Ue–P3 FMCC)

input: $\mathcal{G}$ and $\mathcal{G}^D$ for a weighted Ue–P3 FMCC problem
output: weight of the minimum FMCC
begin
for each of the four feasible 2-partitions of T, find the minimum-weight
    corresponding 2-way cut. Let the weight of the smallest be C^2.
set $C^{\text{man}} = C^{\text{disj}} = \infty$
for each of the eight feasible 3-partitions of T
  label the sets in the partition T_1, T_2, T_3
  find $\text{I}_{\min}(T_1, T_2 \cup T_3)$, $\text{I}_{\min}(T_2, T_1 \cup T_3)$, $\text{I}_{\min}(T_3, T_1 \cup T_2)$ and let C^* be
    the weight of the smallest union of any two of them
  set $C^{\text{disj}} = \min\{C^{\text{disj}}, C^*\}$
  for each choice of faces of $\mathcal{G}$ a and b
    for each division of vertices on the boundaries of a and b into inner
      and outer
      for each choice of T_i and T_j inside a manacle cut and T_ℓ outside,
        $\{i, j, \ell\} = \{1, 2, 3\}$
        set $C_{\text{B}} = \text{I}_{\min}(T_i \cup T_j \cup \text{In}(a) \cup \text{In}(b), T_\ell \cup \text{Out}(a) \cup \text{Out}(b))$
        set $C_{\text{X}} = \text{I}_{\min}(T_i, T_j)$ in $\mathcal{G} - C_{\text{B}}$
        set $C^{\text{man}} = \min\{ C^{\text{man}}$, weight of $C_{\text{B}} + C_{\text{X}}$ $\}$
return $\min\{ C^2 , C^{\text{disj}} , C^{\text{man}} \}$
end

Let $O(\beta)$ be the polynomial bound on finding a minimum isolating cut in a graph (mostly, but not necessarily entirely, planar). Then the complexity of this algorithm is $O(d^4 f^2 \beta)$, where $\mathcal{G}$ has f faces, each bounded by no more than d edges. This gives the following proposition:

Proposition 9 *The three-commodity edge-cut problem in a planar graph is polynomially bounded.*

Proof: Algorithm 3 provides a polynomial procedure for solving this problem. ∎

It seems likely that Ue–Pk FMCC problems will be polynomially bounded for any fixed k. The optimal FMCC in $\mathcal{G}$ must correspond to a bounded number of circuits in $\mathcal{G}^D$. It is merely a matter of enumerating all the ways the circuits may overlap, and then checking all possibilities for circuit junctions and In and Out sets for each feasible partition of T and circuit topology. It also seems likely that the procedure will become very tedious as k increases.

5.4 The Intractability of T-Planar Cuts when k Varies

We now turn to the problem of finding efficient algorithms when k is part of the input. The main result of this section is that the full multicommodity cut problem in T-planar graphs is NP-complete when the number of commodities k is allowed to vary, even under very stringent restrictions on the form of the graph. The unweighted problem is NP-complete even in a rectangular grid with a rectangular boundary; the weighted problem is NP-complete even in a tree.

Ue–PB FMCCs in Rectangular Grids

The burden of this section is the following proposition and its proof:

Proposition 10 *The unweighted T-planar full multicommodity edge-cut problem is NP-hard, even when restricted to rectangular grids with rectangular boundaries.*

Proof: The problem of deciding if there is a subset of p vertices in a graph such that every edge has at least one of them as an endvertex is NP-complete, even when restricted to graphs in which the degree of every vertex is 3 [Garey and Johnson, 1979]. This *vertex cover* problem will be reduced to a Ue–PB FMCC problem in a rectangular grid. Let a vertex cover problem be given with n vertices $v_1, \ldots, v_n$, all of degree 3, and m edges $e_1, \ldots, e_m$, for which it must be decided whether there is a vertex cover of size p or less.

Construction of the Ue–PB FMCC problem. The problem will include the following types of commodities:

Number	Name	Symbol	Source	Sink	
$2n$	"vertex"	$C^{\ell}_{v_j}$	$s^{\ell}_{v_j}$	$t^{\ell}_{v_j}$	$\ell = 1, 2;\ j = 1, \ldots, n$
$2m$	"edge"	$C^{\ell}_{e_i}$	$s^{\ell}_{e_i}$	$t^{\ell}_{e_i}$	$\ell = 1, 2;\ i = 1, \ldots, m$
$2m$	"connecting"	$C_{e_i v_j}$	$s_{e_i v_j}$	$t_{e_i v_j}$	$i \in \{1, \ldots, m\};\ j \in \{1, \ldots, n\}$

Two vertex commodities will be associated with each vertex in the vertex cover problem, and two edge commodities will be associated with each edge. There will be a connecting commodity for each endvertex of each edge: if $e_i = (v_e, v_f)$, then the associated connecting commodities are $C_{e_i v_e}$ and $C_{e_i v_f}$.

All terminals will be on two opposite sides of a rectangular grid, which will be called the *vertex side* and the *edge side*. The distribution is:

Vertex Side	—	Vertex Commodities
		Connecting Commodities (sources)
Edge Side	—	Edge Commodities
		Connecting Commodities (sinks)

A distinctive arrangement of consecutive terminals and nonterminal vertices on the boundary of the grid will be called a *gadget*.

For each vertex v_j, there will be a gadget like Figure 15 on the vertex side of the grid. The two sources for the two commodities $C_{v_j}^1$ and $C_{v_j}^2$ have between them the

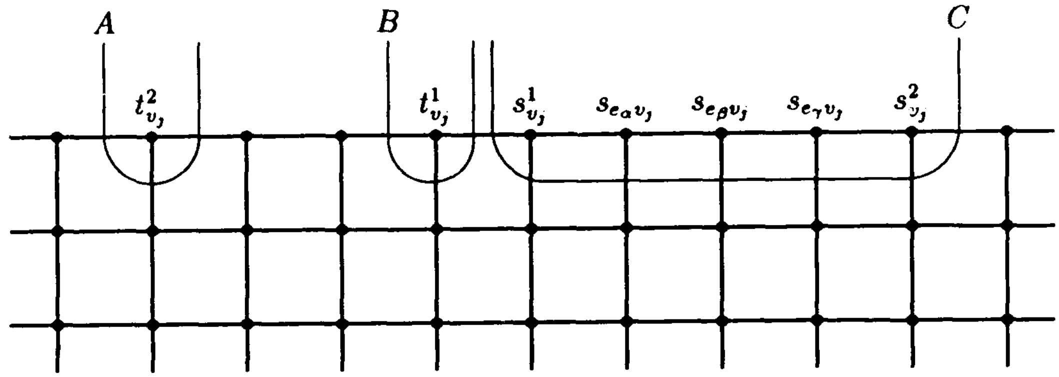


Figure 15: Vertex Gadget for Vertex v_j

three sources for the connecting commodities associated with the three edges $e_\alpha, e_\beta, e_\gamma$ that have v_j as an endvertex. At least two nonterminal vertices are between $t_{v_j}^1$ and $t_{v_j}^2$, and there are at least two nonterminal vertices between consecutive gadgets. No other nonterminal vertices are allowed within the gadget.

On the edge side of the grid there is a gadget like Figure 16 for each edge e_i. The terminals for commodity $C_{e_i}^1$ are at the center, flanked on each side by a sink

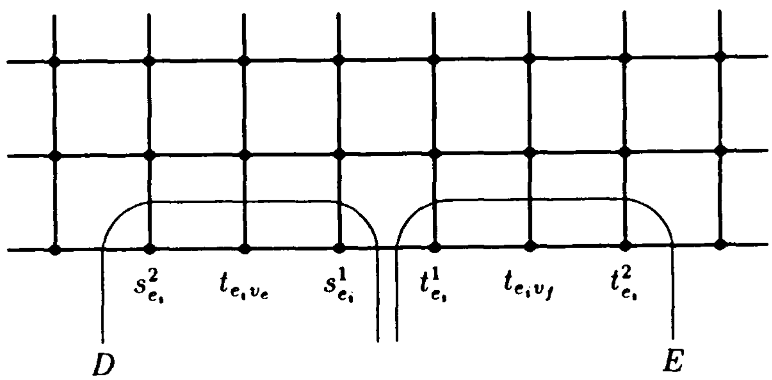


Figure 16: Edge Gadget for $e_i = (v_e, v_f)$

for one connecting commodity and the corresponding terminal for the other edge commodity, as shown in the Figure. The only nonterminal vertices are at least two between gadgets.

The width of the grid in the direction parallel to the vertex and edge sides can be the minimum required to fit in all the gadgets, plus two extra vertices at each end. The depth of the grid should be enough so that the optimal FMCC will not include a cut from the vertex to the edge side (say twice the width).

This FMCC grid is clearly a polynomially-bounded construction. It remains to show that a vertex cover of size p exists if and only if the optimal FMCC for this problem has cardinality less than or equal to a constructed value, which is $6n+5m+p$. Once this is done, the proposition is proved.

FMCC Solution ⇒ Vertex Cover Solution. Suppose we have an optimal solution for the constructed FMCC problem with cardinality no greater than $6n + 5m + p$. Such an FMCC is clearly a union of possibly multiway nonoverlapping minimal cuts; we will see shortly that in fact all the cuts must be simple two-way cuts. The grid is deep enough so that each cut in the FMCC starts and ends (possibly after branching) on either the vertex or the edge side. In fact, each cut passes through the first row of edges normal to one of those sides and isolates a group of consecutive terminals, as cut D does in Figure 16: there is no advantage to going deeper into the grid, and the gaps of two nonterminal vertices between terminals of different commodities means that we can assume that no nonterminal vertices are separated from the rest of the graph.

Note that the vertex commodities can most economically be separated by either two cuts of three edges each isolating the sinks (A and B in Figure 15) or by one cut of seven edges that isolates the sources and also the sources for the associated connecting commodities (C in Figure 15). If in any vertex gadget the given optimal FMCC cuts off one connecting commodity source without cutting off all three, it must use at least seven edges to do so. Replace those edges with a seven-edge-cut around the vertex commodity sources (as C), which will separate the same commodities. If in any vertex gadget the given optimal FMCC separates no connecting commodities, then it must use six edges; replace them if necessary with cuts isolating the vertex commodity sinks (as A and B).

In each edge gadget, the given optimal FMCC must separate each edge commodity with a five-edge cut (like D or E in Figure 16) that also cuts off the sink for one associated connecting commodity. The other connecting commodity sink cannot be isolated by fewer than three edges. Therefore the optimal FMCC must separate that connecting commodity at a cost of at most one additional edge by putting the cut for the corresponding vertex commodity on the sink side (like C in Figure 15), thus cutting off the connecting commodity's source.

Now we will show that a vertex cover of size p or less can be found by taking those vertices whose corresponding vertex commodities are separated by a cut on the source side (like C in Figure 15). There are $6n + 5m$ edges known to be in the FMCC so far: $6n$ for vertex commodity cuts (exclusive of extra edges for source-side cuts) and $5m$ for edge commodity cuts. The remaining p or fewer edges in the FMCC must suffice to separate the connecting commodities. This is most economically done by putting vertex commodity cuts on the source side, so there must be p or fewer vertex commodities with such cuts, and that must separate all connecting commodities. Therefore the proposed vertex cover is of the correct size; it remains to show that it covers every edge. Each edge in the vertex cover problem is associated with an edge gadget in the FMCC problem. That edge gadget is cut on either the source side or the sink side (D or E in Figure 16), but not both. The unseparated connecting commodity sink must have its mate separated by a source-side vertex commodity cut (C in Figure 15). The associated vertex in the vertex cover problem will then cover the edge. Therefore the proposed vertex cover will cover all edges, as required.

Vertex Cover Solution ⇒ FMCC Solution. Given a vertex cover of size p, an FMCC can be constructed as follows. For every vertex v_j in the cover, include the seven-edge cut that cuts off the vertex commodity sources $s^1_{v_j}$ and $s^2_{v_j}$ (as C in Figure 15). For all other vertices, include the three-edge cuts isolating $t^1_{v_j}$ and $t^2_{v_j}$. Include a five-edge cut for each edge commodity (like D or E in Figure 16), if necessary putting it on the side required to separate a connecting commodity not already separated on the vertex side. The correspondence between vertices in the cover and vertex commodities cut on the source side ensures that we cannot have both connecting commodities unseparated. Thus all commodities are cut by an FMCC of $6n + 5m + p$ edges, as required. ∎

Weighted Ue–PB FMCCs in Trees

This section shows that the weighted FMCC problem is NP-hard even when the supply graph $\mathcal{G}$ is a tree (and therefore also T-planar). The disjoint paths construction problem is insoluble in a tree unless none of the unique paths between the terminals of any two commodities overlap, and in that special case the FMCC problem can be solved in linear time. However, if the paths overlap and it is desired to know the optimal way of cutting all of them, the problem becomes intractable.

Proposition 11 *The weighted full multicommodity edge-cut problem is NP-hard even when restricted to trees.*

Proof: As with the previous proposition, proof will be by reduction of a vertex cover problem. Let such a problem be given with n vertexs $v_1, \ldots, v_n$, all of degree 3, and m edges $e_1, \ldots, e_m$, in which it must be decided whether there is a vertex cover of size p.

Construction of the FMCC problem in a tree. The problem will include the following commodities:

Number	Name	Symbol	Source	Sink	
n	"vertex"	C_{v_j}	s_{v_j}	t_{v_j}	$j = 1, \ldots, n$
m	"edge"	C_{e_i}	s_{e_i}	t_{e_i}	$i = 1, \ldots, m$
$2m$	"connecting"	$C_{e_i v_j}$	$s_{e_i v_j}$	$t_{e_i v_j}$	$j \in \{1, \ldots, n\}$; $i \in \{1, \ldots, m\}$

A vertex commodity will be associated with each vertex v_j and an edge commodity with each edge e_i. A connecting commodity will be associated with each endvertex of each edge.

The supply graph $\mathcal{G}$ will be a star graph around a central vertex w. For each cover problem vertex v_j, the star will have two arms as shown in Figure 17. The sink s_{v_j} and the three sinks $s_{e_i v_j}$ for the three edges e_i having v_j as an endvertex will be

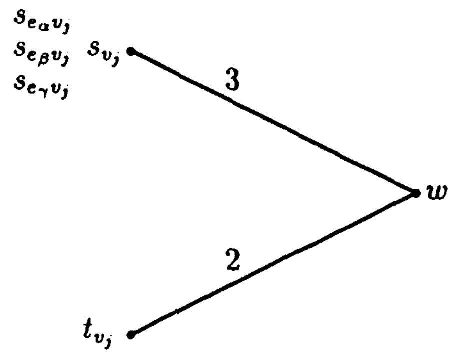


Figure 17: Vertex-associated Arms in Star

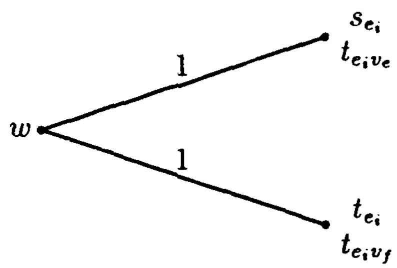


Figure 18: Edge-associated Arms in Star

co-located at the end of an arm of weight 3. The single sink t_{v_j} will be at the end of an edge of weight 2. In any valid FMCC, s_{v_j} will have to be separated from t_{v_j} by a cut of weight 3 or 2 consisting of edge ws_{v_j} or edge wt_{v_j}. The ws_{v_j} cut will correspond to vertex v_j being in the cover.

For each edge $e_i = (v_e, v_f)$, the star will include arms as shown in Figure 18. The source and sink of the edge commodity are at the ends of weight-1 arms, each co-located with a connecting commodity sink. Note that at least one of these two edges must be in any FMCC.

This completes the polynomially-bounded construction of $\mathcal{G}$. It is clearly a tree; in fact it is a star. The FMCC question whose answer will imply an answer to the vertex cover problem is this: is there an FMCC in $\mathcal{G}$ with weight less than or equal to $2n + m + p$? Once it is shown that the vertex cover problem can be satisfied if and only if the optimal FMCC in the constructed tree has a weight no greater than $2n + m + p$, the proposition is proved.

FMCC Solution $\Rightarrow$ Vertex Cover Solution. Suppose an optimal FMCC solution is given with total weight less than or equal to $2n + m + p$. For each v_j, either ws_{v_j} or wt_{v_j} is part of the cut. If both are in it, delete ws_{v_j} from the cut and replace it (if necessary) with the three edges $wt_{e_i v_j}$ for every e_i with v_j as an endvertex; since the total weight of these edges is 3, the result is another optimal FMCC of equal weight. The modified cut must have an edge of weight at least 2 separating each vertex commodity C_{v_j}, for a total of at least $2n$. Every edge commodity C_{e_i} must be separated by one or both of edges ws_{e_i}, wt_{e_i}. If both edges are in the cut for any commodity, delete wt_{e_i}. Suppose $t_{e_i v_f}$ is the connecting commodity sink co-located with t_{e_i}. If $ws_{e_i v_f}$ is not already part of the cut then wt_{v_f} must be; drop the latter and add the former, so that there is no net increase in the size of the cut. At this point there are exactly m weight-1 edges in the cut, so there must be at least $n - p$ weight-2 edges and at most p weight-3 edges.

Let V^c be a candidate vertex cover consisting of every vertex v_j such that the weight-3 edge ws_{v_j} is part of the FMCC as modified. Then $|V^c| \leq p$. Each edge commodity in the constructed problem must have one of the connecting commodities corresponding to its endvertices cut on the vertex commodity side, implying that that endvertex is in V^c. Therefore V^c also covers all edges in the cover problem, as required.

Vertex Cover Solution $\Rightarrow$ FMCC Solution. Suppose there is a vertex cover V^c consisting of p vertices. Then an FMCC can be assembled from the following edges:

Edges	Weight	Separated Commodities
For every vertex $v_j \in V^c$, the weight-3 edge ws_{v_j}	$3p$	Some Vertex Commodities Some Connecting Commodities
For every vertex $v_j \in V - V^c$, the weight-2 edge wt_{v_j}	$2(n-p)$	Remaining Vertex Commodities
For every edge $e_i = (v_e, v_f)$, edge $wt_{e_i v_e}$ if $v_e \in V - V^c$, otherwise $wt_{e_i v_f}$	m	All Edge Commodities Remaining Connecting Commodities

The resulting set of edges is of total weight $2n + m + p$, and it is an FMCC as required. ■

5.5 A Case that is Polynomial in k: Ue–P*BN

The previous section showed that if the number of commodities k is part of the input, then the T-planar full multicommodity cut problem is NP-hard even under severe restrictions on the form of the graph. However, there is a special case in which a polynomial algorithm is possible for general k. Recall that if a T-planar multicommodity flow graph has no two commodities i, j with their terminals in order s_i, s_j, t_i, t_j around the boundary of $\mathcal{G}$ (in either direction), then the terminals are in non-crossing order, and the problem is designated Ue–P*BN. This section shows that this weighted T-planar non-crossing FMCC problem is polynomially bounded even for varying k. The algorithm is based on Algorithm 2, developed in Section 5.2 for the weighted Ue–PBk FMCC problem. A slightly better algorithm is also presented under the additional restriction that the terminals are not nested, *i.e.* that the source and sink for every commodity are consecutive.

In the recursive formulas developed in Section 5.2 for the functions **T**, **S**, **B**, **C**, and **D** (formulas (16), (12), (17), (19), and (20)), the only place that $\mathbf{S}([\sigma_i, \sigma_j], K)$ appeared with $K \neq \phi$ was in the formula for **S** itself. We show that if the terminals are in non-crossing order, $K = \phi$ is the only case that need be considered.

<u>A recursive formula for $\mathbf{S}([\sigma_i, \sigma_j], \phi)$ in Ue–P*BN.</u> Obviously

$$\mathbf{S}([\sigma_i, \sigma_j], \phi) \quad = \quad 0 \qquad \text{if comm}[\sigma_i, \sigma_j] = \phi \tag{21}$$

$$\mathbf{S}([\sigma_i, \sigma_j], \phi) \quad = \quad \infty \qquad \text{if } K_i \cap K_{j-1} \neq \phi \text{ and } \sigma_i \neq \sigma_{j-1} \tag{22}$$

If neither of these conditions hold, let σ_p and σ_q be separation-vertices in $[\sigma_i, \sigma_{j-1}]$ such that there is a commodity that has its terminals at a_p and a_q, and σ_p is as close as possible to σ_i. (Recall from Section 5.2 that our labeling convention is that a terminal vertex a_ℓ in $\mathcal{G}$ falls between separation-vertices σ_ℓ and $\sigma_{\ell+1}$ in $\mathcal{G}^*$.) Then the optimal forest must include at least one tree that separates this commodity. Of all such trees, consider the one with a separation vertex σ_g closest to σ_i. Let the separation-vertex in this tree closest to σ_j be σ_h. Then we must either have σ_g in $[\sigma_{i+1}, \sigma_p]$ and σ_h in $[\sigma_{p+1}, \sigma_{j-1}]$, or σ_g in $[\sigma_{p+1}, \sigma_q]$ and σ_h in $[\sigma_{q+1}, \sigma_{j-1}]$. In either case the weight of the tree is $\mathbf{T}[\sigma_g, \sigma_h]$ (see Figure 19: σ_g and σ_h show the first case and $\sigma_{g'}$ and $\sigma_{h'}$ the second). Furthermore, in either case there is no commodity with one terminal in $[a_i, a_{g-1}]$ and the other in $[a_h, a_{j-1}]$: by the selection of σ_p the mates to terminals in $[a_i, a_{p-1}]$ are in $[a_j, a_{i-1}]$, and if $\sigma_g \in [\sigma_{p+1}, \sigma_q]$ then by the non-crossing

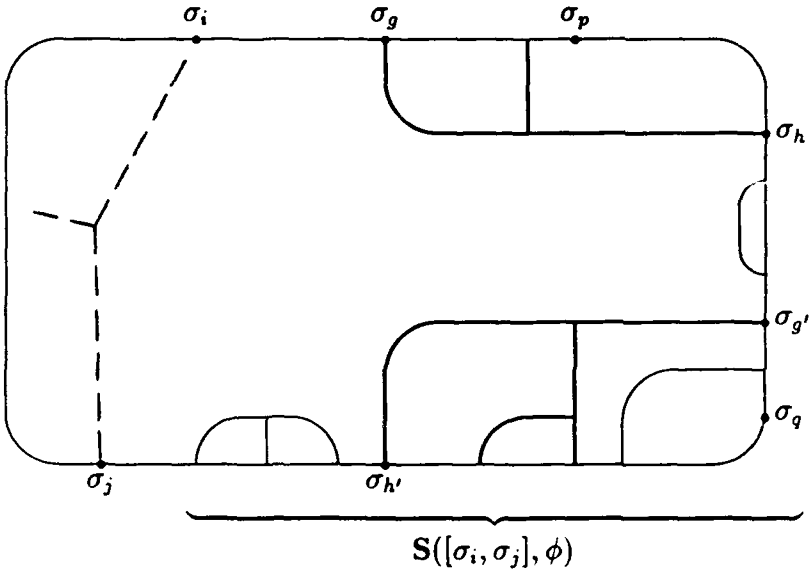


Figure 19: The **S** Function in Ue–P*BN Case (Example)

property the mates to terminals in $[a_p, a_{g-1}]$ are in $[a_p, a_q] \subseteq [a_p, a_{h-1}]$. Therefore any trees on subsets of $[\sigma_{i+1}, \sigma_{g-1}]$ can be constructed independently of trees on subsets of $[\sigma_{h+1}, \sigma_{j-1}]$. This leads to the following recursive formula for **S**:

$$\mathbf{S}([\sigma_i, \sigma_j], \phi) = \min_{(\sigma_g, \sigma_h) \in Q''} \{\mathbf{S}([\sigma_i, \sigma_g], \phi) + \mathbf{T}[\sigma_g, \sigma_h] + \mathbf{S}([\sigma_h, \sigma_j], \phi)\} \qquad (23)$$

where

$$\begin{aligned} Q'' \equiv \{ \quad (\sigma_g, \sigma_h) : \quad & \sigma_g \in [\sigma_{i+1}, \sigma_{q1}] - \{\sigma_q\}, \\ & \sigma_h \in [\sigma_{g+1}, \sigma_j] - \{\sigma_j\}, \\ & \text{at least one of } \sigma_g, \sigma_h \text{ is in } [\sigma_{i+1}, \sigma_p] \cup [\sigma_{q+1}, \sigma_j] - \{\sigma_j\} \quad \} \end{aligned}$$

and σ_p and σ_q are defined as described, being a function of σ_i and σ_j.

