Computational Problems in Telecommunications Networks

Oktay Günlük

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
in the Graduate School of Arts and Sciences

Columbia University
1995

ABSTRACT

Computational Problems in Telecommunications Networks

Oktay Günlük

This dissertation studies some integer programming and combinatorial optimization problems which arise in the design and operation of telecommunication networks. We study three separate but related problems. The first two are relatively new problems and have applications in the so-called lightwave networks and the last one is a version of the capacity expansion problem.

In lightwave networks, nodes are equipped with tunable transmitters and receivers and communication occurs when the frequency of some transmitter is the same as that of a receiver. This technology enables us to update the network topology to respond to changes in traffic patterns. There are two main optimization problems related with this network structure, one being the design of a target graph more suitable to (future) traffic conditions, and the other being the problem of transforming the current network to this target network.

We first study the second problem, i.e. the transition phase when the modifications on the current graph are made through a sequence of intermediate connection networks. In particular, we move from one graph to another by swapping two independent edges in the current graph for two other independent edges not in the current graph, so that the union forms a four-cycle. Given an initial graph and a target graph, we first state the necessary and sufficient conditions for the existence of a transition sequence and then study the properties of a sequence requiring the minimum number of intermediate graphs. We develop upper and lower bounds on the length of a shortest sequence by formulating an integer program and solving its continuous relaxation to optimality. We also consider the case when the intermediate graphs are required to be connected and develop an efficient algorithm for this case.

Next, we consider the design problem related with this network structure. Given a traffic matrix containing amounts to be routed between corresponding nodes, the objective in this problem is to design a network with certain topological features, and to route all the traffic, so that the maximum load (total flow) on any edge is minimized. Even small instances of this combined design/routing problem are extremely intractable. We formulate this problem as a mixed-integer program, and after studying the polyhedral structure of some related but much simpler problems, we develop a cutting plane algorithm. We also report on computational experiments with this cutting plane algorithm.

Lastly, we study a version of the problem known in the literature as the capacity expansion problem. Given a capacitated network and point-to-point traffic demands, the objective is to add capacity to the edges, in integral multiples of various modularities (or "batches"), and to route traffic, so that the overall cost is minimized. Although this problem arises in many applications, relatively little is known regarding its polyhedral structure. We note that this problem is strongly NP-hard as it contains the fixed-charge network design problem, and thus the Steiner-tree problem as a special case.

We first formulate this problem as a mixed-integer program, and then we study the polyhedral structure of this formulation. Next, we develop a cutting-plane algorithm which uses facet defining inequalities to strengthen the linear programming relaxation. The algorithm produces an extended formulation providing both a very good lower bound and a starting point for branch and bound. The overall algorithm appears effective when applied to problem instances using real-life data.

Table of Contents

List of Figures

List of Tables

Acknowledgements

I would first of all like to thank my advisor Daniel Bienstock for providing guidance and encouragement during my years as a graduate student. It was a pleasure to work with him.

I would also like to thank Donald Goldfarb and David Simchi-Levi for their help and friendly support throughout my years at Columbia. I would also like to thank all the members of my committee and in particular to Julien Bramel and Sebastian Ceria for their helpful comments.

I would like to thank all my friends in the department who helped make this process so enjoyable.

I am deeply indebted to my parents for their support and understanding throughout all these years of schooling.

sevgili annecim ve babacım, herşey için teşekkürler...

Chapter 1

Introduction

This dissertation studies some integer programming and combinatorial optimization problems which arise in the design and operation of telecommunication networks. We study three separate but related problems. The first two are relatively new problems and have applications in the so-called lightwave networks and the last one is a version of the capacity expansion problem.

In the next two chapters we address optimization problems related with rearrangeable lightwave networks. Loosely speaking, in rearrangeable lightwave networks, communication nodes are equipped with a (small) number of transmitters and receivers that can be "tuned" to light frequencies. All of the nodes are connected to an optical medium and direct communication occurs when the frequency of some transmitter is the same as that of a receiver. The actual physical process is somewhat more complex and we refer the reader to [16] for details. Since the transmitters and receivers can be tuned to new frequencies, this technology enables us to update the logical topology of the network without changing the underlying physical topology. In particular, if the traffic conditions should change, it may be advantageous to alter the network accordingly.

There are two main optimization problems related with this network structure, one being the design of a target graph more suitable to (future) traffic conditions, and the other being the problem of transforming the current network to this target network. The separation of logical network topology from the the underlying physical infrastructure is a relatively new concept, and consequently, the related optimization problems have not yet attracted the attention they deserve.

First we study the second problem, i.e. the transition phase when the modifications on the current network are made through a sequence of intermediate connection networks. We consider the case when transmitters and receivers of a node operate in pairs (similar to two way radios), and thus if a node can transmit packets to another node, it can also receive packets from that node.

In general, the property of sharing a common frequency (as described above) defines an undirected graph on the node set such that the degree of a node is no more than the number of transmitter and receiver pairs of that node. Without loss of generality, one can assume that in the resulting graph the degrees are always equal to this upper bound (one can introduce a dummy node with degree one to handle the case when degrees add up to an odd number,) and then use this graph for the standard functions of a communications network.

The process of transforming the initial network to the target network can potentially create problems due to packet delays and desequencing. The branch-exchange operation has been identified in [16] and [18] as a "smooth" way of proceeding from a starting network, as opposed to a radical rearrangement of topology. However, one will prefer to use "short" sequences of swaps (or else, the process will be too costly in terms of rerouting traffic, for example). A swap (or, equivalently, branch-exchange) operation can be defined as replacing two matching edges of a graph with two other matching edges on the same vertices.

In Chapter 2, we consider the problem of finding a *shortest* such sequence. After characterizing the necessary and sufficient conditions for the existence of such a sequence, we show that it is NP-hard to compute the length of the shortest sequence. This result clearly implies that it is NP-hard to find the shortest sequence itself. Then, we develop upper and lower bounds on the length of a shortest sequence by formulating an integer program and solving its continuous relaxation to optimality. We also present an efficient algorithm for the case when the intermediate graphs are required to be connected.

Next we study the design problem related with the above described network structure. Given a traffic matrix containing amounts to be routed between corresponding nodes, the objective this time is to design a (directed) network with certain topological features, and to route all the traffic, so that the maximum load (total flow) on any edge is minimized. Even small instances of this combined design/routing problem are extremely intractable.

Applications of this problem are not limited to lightwave networks. An important problem in communication networks is to route existing traffic requests so as to keep congestion levels as low as possible. One way to approach this problem is to route so that the maximum total flow on any edge is as small as possible. This leads to the mathematical problem known as the *maximum concurrent flow* problem, which has received extensive attention. (See [29], [11], [19], and [20] for some computational experiments).

We note that, given a fixed network, the task of routing the commodities to minimize the maximum load is in fact a linear program (this is the unit-capacity maximum concurrent flow problem) and can therefore be efficiently solved. Our problem involves routing and also choosing the network, and is substantially more difficult. (It is NP-hard).

In our model we allow traffic to be routed in a *divergent* manner, i.e. if a certain request specifies that a given amount of traffic is to be sent from one node to another, then it is permitted to use several simultaneous paths, each carrying some fraction of the total desired amount.

In the context of lightwave networks, when the demands are constantly changing, the ideal way to deal with the congestion problem would seem to be to alter the network structure in " real time" to adapt to new conditions. However, in practice, one would not wish to frequently rearrange the existing network, since it would be very disruptive and expensive to continuously reroute existing traffic. In fact, the network should probably not be rearranged more frequently than once every few hours. In practice, one would wish to have a relatively fast heuristic that generates good solutions. We have observed that even small instances can be extremely intractable (see below), so that several hours of running time on a powerful workstation may in fact be necessary to get good solutions for a large instance.

A heuristic approach for this problem is given in [17]. The heuristics generate both solutions and lower bounds (for the min-max load) and are fast. However, as reported in [17], the bounds produced by these heuristics when applied to some small instances (with fully dense demand matrices) were usually rather far apart, with typical gaps between lower and upper bounds of the order of 20% to 30%.

In Chapter 3, we formulate this problem as a mixed integer program, and after studying the polyhedral structure of some related but much simpler problems, we develop a cutting plane algorithm. We also report on computational experiments with this cutting plane algorithm. The algorithm yields good lower bounds and also an extended formulation that appears effective as a starting point for branch-and-bound.

Our computational experience is encouraging: the gaps in the "benchmark" problems in [16] were substantially reduced in most cases (and never worsened), within a few minutes of computation. We also report on similar results for much larger, less than fully dense, randomly generated problems.

The last problem we study in this dissertation is a version of the problem known in the literature as the capacity expansion problem (CEP). Given a capacitated network and point-to-point traffic demands, the objective in CEP is to add capacity to the edges in integral multiples of various modularities (or "batches"), and to route traffic, so that the overall cost (i.e. capacity plus flow cost) is minimized. We note that CEP is strongly NP-hard [10] as it contains the fixed-charge network design problem, and thus the Steiner-tree problem as a special case.

Our primary motivation for studying CEP is that it naturally arises as part of a much larger and complex problem concerning ATM (asynchronous transfer mode) network design. This larger problem is in fact so complex and ill-defined that a direct polyhedral study of it would be impractical and probably not advisable. However, the ATM problem contains several subproblems either identical or closely resembling CEP. We also note that these problems have fully dense traffic matrices (i.e. every node wants to talk to every other node).

We consider CEP when there are two batch sizes. Since it is always possible to scale traffic demands, we assume that the smaller batch size generates unit capacity, and the larger batch size generates capacity equivalent to an integer multiple of this. In our model, we require the total flow on either direction of an edge to be less than or equal to the sum of the existing and the newly added capacity on that edge. This constraint arises in telecommunications models because, generally, one cannot purchase "one-way cables".

We first formulate this problem (CEP), as a mixed-integer program, and then we study the polyhedral structure of this MIP formulation. The polyhedral structure of CEP (or, rather, some closely related variants) has already been previously studied. Magnanti and Mirchandani [22] have studied a special case of CEP in which there is a single commodity to be routed between two special nodes of the network and there is no existing capacity on the network. In this paper, they present some facet defining inequalities and show that this special case of CEP is closely related with the shortest path problem. Another special case, which arises is the context of the lot-sizing problem with constant production capacities, has been studied by Pochet and Wolsey [27]. In this case, the network related with CEP has a special structure and there is a single batch size. In [27], Pochet and Wolsey fully describe the convex hull of a related polyhedron by using a polynomial number of facets.

Some subproblems related with CEP have also attracted attention. Magnanti, Mirchandani and Vachani [23] study the polyhedral structure of a MIP formulation of the network loading problem (NLP) with three nodes and a single batch size. Wolsey and Pochet [26] also study some surrogate problems that arise in network design problems.

Recently, Stoer and Dahl [30] studied a problem similar to ours where the flows are undirected. In their model, there are no flow costs, but the capacities to be added to edges are of a more general form than those we study. (We note that our formulation can be used to model undirected flows). One primary feature of their approach is that (in terms of our model,) they would split the integral variables into sums of $0-1$ variables. As a result, the inequalities they obtain have a rather combinatorial flavor, and when the demands are small, this approach may be effective. Another feature of the approach in [30] is that, they study the projection of the formulation onto the space of the discrete variables, which is possible since they do not have flow costs.

In Chapter 4, we first present facet defining inequalities for the capacity expansion problem and then apply these results to design a cutting-plane algorithm which uses facet defining inequalities to strengthen the linear programming relaxation.

The algorithm produces an extended formulation providing both a very good lower bound and a starting point for branch and bound. The overall algorithm appears effective when applied to problem instances using real-life data. As we mentioned before, our primary motivation for studying CEP is that it arises as a subproblem in ATM network design. Therefore, the computational testing mainly focuses on how effective our inequalities are towards obtaining a strong formulation for CEP, as opposed to developing an algorithm for solving CEP.

Some of the results of this thesis appeared in [3], [4] and [5].

Chapter 2

A Degree Sequence Problem Related to Network Design

2.1 Introduction

In this chapter we study the following problem on graphs, for which a motivation will be given later: Let G_1 and G_2 be two graphs with the same vertex set, and the same number of edges. For two matching edges e_1 and e_3 of $E(G_1)$, let e_1,e_2,e_3 and e_4 form a simple cycle. If $\{e_1, e_3\} = E(G_1)\backslash E(G_2)$ and $\{e_2, e_4\} = E(G_2)\backslash E(G_1)$, then we say that G_2 (respectively, G_1) arises from G_1 (G_2) by a *swap* operation. In other words, a swap operation can be defined as replacing two matching edges of a graph with two other matching edges on the same vertices. In general, if G and H are graphs with the same labeled degree sequence, it is clear that there is a sequence of swaps that maps G into H. In this chapter, we consider the problem of finding a *shortest* such sequence, and some related questions.

Closely related problems arise in the design and operation of so-called "lightwave" networks. Loosely speaking, in lightwave networks each node is equipped with a (small) number of transmitter/receiver (T/R) pairs that can be "tuned" to light frequencies. All of the nodes are connected to an optical medium and direct communication between two nodes can occur if both of them have a T/R pair tuned to the same frequency. To simplify routing, it is assumed that any given frequency can be common to at most two nodes at any given time.

In general, the property of sharing a common frequency defines a graph on the node set such that the degree of a node is no more than the number of T/R pairs of that node. Without loss of generality, one can assume that in the resulting graph the degrees are always equal to this upper bound (one can introduce a dummy node with degree one to handle the case when degrees add up to an odd number) and then use this graph for the standard functions of a communications network. The actual physical process is somewhat complex (see [16] for more details). Since the T/R pairs can be tuned to new frequencies, the topology of the network can be altered. In particular, if the traffic conditions should change, it may be advantageous to alter the network accordingly. The branch-exchange operation has been identified in [16] and [18] as a "smooth" way of proceeding from a starting network, as opposed to a radical rearrangement of topology. However, one will prefer to use "short" sequences of swaps (or else the process will be too costly in terms of rerouting traffic, for example). Thus our problem arises.

The results of this chapter can be summarized as follows:

Theorem 2.3.1 *Let G_i, G_f be two graphs on the same vertex set, and with the same labeled degree sequence. The length of a shortest swap sequence transforming G_i into G_f equals*

$$|E(G_i) \setminus E(G_f)| - |C^*(\tilde{G}_{i,f})|$$

where $\tilde{G}_{i,f} = (V(G_i), E(G_i) \triangle E(G_f))$ and $|C^*(\tilde{G}_{i,f})|$ is the maximum number of edge disjoint circuits in $\tilde{G}_{i,f}$ whose edges alternate between $E(G_i)$ and $E(G_f)$.

Denote by $|S^*(G_i, G_f)|$ the length of a shortest swap sequence mapping G_i into G_f.

Theorem 2.3.2 *It is NP-hard to compute $|C^*(\tilde{G}_{i,f})|$ and thus, it is NP-hard to compute $|S^*(G_i, G_f)|$.*

The problem of computing $C^*(\tilde{G}_{i,f})$ can be viewed as a set-packing, or a hypergraph matching problem. In this hypergraph, there is a vertex for every edge in $E(G_i) \triangle E(G_f)$, where $\triangle$ denotes the symmetric set difference function, and there is an edge for every possible circuit whose edges alternate between $E(G_i)$ and $E(G_f)$. A matching (a collection of pairwise disjoint hyperedges) corresponds to a collection of edge disjoint alternating circuits and consequently, the size of the maximum matching is equal to $|C^*(\tilde{G}_{i,f})|$. Consequently the problem of computing $|C^*(\tilde{G}_{i,f})|$ can be formulated as an integer program. Note that this formulation requires an exponential number of variables, one for each possible circuit. However, its continuous relaxation can be solved in polynomial time, and as we show, this leads to an approximation algorithm for $|S^*(G_i, G_f)|$:

Theorem 2.5.1 *$|S^*(G_i, G_f)|$ can be estimated within a multiplicative error of $7/4$ in polynomial-time.*

Suppose G_i and G_f are both connected. Then in terms of the application, it would be desirable to provide a swap sequence using connected graphs as well. Here we have:

Theorem 2.5.5 *Suppose G_i and G_f are connected. Then in polynomial-time one can compute a swap sequence that maps G_i to G_f, such that all intermediate graphs are connected, and whose length is within a constant bound of $|S^*(G_i, G_f)|$.*

2.2 Preliminaries and Definitions

Throughout our analysis we will work with undirected graphs without loops. Given an initial graph $G_i = (V, E_i)$ and a target graph $G_f = (V, E_f)$ defined on the same vertex set, edges in $E_i \cup E_f$ are partitioned into three disjoint sets as follows:

$$\begin{aligned} \text{Bad Edges} &: \quad B(G_i, G_f) = E(G_i) \setminus E(G_f) \\ \text{Desired Edges} &: \quad D(G_i, G_f) = E(G_f) \setminus E(G_i) \\ \text{Neutral Edges} &: \quad N(G_i, G_f) = E(G_i) \cap E(G_f) \end{aligned}$$

where '\' denotes the ordinary set difference function. The aim of the reconfiguration process is to eventually replace all of the bad edges with the desired ones and thus construct the target graph.

A swap operation s is called *improving* if the number of desired edges introduced by s is more than the number of neutral edges deleted. It is called *perfect* if two bad edges are replaced by two desired ones.

Two graphs are called *accessible* from each other if there exists a sequence of swap operations transforming one of the graphs into the other. Observe that the transformation process is symmetric in the sense that, a sequence mapping G_i into G_f can be used in reverse order to map G_f into G_i. Also observe that swap operations do not change the labeled degree sequence of the initial graph, so that we have the following simple necessary condition for accessibility:

Lemma 2.2.1 *If two graphs are accessible from each other, then they have the same labeled degree sequence.*

Therefore, we will focus our attention on the graphs which share a common degree sequence and given an initial graph, we will assume that the target graph satisfies this requirement. When we combine Lemma 2.2.1 with the fact that the number of edges of any graph equals half the sum of degrees of its vertices, we can conclude that two graphs which are accessible from each other have the same number of edges. Therefore, we have the following corollary:

Corollary 2.2.2 *If G_i and G_f are accessible from each other, then $|B(G_i, G_f)| = |D(G_i, G_f)|$.*

Given $G_i = (V, E_i)$ and $G_f = (V, E_f)$, we define a new graph $\tilde{G}_{i,f}$ as follows. The vertex set of $\tilde{G}_{i,f}$ is V, further, $\tilde{G}_{i,f}$ has two types of colored edges, namely (solid) black edges $B(G_i, G_f)$ and dashed edges $D(G_i, G_f)$. $\tilde{G}_{i,f}$ will be called a *colored* graph. We define the *size* of a colored graph to be the number of colored edges it has, in other words, $\tilde{G}_{i,f}$ is of size $|B(G_i, G_f)| + |D(G_i, G_f)|$.