Using this recursive formula for **S**, we can write our polynomial algorithm for the Ue–P*BN FMCC problem. The following algorithm also makes use of the fact that in such a problem, the terminals for some commodity must have no other terminals between them on the boundary of $\mathcal{G}$. It is to be compared with Algorithm 2 on p. 54, which is for the general Ue–PB FMCC problem, but is not polynomial in k.

Algorithm 4 (Weighted Ue–P*BN FMCC)

input: $\mathcal{G}^{Dm}$ for a weighted Ue–P*BN FMCC problem, with the separation-vertices numbered so that $K_1 \cap K_z \neq \phi$

output: Weight of the minimum forest in $\mathcal{G}^{Dm}$ that corresponds to the optimal FMCC in $\mathcal{G}$

begin
for every $\nu, v \in V^{Dm}$
  find minimum distance $\mathrm{d}(\nu, v)$
for every $\nu \in V^{Dm}$ and every separation-vertex σ_i
  set $\mathbf{C}([\sigma_i, \sigma_i], \nu) = \mathrm{d}(\sigma_i, \nu)$
  set $\mathbf{T}[\sigma_i, \sigma_i] = 0$
for $i = 1, \ldots, z-1$ (size of separation-vertex set, minus one)
  for $j = 1, \ldots, z$ (start of set)
    set $\ell = (j+i) \bmod z$
    compute $\mathbf{S}([\sigma_j, \sigma_\ell], \phi)$ using (21)–(23)
    for every $\nu \in V^{Dm}$
      compute $\mathbf{D}([\sigma_j, \sigma_\ell], \nu)$ using (20)
      compute $\mathbf{B}([\sigma_j, \sigma_\ell], \nu)$ using (17)
      compute $\mathbf{C}([\sigma_j, \sigma_\ell], \nu)$ using (19)
    set $\mathbf{T}[\sigma_j, \sigma_\ell] = \mathbf{B}([\sigma_j, \sigma_\ell], \sigma_j)$
compute $W_{\text{opt}} = \min_{i=2\ldots z} \{\mathbf{T}[\sigma_1, \sigma_i] + \mathbf{S}([\sigma_i, \sigma_1], \phi)\}$
return W_{opt}
end

The complexity of this algorithm is $O(\alpha + k^4 + fk^3 + f^2k^2)$, where again α is the bound on finding all the distances $\mathrm{d}(\nu, v)$ and f is the number of vertices in $\mathcal{G}^{Dm}$. It also provides the proof for the following proposition:

Proposition 12 *The weighted T-planar full multicommodity edge-cut problem is polynomially bounded if the terminals are restricted to be in non-crossing order.*

Proof: Algorithm 4 provides a polynomially-bounded procedure. ∎

An improvement is possible if the terminals are also in non-nested order, *i.e.* the source and sink are adjacent for every commodity. This means that the terminals can be relabeled so that they occur in order $s_1, t_1, s_2, t_2, \ldots, s_k, t_k$ around the boundary of $\mathcal{G}$. In this case, no optimal separating tree in $\mathcal{G}^{Dm}$ will contain more than one vertex of degree three or more. This is because the path joining two such vertices could not serve to separate a commodity, and so would not be part of a minimal tree (see Figure 20). Also, there must be a new tree on at least every other separation-vertex,

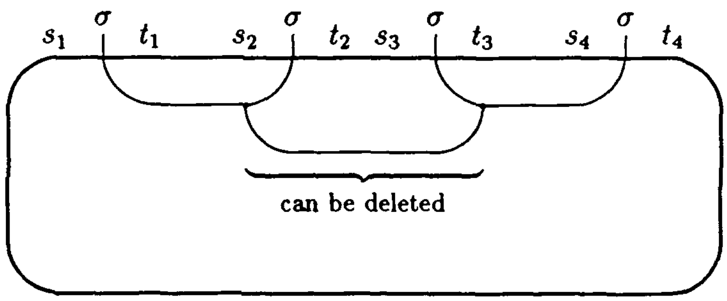


Figure 20: A Non-optimal Configuration in the Non-nested Case

since comm$[\sigma_i, \sigma_{i+3}]$ cannot be empty for any i.

Since all trees in the minimum separating forest have this simple structure, we can adapt Algorithm 4 by dispensing with the **C** and **D** functions and writing new recursive formulas for **S**, **T**, and **B**. The new formulas are:

$$
\mathbf{S}^{nn}[\sigma_i, \sigma_j] = \begin{cases} 0 & \text{if } j = i \text{ or } j = i+1 \\ 0 & \text{if } j = i+2 \text{ and } K_i \cap K_{i+1} = \phi \\ \infty & \text{if } j = i+2 \text{ and } K_i \cap K_{i+1} \neq \phi \\ \min_{(\sigma_p, \sigma_q) \in Q^{nn}} \{\mathbf{T}^{nn}[\sigma_p, \sigma_q] + \mathbf{S}^{nn}[\sigma_q, \sigma_j]\} & \text{otherwise} \end{cases} \tag{24}
$$

where

$$
Q^{nn} \equiv \{(\sigma_p, \sigma_q) : \sigma_p \in \{\sigma_{i+1}, \sigma_{i+2}\}, \sigma_q \in [\sigma_{p+1}, \sigma_{j-1}]\}
$$

$$
\mathbf{T}^{nn}[\sigma_i, \sigma_j] = \begin{cases} 0 & \text{if } i = j \\ \min_{\nu \in V^{Dm}} \mathbf{B}^{nn}([\sigma_i, \sigma_j], \nu) & \text{otherwise} \end{cases} \tag{25}
$$

$$
\mathbf{B}^{nn}([\sigma_i, \sigma_j], \nu) = \begin{cases} \mathrm{d}(\nu, \sigma_i) & \text{if } i = j \\ \mathrm{d}(\nu, \sigma_i) + \min_{\sigma_p \in [\sigma_{i+1}, \sigma_j]} \{\mathbf{S}^{nn}[\sigma_i, \sigma_p] + \mathbf{B}^{nn}([\sigma_p, \sigma_j], \nu)\} & \\ & \text{otherwise} \end{cases} \tag{26}
$$

These lead to the following algorithm, which is to be compared with Algorithm 2 and Algorithm 4.

Algorithm 5 (Weighted Ue–P*BN FMCC, Non-nested Terminals)

input: $\mathcal{G}^{Dm}$ for a weighted Ue–P*BN FMCC problem with non-nested terminals and the separation-vertices numbered so that $K_1 \cap K_z \neq \phi$

output: weight of a minimum forest in $\mathcal{G}^{Dm}$ that corresponds to an optimal FMCC in $\mathcal{G}$

begin

for every $\nu, \upsilon \in V^{Dm}$

  find minimum distance $\mathrm{d}(\nu, \upsilon)$

for every $\nu \in V^{Dm}$ and every separation-vertex σ_i

  set $\mathbf{B}^{nn}([\sigma_i, \sigma_i], \nu) = \mathrm{d}(\sigma_i, \nu)$

  set $\mathbf{T}^{nn}[\sigma_i, \sigma_i] = 0$

  set $\mathbf{S}^{nn}[\sigma_i, \sigma_i] = 0$

for $i = 1, \ldots, z-1$ (size of separation-vertex set, minus one)

  for $j = 1, \ldots, z$ (start of set)

    set $\ell = (j + i) \bmod z$

    compute $\mathbf{S}^{nn}[\sigma_j, \sigma_\ell]$ using (24)

    for every $\nu \in V^{Dm}$

      compute $\mathbf{B}^{nn}([\sigma_j, \sigma_\ell], \nu)$ using (26)

    compute $\mathbf{T}^{nn}[\sigma_j, \sigma_\ell]$ using (25)

compute $W^{nn}_{\text{opt}} = \min_{i=2 \ldots z} \{\mathbf{T}^{nn}[\sigma_1, \sigma_i] + \mathbf{S}^{nn}[\sigma_i, \sigma_1]\}$

return W^{nn}_{opt}

end

The complexity here is $O(\alpha + nk^3)$.

6 PARTIAL MULTICOMMODITY CUTS IN T-PLANAR GRAPHS

This chapter examines the problem if finding partial multicommodity cuts in T-planar graphs. Results are found and algorithms described for three special cases when the terminals are in non-crossing order (UeEP*BN). These are the cases in which $r = k - 1$ (Section 6.2), where $r = k - 2$ (Section 6.3), and where $r = 1$ (Section 6.4). Unlike the previous chapter, only unweighted problems will be considered.

It was noted in Chapter 4 that a partial multicommodity cut problem cannot be easier than the corresponding disjoint paths problem, and that consequently many restricted PMCC problems are NP-hard (Proposition 4). Based on the results of the last chapter (Proposition 10) we can add that the partial multicommodity cut problem is NP-hard even when restricted to planar grids with all terminals on the boundary. Clearly the search for good PMCC algorithms should begin with graph configurations in which the disjoint paths problem is well-characterized. Furthermore, the characteristics of the configurations should persist when edges are deleted from the graph. This makes the many DJP results for Eulerian graphs much less useful. However, Section 6.1 will show that the DJP problem can be characterized for the case of T-planar graphs with the terminals in non-crossing order—the U–EP*BN case. Also, Ue–P*BN is the most general FMCC case for which there exists a polynomial algorithm (Alg. 4). Therefore we consider the class UeEP*BN for partial multicommodity cut algorithms. The next section gives the necessary characterization of disjoint paths in T-planar non-crossing graphs.

6.1 Disjoint Paths in Non-crossing T-Planar Graphs

The following proposition shows that in an undirected graph with all terminals on the boundary in non-crossing order (U–EP*BN), the disjoint paths construction problem has a solution if and only if the general cut condition (GCC; defined on p. 23) is satisfied. This result has been noted before, but it seems that a proof has never been published. Schrijver [n.d.] suggests proving it by reducing it to an instance of the corresponding vertex-disjoint paths problem. What follows is a different proof, which uses a recently-published result by Frank [1990].

Frank showed that in a U–EP* DJP problem in which each commodity has its terminals on one of two faces, the GCC plus an intersection criterion (IC) is necessary and sufficient for the existence of a solution. (This result was reviewed in Chapter 3, p. 25.) For any $X \subset V$, let $s(X)$ be the surplus (defined on p. 24) of $\Gamma(X)$. The IC requires that for any two subsets X and Y of V, if $s(X) = s(Y) = 0$, then $s(X \cap Y)$ must be even. We will need the following lemma to apply this result to the more restricted U–EP*BN case.

Lemma 5 *In a U-EP*BN problem, if the GCC is satisfied but the IC violated, then the IC is violated for some vertex sets* X *and* Y *such that* $X \cap Y$, $X \backslash Y$, $V \backslash (X \cup Y)$, *and* $Y \backslash X$ *all induce simple cuts, and their intersections with the boundary of* $\mathcal{G}$ *occur* *in the given order (either clockwise or counterclockwise).*

Proof: If the IC is violated, then it is violated for some X, Y that induce simple cuts $\Gamma(X)$ and $\Gamma(Y)$ [Frank, 1990, Lemma 2]. These simple cuts correspond to paths in the modified dual graph $\mathcal{G}^{Dm}$ (defined in Section 5.1); the paths run from separation-vertex to separation-vertex across $\mathcal{G}$. Let X and Y be chosen so that the number of common edges in the two paths is maximized. Then the union of the two paths must be a tree. If the union contains a circuit then the number of common edges can be increased by routing both cuts along one side of the circuit. If the union is disconnected then we have $X \subseteq Y$, $Y \subseteq X$, or $X \cap Y = \phi$, and $s(X \cap Y)$ cannot be odd. For this same reason the two paths must cross, so that the union of the two looks like Figure 21a or Figure 21b. So the sets $X \cap Y$, $X \backslash Y$, $V \backslash (X \cup Y)$, and $Y \backslash X$

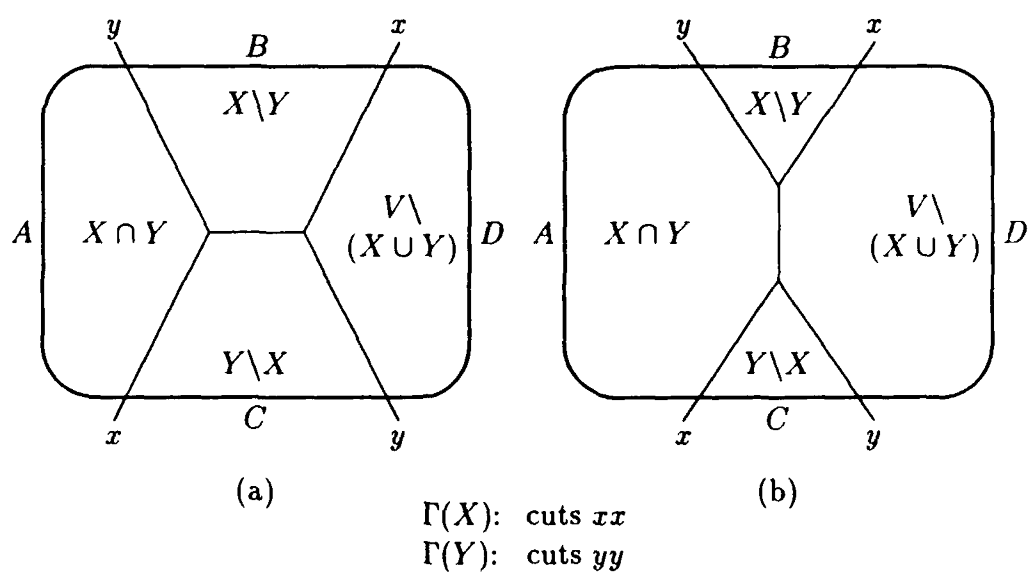


Figure 21: Proof of Lemma 5

are nonempty and in that order around the boundary of $\mathcal{G}$. ∎

Proposition 13 *In an edge-disjoint paths construction problem in a T-planar graph with terminals in non-crossing order, the general cut condition is necessary and sufficient for the existence of a solution.*

Proof: The method is to show that if all the terminals are on exactly one face and in non-crossing order, the GCC implies the IC. The proposition then follows immediately from Frank's result. Suppose that the GCC is satisfied in some U-EP*BN graph $\mathcal{G}$ but the IC is violated. Then by Lemma 5 it must be violated for some vertex sets X and Y whose induced simple cuts divide $\mathcal{G}$ into four parts as described. Let $A \equiv X \cap Y$, $B \equiv X \backslash Y$, $C \equiv Y \backslash X$, and $D \equiv V \backslash (X \cap Y)$, as in Figure 21. Then $s(A)$ is odd, and $s(A) \geq 1$. For $P, Q \in \{A, B, C, D\}$, let $g(P, Q)$ be the number of edges in $\mathcal{G}$ with one endvertex in P and the other in Q, let $h(P, Q)$ be the number of commodities with one terminal in P and the other in Q, and let $s(P, Q) = g(P, Q) - h(P, Q)$.

The crucial observation is that because of planarity and the non-crossing order of the terminals, at least one of $g(A, D)$ and $g(B, C)$ must be zero, and at least one of $h(A, D)$ and $h(B, C)$ must be zero. From Figure 21, it is clear that in configuration (a) there are edges of $\mathcal{G}$ between B and C and in (b) between A and D, but never between both pairs in the same graph, giving the result for g. The non-crossing order of the terminals forbids commodities being shared between A and D and between B and C at the same time, giving the result for h. This observation gives rise to three cases.

Case 1: $h(A, D) = g(A, D) = 0$. From the fact that $s(X) = 0$, we get

$$s(A + B) = 0 \quad (27)$$

$$g(A,C) + g(B,C) + g(B,D) - h(A,C) - h(B,C) - h(B,D) = 0 \quad (28)$$

$$s(A,C) + s(B,C) + s(B,D) = 0 \quad (29)$$

From the fact that $s(Y) = 0$, we get

$$s(A + C) = 0 \quad (30)$$

$$g(A,B) + g(B,C) + g(C,D) - h(A,B) - h(B,C) - h(C,D) = 0 \quad (31)$$

$$s(A,B) + s(B,C) + s(C,D) = 0 \quad (32)$$

Combining (29) and (32), we get

$$s(B) + s(C) = 0 \implies s(B) = s(C) = 0 \quad (33)$$

$s(B) = 0 \implies$

$$s(A,B) + s(B,C) + s(B,D) = 0 \quad (34)$$

Combining (29) and (34),

$$s(A,C) = s(A,B) \quad (35)$$

But $s(X \cap Y) = s(A) = s(A,C) + s(A,B)$, so this implies

$$s(X \cap Y) = 2s(A,C) \quad (36)$$

contradicting the assumption that it is odd.

Case 2: $h(A, D) = g(B, C) = 0$. From the fact that $s(X) = 0$, we get

$$s(A + B) = 0 \quad (37)$$

$$g(A,C)+g(A,D)+g(B,D)-h(A,C)-h(B,C)-h(B,D) = 0 \quad (38)$$

$$s(A,C)+s(B,D)+g(A,D)-h(B,C) = 0 \quad (39)$$

From the fact that $s(Y)=0$, we get

$$g(A,B)+g(A,D)+g(C,D)-h(A,B)-h(B,C)-h(C,D) = 0 \quad (40)$$

$$s(A,B)+s(C,D)+g(A,D)-h(B,C) = 0 \quad (41)$$

Combining (39) and (41), we get

$$\begin{aligned} s(B,D)+s(A,B)-h(B,C)+s(A,C)+s(C,D)-h(B,C) &= -2g(A,D) \\ s(B)+s(C) &= -2g(A,D) \\ s(B)+s(C) &\le 0 \qquad (42) \end{aligned}$$

But $s(B)+s(C)\geq 0$, so we must have $g(A,D)=0$, and this reduces to Case 1.

Case 3: $h(B,C)=0$.

$$s(X)=0 \Rightarrow s(A,C)+s(B,D)+g(B,C)+s(A,D) = 0 \quad (43)$$

$$s(Y)=0 \Rightarrow s(A,B)+s(C,D)+g(B,C)+s(A,D) = 0 \quad (44)$$

$$-s(A)\leq -1 \Rightarrow -s(A,B)-s(A,C)-s(A,D) < 0 \quad (45)$$

$$-s(D)\leq 0 \Rightarrow -s(B,D)-s(C,D)-s(A,D) \leq 0 \quad (46)$$

Adding the above, we get $2g(B,C)<0$, a contradiction.

All cases are accounted for, so the proposition is proved. ■

6.2 The Case of $r=k-1$

We have seen that the general cut condition is a necessary and sufficient condition for the existence of a full set of disjoint paths in a T-planar graph in non-crossing order. The smallest removal of edges that produces an oversaturated cut is therefore the optimal $r = k - 1$ interjacent set. It is easy to find such a set by evaluating the $O(k^2)$ minimum cuts that separate a set of consecutive terminals from their complement in T. It is only necessary to consider simple cuts of this sort, since if the GCC is violated for any non-simple cut it must be violated by a simple cut that is a subset of the cut, and if it is violated for a simple cut then it is violated at least as much for the minimum edge-cut that partitions the terminals in the same way. For any such minimum cut that has the smallest surplus of all such cuts, an optimal interjacent set can be made up of any surplus-plus-one edges in the cut. This suggests the following algorithm:

Algorithm 6 (UeEP*BN $r = k - 1$ PMCC)

input: $\mathcal{G}$ for a UeEP*BN problem with $r = k - 1$
output: cardinality of a minimum interjacent set
begin
set $J_{\text{opt}} = \infty$
for $i = 1$ to $\lfloor |T|/2 \rfloor$
    for every set S of i consecutive terminals in T
        set p equal to the number of unmatched terminals in S
        find the minimum cut C between S and $T - S$
        set $J_{\text{opt}} = \min \{ J_{\text{opt}} , |C| - p + 1 \}$
return J_{opt}
end

The complexity of the algorithm is $O(\gamma k^2)$, where γ is the complexity of finding an edge-cut in a planar graph. We also have the following theorem.

Theorem 1 *The $r = k - 1$ partial multicommodity edge-cut problem with edge-disjoint remaining paths in a T-planar non-crossing graph (UeEP*BN) is polynomially bounded.*

Proof: Algorithm 6 gives a polynomial procedure for this problem. ■

In many cases a similar procedure will help find the optimal PMCC for $r < k - 1$. Let a *smallest-surplus cut* in $\mathcal{G}$ be a cut that has a surplus as small or smaller than any other in the graph. If there is a smallest-surplus cut that separates two or more commodities, any surplus-plus-two edges in the cut will be a feasible $r = k - 2$ interjacent set. From Lemma 3 in Chapter 4 we can also conclude that the interjacent set is optimal. This can be repeated until the smallest-surplus cut with the largest cardinality is exhausted. This leads to the following proposition.

Proposition 14 *In a T-planar partial multicommodity edge-cut problem with the terminals in non-crossing order, suppose that C is a simple cut that contains p edges and separates q commodities, and suppose that no cut in $\mathcal{G}$ has a smaller surplus than C. Suppose further that $k - 1 \geq r \geq k - q$. Then any $p - q + k - r$ edges in C are an optimal interjacent set.*

Proof: An interjacent set as described is clearly feasible, for it prevents $k - r$ of the q commodities separated by C from being connected by disjoint paths. Optimality has already been shown for the case $r = k - 1$. Lemma 3 implies that if a new feasible interjacent set can be found with just one additional edge when r is decremented, then that set must be optimal. By induction, each addition of a new edge from C

to the interjacent set results in a new optimal set for a value of r one smaller, until every edge in C is in the set. ■

However, when $r < k-1$ this proposition will only help us when there is a smallest-surplus cut that separates at least $k-r$ commodities. The general problem becomes significantly more complicated, as we will see in the next section.

6.3 The Case of $r = k - 2$

Let J^* be an optimal interjacent set for a PMCC problem with $r = k - 2$ in a graph $\mathcal{G}$ in which a full set of k disjoint paths can be found. For any $e \in J^*$, the set $\{e\}$ has the property that exactly $k-1$ disjoint paths can be found in $\mathcal{G} - (J^* - \{e\})$. Let J_2 be a maximum-cardinality subset of J^* such that exactly $k-1$ disjoint paths can be found in $\mathcal{G} - (J^* - J_2)$. Then for any $f \in J^* - J_2$, a full set of k paths can be found in $\mathcal{G} - (J^* - (J_2 + \{f\}))$. Let $J_1 \equiv J^* - J_2$, so that J_1 and J_2 are a partition of J^*. If the edges of J_1 are removed from $\mathcal{G}$ one at a time, k disjoint paths are possible until the last edge is removed. J_1 is therefore a minimal PMCC for $r = k - 1$. If the edges of J_2 are then removed, $k - 1$ paths are possible until the removal of the last edge of J_2 (and of J^*).

If we are somehow given the correct J_1, we might expect it to be fairly easy to find J_2 and complete the interjacent set. The method of finding J^* will be to consider the forms that $\mathcal{G} - J_1$ might take. It will turn out that J_2 is indeed easy to find, and furthermore its relationship to J_1 is such that J_1 is also easy to find.

The disjoint paths construction problem has no solution in $\mathcal{G} - J_1$, so the general cut condition must be violated somewhere in it (Proposition 13). However, $k - 1$ paths can be drawn in $\mathcal{G} - J_1$, so it can have no cut with a surplus of -2 or less. Therefore there are one or more smallest-surplus cuts in $\mathcal{G} - J_1$ with a surplus of -1. We can consider two cases: either there is such a negative-surplus cut that contains an edge of $\mathcal{G} - J_1$, or all negative-surplus cuts are determined by components of $\mathcal{G} - J_1$.

Case 1: $\mathcal{G} - J_1$ has a negative-surplus cut that contains an edge not in J_1.

Removing any edge from such a cut will decrease its surplus to -2. Any such edge can then be the sole member of J_2. This means that $J_1 + J_2 = J^*$ is the smallest set of edges whose removal reduces the surplus of some cut in $\mathcal{G}$ to -2. Such a cut can be found by examining all the simple cuts in $\mathcal{G}$ that separate at least two commodities and picking one that has the smallest surplus. J^* is then any surplus-plus-two edges from that cut.

Case 2: All negative-surplus cuts in $\mathcal{G} - J_1$ contain only edges in J_1.

From the minimality of J_1 and the connectedness of $\mathcal{G}$ we know that $\mathcal{G} - J_1$ has exactly two components and that only one commodity has its terminals in separate components. J_1 must therefore be a simple cut in $\mathcal{G}$ that separates exactly one commodity. Consider the problem of finding J_2 once J_1 is given. The commodity separated by J_1 is irrelevant and can be ignored. What is left is a PMCC problem of the form $r = k - 1$, applied to each component of $\mathcal{G} - J_1$. By Proposition 14, J_2 is any subset consisting of surplus-plus-one edges of any cut that in turn has the smallest surplus of a cut in either component. Furthermore, the following two lemmas show that J_2 is almost independent of J_1.

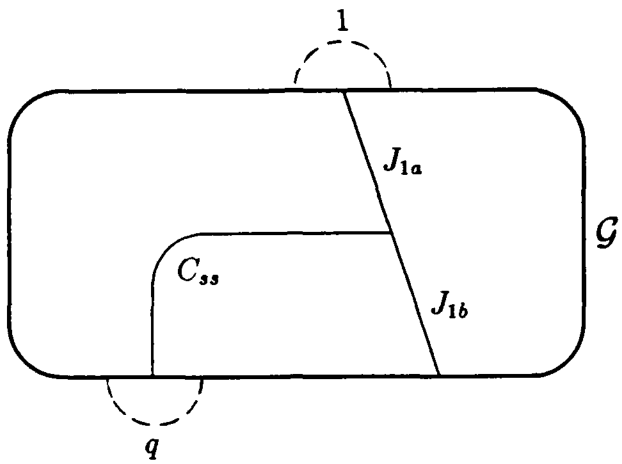


Figure 22: Proof of Lemma 6

Lemma 6 *If $J_1 \subset J^*$ is a simple cut in $\mathcal{G}$ that separates exactly one commodity, then any smallest-surplus cut in $\mathcal{G} - J_1$ is also a full cut in $\mathcal{G}$.*

Proof: Suppose that the smallest-surplus cut in $\mathcal{G} - J_1$ is C_{ss}, containing p edges and separating q commodities, but C_{ss} is not a cut in $\mathcal{G}$. Then C_{ss} must run from the boundary of $\mathcal{G}$ to the cut J_1, inducing a proper partition of J_1 into J_{1a} and J_{1b}, as in Figure 22. Let $|J_{1a}| = a$ and $|J_{1b}| = b$. Then the cardinality of the complete $r = k - 2$ interjacent set will be $a + b + p - q + 1$. But the commodity separated by J_1 must also be separated by either $C_{ss} + J_{1a}$ or $C_{ss} + J_{1b}$; let us say the former. Then $C_{ss} + J_{1a}$ contains $p + a$ edges and separates $q + 1 \geq 2$ commodities, so a better interjacent set can be formed from any $a + p - q + 1$ of those edges. This contradicts the assumption that J_1 is a subset of the optimal interjacent set. ■

Lemma 7 *If J_1 is a simple cut in $\mathcal{G}$ that separates exactly one commodity, then the smallest-surplus cut in $\mathcal{G} - J_1$ does not separate (in $\mathcal{G}$) the commodity separated by J_1.*

Proof: Suppose it does separate that commodity. Let the cut be C_{ss}, let $|C_{ss}| = p$, let $|J_1| = j$, and let C_{ss} separate q commodities in $\mathcal{G} - J_1$. Then the size of the optimal interjacent set is $j+p-q+1$. But C_{ss} by itself separates $q+1$ commodities in $\mathcal{G}$, so any $p-q+1$ edges from it form a smaller interjacent set than the optimal. ∎

It is evident from these lemmas that the best interjacent set in the form of Case 2 can be constructed by examining the minimum cuts C in $\mathcal{G}$ that separate exactly one commodity. Each one may be combined with edges from a smallest-surplus cut in $\mathcal{G}-C$ to form a candidate interjacent set. If the optimal solution is of this form, it will be one of these candidates.

We can conclude that the following algorithm is guaranteed to find an optimal solution to any UeEP*BN PMCC problem with $r=k-2$.

Algorithm 7 (UeEP*BN $r=k-2$ PMCC)

input: $\mathcal{G}$ for a UeEP*BN problem with $r=k-2$
output: cardinality of a minimum interjacent set
begin

(1) Find the surpluses of every minimum cut that separates a set of consecutive terminals from its complement in T.

(2) Examine all cuts with the smallest surplus s_1. If any one of them separates two or more commodities, then $J_{\text{opt}} = s_1 + 2$. Go to Step *5*.

(3) Otherwise:

- *(3a)* Find the smallest surplus s_2 among all cuts separating two or more commodities. Set $J_2 = s_2 + 2$.
- *(3b)* For each cut C_i separating exactly one commodity, find the cut with the smallest surplus s_i that separates only commodities on one side of C_i, and set $J_{C_i} = |C_i| + s_i + 1$. Set $J_3 = \min_i\{J_{C_i}\}$.

(4) set $J_{\text{opt}} = \min\{J_2, J_3\}$

(5) **return** J_{opt}

end

The main work in this algorithm is in Step *1*, where minimum cuts must be found for each partition of T into two consecutive sets. These cuts are equivalent to shortest paths between separation vertices in $\mathcal{G}^{Dm}$ (a planar graph). There are $O(k^2)$ such shortest paths, and if each can be found in $O(\gamma)$, then the complexity of the algorithm is $O(\gamma k^2)$. Alternatively, if the all-pairs shortest paths problem is $O(\alpha)$, this algorithm is also $O(\alpha)$. We also have the following theorem.

Theorem 2 *The $r = k - 2$ partial multicommodity edge-cut problem with edge disjoint remaining paths in a T-planar non-crossing graph (UeEP*BN) is polynomially bounded.*

Proof: Algorithm 7 gives a polynomial procedure for this problem. ■

Unfortunately, this method does not generalize easily to the case in which $r \leq k-3$. The optimal interjacent set J^* can be partitioned into J_3, J_2, and J_1, where J_1 is a minimal interjacent set for $r = k-1$ and $J_1 + J_2$ is minimal for $r = k-2$, but there is no good characterization for $\mathcal{G} - (J_1 + J_2)$, a graph in which all but two commodities can be disjointly connected. Such a characterization needs to be developed before it will be possible to show a minimal way to achieve it.

However, it is possible to solve a UeEP*BN problem for other values of r when r is small. This is the subject of the next section.

6.4 The Case of $r = 1$

Suppose a set of edges J is removed from a T-planar non-crossing multicommodity flow graph $\mathcal{G}$, and suppose that in $\mathcal{R} = \mathcal{G} - J$ only one disjoint s_i–t_i path exists. $\mathcal{R}$ may be made up of any number of components, but all save one of them will have no s_i–t_i paths at all. The remaining component $\mathcal{R}'$ may have many s_i–t_i paths, but no two can be disjoint for different values of i. Those commodities that have both terminals in that component will have them in non-crossing order on its boundary, since that is how the terminals are distributed in $\mathcal{G}$ and the removal of edges will not change this. In order to characterize such components, then, we need to know the properties of a U--P*BN multicommodity flow graph in which many components may be present, but only one disjoint path at a time can be drawn for them. Such a graph will be called a *one-disjoint-path* graph.

A graph can be a one-disjoint-path graph because only one commodity has both terminals in it. Another possibility is that multiple commodities are present, but a single edge lies on every s_i–t_i path, preventing more than one disjoint path. This edge will be a bridge connecting two components, each of which contains one terminal for each commodity. (The edge cannot be in any circuit because the circuit would provide an alternate s_i–t_i path.) However, these two possibilities do not account for all one-disjoint-path graphs, as Figure 23 shows. The graph in this figure has three edges such that every s_i–t_i path contains two of them, so that any two of these edges constitute an FMCC for the graph. Such edges will be called the three *critical edges* of a one-disjoint-path graph. The three possibilities mentioned above characterize all one-disjoint-path graphs.

Theorem 3 *A connected T-planar multicommodity flow graph $\mathcal{R}'$ with its terminals in non-crossing order is a one-disjoint-path graph if and only if at least one of the following statements is true:*

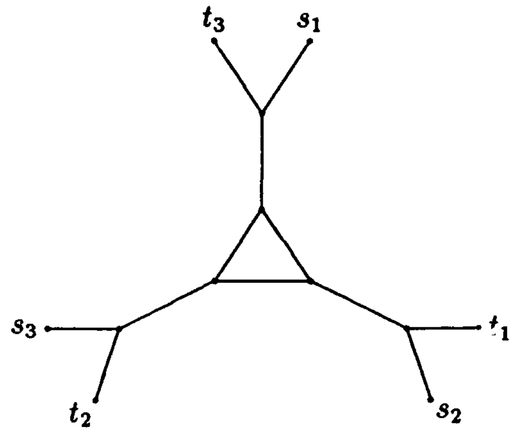


Figure 23: A U--P*BN One-disjoint-path Graph

(a) There is only one commodity in $\mathcal{R}'$.

(b) There is a single edge in $\mathcal{R}'$ that is on every s_i–t_i path.

(c) $\mathcal{R}'$ contains three edges such that every s_i–t_i path contains two of them.

Before proving the theorem, we prove two auxiliary lemmas. Consider $\mathcal{R}'$ as made up of two-edge-connected components $\mathcal{C}_1, \mathcal{C}_2, \ldots$ linked by edges $e_1, e_2, \ldots$. If each component $\mathcal{C}_j$ is contracted to a single vertex, $\mathcal{R}'$ becomes a tree $\mathcal{R}^T$ with edge set $\{e_p : p = 1, 2, \ldots\}$; all terminals in one component $\mathcal{C}_j$ become coincident in $\mathcal{R}^T$. Suppose $\mathcal{R}'$ is a one-disjoint-path graph in which (a) and (b) are false. The following two lemmas will help describe $\mathcal{R}'$.

Lemma 8 *In $\mathcal{R}'$, no commodity has both terminals on any one two-edge-connected component $\mathcal{C}_j$ of $\mathcal{R}'$*

Proof: Suppose s_i and t_i are both on $\mathcal{C}_j$. Then these terminals partition the boundary of $\mathcal{C}_j$ into two disjoint s_i–t_i paths. For any other pair of terminals s_ℓ and t_ℓ, either the terminals themselves or the endpoints of the bridges e_p leading to those terminals must both lie on the same s_i–t_i path. This is because the terminals are in non-crossing order. Therefore there is an s_ℓ–t_ℓ path that uses only a portion of C_j that is a subset of one of the two s_i–t_i paths in C_j's boundary. This will be entirely disjoint from the other s_i–t_i path, a contradiction. ■

Lemma 9 *In $\mathcal{R}'$, every two s_i–t_i paths for distinct commodities share at least one bridge edge e_p.*

Proof: Suppose paths P_i (from s_i to t_i) and P_ℓ (from s_ℓ to t_ℓ) share no bridge edges. Then they must share an edge in a two-edge-connected component C_i. Around the boundary of C_i there must be four vertices (not necessarily all distinct), each of which is either one of the terminals s_i, t_i, s_ℓ, t_ℓ, or the endvertex of a unique bridge edge leading to one of these terminals. By cross-freeness, the four vertices must be in non-crossing order when a bridge edge endvertex is regarded as representing the terminal to which it leads. The boundary of C_i will provide disjoint paths linking s_i to t_i and s_ℓ to t_ℓ. This contradicts the supposition that $\mathcal{R}'$ is a one-disjoint-path graph. ■

Proof of Theorem: Clearly the truth of (a), (b), and/or (c) implies that $\mathcal{R}'$ is a one-disjoint-path graph. The proof is completed by showing that if $\mathcal{R}'$ is such a graph and (a) and (b) are false, then (c) must be true.

In view of the two lemmas, if $\mathcal{R}'$ is a one-disjoint-path graph with (a) and (b) false, then so is the tree $\mathcal{R}^T$ formed by contracting all its two-edge-connected components. Every s_i–t_i path in $\mathcal{R}'$ has a corresponding path in $\mathcal{R}^T$. Any two paths that are disjoint in $\mathcal{R}'$ will be disjoint in $\mathcal{R}^T$, and any two paths for distinct commodities that share an edge in $\mathcal{R}'$ will share an edge in $\mathcal{R}^T$. So if any s_i–t_i path in $\mathcal{R}'$ shares an edge with every other s_ℓ–t_ℓ path for $\ell \neq i$, then the corresponding paths will have the same property in $\mathcal{R}^T$, and *vice versa.*

Suppose in $\mathcal{R}^T$ no one edge covers all paths. Then $\mathcal{R}^T$ must have three or more commodities. This is because in a tree there is only one path between two vertices, and if $\mathcal{R}^T$ has only two commodities and they have non-disjoint s_i–t_i paths they must overlap somewhere. Let P_1, P_2, and P_3 be three s_i–t_i paths for distinct commodities that share no one edge. Let the first and last edges in $P_1 \cap P_2$ in the s_1-to-t_1 direction be e' and e, and let the first and last edges in $P_1 \cap P_3$ be f and f'. By supposition, $P_1 \cap P_2 \cap P_3 = \phi$. Nevertheless P_2 and P_3 must intersect, and they must do so without forming a circuit in $\mathcal{R}^T$. This can only happen if either e and f or e' and f' share a vertex v, and P_2 and P_3 share a third edge g that is also on v. Without loss of generality, let us say that e and f are adjacent, as in Figure 24. Every s_i–t_i path in

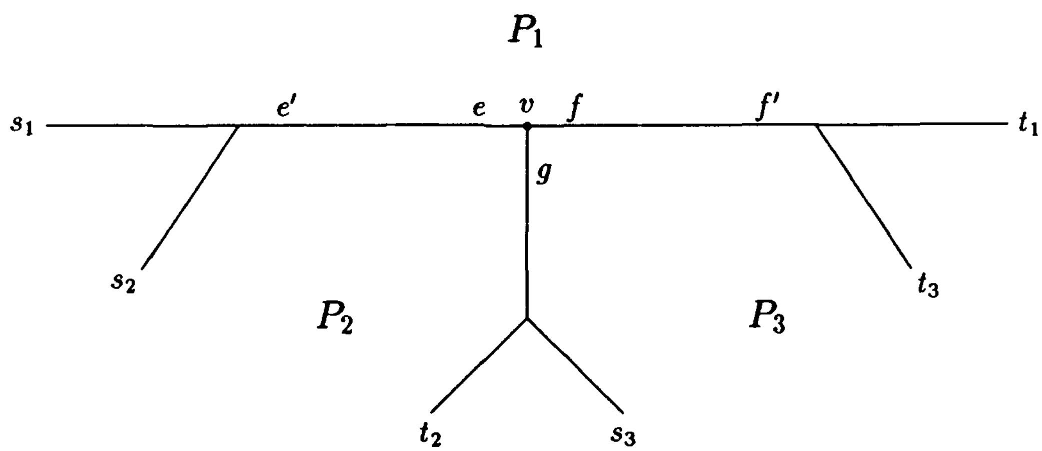


Figure 24: Three Non-disjoint Paths in $\mathcal{R}^T$

$\mathcal{R}^T$ uses exactly two of e, f, and g. For no path can use all three, and if a path uses only one, it must be disjoint from the one of P_1, P_2, P_3 that uses only the other two.

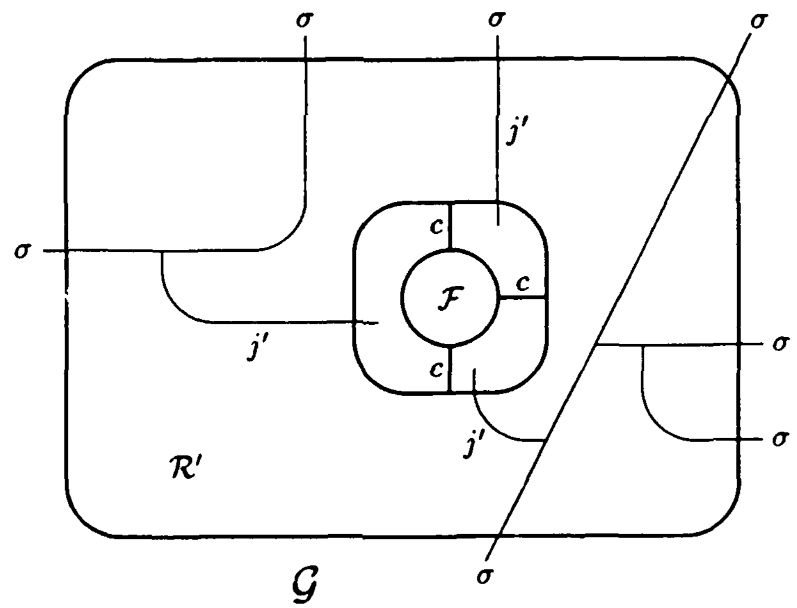
If a path uses none of e, f, and g, then it must be confined to at most one of the three branches from v that start with these edges, so it will also be disjoint from one of P_1, P_2, P_3.

Therefore every s_i–t_i path in $\mathcal{R}^T$ contains two of e, f, and g, which correspond to three bridge edges of $\mathcal{R}'$. Since every s_i–t_i path in $\mathcal{R}'$ goes through the same bridge edges as the corresponding path in $\mathcal{R}^T$, these three edges in $\mathcal{R}'$ cover all s_i–t_i paths, with each path containing at least two of them. This completes the proof of the therorem. ■

If a one-disjoint-path graph $\mathcal{R}'$ is in form (c), we can further assume that no terminals are in the two-edge-connected component of $\mathcal{R}'$ that includes endvertices of all the critical edges. In $\mathcal{R}^T$, such a terminal would be the common endvertex v of the critical edges e, f, and g, and the terminal's mate would have to be on one of the branches from v, so that the path to it would be disjoint from at least one of P_1, P_2, P_3 (see Figure 24).