Example 2.2.3 *A small size example of $\tilde{G}_{i,f}$ related with G_i and G_f is shown in Figure 2.1. Notice that $|B(G_i, G_f)| = |D(G_i, G_f)| = 6$ and $|N(G_i, G_f)| = 3$ so that $|E_i| = |E_f| = 9$ and the size of $\tilde{G}_{i,f}$ is 12.*

The colored graph $\tilde{G}_{i,f}$ does not only have the same number of black (bad) and dashed (desired) edges but also has the nice property that the number of black edges incident with any vertex equals the number of dashed edges incident with it. This fact is due to Lemma 2.2.1. The degree of any vertex in G_i equals the number of bad and neutral edges incident

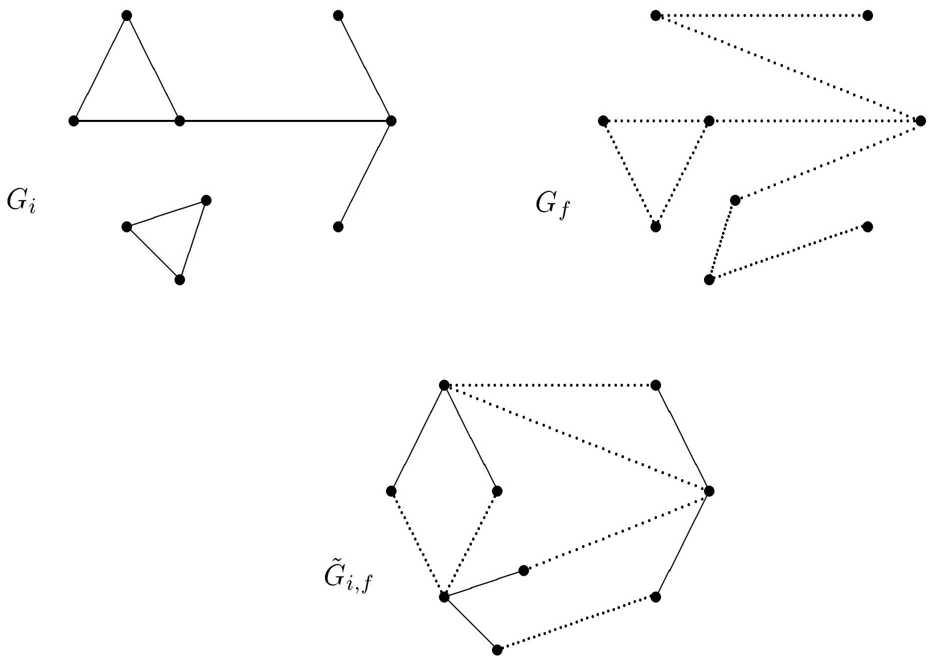

Figure 2.1: The Colored Graph $\tilde{G}_{i,f}$

with it, and similarly in G_f the degree of the same vertex is the number of desired and neutral edges incident with it. Therefore, the necessary condition for accessibility implies the following corollary:

Corollary 2.2.4 *If two graphs G_i, G_f have the same labeled degree sequence, then in the related colored graph $\tilde{G}_{i,f}$, every vertex has the same black and dashed degree.*

There is one more property that $\tilde{G}_{i,f}$ must satisfy, if G_i and G_f satisfy the necessary condition, which is implied by Corollary 2.2.4. We call circuits (not necessarily simple), using colored edges alternatingly, *alternating circuits.*

Corollary 2.2.5 *If two graphs G_i and G_f have the same labeled degree sequence, then the edge set of the related colored graph $\tilde{G}_{i,f}$ can be partitioned into alternating circuits of even length.*

We have stated a necessary condition for the accessibility of two graphs, as we show next, this condition is also the sufficient. The proof is straightforward and we include it for completeness.

Lemma 2.2.6 *Two graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$ are accessible from each other if and only if they both have the same labeled degree sequence.*

Proof. The if part is implied by Lemma 2.2.1 and we will prove the only if part by induction on the size of $\tilde{G}_{1,2}$. Notice that by Corollary 2.2.4 we have $|B(G_1, G_2)| = |D(G_1, G_2)|$, which implies that size of $\tilde{G}_{1,2}$ has to be even.

Using Corollary 2.2.5, and the fact that $B(G_1, G_2) \cap D(G_1, G_2) = \emptyset$, it follows that there exist four distinct vertices x, y, z, w such that either $\{x, y\}$ and $\{z, w\}$ are bad and $\{x, z\}$ is good, or $\{x, y\}$ and $\{z, w\}$ are good and $\{x, z\}$ is bad.

In the first case swapping $\{x, y\}$ and $\{z, w\}$ for $\{x, z\}$ and $\{y, w\}$ yields a graph with fewer bad edges. The second case is similar by symmetry: a swap sequence mapping G_f into G_i yields a sequence mapping G_i into G_f. ■

2.3 Shortest Sequence

In this section we will investigate the properties of a shortest swap sequence transforming one graph to another. We will first show that an alternating cycle representation of edges of the colored graph $\tilde{G}_{i,f}$ corresponds to a swap sequence transforming G_i to G_f and then use this idea to find an upper bound on the length of a shortest swap sequence. Lastly we will show a min-max relationship between the two.

Given an alternating cycle $c = (d_1, b_1, d_2, b_2, \ldots, b_{n-1}, d_n, b_n)$ of length $2n$ (i.e. n black and n dashed edges), on the the colored graph $\tilde{G}_{i,f}$, notice that we can replace the related bad edges of G_i with the desired ones in $n - 1$ swaps. This can be achieved as follows: In the first $n - 2$ swaps we will choose a dashed edge $d_k = \{a, b\}$ on this alternating cycle and swap the neighboring black edges $b_{k-1} = \{c, a\}$ and $b_k = \{b, d\}$ with d_k and $\{c, d\}$. This operation will decrease the length of the cycle by 2 and after the swap operation, the alternating cycle will have the form: $c' = (d_1, b_1, \ldots, d_{k-1}, \{c, d\}, d_{k+1}, \ldots, d_n, b_n)$. Therefore, after $n-2$ swaps, we will have a cycle of length 4, so that we can make a perfect swap operation to replace both of the remaining bad edges with the remaining desired ones.

Therefore, given an alternating cycle representation $C'(\tilde{G}_{i,f})$ of colored edges of $\tilde{G}_{i,f}$, it is possible to replace all of the bad edges of G_i with the desired ones in $|B\left(G_i, G_f\right)| - |C'(\tilde{G}_{i,f})|$ swaps, since we can make a perfect swap operation for each one of the alternating cycles in $C'(\tilde{G}_{i,f})$ as described above. Since this relationship must also hold for a maximum alternating cycle representation $C^*(\tilde{G}_{i,f})$, we can write

$$|S^*\left(G_i, G_f\right)| \leq |B\left(G_i, G_f\right)| - |C^*(\tilde{G}_{i,f})|$$

where $S^*\left(G_i, G_f\right)$ is a shortest swap sequence.

We next show that this upper bound is strict, and thus establish a strong relationship between shortest swap sequences and maximum alternating cycle representations.

Theorem 2.3.1 *Let* G_i *and* G_f *be two graphs accessible from each other. Then:*

$$|S^*\left(G_i, G_f\right)| = |B\left(G_i, G_f\right)| - |C^*(\tilde{G}_{i,f})|.$$

Proof. It suffices to show that $|S^*\left(G_i, G_f\right)| \geq |B\left(G_i, G_f\right)| - |C^*(\tilde{G}_{i,f})|$. Let s be a swap operation transforming G_i to G_s, and let G_f be the target graph. Furthermore let D^- be the set of deleted neutral edges and B^+ be the set of bad edges introduced by s. Define f as follows:

$$f\left(G_i, G_f\right) = |B\left(G_i, G_f\right)| - |C^*(\tilde{G}_{i,f})|.$$

In $C^*(\tilde{G}_{s,f})$ look at the alternating cycles containing elements of D^- or B^+. There can be at most $|D^-| + |B^+|$ of them and the remaining cycles are composed of edges common to both $\tilde{G}_{i,f}$ and $\tilde{G}_{s,f}$. Let $C^- \subseteq C^*(\tilde{G}_{s,f})$ be the set of alternating cycles using these common edges. Therefore:

$$|C^*(\tilde{G}_{s,f})| \leq |C^-| + |D^-| + |B^+|.$$

If $\tilde{G}_{i,f}$ has some desired and bad edges which do not appear on the common cycles then

$$|C^-| + 1 \leq |C^*(\tilde{G}_{i,f})|$$

since those edges have to form at least one more cycle. Therefore,

$$|C^*(\tilde{G}_{s,f})| \leq |C^*(\tilde{G}_{i,f})| - 1 + |D^-| + |B^+|. \quad (2.1)$$

Obviously,

$$|B(G_s, G_f)| = |B(G_i, G_f)| - |B^-| + |B^+| \quad (2.2)$$

where B^- is the set of deleted bad edges and

$$|B^-| = 2 - |D^-| \quad (2.3)$$

so we can subtract (2.2) from (2.1) and substitute (2.3) to get:

$$f(G_s, G_f)| \geq f(G_i, G_f) - 1.$$

On the other hand, if all of the desired and bad edges of $\tilde{G}_{i,f}$ appear on the common cycles then

$$|C^*(\tilde{G}_{s,f})| = |C^-| + 1$$

since s should have introduced two bad edges and two desired edges to form a new cycle. Therefore,

$$f(G_s, G_f) = f(G_i, G_f) + 1 \geq f(G_i, G_f) - 1$$

which shows that $f(G_i, G_f)$ could at most be decreased by 1 after a swap operation.

Knowing that $f(G_f, G_f) = 0$, at least $f(G_i, G_f)$ swap operations are necessary to reach the target graph, and thus

$$|S^*(G_i, G_f)| \geq f(G_i, G_f),$$

which completes the proof. ∎

Having characterized the length of the shortest swap sequence, we next show that it is difficult to find the size of a maximum alternating cycle representation of the colored graph. This result implies that it is also difficult to find the length of a shortest sequence. We will achieve this in two steps. First we will show that the alternating cycle packing problem (ACPP) for arbitrary colored graphs is NP-Hard, and then extend this result to Eulerian colored graphs. ACPP is defined as follows: Given a graph with two different types of edges $\tilde{G} = (V, R, B)$, find a maximum cardinality set C^* such that elements of C^* constitute edge disjoint alternating cycles of $\tilde{G}$. Notice that this packing problem reduces to alternating cycle representation problem if $\tilde{G}$ is Eulerian.

Theorem 2.3.2 *ACPP is NP-Hard.*

Proof. The proof is by transforming the independent set problem for cubic graphs (Cubic-IS) [10] to ACPP.

For any instance of the Cubic-IS problem on $G = (V, E)$ we define a related colored graph $\tilde{G} = (\tilde{V}, R, B)$ as follows.

First, for each vertex v of the original graph, incident, say, with edges $\{v, x\}$, $\{v, y\}$, $\{v, w\}$, $\tilde{G}$ has six vertices $v_{x1}, v_{x2}, v_{y1}, v_{y2}, v_{w1}, v_{w2}$ and three red edges $\{v_{x1}, v_{x2}\}$, $\{v_{y1}, v_{y2}\}$, $\{v_{w1}, v_{w2}\}$ and three black edges $\{v_{x2}, v_{y1}\}$, $\{v_{y2}, v_{w1}\}$, $\{v_{w2}, v_{x1}\}$. We call the alternating cycle formed by these six edges, the vertex cycle related with v.

Next, for every edge $\{v, w\}$ of the original graph, $\tilde{G}$ has six vertices a_{vw1}, a_{vw2}, b_{vw1}, b_{vw2}, c_{vw1}, c_{vw2}, eight black edges $\{v_{w1}, a_{vw1}\}$, $\{a_{vw1}, b_{vw1}\}$, $\{b_{vw1}, c_{vw1}\}$, $\{c_{vw1} w_{v1}\}$, $\{v_{w2}, a_{vw2}\}$, $\{a_{vw2}, b_{vw2}\}$, $\{b_{vw2}, c_{vw2}\}$, $\{c_{vw2}, w_{v2}\}$, and three red edges $\{a_{vw1}, a_{vw2}\}$, $\{b_{vw1}, b_{vw2}\}$, $\{c_{vw1}, c_{vw2}\}$. We say that these 11 edges form a *ladder* joining vertex cycles of v and w.

We denote the set of red edges by R and black edges by B. Figure 2.2 shows a partial application of this transformation for a vertex v and its three neighbors x, y and w.

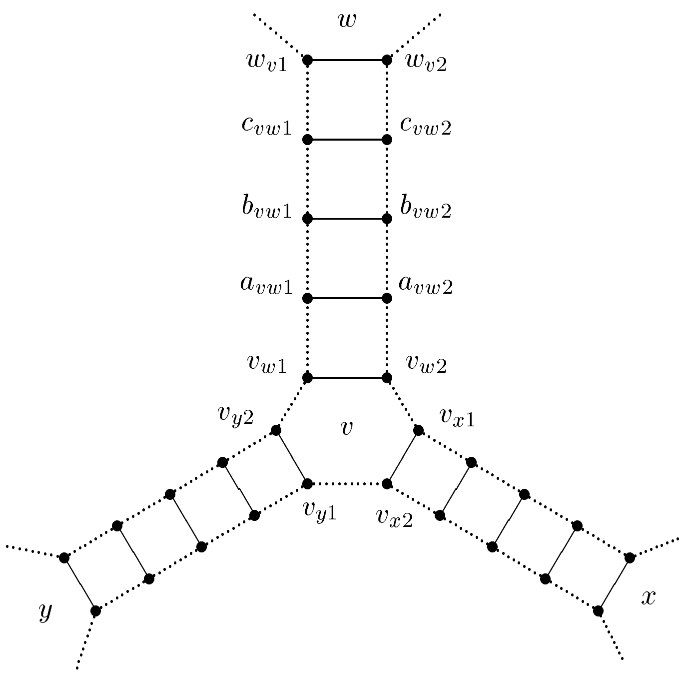

Figure 2.2: Transforming G to $\tilde{G}$

If $|V| = n$ and so $|E| = 3n/2$, then the number of vertices and edges of the colored graph are as follows:

$$\begin{aligned} |\tilde{V}| &= 6n + 6 \cdot 3n/2 = 15n \\ |B| &= (3n + 4 \cdot 3n/2 + 6n) = 15n \\ |R| &= (3n + 3 \cdot 3n/2) = 15n/2 \end{aligned}$$

Now consider a solution of the alternating cycle packing problem on this colored graph $\tilde{G}$. If a black edge related to $v \in V$ is on some alternating cycle then this cycle has to be the vertex cycle of V . This is because, any alternating cycle using one of the black edges but not all of the vertex cycle has to climb up one of the ladders and thus disable 2 possible ladder cycles and so the solution on hand can be improved by deleting this cycle and adding these ladder cycles. Which is not consistent with the optimality of the solution.

Therefore, in the solution we have a number of vertex cycles and some other cycles confined within ladders. It should be obvious that there should be 2 alternating cycles in a ladder if at least one of the adjacent vertex cycles is not in the solution and there should be only one of them if both of the vertex cycles are in the solution. If the latter is the case, then we can arbitrarily drop one of these vertex cycles and increase the number of ladder cycles by one. In this manner, while preserving optimality we can modify the solution not to have adjacent vertex cycles in the solution. Therefore, if we denote the size of a maximum packing by c^*,

$$c^* = k + 3n$$

where k is the number of vertex cycles in the modified solution. Consequently, given the size of the solution to the alternating cycle packing problem, k is the size of a maximum independent set for G and the vertices of this set are the vertices whose vertex cycles are in this solution. Notice that, if we delete the edges that appear in the solution, we obtain a graph which has some vertices with red degree 0 and black degree 1 and the remaining vertices have red degree 1 and black degree 2. Since the transformation is polynomial, the proof is complete. ■

Corollary 2.3.3 *The Maximum alternating cycle representation problem is NP-hard.*

Proof. Extending Theorem 2.3.2 to Eulerian graphs is done by taking two copies of $\tilde{G}$, say, $\tilde{G}_1$ and $\tilde{G}_2$ and then connecting twin vertices with red edges. Obviously this new colored graph $\tilde{G}_E$ is Eulerian and each vertex in this graph has total degree 4.

If we solve the maximum alternating cycle representation problem on $\tilde{G}_E$, in the solution there will be cycles using only $\tilde{G}_1$ edges, others using only $\tilde{G}_2$ edges and the rest using edges connecting $\tilde{G}_1$ to $\tilde{G}_2$. If we respectively denote the number of those cycles by c_1, c_2 and b, the size of the maximum representation c_E^* is $c_1 + c_2 + b$.

It should be obvious that c_1 and c_2 are at most c^*, and that the value of b can not exceed $|\tilde{V}|/2$ since we have only $|\tilde{V}|$ edges connecting $\tilde{G}_1$ to $\tilde{G}_2$ and each cycle of this type uses at least two of them. Therefore, we have:

$$c_E^* = c_1 + c_2 + b \leq 2 \cdot c^* + \frac{|\tilde{V}|}{2}.$$

It is possible to achieve this bound as follows. First we solve the alternating cycle packing problem on $\tilde{G}$ and delete the edges which are used in the alternating cycles. We know that the remaining graph has exactly $|\tilde{V}|/2$ alternating paths starting and ending with black edges. This is because we still have $|\tilde{V}|$ vertices with black degree one more than red degree. Then make a copy of the remaining graph and join twin vertices with red edges. And lastly form $|\tilde{V}|/2$ new alternating cycles by using symmetric alternating paths and red edges joining their end points. Therefore,

$$\begin{aligned} c_E^* &= 2 \cdot c^* + \frac{|\tilde{V}|}{2} \\ &= 2k + 6n + \frac{15}{2}n. \end{aligned}$$

■

This negative result implies that it is not possible to find a shortest swap sequence in polynomial time but the relationship between a shortest swap sequence and maximum alternating cycle representation problems is guiding in the sense that we can find a short swap sequence by finding many alternating cycles first, and then by using the related swap sequence.

2.4 LP Formulation and an Upper Bound on the Length of a Shortest Sequence

A problem of interest is that of approximating the length of a shortest swap sequence in polynomial time. In this section we present a positive result concerning this problem. Namely, we show that in polynomial time we can find a small interval containing the length of a shortest sequence. To this end, we will first find the value of a linear programming relaxation of the alternating cycle representation problem in polynomial time, and then show that

$$(|B| - q) \leq |B| - |C^*(\tilde{G}_{i,f})| \leq \min\left\{|B|, \frac{7}{4}(|B| - q)\right\}$$

where q is the value of this LP relaxation.

As mentioned before, the problem of finding $C^*(\tilde{G}_{i,f})$ can be viewed as a hypergraph matching problem, in which the vertices are the colored edges and the hyperedges are the possible alternating cycles of the colored graph $\tilde{G}$. In this formulation (which requires an exponential number of variables), a matching corresponds to a collection of alternating cycles in the colored graph and consequently, a maximum matching is a maximum alternating cycle representation of the colored graph with $z = |C^*(\tilde{G})|$.

The integer program (IP), its continuous relaxation (LP) and the dual of the relaxed problem (DLP) are as follows, where $\mathbf{C}$ is the set of all possible alternating cycles .

$$
\begin{array}{llll}
(IP) \quad z = & \max \sum_{C \in \mathbf{C}} x_C & \\
s.t. & \sum_{\forall C \ni e} x_C \leq 1 & \forall e \in E = B \cup R \\
& x_C \in \{0, 1\} &
\end{array}
$$

$$
\begin{array}{llll}
(LP) \quad q = & \max \sum_{C \in C} x_C & \\
s.t. & \sum_{\forall C \ni e} x_C \leq 1 & \forall e \in E = B \cup R \\
x_C \geq 0 & &
\end{array}
$$

$$
\begin{array}{llll}
(DLP) \quad q = & \min \sum_{e \in E} y_e & \\
s.t. & \sum_{e \in C} y_e \geq 1 & \forall C \in \mathbf{C} \\
& y_e \geq 0 &
\end{array}
$$

Later in this section, we will show that q can be computed in polynomial-time. This is so because the related separation problem (see [12]) can be solved in polynomial-time. Namely, given $y \in R^{|E|}, y \geq 0$, we will show how to efficiently construct an alternating cycle C such that $\sum_{e \in C} y_e$ is minimum. But let us postpone this till later and instead see how knowing q leads to a good estimate for $|C^*(G_{i,f})|$.