We now have a complete description of the forms $\mathcal{R}$ might take. If J is optimal, it has exactly one component with s_i–t_i paths, and that component either contains both terminals for only one commodity, contains a bridge that separates all its commodities, or contains three edges two of which are on all paths. Let us call these the *one-commodity*, *bottleneck*, and *trifid* forms, respectively. If $\mathcal{R}$ has more than one component, then the one component containing s_i–t_i paths can also be described by these terms. If $\mathcal{R}$ is in one-commodity form when J is optimal, then J can be found by solving an FMCC problem in $\mathcal{G}$ with the commodity in question deleted. If $\mathcal{R}$ is in bottleneck form, then J can be found by solving an FMCC problem with the bridge edge removed. We need a method for finding J if $\mathcal{R}$ is in trifid form.

Let us suppose that for some T-planar non-crossing graph $\mathcal{G}$ the optimal $r = 1$ interjacent set J leaves $\mathcal{R}$ in trifid form. This means that $\mathcal{R}$ has a component $\mathcal{R}'$ that contains three edges, two of which are on every s_i–t_i path in $\mathcal{R}'$. Let J' be the subset of J consisting of edges with both endvertices in $\mathcal{R}'$. The the role of $J - J'$ is to separate $\mathcal{R}'$ from the rest of $\mathcal{G}$ (and possibly to subdivide the latter), while J' partially divides $\mathcal{R}'$ to make it a trifid one-disjoint-path graph. Call J' a *trifid cut* in $\mathcal{R}'$ (see Figure 25).



—	=	trifid partial multicommodity cut J
c	=	critical edge
j'	=	part of trifid cut J' (one of three)

Figure 25: Trifid Cut as Part of a PMCC

Let us suppose that $\mathcal{R}$ consists of a single trifid component $\mathcal{R}'$, so that $J \equiv J'$.

Let J_{+3} be J plus the three critical edges. Clearly J_{+3} divides $\mathcal{G}$ into exactly four components, three of which ($\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{C}_3$) contain terminals and one of which ($\mathcal{F}$) does not (see Figure 26). The fourth component is the two-edge-connected component in $\mathcal{R}'$ with an endvertex of each critical edge. Let J_{+3}^{Dm} be the edges corresponding to J_{+3} in the modified dual graph $\mathcal{G}^{Dm}$ (described in sections 5.1 and 5.2 of Chapter 5). J_{+3}^{Dm} contains exactly three separation-vertices and at most three degree-3 vertices (as in Figure 26; the vertices α, β, γ could coincide with a separation-vertex σ). Since J

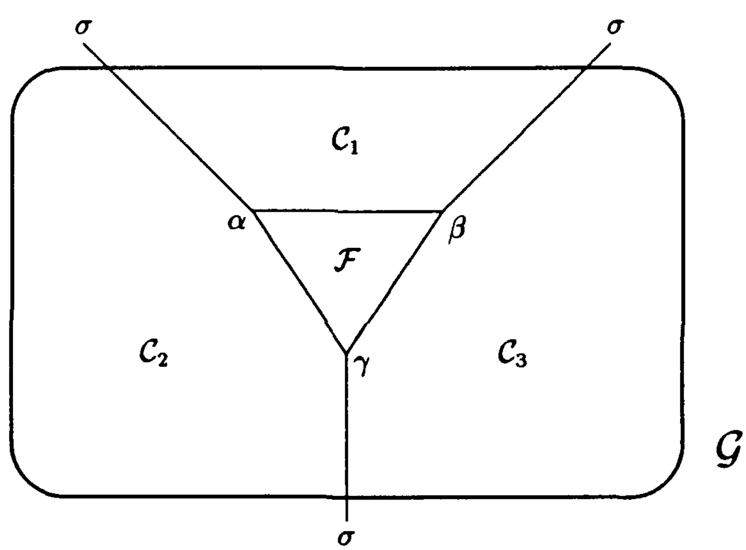


Figure 26: J_{+3}^{Dm} Corresponding to a Trifid Cut J

by itself is a feasible interjacent set, the three separation-vertices in J_{+3}^{Dm} must come between every pair of terminals; otherwise the boundary would provide an s_i–t_i path not using the critical edges. If J is optimal, there must be a commodity linking each pair of the three components of $\mathcal{G} - J_{+3}$ containing terminals. Combining these facts, we see that $\mathcal{G}$ must contain exactly three commodities with adjacent terminals, and the separation-vertices in J_{+3}^{Dm} must fall between these three pairs of terminals. We call these the *key* separation-vertices for this problem.

The smallest four-way cut in the form of J_{+3} could be found by testing each possible trio of branch points α, β, γ, finding the shortest paths in $\mathcal{G}^{Dm}$ between each pair of them and between each one and the three key separation-vertices. The object would be to find the selection of α, β, and γ, and their assignment to key separation-vertices, such that the sum of all six distances is minimized. The optimal trifid cut would then be the sum minus three edges, lacking one edge corresponding to the edges in paths $\alpha\beta$, $\beta\gamma$, and $\gamma\alpha$. But suppose the longest of thses paths is $\alpha\beta$. Then the four-way cut, minus $\alpha\beta$, minus any one edge in $\beta\gamma$ or $\gamma\alpha$, is also a valid $r = 1$ interjacent set, but not a trifid cut. So unless the longest of those paths is only one edge long, the trifid cut will not be uniquely optimal. And since each component of the four-way cut must be nonempty, α, β, and γ cannot coincide. Therefore $\alpha\beta$, $\beta\gamma$, and $\gamma\alpha$ must each contain exactly one edge. So if a trifid cut is uniquely minimal, we can find it by testing every triangle in $\mathcal{G}^{Dm}$. (This corresponds to every set of three edges in $\mathcal{G}$ that separates a component with no terminals from the rest of the graph.) We simply find the shortest distance from each of the triangle's vertices to the three key separation-vertices. The selection of a triangle and assignment of its vertices to key separation-vertices that produces the smallest sum of distances must correspond to the minimum trifid cut.

The following lemma shows that a trifid cut can also be found by finding a full multicommodity cut in a modification of $\mathcal{G}$.

Lemma 10 *Suppose in a T-planar non-crossing multicommodity flow graph $\mathcal{G}$ the* *optimal $r = 1$ partial multicommodity edge-cut J is a trifid cut in which edges e, f, and g are the critical edges in $\mathcal{R}$, and $\mathcal{R}$ has one component. Then J is an optimal full multicommodity cut in $\mathcal{G} - \{e, f, g\}$.*

Proof: J is clearly a feasible FMCC in $\mathcal{G} - \{e, f, g\}$. Suppose that it is not optimal. Then some cut C is the FMCC in $\mathcal{G} - \{e, f, g\}$, and so in $\mathcal{G}$ with a "hole" where the critical edges and the component they separate have been removed (see Figure 27), and $|C| < |J|$. But C is a feasible $r = 1$ interjacent set in $\mathcal{G}$. It cuts all s_i–t_i paths

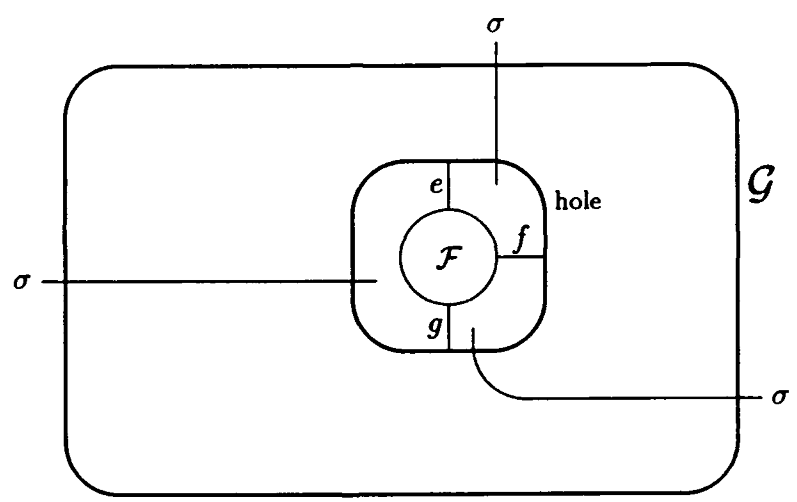


Figure 27: Trifid Cut as Optimal FMCC in $\mathcal{G} - \{e, f, g\}$

in $\mathcal{G} - \{e, f, g\}$, and if those three edges are restored, any s_i–t_i path introduced must use two of them. Therefore $|C| \geq |J|$, a contradiction. ■

So if the critical edges are known, the optimal interjacent set J can be found using the method for full multicommodity cuts in Ue–P*BN graphs (Algorithm 4). If the critical edges are not known, the FMCC problem can be solved repeatedly, once for every trio of edges in $\mathcal{G}$ that could be critical removed (*i.e.* every trio of edges that separates a component with no terminals). This is clearly a polynomially bounded procedure.

Now let us turn to the problem of finding a UeEP*BN $r = 1$ PMCC J when $\mathcal{R}$ may have more than one component, assuming only that J leaves a component in trifid form. First let us suppose that the critical edges are known. Let $\mathcal{G}'$ be the supply graph $\mathcal{G}$ with these edges removed (disconnecting and in effect removing a component $\mathcal{F}$ of $\mathcal{G}$ and leaving a hole—see Figure 25), and let C' be an optimal FMCC in $\mathcal{G}'$. As in the proof of the previous proposition, we can note that C' is also a feasible PMCC in $\mathcal{G}$, that J is a feasible FMCC in $\mathcal{G}'$, and that $|C'| = |J|$. In fact C' must contain a trifid cut in $\mathcal{G}$, just like J. If restoring the critical edges and $\mathcal{F}$ to $\mathcal{G}' - C'$ introduces no paths, then C' minus any edge is a feasible PMCC in $\mathcal{G}$, contradicting $|C'| = |J|$. If it introduces paths, each path must use two of the critical edges, because no terminals are in $\mathcal{F}$. Thus C' leaves a component of $\mathcal{G} - C'$ in trifid form, and with the same critical edges as J, and C' and J are in this sense equivalent. If we use Algorithm 4 to find the FMCC C' in $\mathcal{G}'$, we will have the optimal PMCC in $\mathcal{G}$.

Suppose that the critical edges are not known. There is a polynomial number of candidate trios, and we need only consider those trios that separate some component $\mathcal{F}$ containing no terminals from the rest of $\mathcal{G}$. The optimal interjacent set can be found by solving the FMCC problem in $\mathcal{G}$ with each candidate trio removed in turn.

At this point we have a method for finding the optimal interjacent set J for any UeEP*BN PMCC problem $r = 1$, whether the component of $\mathcal{R}$ containing paths is of the one-commodity, bottleneck, or trifid type. The following algorithm (presented in outline) can solve in polynomial time a general problem of this type.

Algorithm 8 (UeEP*BN $r = 1$ PMCC)

input: $\mathcal{G}$ for a UeEP*BN problem with $r = 1$
output: cardinality of a minimum interjacent set
begin

(1) Do k FMCC problems, deleting one commodity from consideration each time. This is guaranteed to find J_{opt} if it leaves $\mathcal{R}$ in one-commodity form.

(2) Do $|E|$ FMCC problems, deleting one edge from $\mathcal{G}$ each time. This is guaranteed to find J_{opt} if it leaves $\mathcal{R}$ in bottleneck form.

(3) For every set F of three edges that separate $\mathcal{G}$ into two components, one of which has no terminals, find the optimal FMCC in $\mathcal{G} - F$. This will find J_{opt} if it leaves $\mathcal{R}$ in trifid form.

(4) Compare the FMCCs in steps *1, 2,* and *3*, and set J_{opt} to the cardinality of the smallest.

(5) **return** J_{opt}

end

The algorithm requires solving $O(k + m^3)$ FMCC problems (where $m = |E|$). Since we have Algorithm 4 to solve these in $O(\alpha + k^4 + fk^3 + f^2k^2)$ (see p. 73), we have a polynomial bound for Algorithm 8 also. This gives us the following theorem.

Theorem 4 *The $r = 1$ partial multicommodity edge-cut problem with edge-disjoint remaining paths in a T-planar non-crossing graph (case UeEP*BN) is polynomially bounded.*

Proof: Algorithm 8 gives a polynomial procedure for this problem. ∎

It seems likely that a similar approach can find a polynomially bounded algorithm for any fixed r. However, it does not look like the solution will be elegant. For instance, with $r = 2$ we can have in $\mathcal{R}$ two components with s_i–t_i paths, and each of them may be in one-commodity, bottleneck, or trifid form. Also, $\mathcal{R}$ may have only one component with s_i–t_i paths, but so configured that two disjoint paths and no more are possible. Clearly a large number of cases will need to be evaluated.

7 THE EXTENSION TO VERTEX-CUTS

Some of the important results developed in Chapter 5 and Chapter 6 for multicommodity edge-cuts can be extended to vertex-cuts. This chapter shows how this is done. It includes the following:

§7.1 A polynomial algorithm for the weighted T-planar multiterminal vertex-cut problem (*cf.* Corollary 3)

§7.2 A polynomial algorithm for the weighted T-planar multicommodity vertex-cut problem when the number of commodities k is fixed (*cf.* Algorithm 2)

§7.3 Proofs that restricted versions of the T-planar multicommodity vertex-cut problem are NP-hard when k is part of the input (*cf.* Propositions 10 and 11)

§7.4 A polynomial algorithm for the weighted T-planar non-crossing multicommodity vertex-cut problem (*cf.* Algorithm 4)

§7.5 A polynomial algorithm for the T-planar partial multicommodity vertex-cut vertex-disjoint problem when $r = k - 1$ (*cf.* Algorithm 6)

The general method for these results is to construct a sort of dual graph to $\mathcal{G}$ that plays the role of the modified geometric-dual $\mathcal{G}^{Dm}$ in the Ue–PB results. This graph will be called a "face-dual" or $\mathcal{G}^F$. There is a correspondence between paths in $\mathcal{G}^F$ and vertex-cuts in $\mathcal{G}$. The correspondence is not as simple as the direct one-to-one correspondence between paths in $\mathcal{G}^{Dm}$ and cut-sets in $\mathcal{G}$, but it is good enough for the purposes of the algorithms. Once it is established, the methods used in Chapter 5 can at once be seen to apply. The construction of $\mathcal{G}^F$ and the demonstration of its properties, together with slight modifications of the constructions used in the intractability proofs, are all that is required to show that most of the Ue–PB results also apply to the Uv–PB problem.

The next section gives the details of the construction of $\mathcal{G}^F$ and the necessary proofs of its properties, together with their application to the T-planar multiterminal vertex-cut problem. The following sections then extend the method to multicommodity vertex-cut problems.

7.1 The Face-dual $\mathcal{G}^F$ and the Multiterminal Vertex-cut Problem

Let a T-planar multiterminal graph $\mathcal{G}$ be given, along with a positive real-valued weight function $w(v)$ defined on each non-terminal vertex v in $\mathcal{G}$. The *face-dual* of $\mathcal{G}$, denoted $\mathcal{G}^F(V^F, E^F)$, consists of the following:

- One vertex of zero weight for every interior face of $\mathcal{G}$ (*face-vertices*)
- One vertex of zero weight for for each of the paths into which the boundary of $\mathcal{G}$ is partitioned by T (*separation-vertices*)
- All non-terminal vertices of $\mathcal{G}$, with weight unchanged ($\mathcal{G}$*-vertices*)

- An edge between each face-vertex and the $\mathcal{G}$-vertices on the boundary of the corresponding face in $\mathcal{G}$
- An edge between each separation-vertex and the $\mathcal{G}$-vertices on the corresponding path in the boundary of $\mathcal{G}$

An example is shown in Figure 28. $\mathcal{G}^F$ does not contain any edges of $\mathcal{G}$, nor any edges directly corresponding to them (unlike the standard geometric-dual). $\mathcal{G}^F$ is clearly

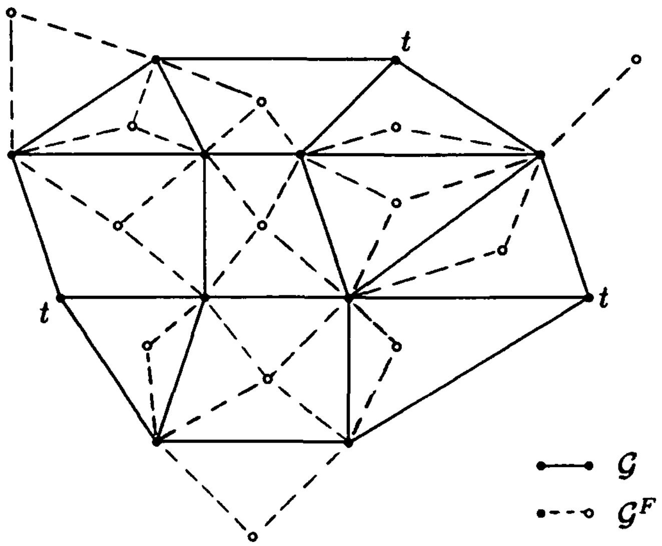


Figure 28: The Face-dual $\mathcal{G}^F$ (Example)

planar. It is also bipartite, since every edge connects a $\mathcal{G}$-vertex with either a face or a separation-vertex. Any path in $\mathcal{G}^F$ that connects exactly two separation-vertices will partition the terminals into two consecutive sets. Such a path contains half as many $\mathcal{G}$-vertices as edges. These $\mathcal{G}$-vertices will form a vertex-cut in $\mathcal{G}$, since by planarity any path in $\mathcal{G}$ between terminals separated by such a path in $\mathcal{G}^F$ will have to use one of those vertices. However, such a vertex-cut will not necessarily be minimal. On the other hand, each minimal vertex-cut in $\mathcal{G}$ corresponds to a path in $\mathcal{G}^F$ between two separation-vertices, as the following proposition shows.

Proposition 15 *Each minimal vertex-cut between the members of a 2-partition of T into consecutive sets corresponds (not necessarily uniquely) to a path in $\mathcal{G}^F$ between two separation-vertices, and every such path in $\mathcal{G}^F$ corresponds to a vertex-cut (unique, but not necessarily minimal) in $\mathcal{G}$.*

Proof: Planarity ensures that the unique set of $\mathcal{G}$-vertices on any such path in $\mathcal{G}^F$

constitute a vertex-cut in $\mathcal{G}$. It remains to show that a minimal vertex-cut induces a path.

Let C be a minimal vertex-cut in $\mathcal{G}$, and let t_1 and t_2 be terminals on opposite sides of the cut. Label all vertices in $\mathcal{G}$ as follows:

c	–	all vertices in C
τ_i	–	all vertices in the same component as t_i in $\mathcal{G} - C$, for $i \in \{1, 2\}$
u	–	all other vertices

Every vertex labeled c is connected in $\mathcal{G}$ to at least one vertex τ_1 and one vertex τ_2; otherwise it could be deleted from the minimal cut C.

Let $\mathcal{T}_1$ be the connected subgraph induced in $\mathcal{G}$ by the vertices labeled τ_1. For each member c_i of C, let T_i be the set of vertices in $\mathcal{T}_1$ that share an edge with c_i. By planarity, all members of all sets T_i must be on the boundary of $\mathcal{T}_1$. By planarity and the minimality of C, the members of different sets T_i cannot be in interleaved or nested order around the boundary of $\mathcal{T}_1$. Therefore we can unambiguously assign numbers so that the sets T_i occur in clockwise order $T_1, T_2, \ldots, T_z$ on the boundary of $\mathcal{T}_1$, with $z = |C|$ and t_1 between T_z and T_1. Also, we must have c_1 and c_z on the boundary of $\mathcal{G}$, each one on a t_1–t_2 path in the boundary. Therefore c_1 and c_z are connected to different separation-vertices in $\mathcal{G}^F$.

It remains to show that c_i and c_{i+1} are on one face of $\mathcal{G}$ for $i \in \{1, \ldots, z-1\}$, and that each of these $z-1$ faces is distinct. This will complete the proof, since in each such face the $\mathcal{G}$-vertices c_i and c_{i+1} are connected via two edges and a face-vertex in $\mathcal{G}^F$, providing a complete path from one separation-vertex to another, as required.

Let t_i^ℓ be the member of T_i that are closest to a member of T_{i+1} clockwise around the boundary of $\mathcal{T}_1$. Then $c_i t_i^\ell$ is an edge of $\mathcal{G}$. This edge separates two faces of $\mathcal{G}$. Follow the boundary of the face on the left (when traveling from c_i to t_i^ℓ) in the c_i-to-t_i^ℓ direction. Beyond t_i^ℓ, the boundary of the face will also be the part of the boundary of $\mathcal{T}_1$ between T_i and T_{i+1}. Continue until a vertex not part of $\mathcal{T}_1$ is encountered (see Figure 29). This vertex cannot be labeled u or τ_2 because it is connected with t_i^ℓ and

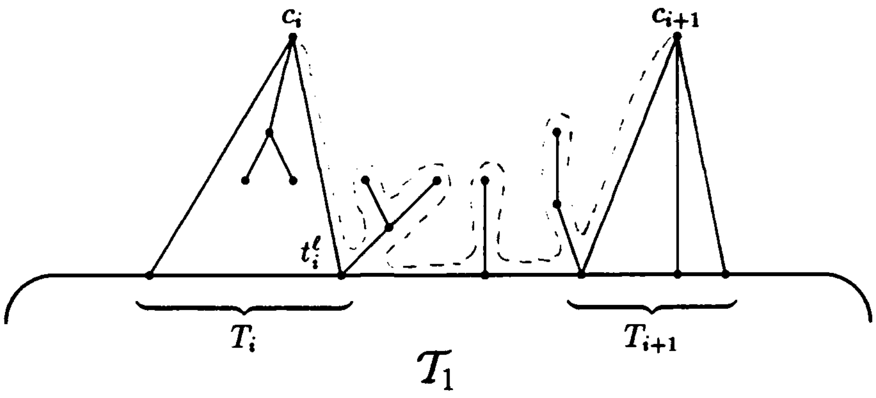


Figure 29: The Common Face of c_i and c_{i+1}

hence with t_1 in $\mathcal{G} - C$. Therefore it must be labeled c. It cannot be c_i again because of the choice of t_i^ℓ. Therefore it must be c_{i+1}, so that c_i and c_{i+1} are on one face of $\mathcal{G}$ and connected via a face-vertex in $\mathcal{G}^F$.