To bound z as a function of q on $|C^*(\tilde{G}_{i,f})|$ we will use one of the results of Aharoni, Erdös and Linial [1] on the hypergraph matching problem. Aharoni et al. study the relation between the optimum value of a hypergraph matching problem and that of its linear programming relaxation, and show that, for any hypergraph with n vertices and m edges

$$\tilde{z} \geq \frac{\tilde{q}^2}{n - \frac{f-1}{m} \cdot \tilde{q}^2} \geq \frac{\tilde{q}^2}{n}$$

where $\tilde{z}$ is the cardinality of the maximum matching, $\tilde{q}$ is the value of the relaxed program and f is the least cardinality of a hyperedge. In our case, using $f \geq 4$, the above result implies:

$$|C^*(\tilde{G}_{i,f})| \geq \frac{q^2 n}{2|B| - (3/|\mathbf{C}|) \cdot q^2} \geq \frac{q^2}{2|B|}. \tag{2.4}$$

Notice that the value of DLP, and consequently, that of LP, can not exceed $|B|/2$, since each alternating cycle contains at least two black edges. Combining 2.4 with this upper bound, we get:

$$\frac{|B|}{2} \quad \geq \quad q \geq |C^*(\tilde{G}_{i,f})| \geq \frac{q^2}{2|B|}$$

implying,

$$\frac{1}{4} \quad \geq \quad \left(\frac{q}{2|B|}\right) \geq \frac{|C^*(\tilde{G}_{i,f})|}{2|B|} \geq \left(\frac{q}{2|B|}\right)^2 \geq 0 \tag{2.5}$$

which bounds the reciprocal of the average cycle length in the optimal solution from above and from below. The length of this interval is:

$$\left(\frac{q}{2|B|}\right) - \left(\frac{q}{2|B|}\right)^2 = \left(\frac{q}{2|B|}\right)\left(1 - \frac{q}{2|B|}\right)$$

attaining its maximum value when $(q/2|B|) = 1/4$, and so in the worst case this interval has length $3/16$.

Similarly, we can bound the average alternating cycle length $\bar{c}$ in terms of $\bar{q} = 2|B|/q$ as:

$$4 \leq \bar{q} \leq \bar{c} \leq \min(\bar{q}^2, 2|B|)$$

which has a length of $(2|B| - \sqrt{2|B|})$ in the worst case.

Furthermore, if we define $r = |C^*(\tilde{G}_{i,f})|/2|B|$ and $\rho = q/2|B|$, we can find an upper bound on the length of the minimal swap sequence as follows (using (2.5) to yield $r \geq \rho^2$):

$$\begin{aligned} 1 \leq \frac{|B| - |C^*(\tilde{G}_{i,f})|}{|B| - q} &= \frac{|B| - 2r|B|}{|B| - 2\rho|B|} \\ &= \frac{1 - 2r}{1 - 2\rho} \\ &\leq \frac{1 - 2\rho^2}{1 - 2\rho}. \end{aligned}$$

If we take the derivative of the last expression with respect to ρ, we get

$$\begin{aligned} \frac{d}{d\rho}\left(\frac{1 - 2\rho^2}{1 - 2\rho}\right) &= \frac{-4\rho(1 - 2\rho) + 2(1 - 2\rho^2)}{(1 - 2\rho)^2} \\ &= \frac{4\rho^2 - 4\rho + 2}{(1 - 2\rho)^2} \end{aligned}$$

which is nonnegative for $\rho \leq 1/4$. Therefore, we can write

$$\frac{|B| - |C^*(\tilde{G}_{i,f})|}{|B| - q} \leq \frac{1 - 2(1/4)^2}{1 - 2(1/4)} = \frac{7}{4}$$

which implies

$$(|B| - q) \leq |B| - |C^*(\tilde{G}_{i,f})| \leq \min\left\{|B|, \frac{7}{4}(|B| - q)\right\}.$$

The interval between the leftmost and rightmost values which contains the length of a shortest swap sequence, has length $\min\{q, (3/4)(|B| - q)\}$, which is equal to $3|B|/7$ in the worst case. Consequently, knowing q, we can compute an estimate of $|B| - |C^*(\tilde{G}_{i,f})|$ within a bound of $7/4$.

Let us now return to the problem of computing q. As stated before, all we need is a polynomial-time algorithm which, given $y \in R^{|E|}$, returns an alternating cycle C such that $\sum_{e \in C} y_e$ is smallest.

Given a colored graph H with edge set $B(G_i, G_f) \cup D(G_i, G_f) = E(G_i) \triangle E(G_f)$, we solve the following family of problems: for each black edge $\{x, y\}$, find a minimum cost alternating cycle containing $\{x, y\}$. These subproblems can be solved using an approach similar to that employed in [14] to find minimum length odd paths.

We construct an auxiliary graph $K_{\{x,y\}}$ related with a black edge $\{x, y\}$ as follows: for all of the black edges $b = \{u, v\}$ different from $\{x, y\}$ define two vertices u_b, v_b and define an edge $b' = \{u_b, v_b\}$. Call this graph $G_{\{x,y\}}$.

For all of the red edges, define two vertices and an edge in a similar way, and call the resulting graph G_r. Notice that for all vertices in V, we have the same number of copies in $G_{\{x,y\}}$ and G_r except for x and y. For these two vertices, the number of copies in $G_{\{x,y\}}$ is one less than that of G_r. Assign weights y_e on the edges defined so far and finally for every $v \in V$ join the copies of v in $G_{\{x,y\}}$ to those in G_r with a complete bipartite graph and assign weight zero to these new edges.

It should be obvious that a minimum-weight perfect matching in $K_{\{x,y\}}$ yields a minimum-weight alternating walk from x to y in H, that starts and ends with red edges.

Therefore, if the value of minimum-weight perfect matching problem on $K_{\{x,y\}}$ is $w^*_{\{x,y\}}$, then $w^*_{\{x,y\}} + y_{\{x,y\}}$ is the weight of a minimum-weight cycle containing edge $\{x, y\}$.

By repeating this procedure for all of the black edges, we can find the length of the shortest cycle and thus solve the separation problem in polynomial time.

2.5 Connected Swap Sequences

In this section we will address a constrained version of the network reconfiguration problem where, given a connected initial graph and a connected target graph, it is desired to find a swap sequence such that the resulting intermediate graphs are also connected.

In communication networks, connectivity is obviously the most important structural property of the network. The motivation behind the configuration process, as stated earlier, is to achieve a connection network which is more suitable to the traffic conditions, and it is only natural to conduct this process while keeping the network connected, so that nodes can continue communication.

We will first investigate the necessary and sufficient conditions to undertake this task for tree structures and then extend this result to general graphs without loops. In both cases, our proofs will be constructive so that we implicitly propose algorithms to network configuration problems with connectivity constraints.

Theorem 2.5.1 *If two trees T_i and T_f are accessible from each other then there exists a swap sequence transforming T_i to T_f such that all intermediate graphs are also trees.*

Proof. We will prove that there is a sequence of at most two swaps that strictly decreases the number of bad edges while maintaining connectivity. Applying this fact inductively yields the theorem.

Thus, let $d_1 = \{x, y\}$ be a desired edge (i.e.. $\{x, y\} \in D(T_i, T_f) = E(T_f) \setminus E(T_i)$). Let P_{xy} be the path between x and y included in T_i . Since swap operations preserve the labeled degree sequence, there are distinct unwanted edges $b_1 = \{w, x\}, b_2 = \{y, z\} \in B(T_i, T_f) = E(T_i) \setminus E(T_f)$. Suppose first that one of b_1, b_2 is in P_{xy}. Then replacing b_1 and b_2 with d_1 and $\{w, z\}$ yields the desired result.

Suppose next that neither b_1 nor b_2 is in P_{xy}. Then P_{xy} must contain some unwanted edge (else T_f contains a cycle), say $b_3 = \{u, v\}$, where u separates v from x in P_{xy}. If b_3 is incident with x or y, then we are back in the first case, so without loss of generality assume that it is not. Then the swap sequence:

(i) replace b_3 and b_2 with $\{y, u\}$ and $\{z, v\}$,

(ii) replace b_1 and $\{y, u\}$ with d_1 and $\{w, u\}$

is as desired.

Finally assume that both b_1 and b_2 are in P_{xy}. Then $T_i - b_1 - b_2$ has three components, one containing x (say T_x), one containing y (say T_y), and the final one containing w and z. Notice that in the tree $T' = T_x + T_y + \{x, y\}$ all vertices have the same degree as they do in T_i , and hence T_f . Consequently, this tree contains an unwanted edge $b_4 = \{u, v\}$ (or else T_f is unconnected), and (say) $b_4 \in T_x$, where u separates x from v in T_x . If $u = x$, we are again back in the first case, so assume that it is not. Then the double swap

(i) replace b_4 and b_1 with $\{x, v\}$ and $\{w, u\}$,

(ii) replace b_2 and $\{x, v\}$ with d_1 and $\{z, v\}$

as desired. ∎

The above construction not only shows that it is possible to preserve connectivity during the reconfiguration process, but also shows that it is possible to achieve this without "disturbing" the neutral edges. Notice that we only delete edges which are either bad or which are introduced by the previous swap. Therefore, during this process, edges in $N(T_i, T_f)$ remain untouched.

Also notice that the proposed algorithm makes at most $4 \cdot |S^*(T_i, T_f)| - 3$ swaps since $|S^*(T_i, T_f)| \geq |B(T_i, T_f)|/2$. Therefore, in the worst case, the length of the proposed swap sequence can be as much as four times the length of the shortest one.

Extending Theorem 2.5.1 to general graphs is more complicated than one would expect. This is mainly because we can not show the existence of a bad edge b_4 for general graphs. We next prove Lemmas 2.5.2 - 2.5.4 which examine structural properties of colored graphs related with two-edge connected graphs.

Lemma 2.5.2 *Let G_i be a two-edge connected graph and G_f be a target graph. If $\tilde{G}_{i,f}$ has a vertex with more than one desired edge incident with it, then there exists an improving* *swap operation such that the resulting graph is still connected.*

Proof. Let v be a vertex with at least two bad and two desired edges incident with it. Let $b_3 = \{v, w\}, b_2 = \{v, z\}$ be bad edges, $d_1 = \{v, y\}$ be good and $b_1 = \{y, x\}$ be bad.

Let G_1 be the graph obtained from G_i after swapping edges b_1 and b_2 with edges d_1 and $\{x, z\}$ and similarly let G_2 be the graph obtained from G_i after swapping edges b_1 and b_3 with edges d_1 and $\{x, w\}$.

Obviously both G_1 and G_2 have more edges in common with G_f than G_i has but they are not feasible if they introduce loops. Due to b_1 and d_1 vertices x, y and v have to be distinct and due to b_2, b_3 and d_1 neither z nor w could be the same as v or y.

Suppose first x, w, z are all distinct. If G_1 is not connected then after removing edges b_1 and b_2 from G_i we end up with two components such that vertices x and z are in one and y, v and w are in the other. This implies that G_2 has to be connected .

If $|\{x, z, w\}| = 2$, then again either G_1 or G_2 is as desired. Finally, if $x = w = z$, then there is a desired edge $d_2 = \{x, q\}$ $(q \neq y, v)$ and thus there is a bad edge $b_4 = \{q, t\}$. So we can either swap b_4 and b_1 with d_2 and $\{t, y\}$ or b_4 and b_2 with d_2 and $\{t, v\}$. The result of the swap will be connected, as is easy to see. ∎

Lemma 2.5.3 *Let G_i be a two-edge connected graph and G_f be a target graph. Consider a decomposition D of $E(\tilde{G}_{i,f})$ into alternating cycles, and suppose D includes two vertex disjoint alternating simple cycles, say C_1 and C_2 . There exists a swap operation transforming G_i into G_j such that G_j is connected and $|E(G_j) \backslash E(G_f)| \leq |E(G_i) \backslash E(G_f)|$ and if equality holds, $E(\tilde{G}_{j,f})$ can be decomposed into $|D| - 1$ alternating cycles.*

Proof. Let $b_1 = \{x, y\}$ be a bad edge on C_1 and $b_2 = \{z, w\}$ be a bad edge on C_2 . Let G_1 be the graph obtained from G_i after swapping edges b_1 and b_2 with edges $\{x, z\}$ and $\{y, w\}$ and let G_2 be defined similarly but this time swapping is done with edges $\{x, w\}$ and $\{y, z\}$.

Since G_i is two-edge connected, after deleting b_1 and b_2, we can get at most two components. Thus either G_1 or G_2 is connected, say G_1 is. Obviously G_1 does not have more bad edges than G_i . Let

$$C_3 = \{C_1 \backslash \{x, y\}\} + \{x, z\} + \{C_2 \backslash \{y, z\}\} + \{y, w\}.$$

C_3 is alternating precisely when $\{x, z\}$ and $\{y, w\}$ are bad. But then $D - C_1 - C_2 + C_3$ is a decomposition of $\tilde{G}_{j,f}$ with one fewer element than D. ∎

Given a colored graph related with G_i and G_f , we define a *double swap* as follows. Let b_1,b_2,$b_3 \in B(G_i, G_f)$ and $d_1 \in D(G_i, G_f)$, such that $b_1 = \{a, b\}, d_1 = \{b, c\}, b_2 = \{c, d\}$ and $b_3 = \{y, z\}$ and all vertices are distinct. First swap edges b_1 and b_3 with $\{a, y\}$ and $\{b, z\}$, then swap edges $\{b, z\}$ and b_2 with d_1 and $\{z, d\}$.

Lemma 2.5.4 *Let G_i be a two-edge connected graph, G_f be a connected target graph and suppose that the bad and desired edges of $\tilde{G}_{i,f}$ form a single alternating simple cycle C . If there are no improving swap operations, then there exists an improving double swap.*

Proof. First, by induction on k, we will show that for any $k \geq 1$, the removal of any k consecutive bad edges $b_1, \ldots, b_k$ of C from G_i creates a graph with k components.

For $k = 1$ the statement follows since G_i is two-edge connected. Let $k > 1$ be such that the statement holds for $k - 1$ and suppose that the removal b_k does not further decompose G_i , i.e. G_i contains a path between both ends of b_k that does not contain any of the edges b_i, $i \leq k$. In particular, this path does not contain b_{k-1}. This fact together with the two-edge connectedness of G_i implies that removing b_{k-1} and b_k does not disconnect G_i and hence the swap of b_{k-1} and b_k with the desired edge between b_{k-1} and b_k on C (and a second edge joining the remaining ends of the deleted edges) preserves connectedness and is improving, a contradiction.

Choose a fixed orientation of C and orient its edges accordingly.

Next, by induction on k, we will show that if there are no improving double swaps, then for any $k \geq 1$, the removal of any consecutive bad edges $b_1, \ldots b_k$ of C from G_i creates a graph with k components $R^0, \ldots, R^{k-1}$, such that

(a) For $j \geq 1$, R^j contains the head of b_j and the tail of b_{j+1}.

(b) Component R^0 contains the head of b_k, and the tail of b_1.

The statement is clear for $k = 1$ since G_i is two-edge connected, so assume it holds for $k \geq 1$. Label vertices of C consecutively $1, 2, \dots$ so that $d_j = \{2j-1, 2j\}$ and $b_j = \{2j, 2j+1\}$ and observe that $d_i \in R^{i-1}$ for $i = 2, \dots, k$. We know that if we further remove the $k+1$'st consecutive bad edge, one of the components $R^0, \dots, R^{k-1}$ will be divided into two, let this component be R^α.

If $\alpha \geq 2$, then the component structure would be inconsistent with the induction assumption (for $k-1$, if we put b_1 back). So $\alpha \in \{0, 1\}$.

If $\alpha = 1$, then after deleting b_{k+1}, R^1 will be divided into two components K_1 and K_2 and let $2k+2 \in K_1$, $2k+3 \in K_2$. Then, to be consistent with the component structure when edges $b_2, \dots, b_{k+1}$ are deleted, $4 \in K_2$ and therefore $3 \in K_1$. But in this case the double swap with: $a = 2$, $b = 3$, $c = 4$, $d = 5$, $z = 2k+3$, $y = 2k+2$ is feasible, therefore, $\alpha = 0$. Let K_3 and K_4 be the components R^0 gets divided into, after further deleting b_{k+1} and let $2k+2 \in K_3$ and $2k+3 \in K_4$. To be consistent with the component structure when edges $b_2, \dots, b_{k+1}$ are deleted, $2k+1 \in K_3$ and thus $2 \in K_4$, which completes the induction.

To complete the proof, it is sufficient to observe that if there are no improving double swap operations, then G_i should have the above structure, which implies that G_f is disconnected, a contradiction. ∎

Theorem 2.5.5 *If two connected graphs $G_i = (V, E_i)$ and $G_f = (V, E_f)$ have the same labeled degree sequence then there exists a sequence of connected intermediate graphs transforming one of them to the other.*

Proof. Suppose we decompose G_i into its two-edge connected components, which we call *clusters*. As is well known, if we contract every cluster we obtain a tree. In the proof we will make use of this property and when considering $\tilde{G}_{i,f}$, we will keep the cluster structure of G_i in mind. Any edge whose removal will disconnect the graph G_i will be called a *tree edge*. The proof has three main steps.

Step 1 :

For any cluster K of G_i, if there is a bad edge $\{a, b\}$ with $a, b \in V(K)$ and a desired edge $d_1 = \{x, z\}$ with $z \in V(K)$ and $x \in V \setminus V(K)$, then we will show that it is always possible to make improving swap operations. Let $b_1 = \{q, x\}$ be a bad edge incident with x. We have the following two cases:

Case 1) : Vertex z has a bad edge $b_0 = \{z, y\}$ incident with it such that $y \in K$.

If $q \neq y$, then swapping b_0 and b_1 with d_1 and $\{q, y\}$ decreases the number of bad edges while preserving connectivity. Therefore, $q = y$, and there is a desired edge $d_2 = \{y, w\}$ incident with y and a bad edge $b_2 = \{w, v\}$ incident with w. The related vertices are shown in Figure 2.3.

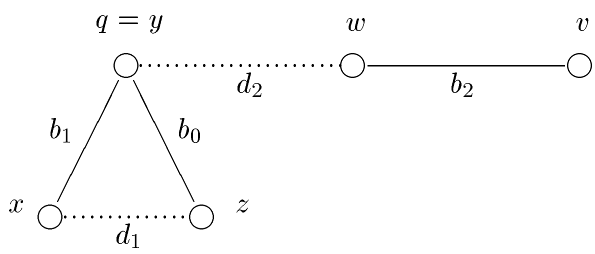

Figure 2.3: Theorem 2.5.5, Case 1, $q = y$

If edge b_2 lies in the same cluster as z (i.e. $w, v \in V(K)$), then swapping b_1 and b_2 with d_2 and $\{x, v\}$ preserves connectivity and is improving.

Therefore, b_2 does not lie in the same cluster as z, and notice that q and z will still be connected after deletion of b_0, b_1 and b_2, since both of the vertices are in the same cluster and neither b_1 nor b_2 lie in this cluster.

Therefore, if $z \neq v$ we can swap b_0 and b_2 with d_2 and $\{z, v\}$, or else we can swap b_1 and b_2 with d_1 and d_2. In both cases, the swap operation is improving and it preserves connectivity.