Furthermore, each such face is distinct for $i \in \{1, \ldots, |C|-1\}$. For suppose c_i and c_j are on the same face and $j > i + 1$. Follow the boundary of this face from c_i to t_i^ℓ and on towards c_j. This will also be the boundary of $\mathcal{T}_1$. A member of T_{i+1} must be encountered, followed by c_{i+1} and t_{i+1}^ℓ, before T_j and c_j are reached. The vertex c_{i+1} cannot be connected to any point labeled τ_2 without violating planarity or dividing the supposed common face of c_i and c_j. This contradiction completes the proof. ■

The construction in this proof also yields the following:

Corollary 4 *A minimal vertex-cut in $\mathcal{G}$ of cardinality p corresponds to a path in $\mathcal{G}^F$ that contains $2p$ edges.*

Let the *weighted-length* of a path in $\mathcal{G}^F$ be the sum of the weights of the $\mathcal{G}$-vertices on the path. Then the following proposition also holds.

Proposition 16 *A minimum-weighted-length path between two separation-vertices in $\mathcal{G}^F$ induces a minimum-weight vertex-cut in $\mathcal{G}$ between a set of consecutive terminals and its complement in T.*

Proof: The separation-vertices at the ends of such a path induce a partition of T into two consecutive sets, and the $\mathcal{G}$-vertices on the path form a vertex-cut between these sets with the same total weight as the path's length. These vertices are a minimum-weight vertex-cut. For suppose a minimal vertex-cut with smaller total weight existed. Then the construction in the proof of Proposition 15 would provide a path between the same separation-vertices, and so partitioning T in the same way, and containing only $\mathcal{G}$-vertices from the smaller cut, so this path would be shorter than the supposed minimum. ■

The T-planar vertex MTC problem is this: for some set of k vertices on the boundary of a planar graph $\mathcal{G}$, find the set of other vertices of minimum weight whose removal from $\mathcal{G}$ will disconnect each terminal from each of the others. As in Section 5.1, we will reformulate the problem into that of finding a tree in $\mathcal{G}^F$ such that the $\mathcal{G}$-vertices in the tree are exactly the minimum-weight multiterminal vertex-cut in $\mathcal{G}$. To ensure that this is possible, we need the following proposition.

Proposition 17 *Given any minimal multiterminal vertex-cut C in a T-planar graph $\mathcal{G}$, a corresponding tree in $\mathcal{G}^F$ exists that includes all separation-vertices and all and only the $\mathcal{G}$-vertices in the set C.*

Proof: The following procedure will construct the required tree:

1. Find a minimal subset of C that separates two terminals t_1 and t_2.
2. Using the construction in the proof of Proposition 15, find a corresponding path in $\mathcal{G}^F$ between two separation-vertices.
3. Repeat steps 1 and 2 for each distinct unordered pair of terminals drawn from T.
4. Let $\mathcal{T}$ be the union of all $k(k-1)$ paths found in this way. Then $\mathcal{T}$ is a subgraph of $\mathcal{G}^F$, it includes all and only the $\mathcal{G}$-vertices from C, and it includes all separation vertices.
5. If $\mathcal{T}$ includes a cycle, delete any edge in the cycle. Repeat until no cycles remain.
6. If $\mathcal{T}$ includes a face-vertex as an endvertex after Step 5, delete it and the associated edge.

$\mathcal{T}$ cannot have a $\mathcal{G}$-vertex as a degree-1 vertex: without the edge on such a vertex it would still contain paths separating all pairs of terminals, so the vertex would be superfluous in the supposed minimal MTC C. The tree constructed by this procedure will not necessarily be the minimum tree on the separation-vertices, nor will it necessarily be unique. However, it will contain exactly the $\mathcal{G}$-vertices in C. ■

We also need to note that a tree on the separation-vertices in $\mathcal{G}^F$ will correspond to a unique multiterminal vertex-cut in $\mathcal{G}$, namely the cut consisting of the $\mathcal{G}$-vertices in the tree. However, there is no guarantee that the minimum-cardinality tree in $\mathcal{G}^F$ will correspond to the minimum-cardinality vertex MTC in $\mathcal{G}$: the number of

$\mathcal{G}$-vertices in a tree in $\mathcal{G}^F$ depends on the location of its branch points, *i.e.* whether they are at $\mathcal{G}$-vertices or face-vertices, as well as on the number of edges it has.

Despite this lack of direct and unique correspondence, trees in $\mathcal{G}^F$ can still be examined to find the optimal vertex MTC in $\mathcal{G}$. We know that there is a tree that corresponds to the every minimal MTC, and we know that an MTC corresponds to every tree. Furthermore, each tree can be assigned a well-defined weight equal to the weight of the MTC it induces: simply the sum of the weights of the $\mathcal{G}$-vertices it contains. Clearly a minimum-weight tree will induce a minimum-weight vertex-cut. Furthermore, the weight of a tree thus defined is the sum of the weight of its parts. This all that is needed to apply an algorithm like that of Provan [1988].

To develop the algorithm, we need a few definitions:

$$w(\nu) \equiv \text{weight of } \nu \in V^F \text{ (zero except for } \mathcal{G}\text{-vertices)} \quad (47)$$

$$\mathrm{d}^V(\nu, v) \equiv \text{length of a minimum-weighted-length path from } \nu \text{ to } v\text{, excluding weights of } \nu \text{ and } v \; (\nu, v \in V^F) \quad (48)$$

$$\mathbf{C}^V_m([\sigma_i, \sigma_j], \nu) \equiv \text{weight (excluding } w(\nu)) \text{ of a minimum-vertex-weight tree on } [\sigma_i, \sigma_j] \text{ and } \nu \in V^F \quad (49)$$

$$\mathbf{B}^V_m([\sigma_i, \sigma_j], \nu) \equiv \text{weight (including } w(\nu)) \text{ of a minimum-vertex-weight tree on } [\sigma_i, \sigma_j] \text{ and } \nu \in V^F \text{ when } \nu \text{ is either a separation vertex or adjacent to two or more edges of the tree} \quad (50)$$

(The m subscripts are applied to these $\mathbf{B}^V$ and $\mathcal{C}^V$ functions for the MTC problem to distinguish them from the similar functions that will be defined in Section 7.2.) It can be shown that at any branch point in the desired tree in $\mathcal{G}^F$, each edge leads to a consecutive set of separation-vertices (Provan [1988]; Erickson, Monma, and Veinott

[1987]). Because of this, the following recursive functions hold (using the convention that $\sigma_i \equiv \sigma_{i \pm k}$):

$$\mathbf{C}_m^V([\sigma_i, \sigma_j], \nu) = \min_{\upsilon \in V^F} \{w(\upsilon) + \mathrm{d}(\nu, \upsilon) + \mathbf{B}_m^V([\sigma_i, \sigma_j], \upsilon)\} \tag{51}$$

and for $i \neq j$,

$$\mathbf{B}_m^V([\sigma_i, \sigma_j], \nu) = w(\nu) + \min_{\sigma_x \in [\sigma_i, \sigma_{j-1}]} \{\mathbf{C}_m^V([\sigma_i, \sigma_x], \nu) + \mathbf{C}_m^V([\sigma_{x+1}, \sigma_j], \nu)\} \tag{52}$$

This leads to the following algorithm, which takes $\mathcal{G}^F$ as input and assumes that it is two-vertex-connected. (If $\mathcal{G}^F$ is not two-vertex-connected, the algorithm can be performed on each two-vertex-connected component, treating articulation vertices as separation-vertices. If $\mathcal{G}$ itself has a terminal as an articulation vertex, then $\mathcal{G}^F$ will have two components, and trees can be found on them separately.)

Algorithm 9 (Weighted Uv–PB MTC)

input: $\mathcal{G}^F$ for a weighted Uv–PB MTC problem
output: weight of an optimal MTC in $\mathcal{G}$
begin
for each $\nu, \upsilon \in V^F$
    find minimum weighted-length distance $\mathrm{d}^V(\nu, \upsilon)$
for each $\nu \in V^{Fm}$ and each separation-vertex σ_i
    set $\mathbf{C}_m^V([\sigma_i, \sigma_i], \nu) = \mathrm{d}^V(\sigma_i, \nu)$
for $i = 2, \ldots, k-1$
    for each set $[\sigma_a, \sigma_b]$ of cardinality i that does not include σ_1
        for every $\nu \in V^F$
            compute $\mathbf{B}_m^V([\sigma_a, \sigma_b], \nu)$ using (52)
            compute $\mathbf{C}_m^V([\sigma_a, \sigma_b], \nu)$ using (51)
return $\mathbf{C}_m^V([\sigma_2, \sigma_k], \sigma_1)$
end

The values $d^V(\nu, v)$ can be found by a simple transformation of the node-weighted problem into an edge-weighted shortest path problem. Since $|V^F|$ is $O(m)$, the complexity of this algorithm is thus $O(\zeta + m^2k^2)$, where ζ is the complexity of finding all the minimum weighted-length $d^V(\nu, v)$ in $\mathcal{G}^F$.

7.2 The T-Planar Multicommodity Vertex-cut Problem with k Fixed

This section will address the problem of finding minimum-weight full multicommodity vertex-cuts in T-planar graphs. The development will only be sketched out, since it is essentially identical to that for the Ue–PB problem addressed in Section 5.2 (p. 43). The only difference is that the face-dual $\mathcal{G}^F$ plays the role of the modified geometric-dual $\mathcal{G}^{Dm}$. The same labeling conventions are taken when using the face-dual in a multicommodity problem as were used with the modified geometric-dual: the distinct terminal vertices in $\mathcal{G}$ are labeled in order clockwise order from a_1 to a_z, and the separation-vertices in $\mathcal{G}^F$ are likewise numbered, so that σ_1 falls between a_z and a_1.

If all the terminals are on the boundary of $\mathcal{G}$, then the optimal FMCC corresponds to a forest in $\mathcal{G}^F$, each tree in which is on some subset of the separation-vertices. Every vertex-cut in $\mathcal{G}$ induces a path between separation-vertices in $\mathcal{G}^F$, and every such path induces a unique vertex-cut. We can conclude that there is a forest in $\mathcal{G}^F$ of disjoint trees on subsets of the separation-vertices, such that that forest induces the optimal vertex FMCC in $\mathcal{G}$. If we define the weight of a tree in $\mathcal{G}^F$ as the sum of the weights of its $\mathcal{G}$-vertices, we can find this forest with the methods used for the Ue–PB problem.

Each tree in the optimal forest is a minimum-vertex-weight tree for the separation-vertices that it contains, and the forest is the minimum-weight combination of such trees that contains at least one path separating each commodity. We can define the following functions in $\mathcal{G}^F$ (where the expressions term$[\sigma_i, \sigma_j]$, comm$[\sigma_i, \sigma_j]$, and $[\sigma_i, \sigma_j]$-feasible are defined as for the Ue–PB problem on p. 48):

$$\mathbf{T}^V[\sigma_i, \sigma_j] \equiv \text{weight of a minimum-vertex-weight } [\sigma_i, \sigma_j]\text{-feasible subforest with } \sigma_i \text{ and } \sigma_j \text{ in the same component} \tag{53}$$

$$\mathbf{S}^V([\sigma_i, \sigma_j], K) \equiv \text{weight of a minimum-vertex-weight } [\sigma_i, \sigma_j]\text{-feasible and (unless } \sigma_{i+1} \equiv \sigma_j) \ [\sigma_{i+1}, \sigma_{j-1}]\text{-feasible subforest that also separates all commodities in } K \tag{54}$$

$$\mathbf{B}^V([\sigma_i, \sigma_j], \nu) \equiv \text{weight (including } w(\nu)) \text{ of a minimum-vertex-weight } [\sigma_i, \sigma_j]\text{-feasible subforest with } \nu, \sigma_i, \text{ and } \sigma_j \text{ in the same component and with } \nu \text{ of degree two or more or identical with } \sigma_i \text{ or } \sigma_j \tag{55}$$

$$\mathbf{C}^V([\sigma_i, \sigma_j], \nu) \equiv \text{weight (excluding } w(\nu)) \text{ of a minimum-vertex-weight } [\sigma_i, \sigma_j]\text{-feasible subforest with } \nu, \sigma_i, \text{ and } \sigma_j \text{ in the same component} \tag{56}$$

$$\mathbf{D}^V([\sigma_i, \sigma_j], \nu) \equiv \text{weight (excluding } w(\nu)) \text{ of a minimum-vertex-weight } [\sigma_i, \sigma_j]\text{-feasible subforest with } \nu \text{ and } \sigma_j \text{ in the same component} \tag{57}$$

Then we can write the following recursive equations:

$$\mathbf{B}^V([\sigma_i, \sigma_j], \nu) = w(\nu) + \min_{\sigma_p \in [\sigma_i, \sigma_{j-1}]} \{\mathbf{C}^V([\sigma_i, \sigma_p], \nu) + \mathbf{D}^V([\sigma_p, \sigma_j], \nu)\} \tag{58}$$

$$\mathbf{C}^V([\sigma_i, \sigma_j], \nu) = \begin{cases} \mathrm{d}(\sigma_i, \nu) & \text{if } i = j \\ \min_{\upsilon \in V^F} \{\mathrm{d}(\nu, \upsilon) + \mathbf{B}^V([\sigma_i, \sigma_j], \upsilon)\} & \text{otherwise} \end{cases} \tag{59}$$

$$\mathbf{D}^V([\sigma_i,\sigma_j],\nu) = \min_{\sigma_q\in[\sigma_{i+1},\sigma_j]}\{\mathbf{S}^V([\sigma_i,\sigma_q],\phi)+\mathbf{C}^V([\sigma_q,\sigma_j],\nu)\} \tag{60}$$

$$\mathbf{S}^V([\sigma_i,\sigma_j],K) = \begin{cases} 0 & \text{if } j=i+1 \\ \infty & \text{if } j\neq i+1 \text{ and } Q'=\phi \\ \min_{(\sigma_a,\sigma_b)\in Q'}\{\mathbf{T}^V[\sigma_a,\sigma_b]+\mathbf{S}^V([\sigma_b,\sigma_j],K')\} & \text{otherwise} \end{cases} \tag{61}$$

where

$$Q' \equiv \{ \quad (\sigma_a,\sigma_b): \quad \begin{array}{l} \sigma_a\in[\sigma_{i+1},\sigma_j]-\{\sigma_{j-1},\sigma_j\}\ , \\ \sigma_b\in[\sigma_{a+1},\sigma_j]-\{\sigma_j\}\ , \\ K\cap \text{term}[\sigma_i,\sigma_a]=\phi\ , \qquad \text{and} \\ \text{comm}[\sigma_i,\sigma_a]=\phi \end{array} \quad \}$$

and

$$K' = (K\cup \text{term}[\sigma_i,\sigma_a])\backslash \text{term}[\sigma_a,\sigma_b])$$

We also have the equations

$$\mathbf{T}^V[\sigma_i,\sigma_j] = \begin{cases} 0 & \text{if } i=j \\ \mathbf{B}^V([\sigma_i,\sigma_j],\sigma_i) & \text{otherwise} \end{cases} \tag{62}$$

and, if W_{opt} is the size of the optimal solution and the separation-vertices are numbeed so that there is one commodity that is in both K_z and K_i,

$$W_{\text{opt}} = \min_{(\sigma_p,\sigma_q)\in Q}\{\mathbf{T}^V[\sigma_p,\sigma_q]+\mathbf{S}^V([\sigma_q,\sigma_p],\phi)\} \tag{63}$$

where

$$Q\equiv\{(\sigma_p,\sigma_q):\sigma_p\in[\sigma_1,\sigma_i],\sigma_q\in[\sigma_{i+1},\sigma_z]\}$$

and σ_i is such that a_i and a_z share a commodity.

Algorithm 2 on p. 54 can be used for this problem virtually as it stands, *mutatis mutandis.* The input is the face-dual $\mathcal{G}^F$ instead of $\mathcal{G}^{Dm}$, $\text{d}^V(\nu,v)$ should be used for d(ν, v), and the V-superscripted versions of functions **S**, **T**, **B**, **C**, and **D** should be used (eqns. (49), (58)–(62)). The complexity is $O(\zeta + 2^k k^4 + mk^3 + m^2k^2)$, where all distances d($\nu, v$) are found in $O(\zeta)$.

7.3 The Intractability of Vertex-cuts when k Varies

This section contains two propositions on the intractability of multicommodity vertex-cuts in T-planar graphs. The problem is NP-hard even when restricted to rectangular grids with rectangular boundaries. The weighted problem is NP-hard even when restricted to trees. These results parallel exactly those in the edge-cut case (Propositions 10 and 11), and the proofs are derived from them.

The vertex-disjoint paths construction problem is not feasible unless the terminals are in non-crossing order (in which case the FMCC problem is polynomial, as we will see in the next section); the edge-disjoint paths problem can be feasible regardless of the order of the terminals. However, it still makes sense to ask what minimum set of vertices will lie on all s_i–t_i paths, even if a full set of disjoint paths is not possible.

Proposition 18 *The unweighted T-planar full multicommodity vertex-cut problem is NP-hard, even when restricted to rectangular grids with rectangular boundaries.*

Outline of Proof: The general method is exactly the same as that for Proposition 10, which makes the same claim for edge-cuts. A vertex cover problem in a graph with all vertices of degree 3 is polynomially reduced to a Uv–PB FMCC problem. The same system of commodities is used: vertex, edge, and connecting. The gadgets must be modified slightly—see figures 30 and 31. A non-terminal vertex must be left between

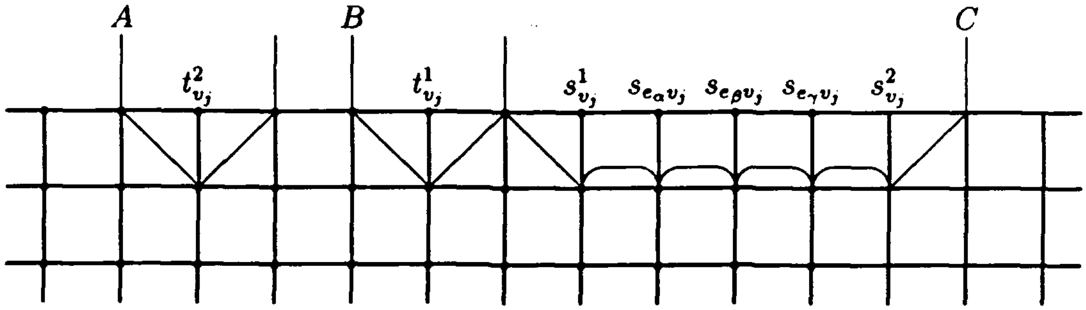


Figure 30: Vertex Gadget for Vertex-cuts (*cf.* Figure 15)

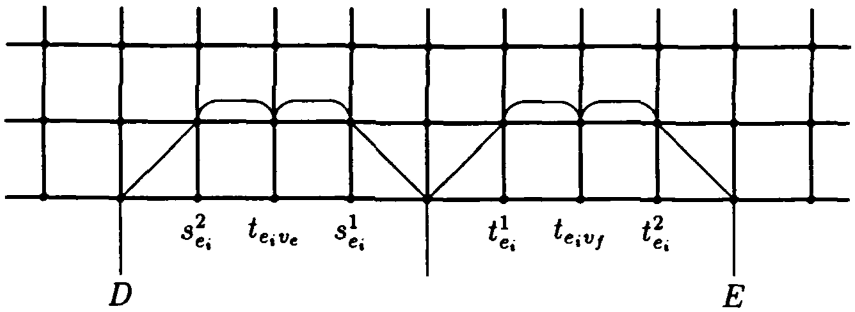


Figure 31: Edge Gadget for Vertex-cuts (*cf.* Figure 16)

each pair of terminals where a cut may need to enter or leave the grid.

In any optimal FMCC in the constructed problem, edge commodities must be separated by vertex-cuts of at least five vertices (as cut D or E in Figure 31). The vertex commodities must be separated either by two cuts of three vertices each isolating the sinks (A and B in Figure 30) or by one vertex-cut of seven vertices isolating the sources and also separating the associated connecting commodities (C). A cut on the vertex commodity source side corresponds to the cover problem vertex v_j being in the cover. The optimal FMCC will then contain the following vertices, where m and n are the number of edges and vertices in the graph for the vertex cover problem:

5 for each of m edge commodities

6 for each of n vertex commodities

1 for each set of three connecting commodities separated by source-side vertex commodity cuts

There is a vertex cover of size p if and only if there is an FMCC in the constructed problem of at most $6n + 5m + p$ vertices. ■

Proposition 19 *The weighted full multicommodity vertex-cut problem is NP-hard even when restricted to trees.*

Proof: The corresponding edge-cut problem is NP-hard (Proposition 11), and it can be reduced to the vertex problem by inserting a vertex into each edge with the edge's weight and assigning an arbitrarily high weight to all other vertices. ■

7.4 A Polynomial Case: T-Planar and Non-crossing

A polynomially-bounded algorithm analogous to Algorithm 4 (for the corresponding edge-cut problem) can be developed for T-planar multicommodity vertex-cuts if the terminals are in non-crossing order. The functions d^V, $\mathbf{T}^V$, $\mathbf{B}^V$, $\mathbf{C}^V$, and $\mathbf{D}^V$ functions developed for the Uv–PB problem (Section 7.2) can be used. The $\mathbf{S}^V([\sigma_i, \sigma_j], K)$ function can be modified slightly because it need be calculated only for $K = \phi$: the "betweenness" argument used in developing Algorithm 4 for Ue–P*BN edge-cuts (Section 5.5) applies without modification. The formula then becomes:

$$\mathbf{S}^V([\sigma_i, \sigma_j], \phi) = \min_{(\sigma_g, \sigma_h) \in Q''} \{\mathbf{S}([\sigma_i, \sigma_g], \phi) + \mathbf{T}[\sigma_g, \sigma_h] + \mathbf{S}([\sigma_h, \sigma_j], \phi)\} \tag{64}$$

where

$$
\begin{aligned}
Q'' \equiv \{ \quad (\sigma_g, \sigma_h) : \quad & \sigma_g \in [\sigma_{i+1}, \sigma_q] - \{\sigma_q\} , \\
& \sigma_h \in [\sigma_{g+1}, \sigma_j] - \{\sigma_j\} , \\
& \text{at least one of } \sigma_g, \sigma_h \text{ is in } [\sigma_{i+1}, \sigma_p] \cup [\sigma_{q+1}, \sigma_j] - \{\sigma_j\} \quad \}
\end{aligned}
$$

and σ_p is such that a member of K_p is also a member of term$[\sigma_{p+1}, \sigma_j]$, say in K_q, and no member of comm$[\sigma_i, \sigma_j]$ is in term$[\sigma_i, \sigma_p]$. Algorithm 4 in Section 5.5 can then be used for the Uv–P*BN problem, *mutatis mutandis*: $\mathcal{G}^F$ used in place of $\mathcal{G}^{Dm}$ and the V-superscripted versions of the d, **S** (eq. 64), **T**, **B**, **C**, and **D** functions used. The complexity comes out to $O(\zeta + k^4 + mk^3 + m^2k^2)$. This gives us:

Proposition 20 *The weighted T-planar full multicommodity vertex-cut problem is polynomially bounded if the terminals are restricted to be in non-crossing order.*

Proof: Outlined in the preceding paragraph. ■

7.5 Partial Multicommodity Vertex-cuts

Just as for edge-cuts, the intractability of the full multicommodity vertex-cut problem implies an additional constraint (beyond those of Proposition 4 in Chaper 4) on the tractability of partial vertex cuts. On the other hand, polynomially bounded algorithms are possible for special configurations. One such algorithm is developed here, for the UvVPB case with $r = k - 1$.