Case 2) : Vertex z does not have a bad edge $b_0 = \{z, y\}$ such that $y \in K$, then the graph has the form shown in Figure 2.4, where vertices $z, a, b \in V(K)$ and $x, y \notin V(K)$, so z, a, b will still be connected after removal of b_0,b_1 and b_2. Also notice that vertices a, b, z, x and

y have to be distinct.

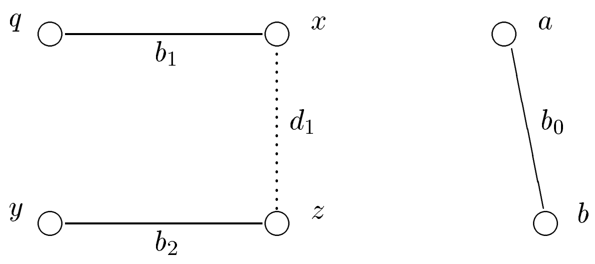

Figure 2.4: Theorem 2.5.5, Case 2

If $q = a$, then we can swap edges b_1 and b_2 with d_1 and $\{y, q\}$. If $q \neq a$, then we can first swap b_1 and b_0 with $\{q, a\}$ and $\{x, b\}$ and then swap $\{x, b\}$ and b_2 with d_1 and $\{b, y\}$. Both operations preserve connectivity and after the second one, at least one more desired edge is placed.

Step 2 :

After performing Step 1 iteratively, we end up with a graph G_j with the property that if a cluster K of G_j contains a bad edge, then there is no desired edge $\{u, v\}$ in $\tilde{G}_{j,f}$ which satisfies $u \in V(K)$ and $v \notin V(K)$. For the clusters containing bad edges, we can apply Lemmas 2.5.2 - 2.5.4 and replace their bad edges with desired ones until the cluster structure of G_j changes. If this change takes place, we go back to Step 1 and iterate.

Step 3 :

We can thus make improvements until no bad edges lie within a cluster. Consequently, the only bad edges are the tree edges lying between clusters. If there are any desired edges within a cluster, then G_f is disconnected, since it is obtained by replacing all bad edges with the desired ones (say, we remove k tree edges, yielding $k+1$ components and adding fewer than k tree edges). So all of the desired edges also join vertices of different clusters.

Consequently the cluster structure of the current graph, say, G_k is the same as that of the target graph G_f(i.e. same clusters, same edges within each given cluster, but different tree edges).

From G_k we can obtain a spanning tree as follows: For each cluster K_i take just enough edges to form a spanning tree T_i of the vertices in that cluster. Notice that there can be no bad edges among the chosen ones since all of the edges inside clusters are in the edge set of the target graph G_f. Let

$$\begin{aligned} T_1 &= \bigcup_{\forall i} T_i + \text{Tree edges of } G_k \\ T_2 &= \bigcup_{\forall i} T_i + \text{Tree edges of } G_f \ . \end{aligned}$$

Both T_1 and T_2 have to be connected since we have chosen enough edges to keep clusters connected and we are taking all of the intra-clusteral edges. Also, notice that the vertices of T_1 and T_2 should have the same labeled degree sequence (since $E(G_k) \triangle E(G_f)$ is made up of tree edges).

Therefore, we can use the previous theorem and claim that there exists a sequence of swap operations preserving connectivity to take us from T_1 to T_2. Using exactly the same swaps we can modify G_k and get G_f, as desired.

In summary, we can always construct a swap sequence of length $O(|B(G_i, G_f)|)$, such that each intermediate graph is also connected. ∎

2.6 Concluding Remarks

A natural extension of this study would be to consider directed graphs where a swap operation is defined as follows: For any two directed edges $a_1 = (x, y)$ and $a_2 = (w, z)$ of the graph, replace a_1 and a_2 with edges (x, z) and (w, y). With minor modifications, and if we allow loops, it is possible to show that the results in Sections 2.3 and 2.4 can be extended to directed graphs. If we translate connectivity into strong connectivity for directed graphs, it is easy to show that we can not always preserve it during the reconfiguration process. For example, there is no swap sequence (preserving strong connectivity) that reverses the orientation of a directed cycle.

Another possible extension is to consider some other structural properties of the graph (e.g. diameter, edge or vertex connectivity of higher orders).

Chapter 3

Computational Experience with a Difficult Mixed-Integer Multicommodity Flow Problem

3.1 Introduction

3.1.1 Problem Definition

Consider the following optimization problem: Given an $n \times n$ matrix T (whose i, j entry is indicated by t_{ij}), and an integer $p > 0$,

1. construct a simple directed graph D with node set $1, \cdots, n$ where each node has indegree and outdegree equal to a fixed number p, and
2. in D, simultaneously route t_{ij} units of flow from i to j, for all $1 \leq i \neq j \leq n$,

so as to minimize the maximum aggregate flow on any edge of D.

In this chapter we first describe valid inequalities for a mixed-integer programming formulation of this problem and then present results of our computational experience with a cutting plane algorithm.

This problem is briefly motivated as follows. An important problem in communication networks is to route existing traffic requests so as to keep congestion levels as low as possible. One way to approach this problem is to route so that the maximum total flow on any edge is as small as possible. This leads to the mathematical problem known as the *maximum concurrent flow* problem, which has received extensive attention (see [29], [11],

[19], and [20] for some computational experiments).

In the operation of lightwave networks it is possible to alter the network topology, within certain limitations. This feature can be used to handle changing traffic patterns. We refer the reader to [17], but in essence the situation is as follows. Each node in the network is equipped with a small number of tunable transmitters and receivers. If a certain node, u, tunes a transmitter to the same frequency that another node v has tuned a receiver, then flow can be sent directly from u to v. It is assumed that no frequency can be shared by more than two nodes at a time. Consequently, the set of ordered node pairs (u, v) corresponding to the frequencies in use yields a directed graph (the *logical network*) which can then be used to route traffic requests. It is further assumed that traffic can be routed in a *divergent* manner, i.e. if a certain request specifies that a given amount of traffic is to be sent from a node u to a node v, then it is permitted to use several simultaneous paths, each carrying some fraction of the total desired amount.

In this framework, the ideal way to deal with the congestion problem seemingly would be to view demands as constantly changing, and to change network structure in " real time " to adapt to new conditions. However, in practice one would not wish to frequently rearrange the existing network, since it would be very disruptive and expensive to continuously reroute existing traffic. In fact, the network should probably not be rearranged more frequently than once every few hours.

In this manner one arrives at the abstract problem described above.

[Remarks: (1)As indicated, the graph D is assumed to be simple (i.e., no parallel edges). If parallel edges are allowed the problem is similar and the cutting planes we will describe later remain valid, but we will not consider this generalization here. (2) In a more general version of the problem, for each node we have specified upper and lower bounds on the indegree and outdegree. This appears to be a much more difficult problem.]

We note here that, given a fixed network D, the task of routing the commodities to minimize the maximum load is in fact a linear program (this is the unit-capacity maximum concurrent flow problem) and can therefore be efficiently solved. Our problem involves routing and also choosing the network, and is substantially more difficult (it is NP-hard even for p=1). In practice, one would wish to have a relatively fast heuristic that generates good solutions. We have observed that even small instances can be extremely intractable (see below), so that several hours of running time on a powerful computer may in fact be necessary to get good solutions for a large instance.

A heuristic approach for this problem is given in [17]. The heuristics generate both solutions and lower bounds (for the min-max load) and are fast. However, as reported in [17] the bounds produced by these heuristics when applied to some small instances (with fully dense demand matrices) were usually rather far apart, with typical gaps between lower and upper bounds of the order of 20% to 30%.

In this chapter, we report on computational experiments with a cutting plane algorithm for a mixed-integer programming formulation of the problem, which is used to obtain good lower bounds. Our experiments are all for the case $p = 2$, motivated by the study in [17]. The cutting plane algorithm yields good lower bounds and also an extended formulation that appears effective as a starting point for branch-and-bound.

There are some noteworthy features about the problem:

(i) The mixed-integer program is extremely difficult, with very large gaps between the continuous relaxations and the integral optima,

(ii) The linear programs to be solved appear to be quite hard, and

(iii) In a practical setting, the time available for computation would be limited, of the order of a few hours, say.

Our computational experience is encouraging: the gaps in the "benchmark" problems in [16] were substantially reduced in most cases (and never worsened), within a few minutes of computation. We also report on similar results for much larger, less than fully dense, randomly generated problems.

3.1.2 Mixed-Integer Programming Formulation

We will work with the following mixed-integer programming formulation of the problem (notation as in 3.1.1). For each ordered pair of nodes i, j there is a $\{0,1\}$-variable x_{ij}, set to 1 if the edge (i,j) is put in the network, and set to 0 otherwise. For each commodity k and ordered pair i, j, there is a continuous variable f_{kij}, that measures the quantity of flow of commodity k on the edge (i,j) (more below on what constitutes a commodity). For a commodity k and a node i, we denote by s_i^k the *net demand* of commodity k at i. Throughout this chapter we assume that there is positive demand for every commodity, that is, $\sum_{i\neq k} s_i^k > 0$ for all k. The overall formulation is:

$$\min z$$

s.t.

$$\sum_{j\neq i} x_{ij} = p, \text{ for all } i \tag{3.1}$$

$$\sum_{j\neq i} x_{ji} = p, \text{ for all } i \tag{3.2}$$

$$f_{kij} \leq M_{ij}^k x_{ij} \quad \text{for all } k,\ i\neq j \tag{3.3}$$

$$\sum_{j\neq i} f_{kji} - \sum_{j\neq i} f_{kij} = s_i^k \quad \text{for all } i,\ k \tag{3.4}$$

$$\sum_k f_{kij} \leq z \quad \text{for all } i \neq j \tag{3.5}$$

$$0 \leq f_{kij}, \quad x_{ij} \in \{0,1\}, \text{ all } k, \text{ all } i \neq j$$

Equations (3.1) and (3.2) are degree constraints. Equation (3.3) indicates that we can route flow on edge (i,j) only if the edge is there (M_{ij}^k is an appropriately large quantity—more on this later). Equation (3.4) is a flow conservation equation, and equation (3.5) measures the load on edge (i,j).

We will denote this mixed-integer program by ICONG(p). Except for variable z, the constraints involving it and the objective function, this can be regarded as a (multicommodity) uncapacitated fixed charge network flow problem, see [24]. There are two important points concerning the formulation that we would like to bring up here.

(a) We may choose commodities to be either disaggregate (i.e. every nonzero entry in the demand matrix yields a distinct commodity—in other words, commodities correspond to source–destination pairs) or aggregate. For example, we may view all demands with the same source as constituting a commodity ([28] uses the terminology "multicommodity" to refer to the disaggregate version, but here we will not because the overall problem is already multicommodity). The" fine grain " disaggregate formulation for general (uncapacitated) fixed-charge network flow problems is stronger, and it may also be possible to use stronger valid inequalities than for the aggregate version. On the other hand, and in particular in the case of ICONG(p) the linear programs arising in the disaggregate version will be *extremely large* and computationally expensive. A system of inequalities, called "dicut collection inequalities", that yields the projection of the disaggregated formulation on the space of the aggregated formulation has been found [28]. But solving the separation problem for these inequalities appears to be rather time consuming. In Table 3.1 we provide information concerning the aggregated and disaggregated formulations for three problem instances of ICONG(2) considered in [17]. For each problem, we list data for four formulations: (1) aggregated, (2) disaggregated, (3) aggregated with some cuts added, and (4) disaggregated with the same cuts as in (3) added.

problem: quasiunif1			
formulation	value	time	quality (%)
agg	9.92	4.47	16.04
disagg	15.82	286.78	25.58
agg+cuts	57.78	16.82	93.45
disagg+cuts	57.96	216.30	93.74

problem: quasiunif2			
formulation	value	time	quality (%)
agg	11.72	4.98	17.89
disagg	17.31	268.25	26.42
agg+cuts	59.64	14.67	91.05
disagg+cuts	60.22	188.50	91.94

problem: ring			
formulation	value	time	quality (%)
agg	33.81	6.15	27.27
disagg	38.19	297.83	30.80
agg+cuts	105.43	21.13	85.02
disagg+cuts	115.98	286.15	93.53

Table 3.1: Comparing the aggregated and disaggregated formulations.

The column labeled "quality" lists the ratio of the LP value to the best upper bound known for the problem (which are, respectively, within 5.5% of optimality for problem quasiunif1, within 3% of optimality for problem quasiunif2, and optimal for problem ring). Times are in seconds on a Sun Sparc2, using CPLEX 2.0 with primal steepest edge pivoting (consistently the best choice). The results above are typical. As expected, the disaggregated formulation is stronger than the aggregated one, but not spectacularly so (even with cuts added). In particular, the disaggregated formulation, by itself, does not cut the gap to the optimum to a few percent (in contrast to the more standard instances of FCNF as reported in [28] and [2]). On the other hand, the disaggregated formulation is substantially more expensive computationally: here we note that the three problems listed above have $n = 8$ (so the aggregated formulation has 449 columns and the disaggregated one, 1233, both after eliminating redundant columns). These results are typical. We would expect the disaggregated formulation to be prohibitively expensive for larger problems, and yet not a substantial improvement on the aggregated formulation. Consequently, we focused our work on the aggregated formulation, and all results reported below are for it.

(b) The choice of the constants M_{ij}^k in the variable upper-bound inequalities (3.3) above is important—in our experiments the quality of the formulation was highly dependent on keeping these numbers as small as possible. The following is a possible choice. For a given node (i.e. commodity) k, and edge (i,j) we set $M_{ij}^k = \sum_{v \neq k} t(k,v)$. Below we will discuss how to strengthen this bound. We also note that variable upper-bound inequalities make a linear program degenerate and so we might expect the LP-relaxation of ICONG(2) to be a difficult linear program, and it is, but not just for this reason. At first glance, replacing all the inequalities (3.3) corresponding to a single edge (i,j) with one inequality of the form $\sum_k f_{kij} \leq Ux_{ij}$ might appear to be a good strategy. But in our experiments, particularly with larger problems, this made the formulation significantly weaker. On the other hand, the number of inequalities (3.3) is very large and typically a small number of them are active. We used these inequalities as cutting planes.

3.1.3 How Difficult Is ICONG(2)?

Even though the gap between the LP relaxation of ICONG(2) and the optimum tends to be very large, it is conceivable that the problem could be solvable using branch-and-bound (possibly after adding some cutting planes) especially in the case of small instances. Early in our research, we made available to several groups of leading researchers the eight-node problem instance labeled **quasunif2** in the table above (56 0-1 variables). [Remark: the number of digraphs on eight nodes, with indegrees and outdegrees equal to 2, exceeds 10^9]. In fact, we made two formulations available. The first one is the standard one as given above, and its LP-relaxation value is 11.72. The second one contains a number of additional valid inequalities, that raise the LP-relaxation value to 59.06. We will refer to these two formulations as the *weak* and the *strong* formulation, respectively. The best lower and upper bound for this problem are 63.99 and 65.67, both obtained by Cook [8] by running his branch-and-bound code on our extended formulation as described in later sections.

Several experimental and commercially available codes were run on both instances. When run on the weak formulation, none of the codes obtained a lower bound higher than 30.00, and when run on the strong formulation, none of the codes improved the LP lower bound, in both cases despite very substantial running times (several days, using large machines) and branch-and-bound trees with hundreds of thousands of nodes.

On the positive side, all of the codes found integral solutions that (*a posteriori*) are within 2 or 3% of the likely optimum. In particular, using a variation of the strong formulation (using some of the inequalities given below) Wolsey [32] very rapidly found a solution of cost 66.20: he did this by fixing the 0-1 variables set to 1 by the linear program (6, out of 16 that will equal 1 in any feasible solution) and *exhaustively* running branch-and-bound on that branch of the tree (the lower bound did not improve, however). The final note in this story is that when the strong formulation was augmented to include many more of our inequalities, several codes significantly tightened the bounds and reduced the computational overhead ([8], [6]).

It is therefore clear that formulation ICONG(2) must be strengthened. Further, branch-and-bound may not be a practicable option when dealing with larger problems. This is due to the fact that the linear programs are very difficult. For example, in several instances with $n = 20$, the solution of each linear program required on the order of 5 minutes when the variable upper bounds were omitted, and this grew very quickly once cutting planes were added to the formulation. As stated before, in the "real-life" setting only a few hours may be available for dealing with a problem. So we would seek an algorithm that yields good bounds while solving a limited number of linear programs.

As a simple example of why these problems are combinatorially difficult, suppose that the demand matrix is given by:

$$t_{ij} = \begin{cases} 1, & \text{if } j = i+1 \text{ and } i < n \quad \text{or } i = 1 \text{ and } j = n, \\ 0, & \text{otherwise} \end{cases} \tag{3.6}$$

where we assume $n \geq 3$. With $p = 2$, it can be shown that the value of the problem is 2/3 if and only if there exists a directed graph of indegrees and outdegrees 2 containing the cycle $1 \cdot 2 \cdot 3 \cdots n-1 \cdot n \cdot 1$ as well as a length-two path from i to $i+1$ for each $i < n$ and also from n to 1. If no such digraph exists, the value of the problem is at least 3/4. Moreover, it can be shown that such a digraph exists only when n is odd. Further, if n is odd the digraph is unique and there is a unique way of routing the commodities to achieve value 2/3.

In summary,

- If n is odd, the value of the problem is 2/3 and there is a unique optimal solution. All feasible solutions which are different from the optimal in the integer variables have value at least 3/4.

- If n is even, the value of the problem is at least 3/4.

This type of problem would be difficult for any algorithm because the instances for n and for $n+1$ are very similar. Moreover, for n odd, until we find the optimal solution we will be more than 10% away from the optimum. While examples of this sort are admittedly contrived, we can expect subtle combinatorial difficulties to arise from the pattern of demand values.

With regards to previous work on valid inequalities for uncapacitated fixed-charge network flow problems, besides the dicut collection inequalities of Rardin and Wolsey [28] mentioned above, which subsume the "basic network inequalities" of Van Roy and Wolsey [31], very little appears to be known concerning polyhedral structure, especially facets, except in special cases, such as the economic lot-sizing problem. Balakrishnan, Magnanti and Wong [2] developed a computationally efficient procedure for solving the LP-relaxation of the disaggregated formulation for uncapacitated fixed-charge network flow problems without any side constraints. The disaggregated formulation for this problem turned out to be very tight.

3.2 Valid Inequalities

In this section we describe some of the inequalities that have proved useful towards obtaining good bounds. We are dealing with the aggregated formulation, so that a commodity will be identified with its source node. In what follows we will say that a digraph is *of degree 2* if the indegree and outdegree of every vertex is 2. For completeness, we state the following result, which is implied by some of the results presented below.

Lemma 3.2.1 *The dimension of* $ICONG(2)$ *is* $n^2(n-1)-(2n-1)+1$. ∎

This lemma simply states that the dimension of $ICONG(2)$ is precisely equal to the number of variables minus the rank of the formulation, that is, there are no additional implied equations. To see this, note that the expression above can be rewritten as $n(n(n-1)-(n-1))+n(n-1)-(2n-1)+1$. The first term here is n times the dimension of a one-commodity network flow polyhedron in a digraph with $n(n-1)$ edges. The next two terms correspond to the x variables. Here note that the degree constraints in $ICONG(2)$ describe a transportation problem in a graph with $2n$ nodes and $n(n-1)$ edges. Lastly, the variable z (which need not satisfy any inequality tightly) contributes one additional unit of dimension.