The T-planar non-crossing PMCC problem with $r = k - 1$ can be handled for vertex-cuts just as it was for edge-cuts in Section 6.2. For the U–VPB disjoint paths construction problem, cross-freeness and the general cut condition are necessary and sufficient for the existence of a solution [Robertson and Seymour, 1986]. If cross-freeness does not hold in $\mathcal{G}$, then the smallest $r = k-1$ interjacent set is the empty set.

Suppose that cross-freeness does hold. Removing vertices from $\mathcal{G}$ will not change that fact. Therefore the optimal interjacent set is the smallest vertex set whose removal from $\mathcal{G}$ will result in a violation of the GCC for some vertex-cut in $\mathcal{G}$. Furthermore, it is necessary to consider only simple vertex-cuts, since some such minimal cut will violate the GCC if a compound cut does. Also, if the GCC is violated for any simple cut, it will be violated as much or more for the minimum vertex-cut that partitions the boundary in the same way. This suggests the following algorithm, which is the counterpart of Algorithm 6 for the edge-cut case:

Algorithm 10 (UvVPB $r = k - 1$ PMCC)

input: $\mathcal{G}$ for a UvVPB problem with $r = k - 1$
output: cardinality of a minimum interjacent set
begin
if $\mathcal{G}$ is not cross-free
    return 0
    end
set $J_{\text{opt}} = \infty$
for $i = 1$ to $\lfloor |T|/2 \rfloor$
    for every set S of i consecutive terminals in T
        set p equal to the number of unmatched terminals in S
        find the minimum vertex-cut C between S and $T - S$
        set $J_{\text{opt}} = \min \{ J_{\text{opt}} , |C| - p + 1 \}$
return J_{opt}
end

The complexity of this algorithm is $O(\delta k^2)$, where δ is the complexity of finding a minimum vertex-cut in a planar graph. Alternatively, since the cuts C correspond to shortest paths between separation vertices in $\mathcal{G}^{Dm}$, the algorithm is $O(\alpha)$, where α is the complexity of the all-pairs shortest paths problem.

If there is a vertex-cut with the smallest surplus in $\mathcal{G}$ that separates two or more commodities, any surplus-plus-two vertices from that cut will be a feasible $r = k - 2$ interjacent set. From Lemma 3, we can also say that this interjacent set is optimal. In fact, we have the following proposition:

Proposition 21 *In a T-planar partial multicommodity vertex-cut problem with the terminals in non-crossing order, suppose that C is a simple vertex-cut that contains p vertices and separates q commoditie :nd suppose that no vertex-cut in $\mathcal{G}$ has a smaller surplus than C. Suppose further that $k-1 \geq r \geq k-q$. Then any $p-q+k-r$ vertices in C are an optimal interjacent set.*

Proof: An interjacent set as descibed is clearly feasible, for it prevents $k - r$ of the q commodities separated by C from being connected by vertex-disjoint paths. Optimality has already been shown for the case $r = k - 1$. Lemma 3 implies that if a new feasible interjacent set can be found with just one additional vertex when r is decremented, then that set must be optimal. By induction, each addition of a new vertex from C to the interjacent set results in a new optimal set for a value of r one smaller, until C is exhausted. ■

8 THE SINGLE-SOURCE PROBLEM

An important special case of the partial multicommodity cut problem is the case in which all commodities have a common source. This is a common situation in a communications or logistics network, where data or goods may travel from one central location to a large number of user locations. It turns out that the PMCC problem is much easier in this case. Even in general graphs, it has a polynomial solution as long as k, r, or $k-r$ is bounded. The method should be practical for answering such questions as, "What is the smallest loss of links that will cut off three or more of the users?"

From the symmetry of the problems, the single-sink problem is equivalent to the single-source. In an undirected graph the s_i and t_i labels can be interchanged, so all that is required is that there be one vertex that is a terminal for each commodity. All these variations will be treated under the name of a single-source problem.

The full multicommodity cut problem ($r = 0$) is easily solved in the single-source case. For example, let us take such a problem in a directed graph. A super-sink t' is added to the digraph, along with the set of *terminal arcs* $\{t_i t' : i = 1, \ldots, k\}$. Call the resulting digraph the *sink-augmented graph* $\mathcal{G}'$. If the terminal arcs are given infinite weight and all other arcs weight 1, the minimum-weight s–t' cut in $\mathcal{G}'$ will then be the required FMCC in $\mathcal{G}$. With well-known reformulations, this applies to either directed or undirected graphs and either edge- or vertex-cuts.

Up to this point in this work only undirected graphs have been considered. In contrast, this section looks at the single-source problem initially as a directed graph problem; this is appropriate to the asymmetry between the single source and the many sinks. We start the development with arc-cuts that leave arc-disjoint paths (the DeE case) with $r = 1$ (Section 8.1), and then generalize to handle any value of r (Section 8.2). Section 8.3 will show how these results can be used with little modification in undirected graphs, when vertex-cuts are required, or both (UeE, DvV, and UvV). The next two sections consider the more complicated *cross-cases* DeV and DvE: here arcs are to be removed from $\mathcal{G}$ until only r vertex-disjoint paths remain, or *vice versa.* The cross-cases are perhaps rarer in actual applications; they are developed here most fully for the $r = 1$ case. Finally, the last section in this chapter returns to undirected and (nearly) T-planar graphs. For this interesting special case, methods similar to those of Chapter 5 are used to find an algorithm that is polynomially bounded for unrestricted r and k in the input.

8.1 Arc-cuts, Arc-disjoint paths, $r = 1$ (DeE)

We consider here the problem: given a multicommodity flow digraph $\mathcal{G}$ in which one node is the source for every commodity, find the smallest set of arcs J whose removal leaves no more than one arc-disjoint path from the source s to any sink t_i. A practical method for solving this problem comes directly from the following lemma.

Lemma 11 *For any feasible solution J to a single-source partial multicommodity cut* *problem in a directed graph $\mathcal{G}$ with $r = 1$, all s–t_i paths in $\mathcal{R} = \mathcal{G} - J$ either go to the same sink t_i, or have at least one arc in common.*

Proof: Take any such feasible solution J. Form the graph $\mathcal{R}'$ from the union of all s–t_i directed paths in $\mathcal{R}$, the super-sink t', and k terminal arcs t_it'. Consider the maximum s–t' flow problem in $\mathcal{R}'$ when all terminal arcs have infinite capacity and all other arcs have capacity 1. The maximum flow must be non-negative and integral. If the maximum flow is 0, then there is no s–t_i path in $\mathcal{R}$ and the lemma is trivially true. If the maximum flow is 1, there must also be a minimum cut of one capacity-1 arc between s and t'. This arc is on every s–t_i path.

Now suppose the maximum flow is greater than 1. Any feasible maximum flow must go to one sink t_j; otherwise there would be disjoint paths to two different sinks. Furthermore, there can be no path in $\mathcal{R}$ that goes to any other sink. For suppose that there is a path P_ℓ in $\mathcal{R}$ that goes to t_ℓ ($\ell \neq j$) . This path must share an arc with each of the disjoint paths to t_j carrying flow in any given maximum flow solution; otherwise it would be a disjoint path to another sink. Call the first such path it shares an edge with P_j^1 and the last P_j^2 (see Figure 32). Then there is a path from s to t_ℓ

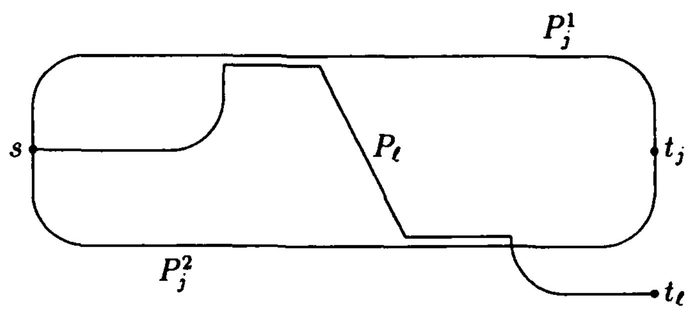


Figure 32: Proof of Lemma 11

that follows P_j^2 until its last common node with P_ℓ, then follows P_ℓ. This path is disjoint from P_j^1 and goes to a different sink, contradicting the feasibility of J. ■

Let $\mathcal{G}'$ be $\mathcal{G}$ plus t' plus the terminal arcs. Note that in $\mathcal{G}'$, any $r = 1$ PMCC has the property that it plus at most one additional arc (either an arc of $\mathcal{G}$ or one of the terminal arcs) is a full s–t' cut. Also, any such full cut that uses no more than one of the terminal arcs can be made into a PMCC by deleting the terminal arc if present, or by deleting any one of the other arcs if it contains no terminal arc. So we can find a minimum PMCC J for the $r = 1$ single-source case by finding the minimum cut C' in $\mathcal{G}'$ (with all arcs, including terminal arcs, at equal weight), with the side condition that no more than one of the terminal arcs can be in C'. J is then formed from C' by deleting one of its members (a terminai arc, if present), and $|J| = |C'| - 1$.

The side condition is of a combinatorial nature, and adding its linear relaxation does not necessarily result in a 0/1 solution when finding the minimum cut via the simplex method. An example is shown in Figure 33. The minimum cut without side

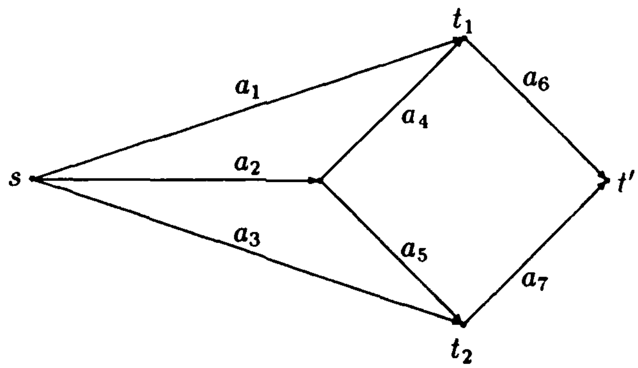


Figure 33: Example of Side Condition Causing a Non-integral Simplex Solution

constraint here is $\{a_6, a_7\}$. If the constraint $x_{a_6} + x_{a_7} \leq 1$ is added, a non-integral cut of size 2.5 results, with $x_{a_1} = x_{a_2} = x_{a_3} = x_{a_6} = x_{a_7} = 0.5$. A proper minimum cut with side constraint is $\{a_1, a_2, a_7\}$.

Nevertheless, the problem can still be solved in polynomial time by solving at most k minimum cut problems in $\mathcal{G}'$, in each one excluding all but one of the terminal cuts by setting their capacities at more than $|E|$. If a general minimum cut problem is $O(\eta)$, the single-source PMCC $r = 1$ problem is $O(\eta k)$.

8.2 Generalizing DeE to $1 < r < k$

Lemma 11 can be generalized in the following way:

Lemma 12 *For any feasible solution J to a single-source partial multicommodity cut problem in a directed graph $\mathcal{G}$, all s-t_i paths in $\mathcal{R} = \mathcal{G} - J$ can be partitioned into two sets such that for some integer p, $0 \leq p \leq r$, all paths in one set can be covered by p edges, and all paths in the other go to no more than $r - p$ sinks t_i.*

Proof: Take any such feasible solution J. Form the graph $\mathcal{R}'$ from the union of all s–t_i directed paths in $\mathcal{R}$, the super-sink t', and k terminal arcs t_it'. Consider the maximum s–t' flow problem in $\mathcal{R}'$ when all arcs have capacity 1. The total flow can be no more than r; otherwise $\mathcal{R}$ contains more than r disjoint paths to different terminals. Therefore the minimum cut in $\mathcal{R}'$ is no more than r. Let such a minimum cut consist of a set P of arcs from $\mathcal{R}$ and a set Q of terminal arcs, so that $|P| + |Q| \leq r$. Partition all s–t_i paths in $\mathcal{R}$ into those containing one or more of the arcs in P and those containing none. The second set of paths must go to one of the $|Q|$ sinks belonging to terminal arcs in Q; otherwise they will not be cut by the minimum cut.

With $p = |P|$, all paths in $\mathcal{R}$ have been partitioned as required. ∎

If the interjacent set J is minimal, this can strengthened:

Lemma 13 *For any* minimal *solution J to a single-source partial multicommodity cut problem in a directed graph $\mathcal{G}$, all s-t_i paths in $\mathcal{R} = \mathcal{G} - J$ can be partitioned into two sets such that for some integer p, $0 \leq p \leq r$, all paths in one set can be covered by p edges, and all paths in the other go to* exactly *$r - p$ sinks t_i.*

Proof: By Lemma 11, all paths can be partitioned into two sets such that Set 1 is covered by p arcs and Set 2 other goes to q distinct sinks, and $p + q \leq r$. It remains to show that if J is optimal then $p + q = r$.

If $p + q < r$, there are fewer than r disjoint s–t_i paths in $\mathcal{R}$: no more than p in Set 1, plus no more than q in Set 2. Adding one edge to $\mathcal{R}$ (*i.e.* removing one edge from J) can add no more than one arc-disjoint path, so the number of disjoint paths will still be feasible if J is reduced by one arc, and it cannot be minimal. ∎

With a single-source PMCC characterized in this way, there is a natural way to find the optimal solution. The following algorithm shows how this is done; the proof of correctness follows. The construction described in the algorithm is illustrated in Figure 34.

Algorithm 11 (Single-source DeE PMCC)

input: $\mathcal{G}$ and r for a single-source DeE partial multicommodity cut problem

output: optimum interjacent set

begin

(1) Augment $\mathcal{G}$ with the super-sink t' and the terminal arcs to form $\mathcal{G}'$.

(2) Solve the s–t' minimum unweighted cut problem in $\mathcal{G}'$, with the side constraint that no more than r of the terminal arcs can be in the cut, to get a set of arcs C'.

(3) Delete r arcs from C', including in the deletion any terminal arcs in C'. Call the resulting set J.

(4) **return** J

end

Proposition 22 *The interjacent set found in Algorithm 11 is an optimal solution for a single-source partial multicommodity arc-cut problem (DeE).*

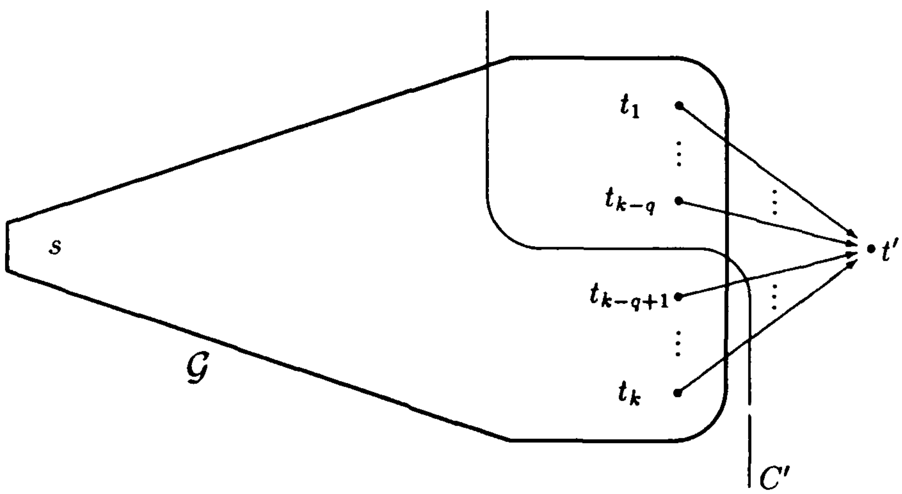


Figure 34: Construction of a Single-source PMCC

Proof: That J constructed in this way is a feasible PMCC is shown as follows. Let C' be a *side-constrained cut* in $\mathcal{G}'$: the minimum cut that contains no more than r

terminal arcs. Let C' be partitioned into a set of terminal arcs Q and a set of other arcs $P' = C' \cap E$. Let $\mathcal{G}^{(-)} = \mathcal{G} - P'$. Then $\mathcal{G}^{(-)}$ contains paths to $|Q| = q$ different terminals. Let P be a set of any $r - q = p$ arcs in P'. Since adding each arc can permit no more than one additional s–t_i path, $\mathcal{G}^{(-)} + P$ can have no more than r disjoint s–t_i paths. But

$$\mathcal{G}^{(-)} + P \; = \; \mathcal{G} - (P' - P) \; = \; \mathcal{G} - J \tag{65}$$

showing that J is feasible.

It remains to show that J is optimal. Suppose there is a smaller interjacent set J^*, and let $\mathcal{R}^* = \mathcal{G} - J^*$. By Lemma 12, we can find a set of arcs P^* and a set of terminal arcs Q^* such that every s–t_i path in $\mathcal{R}^*$ contains a member of P^* or ends at a sink corresponding to a member of Q^*, and $|P^*| + |Q^*| \leq r$. So in $\mathcal{G}'$ the following is an s–t' cut:

$$C^* = J^* + P^* + Q^* \tag{66}$$

Also, C^* contains $|Q^*| \leq r$ terminal arcs. Since C' is the side-constrained minimum cut in $\mathcal{G}'$ and $|P^*| + |Q^*| \leq r$,

$$|C'| \; \leq \; |C^*| \; = \; |J^*| + |P^*| + |Q^*| \; \leq \; |J^*| + r \tag{67}$$

But $|J| = |C'| - r$, so

$$|J| \leq |J^*| \tag{68}$$

contradicting the supposition. ■

To find the smallest side-constrained cut C', it is only necessary to solve the $\binom{k}{r}$ minimum cut problems in $\mathcal{G}'$ with each possible set of r terminal arcs at capacity 1, the balance being at infinite capacity. If the smallest side-constrained cut C' uses fewer than r terminal arcs, it will appear whenever a superset of its terminal arcs is allowed in the solution.

In solving the $\binom{k}{r}$ minimum cut problems, it is necessary to leave the r terminal arcs in as capacity-1 arcs eligible to be part of the cut, rather than making them automatically part of the cut (*i.e.* by setting them at capacity 0). An example of this is shown in Figure 35. Using the procedure of Proposition 22, there are three

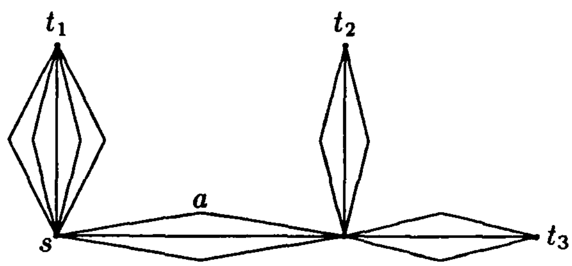


Figure 35: A PMCC Problem with $k = 3$, $r = 2$

minimum cut problems to solve, each with one of the three terminal arcs set at infinite capacity. If the other terminal arcs are set at zero capacity, the best PMCC found is three arcs. If they are set at capacity 1, the correct PMCC of any two of the three arcs at a is found.

If r is fixed, $\binom{k}{r}$ is $O(k^r)$, and so Algorithm 11 is polynomially bounded. If k is fixed, the algorithm is also bounded, since $r < k$. If both r and k vary but their difference is bounded, then the algorithm is bounded by k^{k-r} times the work to find a minimum cut in a general digraph. Only when $k \to \infty$, $r \to \infty$, and $k - r \to \infty$ does this method of solution increase exponentially. Thus we have the following theorem.

Theorem 5 *The single-source partial multicommodity arc-cut problem with arc-disjoint remaining paths (case DeE) is polynomially bounded if r or $k-r$ is bounded.*

Proof: The preceding discussion shows that Algorithm 11 is a polynomial method for this problem. ■

8.3 Generalizing to Cases UeE, DvV, and UvV

With standard reformulations, the method developed in the previous two sections for partial arc-cuts can also be used for edge-cuts in undirected graphs and for node- or vertex-cuts. The only restriction is that the disjointness of the remaining paths must be determined by the same graph elements that make up the cut; the cross-cases are addressed in following sections. This section shows the applicability of the last section's method to UeE, to DvV, and finally to UvV single-source partial multicommodity cuts.

The standard way to reformulate an undirected cut problem as a directed one is to replace each edge with two oppositely directed arcs. It is well-known that the minimum arc-cut will never contain a pair of such opposite arcs, and the set of edges corresponding to the arcs in the minimum directed cut is a minimum undirected cut of the same cardinality. This reformulation also works for partial multicommodity cuts, as the following lemma shows.