3.2.1 A Basic Facet

For any commodity k and subset S of nodes write $t^k(S) = \sum_{i\in S} t_{ki}$. The main result in this Section is:

Theorem 3.2.2 *Let* k *be a commodity and* $S \subset \{1,\cdots,n\} \setminus k$. *Write* $T = \{1,\cdots,n\} \setminus (S \cup k)$. *Then inequality*

$$\sum_{i\in T, j\in S} f_{kij} \geq \left(1 - \sum_{i\in S} x_{ki}\right) t^k(S) \tag{3.7}$$

is a facet of ICONG(2) provided $2 \le |S| \le n-3$, $t^k(S) > 0$ *and* $t^k(T) > 0$. ■

Validity of (3.7) is easy to see when $\sum x \ge 1$, and if $\sum x = 0$, then (3.7) is satisfied whenever the total flow of commodity k on the edges (i,j), $i \in S, j \in T$ is sufficient to satisfy the demand of S.

Before showing that (3.7) is a facet of ICONG(2), we first study a related but simpler polyhedron and show that (3.7) is facet defining for this polyhedron. We denote by B^k the set of k-vectors all of whose coordinates are 0 or 1. A digraph is called *strong* if it contains a directed path from each vertex to every other vertex. If $x \in B^{n(n-1)}$ we let $G[x]$ be the digraph whose edge set has incidence vector x. For any subset K of commodities, let P^K denote the convex hull of points satisfying the following constraints:

$$\sum_{j \ne i} x_{ij} = 2, \quad \text{for all } i \tag{3.8}$$

$$\sum_{j \ne i} x_{ji} = 2, \quad \text{for all } i \tag{3.9}$$

$$x_{ij} = 1 \quad \text{if} \quad f_{kij} > 0 \quad \text{for all } k \in K \text{ and } i \ne j \tag{3.10}$$

$$\sum_{j \ne i} f_{kji} - \sum_{j \ne i} f_{kij} = t_{ki} \quad \text{for all } i \text{ and } k \in K \tag{3.11}$$

$$G[x] \quad \textit{strong} \tag{3.12}$$

$$0 \le f_{kij}, \text{ all } k \in K \text{ and } i \ne j, \;\; x_{ij} \in \{0,1\}, \text{ all } i \ne j$$

In essence this is the formulation for ICONG(2) restricted to commodities in K, without the variable z, and with the added restriction that the digraph we use must be strong. We abbreviate $P^{\{k\}}$ as P^k and points in P^k will be given as pairs(x, f_k). In what follows we assume that commodity k and set S satisfy the conditions of Theorem (3.2.2).

Lemma 3.2.3 *Inequality (3.7) defines a facet of* P^k *if the conditions in Theorem (3.2.2) hold. Moreover,* $dimP^k = 2n(n-1) - (n-1) - (2n-1)$

Proof. By construction we will show that the related face $F = \{(x, f) \in P^k : (x, f_k)$ satisfy (3.7) with equality $\}$ is not empty and then by contradiction, we will show that it is a facet.

To simplify notation, assume $k = n$, $S = \{1, \ldots, s\}$ and $T = \{s+1, \ldots, n-1\}$. Let C_S and C_T be the directed cycles $n \cdot 1 \cdot 2 \cdots s \cdot n$ and $n \cdot s+1 \cdot s+2 \cdots n-1 \cdot n$ respectively. Furthermore let $\bar{C}_S$ and $\bar{C}_T$ be $s \cdot s-1 \cdots 1 \cdot s$ and $n-1 \cdot n-2 \cdots s+1 \cdot n-1$ respectively. We define G_0 to be the graph consisting of C_S, C_T, $\bar{C}_S$ and $\bar{C}_T$. If we denote the incidence vector of $E(G^0)$ by x^0, then clearly x^0 satisfies (3.8), (3.9) and (3.12).

In G^0, we first route the demands using the edges in C_S and C_T, and then increase the flows on all edges in $E(G^0)$ by a small amount $\epsilon > 0$. If we call the resulting flow vector f^0, it is clear that $p^0 = (x^0, f^0) \in F$ and $f^0_{nij} > 0$ whenever $x_{ij} = 1$.

Assume that F is not a facet of P^k (or $dimP^k < 2n(n-1) - (n-1) - (2n-1)$), then there is an equation of the form

$$\alpha x + \beta f = \pi \tag{3.13}$$

satisfied by all points $p = (x, f) \in F$, where α and β are vectors of appropriate dimension and π is a real number.

Let T be the directed tree with edge set $A = C_S \cup C_T \setminus \{(s, n), (n-1, n)\}$. If necessary by subtracting a linear combination of the flow-balance equalities (3.11) from (3.13) we can assume that $\beta_a = 0$ for all $a \in A$. Notice that we can perturb p^0 by sending circulation flows along the cycles C_S, C_T, $\bar{C}_S$ or $\bar{C}_T$ and obtain new points on. Therefore, $\beta_{sn} = \beta_{(n-1)n} = \beta_{s1} = \beta_{(n-1)(s-1)} = 0$. Furthermore, for all $(i, j) \in E(G^0)$, if $(j, i) \in E(G^0)$ then we can increase flows on (i, j) and (j, i) simultaneously to obtain new points, implying $\beta_a = 0$ for all $a \in E(G^0)$.

Next, for all $(i, j) \notin E(G^0)$, $i \neq n$, if $(j, i) \notin E(G^0)$ then we define the cycle C^{ij} to be the directed cycle that spans the nodes in $S \cup T \setminus \{i, j\}$ in decreasing order and use G^{ij} to denote the digraph consisting of the cycles C_S, C_T, C^{ij} and edges (i, j) and (j, i). If we denote the related incidence vector by x^{ij}, then clearly $p^{ij} = (x^{ij}, f^0) \in F$.

For all $i \in S$, $j \in S \cup T$, $j > i$, if $(i,j),(j,i) \notin E(G^0)$, then we can perturbate p^{ij} by sending circulation flows along the cycle $n \cdot 1 \cdot 2 \cdots i-1 \cdot i \cdot j \cdot j+1 \cdots n$ to obtain a new point in F. We can also send circulation flows on the two cycle $i \cdot j \cdot i$. Since the resulting points are on the face, $\beta_{ij} = 0$ for all $i \in S$, $j \in S \cup T$. Similarly, we can extend this idea to include $i = n$ and $j \in S \cup T$ and show that $\beta_{ij} = 0$ for all $i \neq j$ unless $i \in T$ and $j \in S$.

Let C^n be the directed cycle $n \cdot 1 \cdot 2 \cdots n-1 \cdot n$ and $\bar{C}^n$ be the cycle obtained by reversing the orientation of C^n. If necessary by subtracting a linear combination of the degree equalities (3.9) and (3.8) from (3.13) we can assume that $\alpha_a = 0$ for all $a \in C^n \cup \bar{C}^n \setminus (n,1)$. Using C^n and $\bar{C}^n$, we can construct a point on F such that all flow uses C^n edges, implying $\alpha_{n,1} = \pi$.

Let $C_{1,s+1}$ be the directed cycle $1 \cdot 2 \cdots s \cdot s+1 \cdot 1$. For any $i > j$, we next construct E^{ij} such that E^{ij} contains C_T and $C_{1,s+1}$. Furthermore, E^{ij} contains (i,j), if $(i,j) \notin C_T \cup C_{1,s+1}$. Let $C^{i,j}$ be the simple cycle formed by (i,j) and some of the edges in $C_T \cup C_{1,s+1}$, and let $\bar{C}^{i,j}$ be the cycle obtained by reversing the orientation of $C^{i,j}$. It is easy to see that using E^{ij} and $E^{ij} \setminus C^{i,j} \cup \bar{C}^{i,j}$, we can construct points on F and thus show that $\alpha_{ij} = \alpha_{ji}$ unless $j \in S$ and $i = n$.

Next, by means of swap operations, we will show that $\alpha_{ij} = 0$ unless $j \in S$ and $i = n$. Note that for any four distinct vertices u, v, j, $k \in S \cup T$, we can construct a point $p^{uvjk} = (\bar{x}, \bar{f})$ on F such that $\bar{x}_{uv} = \bar{x}_{jk} = 1$, $\bar{x}_{uk} = \bar{x}_{jv} = 0$ and $\bar{f}_{nuv} = \bar{f}_{njk} = 0$. Fix two vertices $u, v \in T$ and for any $i \in S \cup T \setminus \{u,v\}$ let $a_i = \alpha_{ui}$, $b_i = \alpha_{iv}$, $a_u = b_v = 0$ and $\alpha_{uv} = a_v = b_u = \Delta$. Notice that we can perturbate p^{uvjk} without changing the flow vector by swapping edges $\{uv\}$ and $\{jk\}$ with $\{uk\}$ and $\{jv\}$ implying that $\alpha_{jk} = a_k + b_j - \Delta$. Similarly, we can construct p^{uvkj} and swap $\{uv\}$ and $\{kj\}$ with $\{uj\}$ and $\{kv\}$ to show that $\alpha_{kj} = a_j + b_k - \Delta$. If we let $c_i = (a_i + b_i)/2$, then using $\alpha_{kj} = \alpha_{jk}$, we conclude that $\alpha_{kj} = c_j + c_k - \Delta$. Next, using $\alpha_{i(i+1)} = \alpha_{(i+1)(i+2)} = 0$ we conclude that $c_i = c_{i+2}$. This means that if we fix $u < n-1$ and choose v to be $u+1$, then $c_u = c_{u+1} = \Delta = 0$ implying $\alpha_{jk} = 0$ for all $j, k \in S \cup T$.

To show that $\alpha_{nj} = \pi$ for all $j \in S$, we construct points of the form p^0 such that instead of C_S we use cycles that span S using a different order. Similarly, we can show that $\alpha_{nj} = 0$ for all $j \in T$. Lastly, for all $i \in T$, $j \in S$, we construct a solution $\bar{p}^{ij} = (\bar{x}^{ij}, \bar{f}^{ij})$ with the following property. The edge set related with $\bar{x}^{ij}$ is such that it contains $\bar{C}_S$, $\bar{C}_T$ and a cycle that, starting from node n, spans all nodes in $S-i$ then spans nodes i and j and then spans all nodes in $T-j$. Clearly, for these points $\sum_{v \in S} \bar{x}^{ij}_{nv} = 0$ and $\sum_{u \in T, v \in S} \bar{f}^{ij}_{nuv} = \bar{f}^{ij}_{nij} = t(S)$, implying $\beta_{ij} = \pi/t(S)$. Therefore, we conclude that (3.13) is a linear combination of (3.7), flow-balance equalities (3.11) and degree constraints (3.8) and (3.9). ■

Using Lemma 3.2.3 we next prove two more lemmas and show that (3.7) defines a facet of $P^{\{k,h\}}$ for any $h \neq k$.

Lemma 3.2.4 *Let $h \neq k$ be a commodity. There exist affinely independent points $(x^i, f^i_k) \in P^k$, $1 \leq i \leq d$ (for some d) satisfying (3.7) with equality, such that:*

(1) For $1 \leq i \leq d$, $G[x^i]$ is strong and of degree 2,

(2) For $1 \leq i \leq d$, $G[x^i]$ contains an arborescence A^i rooted at h, such that for every edge $e \notin A^1$ there exists $1 \leq i \leq d$ with $e \in G[x^i] \setminus A^i$ but $e \notin G[x^j]$ for each $j < i$.

(3) For $2 \leq i$, $G[x^i]$ contains an edge not in $\cup_{j<i} G[x^j]$.

Proof. To simplify notation, assume $k = n$, $h \neq n-1$, $S = \{1, \ldots, s\}$ and $T = \{s+1, \ldots, n-1\}$. Let H be the digraph consisting of the cycle $1 \cdot 2 \ldots s \cdot n \cdot n-2 \cdot n-3 \ldots s+1 \cdot 1$ and the edge $(n, n-1)$. We have

(a) If $G \supseteq H$ is a strong digraph of degree 2 then there is a point $(x, f) \in P^n$ satisfying (3.7) with equality, such that $G = G[x]$,

(b) For every edge $e = (u, v) \notin H$ with $u \neq n$, there is a strong digraph G of degree 2 including $H \cup e$.

To see that (a) holds, notice that in the arborescence $H \setminus (s, n)$ all vertices of T are reached from n before the vertices of S, and that if G is as in (a) and $G = G[x]$ (for some

$\{0,1\}-$vector x) then $\sum_{i\in S} x_{ni} = 0$. To see that (b) holds, notice that if we add to H the edge $(n-1,n)$ and the cycle $C = s \cdot s-1 \cdots 2 \cdot 1 \cdot n-1 \cdot s+1 \cdots n-3 \cdot n-2 \cdot s$ we obtain a strong digraph of degree 2. So if e is in this digraph we are done. If not, then it is easy to see how to break up the cycle C to conclude that (b) holds. Let A be the arborescence obtained by removing from H the edge entering h. We can construct points $(x^i, f_k^i) \in P^k$, such that conditions (1) and (3) hold, and condition (2) holds for each edge e whose tail is not n (using $A^i = A$). It is easy to see that the same can be done for each edge of the form (n,t). The fact that the points (x^i, f_k^i) are affinely independent follows from (2). This concludes the proof of the Lemma. ∎

Lemma 3.2.5 *For any $h \neq k$ inequality (3.7) defines a facet of $P^{\{k,h\}}$.*

Proof. Let F be the face of $P^{\{k,h\}}$ induced by (3.7). To prove this lemma, we will construct $dimP^k + n(n-1) - (n-1)$ affinely independent points in F (thereby also showing that $dimP^{\{k,h\}} = dimP^k + n(n-1) - (n-1)$. The result will follow from this because the right-hand side in this expression is certainly an upper bound on $dimP^{\{k,h\}}$). For convenience, denote $U = dimP^{\{k,h\}} = dimP^k + n(n-1) - (n-1)$.

For $1 \leq j \leq U$ the jth point we will construct will be denoted by v_j. Let d be the quantity produced by Lemma 3.2.4. The points we construct will be of two types:

- Type 1. For $1 \leq i \leq d$, let $(x^i, f_k^i) \in P^k$, be the points produced by Lemma 3.2.4. For $1 \leq i \leq d$, let $C(i) = G[x^i] \setminus (A^i \cup_{j<i} G[x^j])$, and $c(i) = |C(i)|$, and write $G[x^i] = G^i$. For each $1 \leq i \leq d$, we will construct a set $F(i)$ of $1 + c(i)$ points in F, each of them having projection (x^i, f_k^i) in the (x, f_k) space. By construction, this will yield

$$\sum_{i=1}^{i=d}(1 + c(i)) = d + \sum_{i=1}^{i=d} c(i) = d + n(n-1) - (n-1) \tag{3.14}$$

$$= U - (dimP^k - d) \tag{3.15}$$

points.

- Type 2. If $d < dimP^k$, for each j with $U - (dimP^k - d) \leq j \leq U$ we construct an additional point v_j with the following property. If w_j is the projection of v_j in the (x, f_k) space, then we require that the family of points

$$\{(x^i, f_k^i) \,:\, 1 \leq i \leq d\} \cup \{w_j \,:\, U - (dimP^k - d) \leq j \leq U\} \tag{3.16}$$

be affinely independent.

First we handle the Type 1 points. Choose a fixed i, $1 \leq i \leq d$. For simplicity, when describing the points in $F(i)$ we will only give their f_h coordinates. Let g^i be a circulation flow (in the space of commodity h), such that for any edge $e \in G^i$, $g_e^k > \sum_j t_{hj}$. Then:

- We obtain one point of the form $g^i + a^i$, where a^i is the flow vector obtained by routing commodity h on A^i (this will be called the *first* point in F(i))
- For each $e \in C(i)$ we obtain an additional point of the form $g^i + a^i + \theta_e^i$, by pushing a small amount of flow along the cycle obtained by adding e to A^i.

Proceeding in this manner we produce $1 + c(i)$ points as desired.

Next, we construct the Type 2 points. By Lemma 3.2.3 it is clear that we can find points $w_j \in P^k$ satisfying (3.7) with equality so that the affine independence condition in the definition of Type 2 points is satisfied. By definition of P^k, the digraph corresponding to each of these points is strong, and so we can route commodity h arbitrarily, yielding points $v_j \in F$ as desired.

We claim that the points v_j, $1 \leq j \leq U$ are affinely independent. For suppose that for some coefficients α, $\sum_j \alpha_j v_j = 0$. By construction of the Type 2 points, we conclude $\alpha_j = 0$ for $j > d$. So we just have to show that the vectors in $\cup_{i \leq d} F(i)$ are affinely independent. For $1 \leq i \leq d$, subtract the first vector in $F(i)$ from the other vectors in $F(i)$, obtaining a family $F'(i)$. By construction, the last $c(i)$ vectors in $F'(i)$ will have projection $(0, 0)$ in the (x, f_k) space, but for each $e \in C(i)$ precisely one of these vectors will have a positive coordinate on edge e. Recall that for any such e we have $e \notin G^j$ for every $j < i$ (property (2) of Lemma 3.2.4). So if there is a nontrivial linear combination of the vectors in $\cup_{i \le d} F'(i)$ that adds to zero, then only the first vector from each $F(i)$ can have a nonzero coefficient. But these are affinely independent because their projections in the (x, f_k) space are, again by Lemma 3.2.4. This concludes the proof. ■

Proof (Theorem 3.2.2). By applying the same technique as in Lemma 3.2.5, one commodity at a time, we conclude that inequality (3.7) defines a facet of $P^{\{1,\cdots,n\}}$. That is, we can construct a family $\mathcal{T}$ of

$$dim P^k + (n-1)(n(n-1) - n + 1) \quad = \quad n(n+1)(n-1) - n(n-1) - (2n-1)$$

affinely independent points in $P^{\{1,\cdots,n\}}$, each satisfying (3.7) with equality. Each of the points of $\mathcal{T}$ yields a point in ICONG(2) by setting $z = \max_{ij} \sum_k f_{kij}$. Consequently, (3.7) defines a facet of ICONG(2) if we can find one additional (affinely independent) point. This is easy: we simply take one of the points in $\mathcal{T}$ and set the $z-$coordinate larger than $\max_{ij} \sum_k f_{kij}$. This concludes the proof of Theorem 3.2.2, modulo the proof of Lemma 3.2.3. ■

Notes:

(1) In the proof of Lemma 3.2.5 as given above, we need to allow positive flows of commodity h on edges entering h. On the other hand, when solving problem ICONG(2) we can set all variables of the form f_{hih} to 0. The proof of Theorem 3.2.2 can be adapted to take this into account.

(2) Given a facet of P^k, under what conditions is it also a facet of ICONG(2)? Of course one expects that this is always the case. It can be shown that if in the definitions of all the polyhedra above, the degree equations $\sum_j x_{ij} = \sum_j x_{ji} = 2$ are replaced with inequalities (≤ 2) the corresponding result holds.

One possible variant of inequality (3.7) is the following. Let k, S and T be as above, and let $U \subset S$. Then

$$\sum_{i \in T, j \in S} f_{kij} + t^k(S) \sum_{i \in S \setminus U} x_{ki} + \sum_{i \in U} f_{kki} \geq t^k(S) \tag{3.17}$$

is valid. However, computationally it has usually been the case that the variable upper-bound inequalities are tight for a given commodity k and edges (k,i). As a consequence, f_{kki} tends to be close to $(t^k(S) + t^k(T))x_{ki}$ with the result that (3.7) dominates (3.17).

It is difficult to separate over inequalities (3.7) (or in general, over dicut collection inequalities, of which (3.7) is a special case). Moreover, we remind the reader of the result in [28], that using all dicut collection inequalities yields the projection of the disaggregated formulation. Further, as we saw in the previous section, this formulation cannot be expected to be substantially stronger. On the positive side, it can be shown that (under fairly general conditions) (3.7) remains facet-defining even after fixing some of the x_{ij} variables.

To close this section, we point out the following (perhaps curious) result.

Proposition 3.2.6 *Let* $P = conv\{x \in B^{n(n-1)} : G[x]$ *of degree 2*$\}$ *and* $Q = conv\{x \in B^{n(n-1)} : G[x]$ *is strong and of degree 2* $\}$. *For* $n \geq 4$, $dimP = dimQ$. ∎

3.2.2 Source Inequalities

In this section we study three polyhedra related to variable upper-bound flow models (also see [25] and [24], Sections II.2.4 and II.6.4 for similar models.)

For any fixed $F > 0$ and integer $n > 2$, let $P_n(2F)$ be the convex hull of points $(x, f, z) \in R^{2n+1}$ (where $x \in R^n, f \in R^n$ and $z \in R$) satisfying:

$$\sum_{i=1}^{n} f_i = 2F \tag{3.18}$$

$$\sum_{i=1}^{n} x_i = 2 \tag{3.19}$$

$$x_i = 1 \text{ if } f_i > 0 \quad , \quad i \in \{1, \cdots, n\} \tag{3.20}$$

$$f_i \leq z \quad , \quad i \in \{1, \cdots, n\} \tag{3.21}$$

$$0 \leq f_i \quad , \quad i \in \{1, \cdots, n\} \tag{3.22}$$

$$x_i \in \{0, 1\} \quad , \quad i \in \{1, \cdots, n\} \tag{3.23}$$

In other words: we have a supply of $2F$ units of flow, that must be pushed through two edges chosen from among n candidates. The largest of all flows is (a lower bound for) z.

In what follows $e^i \in R^n$ will be the i'th unit vector, and $e^{ij} = e^i + e^j$. We have:

Lemma 3.2.7 *The dimension of $P_n(2F)$ is $2n - 1$ for $n > 2$.*

Proof. Consider the following $2n$ points in $P_n(2F)$: (e^{1i}, Fe^{1i}, F) for $2 \leq i \leq n$, $(e^{1i}, 2Fe^1, 2F)$ for $2 \leq i \leq n$, $(e^{12}, 2Fe^2, 2F)$, and (e^{23}, Fe^{23}, F). The $2n \times (2n+1)$ matrix formed by these vectors is

$$\left(\begin{array}{c|ccc|c|ccc|c}
1 & & & & F & & & & F \\
\vdots & & I_{n-1} & & \vdots & & FI_{n-1} & & \vdots \\
1 & & & & F & & & & F \\
\hline
1 & & & & 2F & & & & 2F \\
\vdots & & I_{n-1} & & \vdots & & 0 & & \vdots \\
1 & & & & 2F & & & & 2F \\
\hline
1 & 1\,0\,0 & \cdots & 0 & 0 & 2F\,0\,0 & \cdots & 0 & 2F \\
0 & 1\,1\,0 & \cdots & 0 & 0 & F\,F\,0 & \cdots & 0 & F
\end{array}\right) \tag{3.24}$$

and it can be seen that the rows of this matrix are affinely independent. (Actually, the matrix has full row rank). ∎

Because of equations (3.18) and (3.19), this result implies that the dimension of $P_n(2F)$ is precisely equal to the number of variables minus the rank of the formulation.

In what follows, for any vector v and $S \subseteq \{1, \cdots, n\}$, $v(S) = \sum_{i \in S} v(i)$. Consider the following inequalities:

$$z - f(S) + Fx(S) \geq F, \tag{3.25}$$

for all $S \subset \{1, \cdots, n\}$ with $1 \leq \mid S \mid < n$,

$$z - 2f(S) + 2Fx(S) - f_i \geq 0, \tag{3.26}$$

for all $S \subset \{1, \cdots, n\}$ with $1 \leq \mid S \mid \leq n-2$ and $i \notin S$, and

$$f_i \leq 2Fx_i \tag{3.27}$$

for all $i \in \{1, \cdots, n\}$. Below we will show that these are facet defining. Inequalities (3.25) and (3.26) strengthen the inequality $z \geq F$, which is valid because $2F$ units of flow must be routed on 2 edges. Typically, the linear programming formulation (once we replace (3.21) by an appropriate variable upper bound inequality) will "cheat" by spreading the $2F$ units of flow over more than 2 terms somewhat unevenly, while keeping z small. That is to say, for some indices i we may have $f_i > Fx_i$. Inequalities (3.25) and (3.26) cut off such fractional points. It would be interesting to obtain these facets using the MIR procedure ([24]).

Lemma 3.2.8 *Inequalities (3.25) - (3.27) define facets of $P_n(2F)$.*

Proof. Consider inequality (3.25) for a given subset S. If $x(S) = 0$ or if $x(S) = 2$ the inequality is valid, since in either case $f(S) - Fx(S) = 0$. If $x(S) = 1$, say $x_i = 1$ for some $i \in S$, then $z \geq f_i = f_i - F + F = f(S) - Fx(S) + F$. To see that (3.25) is facet inducing, assume w.l.o.g. that $S = \{1, \cdots, |S|\}$, and that $|S| \leq n-2$ (the case $|S| = n-1$ is similar). Write $s = |S|$. Consider the $2n-1$ points (e^{1i}, Fe^{1i}, F) for each $i \notin S$, $(e^{i,s+1}, Fe^{i,s+1}, F)$ for each $i \in S \setminus 1$, $(e^{1i}, 2Fe^1, 2F)$ for each $i \notin S$, $(e^{i,s+1}, 2Fe^i, 2F)$ for each $i \in S \setminus 1$ and $(e^{s+1,s+2}, Fe^{s+1,s+2}, F)$. Each of these points satisfies (3.25) with equality, and together they make up the matrix

$$\left(\begin{array}{c|c|c|c|c|c|c}
1 & 0 \;\cdots\; 0 & & F & 0 \;\cdots\; 0 & & F \\
\vdots & \vdots \;\cdots\; \vdots & I_{n-s} & \vdots & \vdots \;\cdots\; \vdots & FI_{n-s} & \vdots \\
1 & 0 \;\cdots\; 0 & & F & 0 \;\cdots\; 0 & & F \\ \hline
0 & & 1\,0 \;\cdots\; 0\,0 & 0 & & F\,0\,0 \;\cdots\; 0 & F \\
\vdots & I_{s-1} & \vdots \;\cdots\; \vdots & \vdots & FI_{s-1} & \vdots \;\cdots\; \vdots & \vdots \\
0 & & 1\,0 \;\cdots\; 0\,0 & 0 & & F\,0\,0 \;\cdots\; 0 & F \\ \hline
1 & 0 \;\cdots\; 0 & & 2F & 0 \;\cdots\; 0 & 0\,0\,0 \;\cdots\; 0 & 2F \\
\vdots & \vdots \;\cdots\; \vdots & I_{n-s} & \vdots & \vdots \;\cdots\; \vdots & \vdots \;\cdots\; \vdots & \vdots \\
1 & 0 \;\cdots\; 0 & & 2F & 0 \;\cdots\; 0 & 0\,0\,0 \;\cdots\; 0 & 2F \\ \hline
0 & & 1\,0 \;\cdots\; 0\,0 & 0 & & 0\,0\,0 \;\cdots\; 0 & 2F \\
\vdots & I_{s-1} & \vdots \;\cdots\; \vdots & \vdots & 2FI_{s-1} & \vdots \;\cdots\; \vdots & \vdots \\
0 & & 1\,0 \;\cdots\; 0\,0 & 0 & & 0\,0\,0 \;\cdots\; 0 & 2F \\ \hline
0 & 0 \;\cdots\; 0 & 1\,1 \;\cdots\; 0\,0 & 0 & 0 \;\cdots\; 0 & F\,F\,0 \;\cdots\; 0 & F
\end{array}\right) \tag{3.28}$$

from which it is easy to argue that the points are affinely independent, as desired. The proofs that (3.26) and (3.27) are facet inducing are similar. ∎

Proposition 3.2.9 *The facet defining inequalities (3.25), (3.26) and (3.27), together with (3.18), (3.19), (3.22) as well as the bounds* $0 \leq x_i \leq 1$ *(for every* i*) yield the polyhedron* $P_n(2F)$ *.* ∎

In terms of problem ICONG(2), inequalities (3.25) and (3.26) and (3.27) cannot be used directly as cutting planes (for any given node k) because the value of F is not fixed. But we obtain valid inequalities by (1) replacing $x(S)$ with $2 - x(V \setminus S)$, (2) replacing F in all linear terms with the total flow leaving k ($\sum_{i,h} f_{hki}$), and (3) in the terms of the form $-Fx(S)$ by replacing F with a lower bound. Such a lower bound is available: half of the sum of all demands with source node k. The resulting inequalities can be improved if an upper bound on F is available, or better, if an upper bound u on the individual flows f_i

is available. Such an upper bound is frequently available if we have an upper bound on the value of the instance of ICONG(2). A different perspective on this is the following. Notice that in the preceding proofs we constructed points with $f_i = 2F$ for certain indices i. That is, all flow is routed on one edge. In computation, we would expect that this might not happen, i.e. flow would be split at least among two edges so as to keep z small. This suggests that the above inequalities may not necessarily be active, especially after a few rounds of cutting planes. What might be needed, instead, are inequalities that handle cases where z is larger than F but much smaller than $2F$.

To that effect we first consider, for given $0 < F \leq u$ the polyhedron $Q_n(2F, u)$ obtained by replacing (in the formulation of $P_n(2F)$) (3.20) with $f_i \leq ux_i$. This is of interest when $F < u$. By adapting the proof of Lemma 3.2.8 one has:

Lemma 3.2.10 *For all* $0 < F < u$ *(3.25) and (3.26) induce facets of* $Q_n(2F, u)$ *.* ∎

We omit this proof because it is very similar to the one for Lemma 3.2.8. All one has to do is replace the points with $z = 2F$ with points where $z = u$, which is possible because $F < u$.

There is another way of improving on the above polyhedra, which arises by stating that we have to push *at least* $2L$ units of flow, rather than exactly $2F$ units, for some L. More precisely, consider for fixed $0 < L < u$ the polyhedron $R_n(2L, u)$ obtained by replacing (in the formulation of $Q_n(2L, u)$) (3.18) with $\sum_i f_i \geq 2L$. Computationally, this has been most useful when $L < u < 2L$, in which case the variable upper bound inequalities become active. One can obtain facets for $R_n(2L, u)$ by starting with facets for $Q_n(2L, u)$ and "lifting" them so as to be tight for at least one point in $Q_n(2u, u)$. In this way one has (where $V = \{1, \cdots, n\}$):

Lemma 3.2.11 *For every* $0 < L < u$ *the following inequalities define facets of* $R_n(2L, u)$*:*

$$z \;\geq\; \frac{1}{2}f(S) - \frac{1}{2}f(V \backslash S) - Lx(S) + 2L, \tag{3.29}$$

for all $S \subset V$ *with* $2 \leq | S | \leq n - 1$, *and*

$$z \geq 2\left(Lx(S) - f(S)\right) + f_j + \min\left\{1, \frac{u}{2(u-L)}\right\}\left(f(V) - 2L\right), \tag{3.30}$$

for all $S \subset V$ with $2 \leq \mid S \mid \leq n-2$ and $j \in S$, and

$$z \geq f_j - Lx_j + L \tag{3.31}$$

$$f_j \geq (2L - u)x_j \tag{3.32}$$

$$z \geq f(V) - f_j - u(1 - x_j) \tag{3.33}$$

■

We obtained these inequalities using the lifting procedure described above, which guarantees that they are facets. Inequality (3.29) is especially active when we have a set S with $x(S) = 1$ (or close to it) and $f(S) > f(V \backslash S)$. The inequality says that in an integral solution *the* edge used in S must carry flow at least $\frac{1}{2}f(S) - \frac{1}{2}f(V \backslash S) + L$ (which is equal to $2L - f(V \backslash S)$, thus showing validity), a value strictly larger than L. As a result, the inequality tends to cut off fractional points with slight imbalances among the variables f_i. The other inequalities above have similar interpretations.

3.2.3 Flux Inequalities

In [17] the following procedure was used to obtain a lower bound for z in an instance of ICONG(2): if F^* is a lower bound on $\sum_k \sum_{i,j} f_{kij}$ then clearly $z \geq F^*/2n$. To compute such a lower bound F^*, we denote by F^1 the sum of the $2n$ largest demands, and for $d > 1$ F^d the sum of the $2^{d-1}n + 1$ through $2^d n$ largest traffic demands. Then

$$F^* = \sum_d dF^d \tag{3.34}$$

is a valid lower bound, since in a digraph with outdegrees at most 2, at most $2^d n$ pairs of vertices are at distance d, for any d. This lower bound tends to be weak because it averages over all commodities. In particular, for any given node i at most 2^d nodes can be at distance d from i, yet the quantity F^d may include more than 2^d traffic elements, all with source node i. A heuristic way of improving the lower bound is given in [17].

Equation (3.34) can be strengthened if we adapt it to work one commodity at a time. In what follows we consider a fixed commodity (i.e. source node) k. For a subset of edges S, write $f_k(S) = \sum_{i,j \in S} f_{kij}$. We call $f_k(S)$ the *flux* of commodity k on S. Denote by E the set of all edges, i.e. the set of all ordered pairs of nodes. Then

$$f_k(E) \geq \sum_d dT^d \tag{3.35}$$

where T^1 is the sum of the two largest demands with source k, and for $d \geq 1$, T^d is the sum of the $2^{d-1}+1$ through 2^d largest such demands. This inequality can be strengthened by noting that for any node $i \neq k$, if i is at distance one from k then i contributes t_{ki} to $f_k(E)$ (and otherwise i contributes at least $2t_{ki}$). So we can write

$$f_k(E) \geq \sum_{i \neq k} (2 - x_{ki}) t_{ki} + \sum_{d>2} \left[(d-2) \sum \{ t_{ki} : i \text{ at distance } d \text{ from } k \} \right] \tag{3.36}$$

and so

$$f_k(E) \geq \sum_{i \neq k} (2 - x_{ki}) t_{ki} + \sum_{d>2} (d-2) T^d. \tag{3.37}$$

This inequality can further be strengthened as follows. For any d let $V(d)$ be the set of nodes i such that t_{ki} is included in T^d. For any $d > 1$, let

$$s(d) = \sum_{2 \leq h < d} min\{ t_{ki} : i \in V(h) \}. \tag{3.38}$$

Then:

$$f_k(E) \geq \sum_{i \neq k} (2 - x_{ki})t_{ki} + \sum_{d>2} (d-2)T^d + \sum_{d>2} \sum_{i \in V(d)} (s(d) - (d-2)t_{ki})x_{ki} \tag{3.39}$$

is a valid inequality. To see this, notice that if $x_{ki} = 1$ for some $i \in V(d)$ with $d > 2$, then the right-hand side of inequality (3.36) exceeds that of inequality (3.37) by at least $s(d) - (d-2)t_{ki}$.

Example 3.2.12 *Suppose the nonzero demands t_{ki} are $100, 90, 75, 70, 68, 62$ and 30, corresponding to nodes $i = 1, \cdots, 7$. Then inequalities (3.35), (3.37) and (3.39), respectively, are*

$$f_k(E) \geq 830, \tag{3.40}$$

$$f_k(E) \geq 1020 - 100x_{k1} - 90x_{k2} - 75x_{k3} - 70x_{k4} - 68x_{k5} - 62x_{k6} - 30x_{k7}, \tag{3.41}$$

$$f_k(E) \geq 1020 - 100x_{k1} - 90x_{k2} - 75x_{k3} - 70x_{k4} - 68x_{k5} - 62x_{k6} + 2x_{k7}. \tag{3.42}$$

In the rest of this section we discuss valid inequalities involving quantities $f_k(S)$ as well as further ways of strengthening (3.37) and (3.39). First we need an auxiliary result. The following inequality, while not globally valid, does not cut off at least one optimal integral solution; and at the same time it is fairly useful towards strengthening the formulation.

Proposition 3.2.13 *Let z^L be a known lower bound on the value of a problem. For any commodity k and node i the following inequality is valid, without loss of generality:*

$$f_{kki} \geq min\{t_{ki}, z^L\}x_{ki} \tag{3.43}$$

Proof. If $x_{ki} = 0$ validity is clear, and so assume $x_{ki} = 1$, and that $f_{kki} < t_{ki}$. In this case there is a path P from k to i carrying positive flow of commodity k. Since $f_{kki} < z$, we can reroute a small amount of flow from this path to the edge (k, i) (and if necessary reroute flow of other commodities from (k, i) to P) without increasing the value of z. ■

There are many reasons why inequalities (3.35) or (3.39) may fail to be tight, and we discuss two of them next. Notice that for these inequalities to be tight or nearly so (at an integral solution) commodity k must be routed using a shortest path tree that is balanced. Such a tree will not exist if the graph contains certain subgraphs, which we might call "obstructions". For example, if the graph contains edges (k, i) and (i, k) for a given i, then there will be at most three vertices at distance 2 from k (rather than four) and this will affect the number of vertices at all distances $d \geq 2$ from k. Thus in principle inequality (3.35) can be strengthened as follows:

$$f_k(E) \geq \sum_d dT^d + \sum_{i,j} \alpha_{ij}(x_{ij} + x_{ji} - 1)^+ \tag{3.44}$$

for appropriate parameters α_{ij}. Similar inequalities can be used with other obstructions. By themselves, inequalities of this type are rather weak. However, it can be the case that one can argue that at least some number of obstructions *must* occur in any digraph of degree two. For example, it can be shown that for $n = 8$ at least one of three obstructions must occur: at least one pair of parallel edges (i, j), (j, i), or a triple of edges (i, j), (i, h), (h, j), or a quadruple (i, j), (i, h), (j, g), (h, g). This fact can be stated with an appropriate valid inequality. In conjunction with inequalities (3.44), inequalities of this type can have a powerful effect on the quality of the lower bound, especially near optimality. On the other hand, determining the numbers of obstructions that must occur is a very difficult combinatorial problem and so this technique can be used strictly in an ad hoc manner.

The other reason why inequalities (3.35), (3.39) can be weak can be handled computationally, and it occurs when some demand is large compared with some known upper bound on the value of the problem and with the other demands with same source. As a simple example, suppose that the (positive) demands with source k are: $200, 50, 30, 20, 14, 10, 8$ corresponding to nodes $1, \cdots, 7$ and that we already know that 188 is an upper bound on the value of the problem. Since t_{k1} is so large the linear program will usually set $x_{k1} = 1$, and so it is important to strengthen our inequalities in this case. But for (3.35) or (3.39) to be tight we would again need a balanced tree, i.e. one where the demands of at least two nodes are routed on edge $(k, 1)$. But $t_{k1} > 188$ and so less than t_{k1} units of flow are routed on this edge and the tree becomes unbalanced. If $x_{k1} = 1$ it is easily seen that

$$f_k(E) \geq 188 + 50 + 2(30 + 20) + 3(14 + 10 + 8 + 12) = 470 \tag{3.45}$$

(whereas the right-hand side of (3.35) is 422). If $x_{k1} = 0$ then $f_k(E) \geq 602$. This can be argued as follows. The inequality is clearly valid if node 1 is at distance three or greater from k. If it is at distance two, the fact that $z \leq 188$ implies that at most three nodes can be at distance two from k, and consequently

$$f_k(E) \geq 50 + 30 + 2(200 + 20 + 14) + 3(10 + 8) = 602 \tag{3.46}$$

as desired. As a result we have that $f_k(E) \geq 602 - 132x_{k1}$ is a valid inequality which will dominate (3.39), and which can itself be improved in the same way that (3.39) improves over (3.35).