Lemma 14 *Given a single-source partial multicommodity edge-cut problem in an undirected graph $\mathcal{G}$, let J be the set of edges corresponding to the members of an optimal solution to the corresponding directed cut problem formed by replacing every* *edge in $\mathcal{G}$ with two oppositely directed arcs. Then J is an optimal solution to the original undirected problem.*

Proof: Let J be an interjacent set formed as described. J must be feasible, since the undirected graph cannot have more s–t_i paths in it than the corresponding directed graph. Suppose there is a smaller feasible interjacent set J^*. We can apply the partition-of-paths argument used on the proof of Lemma 12 to show that there must be in $\mathcal{G} - J^*$ a set of edges P^* and a set of terminals Q^* such that each s–t_i path in $\mathcal{G} - J^*$ is covered by either P^* or Q^*, and $|P^*| + |Q^*| \leq r$. Let

$$C' = J^* + P^* + \{t_i t' : t_i \in Q^*\} \tag{69}$$

Then C' is a full s–t' cut in the sink-augmented undirected graph $\mathcal{G}'$, and divides $\mathcal{G}'$ into at least two pieces, one of which contains s and another t'. Let $\mathcal{G}'_{\text{dir}}$ be the sink-augmented graph for the corresponding directed cut problem, and let C'_{dir} be the set of arcs corresponding to C' in the s-to-t' direction. This is well-defined because of the way C' divides $\mathcal{G}'$. Also, C'_{dir} contains $|Q^*| \leq r$ terminal arcs, and

$$|C'_{\text{dir}}| \;=\; |C'| \;\leq\; |J^*| + r \;<\; |J| + r \tag{70}$$

But by the method of construction of J and Proposition 22, the smallest such side-constrained s–t' cut in $\mathcal{G}'_{\text{dir}}$ is of cardinality $|J| + r$. This is a contradiction. ■

The directed node-cut, node-disjoint problem DvV can be reformulated as a capacitated version of the DeE problem in the standard way by splitting each node into an in-node and an out-node connected by a capacity-1 *node arc*; all the other arcs of $\mathcal{G}$ then go from an out-node to an in-node and have infinite capacity. The arguments of lemmas 11 and 12 and Proposition 22 apply equally well to this reformulated problem. Let $\mathcal{G}_N$ be $\mathcal{G}$ reformulated as described, and let $\mathcal{G}'_N$ be that digraph augmented with t' and capacity-1 terminal arcs. The optimal partial node-cut must correspond to the minimum arc-cut in $\mathcal{G}'_N$ that consists of only node arcs and terminal arcs, and no more than r of the latter. This minimum cut can be found by evaluating $\binom{k}{r}$ such arc-cuts. Thus the node version of the single-source PMCC problem is polynomially bounded under the same restrictions as the arc version.

The remaining non-cross-case is UvV. Using the same methods, it is easy to see that for an optimal J, every s–t_i path in $\mathcal{R}$ must either be covered by a set of vertices P or go to a set of sinks Q, with $|P| + |Q| = r$. Therefore J can be formed from the smallest s–t' vertex-cut C' in $\mathcal{G}'$ that includes no more than r of the k vertices t_i: J is C' less any subset of r vertices that includes all vertices t_i in C'. Finding the optimal solution in the general case requires evaluating $\binom{k}{r}$ minimum vertex-cuts in $\mathcal{G}'$.

We summarize this section in the following theorem.

Theorem 6 *The single-source partial multicommodity cut problem with r or $k - r$ bounded is a polynomially bounded problem for node-cuts and node-disjoint paths in directed graphs (DvV), and for either edge-cuts and edge-disjoint paths or vertex-cuts and vertex-disjoint paths in undirected graphs (UeE and UvV).*

Proof: Lemma 14 shows that a UeE problem can be reformulated as a DeE problem, which has been shown to be polynomial under the given restrictions (Theorem 5). The foregoing discussion shows that DvV and UvV problems can be solved by evaluating $\binom{k}{r}$ vertex-cut problems, so these cases are also polynomial. ■

8.4 The Cross-case DeV

We next consider the problem of removing the smallest number of arcs from a digraph so that there are no more than r node-disjoint paths left between a source s and distinct members of the set of sinks $\{t_1, \ldots, t_k\}$. This might represent a situation in a pipeline network in which the factor limiting throughput was pumping capacity at the nodes, but it is desired to know the consequences of closing some of the lines. We could imagine a question like, "What is the smallest number of lines closed that will result in only r of my customers being served?" This problem would be formulated as a DeV partial multicommodity cut problem.

Let such a PMCC problem be given with $r = 1$. Without loss of generality we can assume that s has zero in-degree and all nonterminal nodes have non-zero in- and out-degrees. The sinks t_i must have non-zero in-degree, but may or may not have out-arcs.

Candidate PMCCs can be found by finding the minimum arc-cuts between s and each of the k sets of all sinks but one. Suppose that the smallest of these k candidate PMCCs is of cardinality j. If this size-j candidate is not the optimum PMCC, there must be another set of arcs J^* such that $|J^*| < j$ and all s–t_i paths in $\mathcal{G} - J^* = \mathcal{R}$ go through one node v (which could be a t_i). Let all the nodes that can be reached from s in $\mathcal{R} - v$ be called the *s-side* of $\mathcal{G}$ with respect to J^*, and let all the nodes from which a sink can be reached be called the *t-side*. All the arcs on v can be partitioned into *s-arcs* from or to the s-side, *t-arcs* from or to the t-side, and all other arcs. Let a *supreme directed cut* in a multicommodity flow digraph be a set of arcs or nodes that covers all s_i–t_i paths. Then J^* plus either all arcs to the t-side or all arcs from the s-side forms a supreme directed arc-cut in $\mathcal{G}$. In other words, there is a supreme cut of cardinality ℓ with one node covering b of its arcs, and $\ell - b < j$.

Now one node can cover only as many arcs in a minimum arc-cut as the lesser of its in-degree and out-degree. If we know that the largest such *lesser degree* for any node in $\mathcal{G}$ is g, we can use the following algorithm to find the cardinality of the smallest DeV PMCC in an $r = 1$ single-source problem.

Algorithm 12 (Single-source DeV $r = 1$ PMCC)

input: $\mathcal{G}$ for a single-source DeV PMCC problem with $r = 1$ and largest lesser degree g

output: cardinality of a minimum interjacent set

begin

(1) Find the smallest of the k cuts between s and all sinks but one. Set j equal to its cardinality.

(2) Find the smallest supreme directed cut. Let its cardinality be ℓ, and set $a = \ell - j$.

(3) For each node v, in order of decreasing lesser degree from g to $a + 1$:

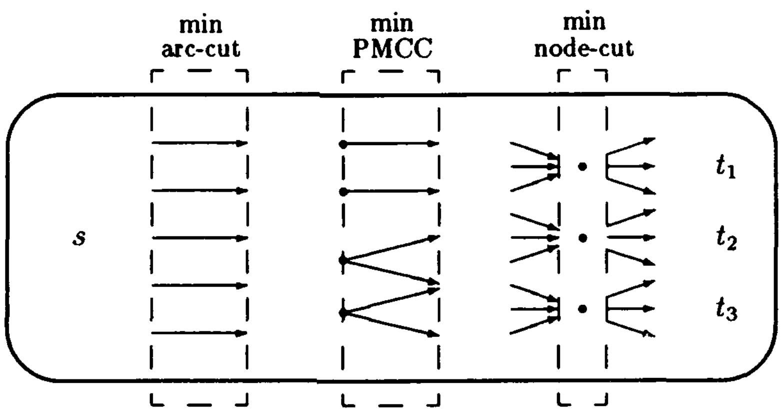
(a) Find the smallest supreme directed cut in $\mathcal{G} - v$. Let its cardinality be j_v.
(b) **if** $j_v < j$, set $j = j_v$
set $a = \ell - j_v$ (changing the limits of iteration in Step *3*)

(4) **return** j

end

If g is not known, then Step 3 can be executed for every node, not just those of lesser degree greater than a. In either case, the work involved is $O(n\eta)$, where $n = |V|$ and η is the work to find a minimum arc-cut.

Now we would like to generalize the method to handle any value of r. Unfortunately, this offers some difficulties. Each supreme arc-cut induces a candidate PMCC: given such a cut, find the set of r nodes that covers the maximum number of arcs in it, and form the interjacent set from the uncovered arcs. Similarly, a supreme node-cut also induces a candidate PMCC. For a node v in a supreme node-cut C, define $mindeg(v, C)$ as the minimum degree of v directed across C, *i.e.* as the lesser of the number of arcs into v from the s-side and the number of arcs out of v to the t-side of $\mathcal{G} - C$. Given such a node-cut C, for all $v \in C$ except the r of highest mindeg, put in the interjacent set the mindeg(v, C) arcs that connect v to one side of the node-cut C and are oriented in the s-to-t direction. However, neither the minimum supreme arc-cut nor the minimum supreme node-cut necessarily induces the minimum DvE PMCC, as Figure 36 shows.



	min arc-cut	min PMCC	min node-cut
Arc-cut size	5	6	9
Node-cut size	5	4	3
DeV PMCC size	3	2	3

Figure 36: Single-source DeV Problem, $k = 3$, $r = 2$

To find the optimal interjacent set for $r \geq 2$, we will need the following two propositions. The first is the equivalent of Lemma 13 for the DeE case.

Proposition 23 *Suppose J is an optimal partial multicommodity directed cut in a single-source, arc-cut, node-disjoint problem. Then all s–t_i paths in $\mathcal{G} - J = \mathcal{R}$ can be partitioned into two sets as follows. One set consists of paths to a member of a set of sinks Q, the other set is covered by a set of nodes P, and $|Q| + |P| = r$. Furthermore, there are no such sets Q' and P' that cover all paths and have $|Q'| + |P'| < r$.*

Proof: Let $\mathcal{R}'$ be the union of all s–t_i paths in $\mathcal{R}$, plus t' and the terminal arcs $t_i t'$, and with all nodes (except the terminals and t') split into an in-node and an out-node. Put capacity 1 on the node arcs and terminal arcs, let other arcs have unlimited capacity, and find a feasible maximum s–t' flow. The total flow can be no more than r, so the minimum cut is no more than r. Neither can the flow be less than r: if it is, one arc can be deleted from J and added to $\mathcal{R}$ and result in a maximum flow of no more than r in $\mathcal{R}'$, contradicting the optimality of J. Then the minimum cut in $\mathcal{R}'$ must be exactly r, and it must correspond to a set Q of sinks plus a set P of nodes, where P and Q together cover all paths and $|Q| + |P| = r$. Furthermore, if there were a set of sinks Q' and a set of nodes P' with $|Q'| + |P'| < r$ that covered all s–t_i paths in $\mathcal{R}$, then the maximum flow in $\mathcal{R}'$ would also have to be less than r. ∎

Proposition 24 *Given J, P, and Q as defined in Proposition 23, J is a minimum arc-cut between s and $T - Q$ in $(\mathcal{G} - P) - Q$.*

Proof: Suppose C^* is a minimum arc-cut as described and $|C^*| < |J|$. Then in $\mathcal{G} - C^*$ any s–t_i path must either go to a member of Q or pass through a member of P. There can be no more than $|P| + |Q| = r$ node-disjoint such paths to different sinks. Therefore C^* is a feasible PMCC, and $|C^*| \geq |J|$, a contradiction. ■

From Proposition 24, it is clear that for some set N consisting of r nodes in $\mathcal{G}$ (excluding s, but possibly including sinks), the optimal PMCC J is the minimum supreme arc-cut in $\mathcal{G} - N$. This means we can find J by evaluating $\binom{n}{r}$ minimum cut problems ($n = |V|$). This involves $O(n^r \eta)$ or $O(n^{n-r} \eta)$ work, where a minimum cut can be found in $O(\eta)$ work, and is polynomial if $n - r$ or r is bounded.

Though it will not change the bound, some of the evaluations may be deleted by considering the non-sink nodes in order of decreasing lesser degree. For any candidate PMCC J and its sets P and Q as defined in Proposition 23, a supreme cut is formed by J plus the in-arcs on the members of Q plus either the in-arcs or the out-arcs of each member of P. If we define the *lesser degree of a sink* as its in-degree and *lesdeg*(v) as the lesser degree of v, the cardinality of this supreme cut is

$$|J| + \sum_{v \in P} \text{lesdeg}(v) + \sum_{v \in Q} \text{lesdeg}(v) = c_{\text{sup}} \tag{71}$$

If the smallest supreme cut in $\mathcal{G}$ is of size c, we can conclude that

$$c_{\text{sup}} \geq c \tag{72}$$

Once we have a candidate PMCC of size j, to have $|J| < j$ we must have

$$j \;>\; c_{\text{sup}} - \sum_{v \in P \cup Q} \text{lesdeg}(v) \;\geq\; c - \sum_{v \in P \cup Q} \text{lesdeg}(v) \tag{73}$$

$$\sum_{v \in P \cup Q} \text{lesdeg}(v) \;>\; c - j \tag{74}$$

Thus we only need to consider sets P and Q whose total lesser degree is at least $c - j + 1$.

8.5 The Cross-case DvE

This is the problem of removing the smallest number of nodes from a digraph so that there are no more than r arc-disjoint paths left between the source and distinct members of the set of sinks. This might represent a pipeline network in which capacity was limited by the pipes, but questions are being asked about the consequences of failures at pumping stations at the nodes. Without loss of generality we can assume that s has zero in-degree, the sinks have positive in-degree, and all terminals (except possibly the sinks) have positive out-degree.

Let us first consider the case of $r = 1$. Candidate PMCCs can be found by finding the minimum node-cuts between s and each of the k sets of all terminals but one. Suppose the smallest of these is of cardinality j. The only possibility for a smaller interjacent set is that there might be a full multicommodity node-cut also of cardinality j that included a node that was either the head of exactly one arc from the s-side of the cut or the tail of exactly one arc to the t-side of the cut. Let us call such a node a *mindeg-1* node in relation to a given node-cut (see Figure 37). If a mindeg-1 node is deleted from an FMCC, the remaining graph $\mathcal{R}$ can contain only one arc-disjoint path from s to any t_i. This PMCC would then be optimal, at cardinality $j - 1$.

A cut containing a mindeg-1 node, if it exists, can be found with one more minimum cut evaluation in the following way. Reformulate the problem into an arc-cut problem in the standard way by splitting each node into an in-node and an out-node connected by a weight-1 node arc, but weight all the other arcs at $1 - \epsilon$ (where

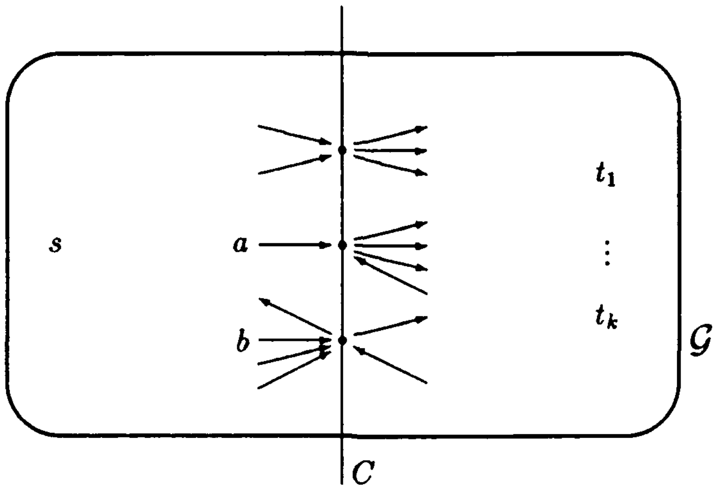


Figure 37: Mindeg-1 Nodes (a and b)

$\epsilon < 1/|V|$) instead of infinity when finding the minimum s–t' supreme arc-cut. With that weighting, any node-cut of size j with ℓ mindeg-1 nodes will appear as a size $j - \ell\epsilon$ cut (see Figure 38).

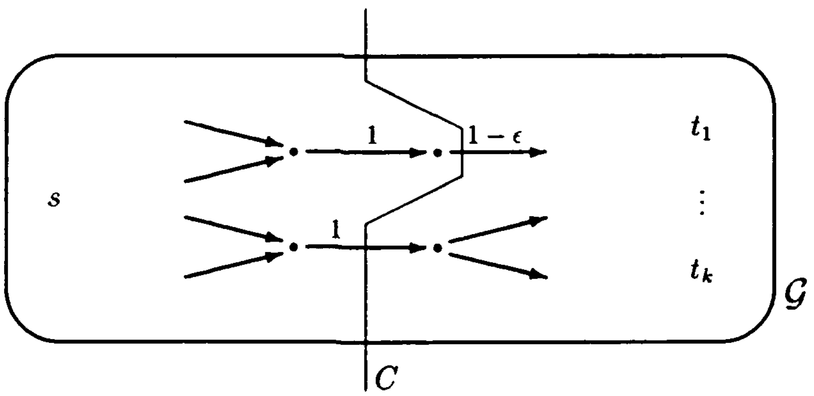
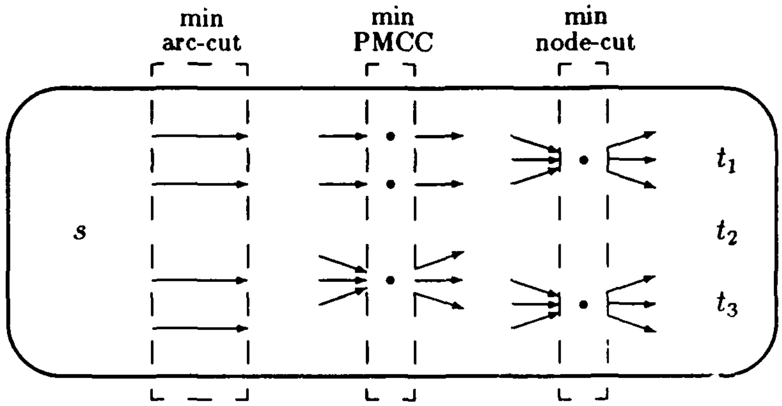


Figure 38: Finding a Node-cut Containing a Mindeg-1 Node

Thus, the optimum PMCC in the DvE $r = 1$ case can be found by evaluating and comparing $k + 1$ weighted minimum cuts. Unfortunately, this method of finding FMCC-based PMCCs will not work with $r > 1$.

Let us consider the problem with $r \geq 2$. As with the DeV case in the previous section, the minimum supreme arc-cut and the minimum supreme node-cut can each be used in a straightforward way to generate a candidate DvE interjacent set. However, as Figure 39 shows, these candidates are not necessarily optimal.



Arc-cut size	4	5	6
Node-cut size	4	3	2
DvE PMCC size	2	1	2

Figure 39: Single-source DvE Problem, $k = 3$, $r = 2$

If the minimum in- and out-degrees of every node in $\mathcal{G}$ are greater than r, then it is clear that any PMCC will have to be an FMCC also. In general, since the removal of one node can mean the removal of any number of arc-disjoint paths, $\mathcal{R}$ may have strictly fewer than r disjoint s–t_i paths. (Lemmas 3 and 4 do not apply in the DvE case.) Consequently the same interjacent set may be optimal for different values of r.

Using the same approach as for the other cases of the single-source problem, it is clear that for some set of arcs P and some set of sinks Q, with $|P| + |Q| = r$, the optimal DvE PMCC will also be the minimum full s–t_i node-cut in $(\mathcal{G} - P) - Q$. One way to find the optimal PMCC is to evaluate all $\binom{m+k}{r}$ such node-cuts

($m = |E|$). This method results in a combinatorial explosion unless $m + k$, r, or $m + k - r$ is bounded.

8.6 Undirected Nearly T-Planar Graphs (UeEP)

If a single-source undirected multicommodity flow graph $\mathcal{G}$ is planar and has all its sinks on the boundary, call it *sink-planar.* This section will develop a polynomially-bounded algorithm for sink-planar graphs.

Without loss of generality, number the sinks from t_1 to t_k clockwise around $\mathcal{G}$. Form $\mathcal{G}'$ from $\mathcal{G}$ by adding the super-sink t' and the k terminal edges t_it' (see Figure 40; note that $\mathcal{G}'$ is planar.) From Lemma 14, we know that the smallest PMCC J corresponds to the smallest s–t' cut C' that contains no more than r terminal edges, and that J is found by deleting from C' any r edges, to include any terminal edges in C', so that $|J| = |C'| - r$. The following characerizes C', assuming it contains terminal edges.

C' must be a minimal cut in $\mathcal{G}'$, corresponding to a circuit in the geometric-dual of $\mathcal{G}'$. It will consist of alternating sets of edges $A_1, T_1, A_2, T_2, \ldots, A_\ell, T_\ell$ ($\ell \leq k/2$), where each A_i separates from the rest of $\mathcal{G}$ a set of terminals from some t_a clockwise to t_b (in Figure 40, A_1 separates t_3 to t_5), and each T_i consists of a set of consecutive terminal arcs (in Figure 40, T_1 consists of t_6t' through t_9t'). The sets of terminals associated with each of the A_i and T_i form a partition of the set of sinks $T - \{s\}$. As in previous chapters, let us use the conventions that $t_i \equiv t_{i \pm k}$, $[t_i, t_j]$ means the set of terminals from t_i clockwise around $\mathcal{G}$ through t_j, and $[t_i, t_{i-1}] = \phi$.

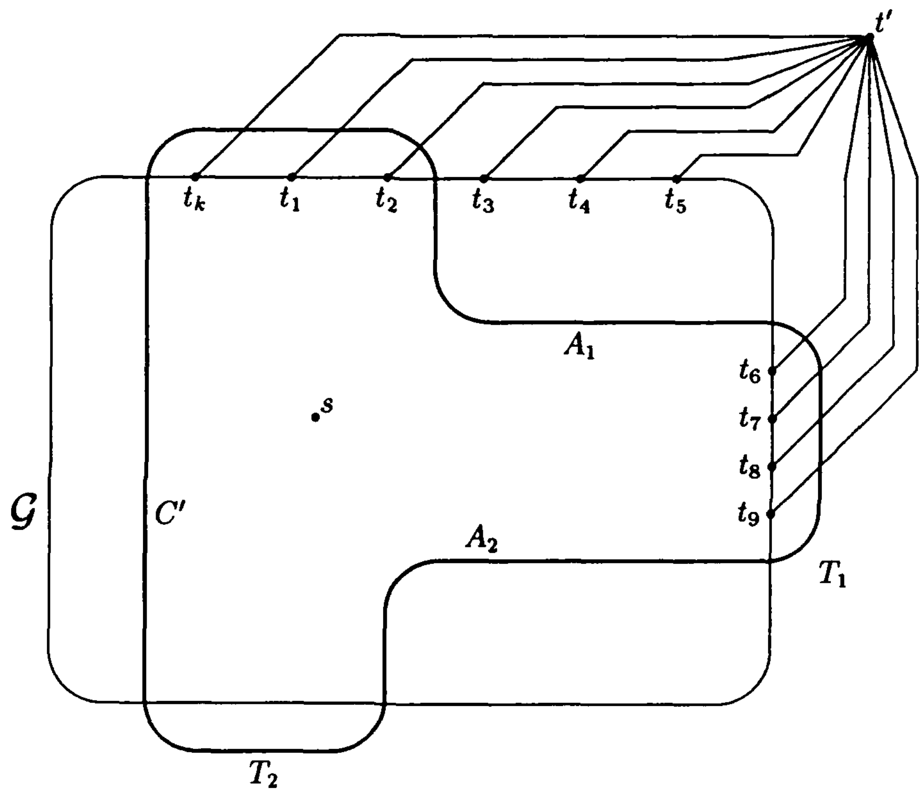


Figure 40: FMCC Corresponding to Optimal PMCC in Sink-planar Graph

We know that

$$r \geq \sum_i |T_i| \equiv q \tag{75}$$

Also,

$$\begin{aligned} |J| &= |C''| - r \\ &= \sum_i |A_i| + \sum_i |T_i| - r \\ &= \sum_i |A_i| + q - r \end{aligned} \tag{76}$$

and it is characteristic of the optimal C'' that this quantity is minimized over all similar cuts. Since r is constant, C'' will minimize $\sum |A_i| + q$, the size of the union of all the cuts A_i plus the number of terminals not separated from s by any of them.