The general procedure for handling this kind of situation is given next. We use the following notation. Let $D = \{d_1, d_2, \cdots\}$ be a list of numbers. If $\mathcal{Q} = \{Q_1, Q_2, \cdots\}$ is an ordered partition of D,

$$w^j(\mathcal{Q}) = \sum_{d_h \in Q_j} d_h, \tag{3.47}$$

and

$$W(\mathcal{Q}) = \sum_j j w^j(\mathcal{Q}). \tag{3.48}$$

If $S \subseteq D$ we let $\mathcal{R}(D, S)$ be an ordered partition $\{D_1, D_2, \cdots\}$ of D such that $D_1 = S$, and for $j \geq 2$, D_j contains the $2(2^{j-2} - 1)|S| + 1$ through $2(2^{j-1} - 1)|S|$ largest elements of

$D \setminus S$, and we write

$$L(D,S) = W(\mathcal{R}(D,S)), \tag{3.49}$$

$$L^i(D) = \min_{|S|=i}\{L(D,S)\}, \tag{3.50}$$

(note: the right-hand side of (3.35) equals $L^2(D)$, where D is the list of all demands with source k). In what follows, z^U and z^L denote known upper and lower bounds on the value of a given instance of ICONG(2). Now we have:

Lemma 3.2.14 *Suppose that for some commodity k and node $i \neq k$ we have $z^U < t_{ki}$. Let D be the list of demands with source node k, and let $\mathcal{Q} = \{D_1, D_2, \cdots D_m\}$ be the ordered partition that attains $L^1(D \setminus t_{ki})$. Define:*

$$l_d = min\{t_{kj} : t_{kj} \in D_d\}, \tag{3.51}$$

for $d < m$ and also for $d = m$ if D_m has 2^{m-1} members; $l_m = 0$ otherwise. Also, write $l_m + 1 = 0$. Set

$$i^1 = min\{s : 2 \leq s \leq m+1,\ l_s < t_{ki} - z^L\} \tag{3.52}$$

$$i^2 = min\{s : s \geq i^1,\ l_s \leq t_{ki} - z^U\}. \tag{3.53}$$

Let $\mathcal{P}$ be the ordered partition that attains

$$min\{L(D,S) : |S| = 2, t_{ki} \notin S\}. \tag{3.54}$$

Then :

$$
\begin{aligned}
f_k(E) \quad \geq \quad & \textstyle\sum_j (2 - x_{kj})t_{kj} + \left(\sum_{d>2} (d-2)w^d(\mathcal{Q}) + \sum_{d \geq i^2} l(d)\right) x_{ki} \\
& +(i^2 - i^1 - 1)(t_{ki} - z^U)x_{ki} + (i^1 - 1)(t_{ki}x_{ki} - f_{kki}) \\
& + \left(\textstyle\sum_{d>2} (d-2)w^d(\mathcal{P})\right)(1 - x_{ki})
\end{aligned}
\tag{3.55}
$$

is valid.

Proof. Suppose first that $x_{ki} = 0$. Then the right-hand side of the inequality is $\sum_j (2 - x_{kj})t_{kj} + \sum_{d>2} (d-2)w^d(\mathcal{P})$ and as in (3.39) it is easy to see that this is valid. Next, suppose $x_{ki} = 1$. By Proposition 3.2.13 $f_{kki} \geq z^L$. So, writing $\epsilon = t_{ki} - f_{kki}$,

$$\epsilon \leq t_{ki} - z^L. \tag{3.56}$$

Let h be such that $l_h \leq \epsilon$, and either $\epsilon < l_{h-1}$ or $h = 2$. Since $\epsilon > 0$, $f_k(E) - f_{kki}$ is lower bounded by a quantity of the form (c.f. (3.49)) $L(D', \{t_{ks}\})$, where D' is obtained from D by replacing t_{ki} with ϵ and $s \neq i$. The assumption on ϵ implies that the partition that attains $L(D', \{t_{ks}\})$ can be obtained from $\mathcal{Q}$ by putting ϵ in set h, and moving l_h to set $h+1$, l_{h+1} to set $h+2$, and so on. So we can write:

$$f_k(E) - f_{kki} \quad \geq \quad \sum_d dw^d(\mathcal{Q}) + h\epsilon + \sum_{d \geq h} l_d \tag{3.57}$$

and arguing as when obtaining (3.37) from (3.35), this inequality can be strengthened as follows:

$$
\begin{aligned}
f_k(E) - f_{kki} \quad \geq \quad & \sum_{j \neq i} (2 - x_{kj})t_{kj} \\
& + \sum_{d>2} (d-2)w^d(\mathcal{Q}) + h\epsilon + \sum_{d \geq h} l_d.
\end{aligned}
\tag{3.58}
$$

Further, $f_{kki} \leq z^U$ implies that $\epsilon \geq t_{ki} - z^U$, and since by definition of i^2 we have that

$h \leq i^2$, we conclude

$$h\epsilon + \sum_{d \geq h} l_d \geq i^1\epsilon + \sum_{d \geq i^2} l_d + (i^2 - i^1 - 1)(t_{ki} - z^U)x_{ki}. \tag{3.59}$$

Moreover, $i^1\epsilon = (i^1 - 2)(t_{ki}x_{ki} - f_{kki}) + 2(t_{ki} - f_{kki})$ and so

$$\sum_{j \neq i} (2 - x_{kj})t_{kj} + i^1\epsilon = \sum_j (2 - x_{kj})t_{kj} + (i(1) - 2)(t_{ki}x_{ki} - f_{kki}) + t_{ki}x_{ki} - 2f_{kki}, \tag{3.60}$$

and combining (3.58), (3.59) and (3.60) one obtains the right-hand side of (3.55). ∎

Example 3.2.15 *Let* $D = \{300, 80, 70, 10, 10, 9, 7, 6, 5, 5, 5, 4, 3, 3, 2, 1\}$ *(where* $t_{k1} = 300$*). Suppose* $z^U = 291$ *and* $z^L = 262$*. Then*

$$\mathcal{Q} \;=\; ((80), (70, 10), (10, 9, 7, 6), (5, 5, 5, 4, 3, 3, 2, 2), (1)) \tag{3.61}$$

$l(2) = 10$, $l(3) = 6$, $l(4) = 2$, $l(5) = 0$, $i^1 = 2$, $i^2 = 3$, *and*

$$\begin{aligned} \mathcal{P} &= ((80, 70), (300, 10, 10, 9), (7, 6, 5, 5, 5, 4, 3, 3), (2, 2, 1)) \\ \textstyle\sum_{d>2} (d-2)w^d(\mathcal{Q}) &= 92 \\ \textstyle\sum_{d \geq i^2} l_d &= 8. \end{aligned} \tag{3.62}$$

The valid inequality is:

$$\begin{aligned} f_k(E) &\geq \textstyle\sum_j (2 - x_{kj})t_{kj} + 100x_{k1} + 300x_{k1} - f_{kk1} + (38 + 10)(1 - x_{k1}) \\ &= \textstyle\sum_j (2 - x_{kj})t_{kj} + 352x_{k1} - f_{kk1} + 48. \end{aligned} \tag{3.63}$$

A similar type of valid inequality can be used when $t_{ki} < z^U$, although the situation is more complicated in this case. We will need some further notation. Let D be a list of numbers, and consider an element $d_i \in D$ and a number $u \geq d_i$. Then we write

$$W(D, d_i, u) = \min_K \left\{ \ \min_{A,B}\{L(A, d_i) + L^1(B)\} \ \right\}, \tag{3.64}$$

where

(1) K is any list obtained from D by replacing, for some $j \neq i$, d_j with two numbers of the form $d_j - \epsilon$ and ϵ, where $0 \leq \epsilon \leq d_j$, and

(2) A and B are lists whose union is K, such that $d_i \in A$ and the sum of the elements of A is at most u.

Example 3.2.16 *Let* $D = \{20, 20, 9, 8, 5\}$, $d_i = 20$ *and* $u = 23$. *Then the minimum is attained by setting* $K = \{20, 20, 9, 8, 3, 2\}$, $A = \{20, 3\}$ *and* $B = \{20, 9, 8, 2\}$.

Remark. The knapsack problem is a special case of that of computing quantities $W(D, d_i, u)$.

Inequality (3.35) can be strengthened as follows:

Proposition 3.2.17 *Consider an instance of ICONG(2) where* z^U *is a known upper bound on the value of the problem such that for some commodity* k *and node* i, $t_{ki} \leq z^U$. *Let* D *be the list of all demands with source node* k. *Then*

$$f_k(E) \geq W(D, t_{ki}, z^U)x_{ki} + (min_S L(D, S))(1 - x_{ki}), \tag{3.65}$$

where the minimum is taken over all pairs S *with* $t_{ki} \notin S$. ■

The proof of this proposition follows easily from the definitions given above. We also note that (3.65) can be strengthened by adding to the right-hand side terms involving variables x_{kj}, $j \neq i$.

Example 3.2.18 *Let* $D = \{300, 80, 50, 30, 25, 20, 16, 12\}$, $t_{ki} = 300$ *and* $z^U = 305$ *(note* *that the sum of entries in D is* 533*). Then we have:*

$$f_k(E) \geq 561 + \sum_j (1 - x_{kj})t_{kj} - f_{kki} + 345x_{ki} \tag{3.66}$$

To see why this holds, notice that if $x_{ki} = 0$ *the last two terms disappear in the above inequality disappear and we obtain inequality (3.37). If* $x_{ki} = 1$*, then we can write* $f_{kki} = 300 + \epsilon$*, where* $\epsilon \leq 5$*. It is easy to verify that in this case* $f_k(E) \geq 561 + \sum_j (1 - x_{kj})t_{kj} - \epsilon + 25 + 20 = 561 + \sum_j (1 - x_{kj})t_{kj} - f_{kki} + 345$*.*

The final type of flux inequalities that we use involve arbitrary node subsets S. Then it is possible to write an inequality of the form

$$\sum_{j \in S} f_{kij} \quad \geq \quad L + \sum_{j \in S} \alpha_{ij} x_{ij} + \sum_{i \notin S, j \in S} \beta_{ij} f_{kij}, \tag{3.67}$$

much in the same way that inequalities (3.35), (3.37) where generated. It is computationally difficult to separate over inequalities of the form (3.67) and we have used them strictly in an ad hoc manner (see Section 3.3). Nevertheless, these inequalities are experimentally very useful when there are *clusters*: a cluster is a subset of nodes C such that the traffic demands are large amongst members of C, but low between C and its complement. In such a case we would use inequality (3.67) with $S = C$, or $S = C \setminus i$ for some $i \in C$.

3.2.4 Other Inequalities

Here we describe various simple inequalities that appear useful in computation.

The first such inequality is (3.43) described in the previous section: for any commodity k and node i, $f_{kki} \geq \min\{t_{ki}, z^L\}x_{ki}$, for any lower bound z^L on the value of the problem.

The next class of inequalities are intended to strengthen the variable upper-bound inequalities (3.3). Consider any commodity k. In any feasible integer solution solution there will be two edges of the form (k, h), say (k, a) and (k, b) where $t_{ka} \leq t_{kb}$. Since we must route commodity k to satisfy the demand of each of a and b, as before one to argue that without loss of generality *both* (k, a) and (k, b) carry an amount of commodity k of value at least t_{ka}. Furthermore, for any edge (i,j), the flow of commodity k on this edge does not include any flow destined to satisfy the demand at i.

As a consequence, in inequality (3.3) we can set

$$M_{ij}^k = \sum_h t_{kh} - t_{ki} - \min_{h \neq i}\{t_{kh}\}. \tag{3.68}$$

A different (and usually more effective) way of tightening the variable upper bounds is given by the following inequality, which is one of the "basic network inequalities" of [31]:

$$f_{kij} \leq t_{kj}x_{ij} + \sum_{h \neq i} f_{kjh}, \tag{3.69}$$

and whose validity is clear. In general, if S is a subset of nodes not containing node j, we can write

$$\sum_{i \in S} f_{kij} \leq \min\{1, \sum_{i \in S} x_{ij}\}t_{kj} + \sum_h f_{kjh}. \tag{3.70}$$

The following inequality can be used to tighten inequality (3.5) of the original formulation. For any pair i, j we have:

$$z \geq \sum_k f_{kij} + z^L(1 - x_{ij}). \tag{3.71}$$

Experimentally, this inequality has been very effective in terms of improving the lower bound z^L.

3.3 Computational Results

In our computational experiments, the initial formulation consisted of the degree equations, the flow conservation equations, and inequalities (3.71) to measure the maximum load (a lower bound on the value of the problem is always available as discussed before). The variable upper bounds strengthened as in (3.68) were used as cutting planes. Further, in many cases it proved advantageous to introduce new variables that aggregate others, in this way obtaining a sparser formulation. For example, for any edge (i, j) we can add the equation $F_{ij} = \sum_k f_{kij}$ and use F_{ij} instead of the right-hand side elsewhere. This strategy usually resulted in faster solving linear programs. The inequalities that we used were (3.7), those described in Lemma 3.2.11, (3.39), (3.2.13), (3.55), (3.65), (3.69), (3.70), as well as some of the inequalities strengthening (3.35), such as (3.67). These, although very strong, are difficult to separate and we typically added some of them to the formulation before running the automatic part of the algorithm.

We first applied the algorithm to the eight-node problems from [16] (also see [17]). As stated before, the resulting mixed-integer programs have 56 $\{0, 1\}$ variables, approximately 400 columns and start with approximately the same number of rows; typically we would end up with roughly 2000 rows. Each of the linear programs took (on the average) 7 to 10 seconds on a SPARC2 machine, using Cplex 2.0. We observed that the improvement in the lower bounds would taper off rather quickly after three or four iterations, and thus the overall algorithm would run in less than one minute. Some of our inequalities benefit from having good lower and upper bounds on the value of the problem. As a result, in all but one of the cases we ran the cutting-plane algorithm twice: once to get a lower bound (and an upper bound, see below) and then again with the bounds in place (in one of the cases the algorithm was run three times).

Table 3.2 summarizes our experience with the eight-node problems. For each problem, the column labeled "LP relax." gives the LP-relaxation value of ICONG(2) including all variable upper-bound inequalities, the column labeled "strong ineqs." gives the best lower bound obtained by the cutting plane algorithm, "upper bound" is our upper bound and "GAP" is the percentage gap between our bounds.

We were able to solve problems **ring** and **disconn** by running branch-and-bound on our extended formulation (with all variable upper bound inequalities added). This required several tricks. For example, problem **disconn** has the following structure: the nodes are partitioned into two classes, such that traffic demands are large between nodes of the same

problem	LP-relax.	strong ineqs.	upper bound	**GAP** (%)
uniform2	10.00	66.25	66.67	0.63
quasunif1	9.92	58.63	61.83	5.46
quasunif2	11.72	62.70	65.67	4.73
ring	33.81	113.74	124.00*	9.02
central	95.71	335.00	335.00*	0.00
disconn	62.50	255.80	275.40*	7.66

Table 3.2: Performance of cutting-plane algorithm and heuristics for design-routing problem (* = optimally solved).

class and small otherwise. The linear program will set the sum of the x variables going from one class to the other to a value between 1 and 2. So we can branch, by setting this sum of variables to 1, or 2, or at least 3. In each of these cases we can significantly strengthen the formulation by using inequalities of type (3.67) as well as others. Each of the resulting three problems had an LP-relaxation value (after running the cutting plane algorithm) within 3% of the optimum, and each was separately solved to optimality using branch-and-bound (requiring a few hours in each case). A similar trick solved problem **ring**. Problem **central** is quite easy and the cutting plane algorithm quickly found the integer optimum.

The upper bounds in Table 3.2 were obtained by us by branch-and-bound, except for the bound for problem **quasiunif2** which was obtained by W. Cook, by running his branch-and-bound algorithm on the extended formulation (which required several days of computing). To obtain upper bounds, the strategy suggested by L. Wolsey worked rather well for all problems: fix at 1 all $\{0,1\}$-variables that are set that way by the cutting-plane algorithm, and run branch-and-bound on the remaining variables. This approach always found good solutions very quickly, sometimes in a few seconds. Typically these quick solutions were no more than 5% away from the optimum (and sometimes closer than that). We also used a (slower) heuristic to obtain better upper bounds that would run the cutting-plane algorithm, round to 1 some of the x_{ij} variables (based on their fractional value and using a randomized rule), and repeat until an integral solution was found. The overall process ran relatively quickly and we ran it several times for each problem.

In order to further test the strength of our lower bounds, we randomly generated sparse 20-node problems. In these problems the nodes are partitioned into clusters of four or five nodes each, such that traffic demands between two nodes are positive if and only if both nodes are in the same cluster. This structure makes it easy to find upper bounds – we solve the problem restricted to each cluster separately. For problems of this type, it is often the case that the optimal solution has edges between clusters and potentially these upper bounds could be crude. As discussed below, however, the linear programs that arose in these experiments were extremely difficult, and consequently we did not run branch-and-bound or our randomized heuristic on these problems, with the result that the simple upper bounds were all we had.

When running the cutting-plane algorithm, we employed a similar strategy as for the small problems, with one exception: the facet-defining inequalities (3.7) were used for (i.e. separated over) all small subsets, and heuristically for large subsets. We do not include a table of results for the randomly generated problems, but they can be summarized as follows. In all but one of these experiments the lower bounds we obtained were within 8 to 9% of our upper bounds. In one problem with clusters of size 5, the gap was approximately 11%.

As stated before, the linear programs arising here are quite difficult. Typically, the first linear program solved by the cutting-plane algorithm (i.e. one without the variable upper-bounds) required on the order of 5 minutes (still using Cplex 2.0 on a SPARC2). After adding some cutting-planes, the solution time would grow very quickly, sometimes to over one hour per LP. It is not clear precisely what makes these linear programs difficult. It is known that vubs (variable upper-bounds) make a formulation degenerate, and at first glance that could be a problem here. However, very few vubs would be explicitly added to the formulation in the course of the algorithm. Moreover, the approach of randomly perturbing the vub coefficients did not seem to help at all. A similar approach would be to perturb the objective function and run a dual simplex algorithm; this helped if the perturbation was rather large, but undoing the perturbation proved just as hard as solving the problem itself.

Table 3.3 displays information concerning the solution of one of the initial linear programs using various pivoting strategies, now running Cplex 2.1 (still on a SPARC2).

	primal steepest edge	primal devex	primal red. cost	dual steepest edge	dual std. pricing
iterations	3244(790)	8040(760)	25571(2573)	9288(2532)	18613(301)
time	275.60	484.03	445.57	846.38	1684.88

Table 3.3: LP solution statistics

This linear program has 8021 columns, 1320 rows and 32520 nonzeros. In the iterations row, the data in parenthesis indicates Phase I pivots. Times are in seconds.

This data is typical (perhaps even a bit conservative) in that it shows that primal steepest edge pivoting is the best strategy (at least using our LP solver). The interior point code OB1 was also run on this problem, with negative results [21].

It is clear that in order to make branch-and-bound practicable (or even to run a traditional cutting-plane algorithm) a way must be found to speed-up the solution of these linear programs. (Presumably the LPs for problems on more than twenty nodes will be even more difficult). One difficulty is that the "good" cutting-planes tend to be quite dense and of different types, and adding just a few of them can significantly change the linear program, so that starting from the previous optimal basis may not be helpful. This is an important area of work to be tackled.