Another characterization is in the following proposition.

Proposition 25 *In a sink-planar problem with an optimal solution J corresponding to s–t' cut C'' in $\mathcal{G}'$, if a minimal cut $A_i \subset C''$, $A_i \subset E$, separates exactly $[t_a, t_b]$ from s, then it is a minimum cut in $\mathcal{G}$ between s and $[t_a, t_b]$.*

Proof: Let A be the minimum s–$[t_a, t_b]$ cut in $\mathcal{G}$, and suppose $|A| < |A_i|$. A must contain edges in the boundary of $\mathcal{G}$, for otherwise it would be a feasible s–t' cut with cardinality smaller than that of $C'' \supset A_i$. A must therefore separate a set of terminals $[t_c, t_d]$ from s, with $[t_a, t_b] \subseteq [t_c, t_d]$ (see Figure 41). If $a = c$ and $b = d$, then a C''' can be formed from C'' by substituting A for A_i, with $|C'''| < |C''|$, contradicting the optimality of C''. Suppose $a \neq c$ or $b \neq d$. As previously noted, C'' can be represented by a simple closed curve that intersects $\mathcal{G}$ only on members of C''. A can similarly be represented by a curve across $\mathcal{G}$. Form C''' from the edges in the portion of A in

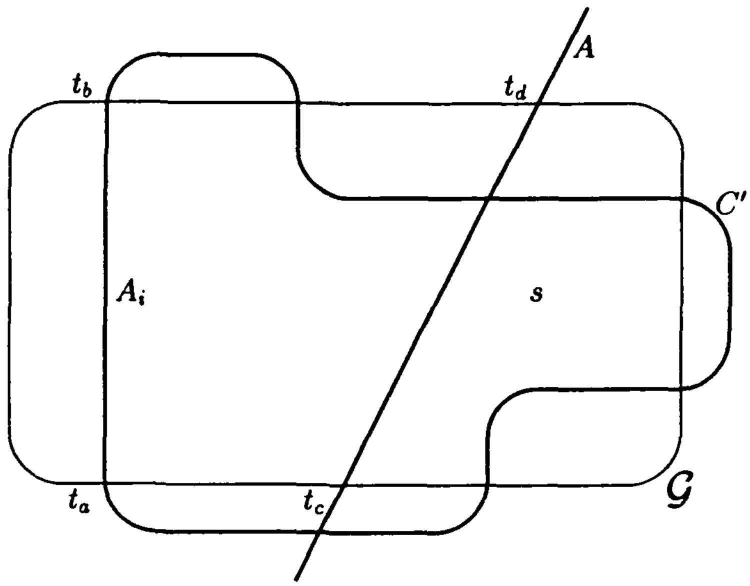


Figure 41: Proof of Proposition 25

the interior of C', plus those in the loop of C' on the side of A that includes s (see Figure 41). C'' is a valid s–t' cut that includes no more than r terminal edges. Also, $|A| < |A_i| \Rightarrow |C''| < |C'|$, since A_i is a subset of the portion of C' deleted to make C''. This again is a contradiction. Therefore A must be no smaller than A_i, so that A_i is a minimum s–$[t_a, t_b]$ cut. ■

Let $M(a, b)$ be the cardinality of the minimum cut between s and $[t_a, t_b]$ if that cut leaves all of $[t_{b+1}, t_{a-1}]$ connected to s, and arbitrarily large if it does not. Then the size of each A_i in C' can be found once its span is determined by finding $M(a, b)$, without reference to the rest of C'. The size of each possible cut A_i can be found by solving $O(k^2)$ planar minimum cut problems. The algorithm that is developed in this section compares every feasible combination of cuts A_i and picks the combination that corresponds to the smallest s–t' cut that uses no more than r terminal edges, which in turn corresponds to a minimum PMCC. The feasible combinations of cuts

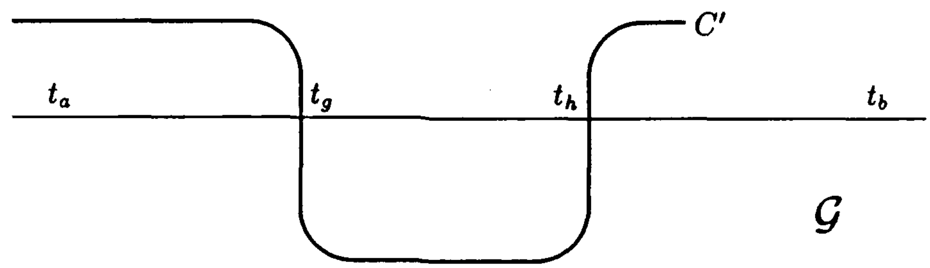


Figure 42: The Recursive Formula for $\mathbf{A}([t_a, t_b], q)$

will be found using a dynamic programming recursion.

Suppose that for some set of terminals $[t_a, t_b]$, no cut A_i separates terminals both inside and outside the set from s. Also, assume that no more than q terminals in $[t_a, t_b]$ can be left unseparated from s, *i.e.* that $r - q$ terminals are unseparated by C' in $[t_{b+1}, t_{a-1}]$. Make the definition

$$\mathbf{A}([t_a, t_b], q) \equiv \text{cardinality of a part of } C' \text{ that runs from } t_a \text{ to } t_b, \text{ assuming } r-q \text{ terminals in } [t_{b+1}, t_{a-1}] \text{ are unseparated} \quad (77)$$

The function $\mathbf{A}$ can be calculated recursively as follows. Suppose the first cut A_i in $[t_a, t_b]$ cuts off $[t_g, t_h]$ from s (see Figure 42). Then the following hold:

- Terminals t_a-t_g-t_h-t_b occur in that order clockwise around the perimeter of $\mathcal{G}$, except that possibly $a = g$, $g = h$, and/or $h = b$.
- The terminal arcs on $[t_a, t_{g-1}]$ are in C'. Since there can be no more than q such arcs, we must have $|[t_a, t_{g-1}]| \le q$.
- $|A_i| = M(g, h)$
- The cardinality of all cuts A_j on $[t_{h+1}, t_b]$ is $\mathbf{A}([t_{h+1}, t_b], q - |[t_a, t_{g-1}]|)$.

So we have following recursion for **A**:

$$\mathbf{A}([t_a, t_b], q) = \min_{t_g \in Q^t} \left\{ D_{[a,g)} + \min_{t_h \in [t_g, t_b]} \left\{ M(g, h) + \mathbf{A}([t_{h+1}, t_b], q - D_{[a,g)}) \right\} \right\} \quad (78)$$

where

$$D_{[a,g)} \equiv \|[t_a, t_{g-1}]\|$$

$$Q^t \equiv \{t_g : t_g \in [t_a, t_{b+1}] \ , \ D_{[a,g)} \leq q\}$$

and min over a null set is taken to be zero. The recursion can be initiated with

$$\mathbf{A}(\phi, q) = 0 \quad \text{for all } a, \ \ q \geq 0 \quad (79)$$

The function **A** can be used to find C' if the latter includes at least one terminal edge. If C' includes $t_i t'$, then

$$|C'| = \mathbf{A}([t_{i+1}, t_{i-1}], r - 1) \quad (80)$$

If C' includes no terminal edges, then it is the minimum supreme cut. In either case, r edges (including any terminal edges) may be deleted from C' to find an optimal PMCC. Combining both possibilities, we get the following algorithm.

Algorithm 13 (Single-source Sink-planar UeEP PMCC)

input: $\mathcal{G}'$ and r for a single-source sink-planar UeEP PMCC problem

output: cardinality of an optimal interjacent set J in $\mathcal{G}$

begin

compute $c_o' =$ size of minimum supreme cut

for every set $[t_a, t_b]$ of cardinality less than k

  compute $M(a, b)$

for $q = 0, \ldots, r$

  set $\mathbf{A}(\phi, q) = 0$

for $i = 1, \ldots, k-1$

  for every set $[t_a, t_b]$ of cardinality i

    for $q = 0, \ldots, r$

      compute $\mathbf{A}([t_a, t_b], q)$ using (78)

return $\min\left\{c_o' \ , \ \min_{1 \leq j \leq k}\{\mathbf{A}([t_{j+1}, t_{j-1}], r-1)\}\right\} - r$

end

The complexity is $O(k^2\gamma + r^2k^3)$, where γ is the complexity of finding a planar minimum cut. We conclude with the following theorem.

Theorem 7 *The single-source sink-planar partial multicommodity edge-cut problem with edge-disjoint remaining paths is polynomially bounded.*

Proof: Algorithm 13 provides a polynomial procedure for this problem. ■

9 CONCLUSIONS AND FUTURE WORK

This chapter summarizes the work and points out some likely directions for future investigations. The results in Chapter 4 showed that the general MCC problem was intractable, so investigation was focused on restricted special cases that offered promise. Following are discussions of the four major lines of research on multicommodity cuts: full edge-cuts, partial edge-cuts, and vertex-cuts, all in planar graphs; and the single-source partial cut problem in general graphs.

In the area of full multicommodity cuts in T-planar graphs the research seems to be most nearly complete. There are polynomial algorithms for the weighted problem with fixed k (Alg. 2) and in non-crossing graphs with varying k (Alg. 4), and in view of Proposition 10 there is little more to be expected, except perhaps in more restricted cases. It is perhaps something of a disappointment that the T-planar FMCC problem should turn out to be NP-complete even in classes of graphs as restricted as grids and weighted trees. Its status in unweighted trees would be interesting to know, but this is still open. If the terminals are not restricted to the boundary the problem can certainly be no easier than the T-planar version, so polynomial algorithms for fixed k are what is to be hoped for. Algorithm 3 was developed for the $k = 3$ case. It would be good to find a unifying treatment that would work for any fixed k in general planar graphs.

Prospects for efficient algorithms for partial multicommodity edge-cuts are severely limited by the intractability of the closely associated full multicommodity cut and disjoint paths problems. This work had some success with the PMCC problem in T-planar non-crossing graphs for $r = k - 1$ (Alg. 6), $r = k - 2$ (Alg. 7), and $r = 1$ (Alg. 8). It would be good to have a unifying treatment that would close the gap ($k - 3 \geq r \geq 2$). It seems clear that what is needed is an account of the characteristics of T-planar *q-disjoint-path* graphs, *i.e.* multicommodity flow graphs in which only q disjoint s_i–t_i paths can be found. Also, only the unweighted PMCC problem was investigated. The corresponding weighted problem is still open.

Chapter 7 showed that some of the major results for multicommodity edge-cuts can be extended readily to vertex-cuts. It seems likely that efforts to extend the remaining results (full three-commodity cuts in general planar graphs; partial cuts in T-planar graphs with $r \leq k - 2$) would be rewarded.

The single-source problem turned out to be relatively easy, and it was handled successfully in the greatest variety of cases. However, some interesting open questions remain. The question of greatest theoretical interest is whether the general problem is NP-complete for arbitrary k and r; it was shown to be polynomial for r or $k - r$ bounded (Proposition 22). Also, the cross-cases DvE and DeV were handled satisfactorily only for $k = 1$, and the undirected cross-cases were not addressed. The weighted single-source problem remains open. Finally, it would be interesting to see if the specialized algorithm for the undirected PMCC problem in sink-planar graphs (Alg. 13) could be extended to directed graphs.

Except for single-source problems, this research concentrated on multicommodity cuts in undirected graphs. It would be interesting to know if these or similar methods could be used to find full or partial multicommodity cuts in directed planar or T-planar graphs.

REFERENCES

[1] Aneja, Y. P. and R. R. Vemuganti. 1977. A Row-generation Scheme for Finding a Multi-commodity Minimum Disconnecting Set. *Management Science*, **23**, 652-659.

[2] Assad, A. A. 1978. Multicommodity Network Flows—A Survey. *Networks*, **8**, 37-91.

[3] Becker, M. and K. Mehlhorn. 1986. Algorithms for Routing in Planar Graphs. *Acta Informatica*, **23**, 163-176.

[4] Bellmore, M., H. J. Greenberg, and J. J. Jarvis. 1970. Multi-commodity Disconnecting Sets. *Management Science*, **16**, B427-B433.

[5] Dahlhaus, E., D. S. Johnson, C. H. Papadimitriou, P. D. Seymour, and M. Yannakakis. 1992. The Complexity of Multiway Cuts. *Proceedings of the Twenty-fourth Annual ACM Symposium of the Theory of Computing* (New York: ACM Press), 241-251.

[6] Ding, G., A. Schrijver, and P. D. Seymour. 1992. Disjoint Paths in a Planar Graph—A General Theorem. *SIAM Journal on Discrete Mathematics*, **5**, 112-116.

[7] Erickson, R. E., C. L. Monma, and A. F. Veinott. 1987. Send and Split Method for Minimum-concave-cost Network Flows. *Mathematics of Operations Research*, **12**, 634-664.

[8] Evans, J. R. and J. J. Jarvis. 1979. Network Topology and Integral Multicommodity Flow Problems. *Networks*, **8**, 107-119.

[9] Even, S., A. Itai, and A. Shamir. 1976. On the Complexity of Timetable and Multicommodity Flow Problems. *SIAM Journal on Computing*, **5**, 691-703.

[10] Floyd, R. W. Algorithm 97: Shortest Path. *Communications of the ACM*, **5**, 345.

[11] Ford, L. R. and D. R. Fulkerson. 1956. Maximal Flow through a Network. *Canadian Journal of Mathematics*, **8**, 399-404.

[12] Fortune, S., J. Hopcroft, and J. Wyllie. 1980. The Directed Subgraph Homeomorphism Problem. *Journal of Theoretical Computer Science*, **10**, 111-121.

[13] Frank, A. 1985. Edge-disjoint Paths in Planar Graphs. *Journal of Combinatorial Theory*, Series B, **39**, 164-178.

[14] Frank, A. 1990a. On Disjoint Homotopic Paths in the Plane. *Polyhedral Combinatorics*, W. Cook and P. D. Seymour, eds. (Morristown, NJ: American Mathematical Society), 163–170.

[15] Frank, A. 1990b. Packing Paths, Circuits, and Cuts—A Survey. *Paths, Flows and VLSI-layout*, B. Korte, L. Lovász, H. J. Prömel, and A. Schrijver, eds. (New York: Springer-Verlag), 47–100.

[16] Frank, A. 1990c. Packing Paths in Planar Graphs. *Combinatorica*, **10**, 325–331.

[17] Frank, A. and A. Schrijver. 1990. Vertex-disjoint Simple Paths of Given Homotopy in a Planar Graph. *Polyhedral Combinatorics*, W. Cook and P. D. Seymour, eds. (Morristown, NJ: American Mathematical Society), 139–161.

[18] Frederickson, G. N. 1987. Fast Algorithms for Shortest Paths in Planar Graphs, with Applications. *SIAM Journal on Computing*, **16**, 1004-1022.

[19] Garey, M. R., and D. S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness* (New York: W. H. Freeman and Company).

[20] Garey, M. R., D. S. Johnson, and L. Stockmeyer. 1976. Some Simplified *NP*-Complete Graph Problems. *Theoretical Computer Science*, **1**, 237-267.

[21] Hassin, R. 1984. On Multicommodity Flows in Planar Graphs. *Networks*, **14**, 225–235.

[22] Hu, T. C. 1963. Multi-commodity Network Flows. *Operations Research*, **11**, 344-360.

[23] Jarvis, J. J. and J. B. Tindall. 1972. Minimal Disconnecting Sets in Directed Multi-commodity Networks. *Naval Research Logistics Quarterly*, **19**, 681–690.

[24] Kennington, J. L. 1978. A Survey of Linear Cost Multi-commodity Network Flows *Operations Research*, **26**, 209–236.

[25] Kohn, M. C. and R. D. Lemieux. 1991. Identification of Regulatory Properties of Metabolic Networks by Graph Theoretical Modeling. *Journal of Theoretical Biology*, **150**, 3–25.

[26] Lai, T.-H. and A. Sprague. 1987. On the Routability of a Convex Grid. *Journal of Algorithms*, **8**, 372–384.

[27] Larman, D. G. and P. Mani. 1970. On the Existence of Certain Configurations within Graphs and the 1-Skeleton of Polytopes. *Proceedings of the London Mathematical Society*, **20**, 144–160.

[28] Matsumoto, K., T. Nishizeki, and N. Saito. 1985. An Efficient Algorithm for Finding Multicommodity Flows in Planar Networks. *SIAM Journal on Computing*, **14**, 289–302.

[29] Mehlhorn, K. and F. P. Preparata. 1986. Routing through a Rectangle. *Journal of the ACM*, **33**, 60–85.

[30] Middendorf, M. and F. Pfeiffer. 1990. On the Complexity of the Disjoint Paths Problem. *Polyhedral Combinatorics*, W. Cook and P. D. Seymour, eds. (Morristown, NJ: American Mathematical Society), 171–178.

[31] Okamura, H. 1983. Multicommodity Flows in Graphs. *Discrete Applied Mathematics*, **6**, 55–62.

[32] Okamura, H. and P. D. Seymour. 1981. Multicommodity Flows in Planar Graphs. *Journal of Combinatorial Theory*, Series B, **31**, 75–81.

[33] Nagamochi, H. and T. Ibaraki. 1989. On Max-flow Min-cut and Integral Flow Properties in Directed Networks. *Information Processing Letters*, **31**, 279–285.

[34] Perl, M. and Y. Shiloach. 1978. Finding Two Disjoint Paths between Two Pairs of Vertices in a Graph. *Journal of the ACM*, **25**, 1–9.

[35] Provan, J. S. 1983. A Polynomial Algorithm for the Steiner Tree Problem on Terminal-planar Graphs. Technical Report 83/10, Department of Operations Research, University of North Carolina, Chapel Hill, NC.

[36] Provan, J. S. 1988. Convexity and the Steiner Tree Problem. *Networks*, **18**, 55-72.

[37] Robertson, N. and P. D. Seymour. 1985. Disjoint Paths—A Survery. *SIAM Journal on Algebraic and Discrete Methods*, **6**, 300–305.

[38] Robertson, N. and P. D. Seymour. 1986. Graph Minors VI: Disjoint Paths across a Disc. *Journal of Combinatorial Theory*, Series B, **41**, 115–138.

[39] Robertson, N. and P. D. Seymour. 1988. Graph Minors VII: Disjoint Paths on a Surface. *Journal of Combinatorial Theory*, Series B, **45**, 212–254.

[40] Rothschild, B. and A. Whinston. 1966. On Two Commodity Network Flows. *Operations Research*, **14**, 377–387.

[41] Sakarovitch, M. 1973. Two Commodity Network Flows and Linear Programming. *Mathematical Programming*, **4**, 1–20.

[42] Schrijver, A. 1983. Min-max Results in Combinatorial Optimization. *Mathematical Programming*, A. Bachem, M. Grötschel, and B. Korte, eds. (New York: Springer-Verlag), 439–500.

[43] Schrijver, A. 1989. The Klein Bottle and Multicommodity Flows. *Combinatorica*, **9**, 375–384.

[44] Schrijver, A. 1990. Homotopic Routing Methods. *Paths, Flows, and VLSI-layout*, B. Korte, L. Lovász, H. J. Prömel, and A. Schrijver, eds. (New York: Springer-Verlag), 329-371.

[45] Schrijver, A. 1991. Short Proofs on Multicommodity Flows and Cuts. *Journal of Combinatorial Theory*, Series B, **53**, 32–39.

[46] Schrijver, A. No date. Multicommodity Flows and Disjoint Paths. Unpublished MS.

[47] Sebő, A. 1990. The Cographic Multiflow Problem: An Epilogue. *Polyhedral Combinatorics*, W. Cook and P.D. Seymour, eds. (Morristown, NJ: American Mathematical Society), 203–234.

[48] Seymour, P. D. 1980a. Disjoint Paths in Graphs. *Discrete Mathematics*, **29**, 293–309.

[49] Seymour, P. D. 1980b. Four-terminus Flows. *Networks*, **10**, 79–86.

[50] Seymour, P.D. 1981a. Matroids and Multicommodity Flows. *European Journal of Combinatorics*, **2**, 257–290.

[51] Seymour, P. D. 1981b. On Odd Cuts and Plane Multicommodity Flows. *Proceedings of the London Mathematical Society*, Series 3, **42**, 178–192.

[52] Shiloach, Y. 1980. A Polynomial Solution to the Undirected Two Paths Problem. *Journal of the ACM*, **27**, 445–456.

[53] Suzuki, H., A. Ishiguro, and T. Nishizeki. 1990. Edge-disjoint Paths in a Grid Bounded by Two Nested Rectangles. *Discrete Applied Mathematics*, **27**, 157–178.

[54] Thomassen, C. 1985. The 2-Linkage Problem for Acyclic Digraphs. *Discrete Mathematics*, **55**, 73–87.

[55] Watkins, M. E. 1968. On the Existence of Certain Disjoint Arcs in Graphs. *Duke Mathematical Journal*, **35**, 231–246.

[56] Wilson, R. J. 1985. *Introduction to Graph Theory*, 3rd ed. (London: Longman Scientific & Technical).