3.4 Concluding Remarks

There are several strategies for dealing with the problems discussed in this chapter that appear promising, although whether they are computationally practicable is not clear.

One approach would be to selectively (and dynamically) disaggregate the problem (so that, for example, we may have several commodities corresponding to a given source node). Such a formulation could be strengthened in particular by using appropriate versions of our "flux" inequalities (in fact, our inequalities (3.67) are an attempt to do something like that). Of course, this would have to be done with care so as to avoid blowing up the formulation, and we do not have an automatic criterion for doing this.

Another approach would be to develop a branch-and-cut algorithm with "local" cuts at any node. In other words, it is usually the case for this problem that the formulation can be significantly strengthened at a branch. In particular, the flux inequalities and the variable upper-bound inequalities can be strengthened. However, we do not know how to do this automatically and further, the cuts involved would not be globally valid and very different types of cuts would be used at different nodes.

One approach that worked, but appears very difficult to implement, is that of carefully adding new variables with a combinatorial interpretation. As mentioned above, we did this for variables of the form $(x_{ij} + x_{ji} - 1)^+$ and in particular this worked well for problem **uniform2**: it narrowed the gap from 2.5 % to less than 1 %.

A final approach that does seem computationally efficient is the following. For a node i, denote $F(i) = \sum_{k,j} f_{kij}$. We know that $\sum_j t_{ij} \leq F(i)$ and if z^u is a known upper bound on the value of the problem some of our inequalities will force $F(i) \leq 2z^u$. One can proceed as follows: first partition the interval $[\sum_j t_{ij}, 2z^u]$ into a set of subintervals, each of a given size δ. Corresponding to any such subinterval we can branch, by forcing an inequality of the form $a \leq F(i) \leq a + \delta$. The rationale for this general approach is that the formulation can be tightened if we roughly know the value of $F(i)$. In particular, we may be able to reject a subinterval in one run of the cutting-plane algorithm. Further, in our experiments the interval $[\sum_j t_{ij}, 2z^u]$ was not very large, which would allow δ to be chosen rather small without generating many subintervals.

We have done some preliminary work in this area. However, we do not know how to automatically choose the quantities δ for an arbitrary problem (they should depend on the node i and should be dynamically adjusted) and also how to branch in an intelligent way.

Finally, it is clear that we must be able to solve the linear programs substantially faster.

Chapter 4

Capacitated Network Design - Polyhedral Structure and Computation

4.1 Introduction and Formulation

In this chapter we study the polyhedral structure of a mixed-integer programming formulation of the capacity expansion problem (CEP) and present computational results related with a cutting-plane algorithm which uses facet defining inequalities to strengthen the linear programming relaxation.

Given a capacitated network and point-to-point traffic demands, the objective in CEP is to add capacity to the edges, in integral multiples of various modularities (or "batches"), and route traffic, so that the overall cost is minimized. We note that CEP is strongly NP-hard [10] as it contains the fixed-charge network design problem, and thus the Steiner tree problem as a special case.

We assume that, for any fixed edge $\{i, j\}$ of the network, flows on directed edges (i, j) and (j, i) do not interfere with each other and thus we require that the total flow on (i, j) (and on (j, i)) is at most the capacity of the edge $\{i, j\}$. This constraint arises in telecommunications models because, generally, one cannot purchase "one-way" cables.

Here we study CEP when there are two batch sizes. We will assume that the larger batch size is an integral multiple of the smaller one (again a realistic assumption). By rescaling demands, we may assume that the smaller batch size is 1. We call the batch sizes unit-batches and λ-batches, where $\lambda > 1$ is the capacity of the larger batch size. Given a connected undirected graph $G = (V, E)$ with existing capacities $C_e \geq 0$ for all $e \in E$, and point-to-point traffic demand between various pairs of nodes, let $\mathcal{P}^{\mathcal{X}}$ denote the convex hull of feasible solutions to CEP. Then,

$$\mathcal{P}^{\mathcal{X}} = \text{conv}\Big\{ \quad f \in \mathcal{R}^{|K|\times 2\times|E|} \quad , \quad x,\ y \in \mathcal{Z}^{|E|} \ :$$

$$\sum_{\{i,j\}\,\in\,E} f^k_{ji} - \sum_{\{i,j\}\,\in\,E} f^k_{ij} \;=\; t_{ki} \qquad i \in V,\ k \in K \ \ i \neq k \tag{4.1}$$

$$\sum_{k\in K} f^k_{ij} \;\leq\; C_{i,j} + x_{i,j} + \lambda y_{i,j} \quad \{i,j\} \in E \tag{4.2}$$

$$\sum_{k\in K} f^k_{ji} \;\leq\; C_{i,j} + x_{i,j} + \lambda y_{i,j} \quad \{i,j\} \in E \tag{4.3}$$

$$x_{i,j},\ y_{i,j},\ f^k_{ij} \;\geq\; 0 \quad \Big\}$$

where K denotes the set of commodities related with the traffic demands, t_{ki} is the net demand of commodity k at i and f, x and y are the variable vectors related with flow, unit-batches and λ-batches, respectively. In this formulation equation (4.1) is a flow conservation equation, and equations (4.2) and (4.3) indicate that total flow on directed edge (i, j) or (j, i) can not exceed total capacity of the related edge $\{i, j\}$.

We note that the dimension of $\mathcal{P}^{\mathcal{X}}$ is equal to the number of variables minus the rank of the formulation, that is, there are no additional implied equations. Although we do not prove it explicitly, this result is implied by some of the polyhedral results presented in the following sections.

Throughout this chapter, we will use $x_{i,j}$ and $x_{j,i}$ interchangeably to denote the same variable x_e when $e = \{i, j\}$ and we will do the same for variables y and existing capacities C as well.

In the literature on multicommodity network flow problems, there are two main approaches related with the definition of the commodities. The first approach is to define a separate commodity for every non-zero point-to-point demand, resulting in $O(|V|^2)$ commodities in general. The second approach is to aggregate the demands with respect to their source (or destination) nodes and define a commodity for each node with positive supply (or demand). The aggregated formulation has $O(|V|)$ commodities.

For some problems similar to CEP (fixed charge network flow problem, for example) the "fine grain" disaggregated formulation results in a stronger LP-relaxation. The number of variables in this formulation is $O(|E||V|^2)$ as opposed to $O(|E||V|)$ of the aggregated formulation and, as noted in Chapter 3 and [5], when developing a cutting-plane algorithm, it can be prohibitively expensive to use the disaggregated formulation. Although it is possible to project the disaggregated formulation on the space of the aggregated formulation by using a family of inequalities, called "dicut collection inequalities" [28], the related separation problem appears to be very difficult. Here we adopt the second approach (aggregated version) and define a commodity for each supply node. We also note that for CEP, the LP-relaxations for both of the formulations have the same value.

Our primary motivation for studying CEP is that it naturally arises as part of a much larger and complex problem concerning ATM (asynchronous transfer mode) network design that we are separately studying. This larger problem is in fact so complex and ill-defined that a direct polyhedral study of it would be impractical and probably not advisable. However, the ATM problem contains several subproblems either identical or closely resembling CEP. These problems have fully dense traffic matrices (i.e. every node wants to talk to every other node) and this is the main reason why we are using the aggregated formulation. Our strategy to solve the ATM problem is to tighten-up formulations involving CEP, and that is our primary concern here. Thus, our computational testing will focus on how effective our inequalities are towards obtaining a strong formulation for CEP (as opposed to developing an algorithm for solving CEP).

The polyhedral structure of CEP (or, rather, some closely related variants) has already been previously studied. Magnanti and Mirchandani [22] have studied a special case of

CEP in which there is a single commodity to be routed between two special nodes of the network and there is no existing capacity on the network. In this paper, they present some facet defining inequalities and show that this special case of CEP is closely related with the shortest path problem. We will describe the results in [22] more completely later in this chapter. Another special case, which arises is the context of the lot-sizing problem with constant production capacities, has been studied by Pochet and Wolsey [27]. In this case, the network related with CEP has a special structure and there is a single batch size. In [27], Pochet and Wolsey fully describe the convex hull of a related polyhedron by using a polynomial number of facets.

Some subproblems related with CEP have also attracted attention. Magnanti, Mirchandani and Vachani [23] study the polyhedral structure of a MIP formulation of the network loading problem (NLP) with three nodes and a single batch size. In [23], Magnanti et al. present a complete characterization of the projection of the related polyhedron on the space of discrete variables.

In [26], Pochet and Wolsey study how to strengthen inequalities of the form $\sum C_j x_j \geq b$ and $\sum C_j x_j \geq y$, for $y \in R_+$ and $x_j \in Z_+^n$, essentially using the so-called MIR procedure. Inequalities of this form arise in our problem and we use some of their techniques.

Recently, Stoer and Dahl [30] studied a problem similar to ours where the flows are undirected, there are no flow costs but the capacities to be added to edges are of a more general form than those studied here. (We note that our formulation can be used to model undirected flows). One primary feature of their approach is that (in terms of our model) they would split the integral variables into sums of $0 - 1$ variables. As a result the inequalities they obtain have a rather combinatorial flavor and when the demands are small, this approach may be effective. Another feature of the approach in [30] is that they study the projection of the formulation onto the space of the x and y variables, which is possible since the problem in [30] does not have flow costs. Feasibility is achieved by means of cutting planes that are generated algorithmically. A second class of models considered in [30] can in addition handle side constraints, such as survivability constraints.

Next, we briefly introduce the notation used in this chapter. In what follows, the set of all real numbers is denoted by R, and non-negative real numbers by R^+. Similarly Z and Z^+ denote the set of integers and non-negative integers respectively. We use "\" to denote the ordinary set difference function and when it is not ambiguous, we denote $\{i\}$ by i.

For any vector v and a subset S of its indices, we define $v(S) = \sum_{i \in S} v_i$. Similarly, for a set A of directed edges and a set Q of commodities, we define $f^Q(A) = \sum_{k \in Q} \sum_{a \in A} f_a^k$.

We define $(\alpha)^+$ to be $\max\{0, \alpha\}$ and $r(\cdot,\cdot)$ to be

$$r(\alpha, \beta) = \begin{cases} \alpha - \beta(\lceil \alpha/\beta \rceil - 1) & \text{if } \alpha, \beta > 0, \\ 0 & \text{otherwise} \end{cases}$$

so that $\alpha = \beta(\lceil \alpha/\beta \rceil - 1) + r(\alpha, \beta)$ and $\beta \geq r(\alpha, \beta) > 0$ if $\alpha, \beta > 0$. We will abbreviate $r(\alpha)$ for $r(\alpha, 1)$.

Let $\delta(W) = \{e = \{i,j\} \in E : i \in W, j \notin W\}$ for $W \subset V$. Given $W \subset V$, we denote the net traffic of W by $T(W)$ where

$$T(W) = \left(\max \left\{ \sum_{i \in W} \sum_{j \in V \setminus W} t_{ij}, \sum_{i \in V \setminus W} \sum_{j \in W} t_{ij} \right\} - C(\delta(W)) \right)^+ .$$

For a feasible solution $\bar{p} = (\bar{x}, \bar{y}, \bar{f}) \in \mathcal{P}^{\mathcal{X}}$, edge $\{i,j\} \in E$ is said to be *"saturated "* if total flow on the directed edge (i,j) or (j,i) is equal to the total capacity of $\{i,j\}$, in other words if $\max\{\bar{f}_{ij}^K, \bar{f}_{ji}^K\} = \bar{x}_{i,j} + \lambda \bar{y}_{i,j} + C_{i,j}$.

4.2 Cut-set Facets

We start with a generalization of the "cut-set" inequalities studied in [22] for the single commodity problem. Given a set $S \subset V$, remember that $T(S)$ gives a lower bound on the capacity to be added across the cut separating nodes in S from the rest of the network. When the value of this lower bound (implied by flow-conservation equations and capacity constraints) is fractional, the LP-relaxation can be strengthened by forcing the added capacity across the cut to be at least $\lceil T(S) \rceil$. These valid inequalities do not define facets of the the CEP polytope unless the set S satisfies certain properties. We next state these properties.

Definition 4.2.1 *Given a connected graph $G = (V, E)$, a set S is called a* **"strong subset"** *of V with respect to G if it is a proper subset of V and both $G_S = (S, E(S))$ and $G_{\overline{S}} = (V \setminus S, E(V \setminus S))$ are connected.*

Before proceeding with the facet proof we note that, given $S \subset V$, the related cut-set inequality is dominated by other cut-set inequalities whenever G_S or $G_{\overline{S}}$ is disconnected.

Theorem 4.2.2 *Given a strong subset S of V*

$$x(\delta(S)) + \lambda y(\delta(S)) \geq \lceil T(S) \rceil \tag{4.4}$$

defines a facet of $\mathcal{P}^{\mathcal{X}}$ provided $\lceil T(S) \rceil > T(S)$ and $\lceil T(S) \rceil \geq \lambda$.

Proof. Validity of (4.4) is obvious. To simplify notation, let $E' = \delta(S)$ and $\bar{T} = \lceil T(S) \rceil$. By construction we will show that the related face $F = \{(x, y, f) \in \mathcal{P}^{\mathcal{X}} : x(E') + \lambda y(E') = \bar{T}\}$ is not empty and then by contradiction, we will show that it is a facet.

For a fixed $e_0 \in E'$ consider $\bar{p} = (\bar{x}, \bar{y}, \bar{f})$ where

$$\bar{x}_e = \begin{cases} M & e \notin E' \\ \bar{T} & e = e_0 \\ 0 & \text{otherwise} \end{cases} \qquad \bar{y}_e = \begin{cases} M & e \notin E' \\ 0 & \text{otherwise} \end{cases}$$

(M is a large enough number) and $\bar{f}$ is such that all traffic between nodes in S $(V \setminus S)$ is sent using $E(S)$ $(E(V \setminus S))$ edges and traffic crossing the cut is sent using edges with positive existing capacity and the remaining through e_0. Since both G_S and $G_{\overline{S}}$ are connected and $x(E') > T(S)$, $\bar{f}$ is feasible and thus $\bar{p} \in F$.

Notice that the edges in $E \setminus E'$ are not saturated. Therefore, without saturating them, it is possible to increase flow by a small amount for all commodities. We can do the same for e_0 as well, so without loss of generality we will assume that $\bar{f}^k_{ij}, \bar{f}^k_{ji} > 0$ for all $k \in K$

for edges with positive $\bar{x}_{i,j}$. Assume that F is not a facet of $\mathcal{P}^{\mathcal{X}}$, then there is an equation of the form

$$\alpha x + \beta y + \gamma f = \pi \tag{4.5}$$

satisfied by all points $p = (x, y, f) \in F$, where α, β and γ are vectors of appropriate dimension and π is a real number.

For all $e \notin E'$, it is possible to modify $\bar{p}$ by keeping $\bar{f}$ same and increasing $\bar{x}_e$ or $\bar{y}_e$ to obtain another point in F, which implies that $\alpha_e = \beta_e = 0$. We can also decrease $\bar{x}_{e_0}$ by λ and increase $\bar{y}_{e_0}$ by 1 to get a new point in F. Therefore $\alpha_{e_0} = (1/\lambda)\beta_{e_0}$ and since $e_0 \in E'$ is arbitrary, $\alpha_e = (1/\lambda)\beta_e$ for all $e \in E'$.

For any $k \in K$, it is possible to obtain new points in F by modifying $\bar{p}$ by simultaneously increasing $\bar{f}^k_{i,j}$ and $\bar{f}^k_{j,i}$ by a small amount for edges $\{i, j\}$ with positive $\bar{x}_{i,j}$. Since e_0 is arbitrary, we can conclude that $\gamma^k_{ij} = -\gamma^k_{ji}$ for all $\{i, j\} \in E$ and $k \in K$.

To show that $\gamma = 0$, we will first choose a spanning tree $T = (V, E'')$ of G using edges in $E \setminus E'$ and edge e_0 and then arbitrarily direct its edges to obtain the directed tree $T' = (V, A)$. If necessary by subtracting a linear combination of the flow-balance equalities (4.1) of $\mathcal{P}^{\mathcal{X}}$ from (4.5) we can assume that $\gamma^k_a = 0$ for all $k \in K$ and $a \in A$. Since $\gamma^k_{ij} = -\gamma^k_{ji}$ for any $\{i, j\} \in E$, this implies that $\gamma^k_{ij} = 0$ for $\{i, j\} \in E''$ and $k \in K$.

For $\{i, j\} \in (E \setminus E') \setminus E''$ we can find the unique cycle formed by $\{i, j\}$ and the edges in E''. Notice that e_0 will not appear on this cycle since it is the only edge crossing the cut. Since flows on the tree edges are positive in both directions for all commodities, we can send small circulation flows of each commodity on this cycle and conclude that $\gamma^k_{ij} = 0$ for $\{i, j\} \in (E \setminus E') \cup e_0$ and $k \in K$.

If $|E'| = 1$, then the proof is complete. On the other hand if $|E'| \geq 2$, then we choose an edge $\{u, v\} = e_1 \in E'$ different from e_0. Next, we modify $\bar{p}$ by increasing $\bar{x}_{e_1}$ by 1 and decreasing $\bar{x}_{e_0}$ by 1 and rerouting flow so that neither e_0 or e_1 is saturated and flows on both e_0 and e_1 are positive for all commodities. Obviously this new point is on the face. Now we find the unique cycle formed by e_1 and the edges in E'' and send circulation flows to argue that $\gamma^k_{uv} = 0$ for all $k \in K$. Since e_1 is arbitrary, we can conclude that $\gamma = 0$.

Lastly, modifying $\bar{p}$ as above also implies that if $|E'| > 1$, then there is a number $\bar{\alpha} \in R$ such that $\alpha_e = \bar{\alpha} = (1/\lambda)\beta_e$ for all $e \in E'$. Therefore, (4.5) is a multiple of (4.4) (plus a linear combination of flow-balance equations). ∎

Usually, inequalities of the form (4.4) are accompanied by other valid inequalities (obtained by means of the MIR procedure, see [24]) that exploit the following fact: If no capacity is added across a cut using unit-batches, then enough capacity should be added using an integer number of λ-batches.

Example 4.2.3 *Consider the instance of CEP with $V = \{1, 2\}$ and $E = \{1, 2\}$. Let $\lambda = 4$, $t_{12} = 7.2$, $t_{21} = 5.7$ and $C_{1,2} = 0.8$. The cut-set inequality for this case is:*

$$x_{1,2} + 4y_{1,2} \geq 7 \tag{4.6}$$

since $\lceil \max\{7.2, 5.7\} - 0.8 \rceil = 7$. Now assume that the flow costs are zero, the cost of a unit-batch is $C_1 = 1$ and the cost of a λ-batch is $C_\lambda = 3$ (so that $C_1 > C_\lambda/\lambda$). After including (4.6) to the LP-relaxation of the problem, the optimal solution has $x_{1,2} = 0$ and $y_{1,2} = 7/4$, not an integral solution. Notice that if $y_{1,2} < 2$ then $y_{1,2} \leq 1$, implying $x_{1,2} \geq 3$, and thus,

$$x_{1,2} \geq 3(2 - y_{1,2})$$

is a valid inequality which cuts off the above fractional solution from the set of feasible solutions.

We next generalize this idea and introduce a new family of cut-set facets.

Theorem 4.2.4 *Given a strong subset S of V such that $\lambda > r(\lceil T(S) \rceil, \lambda) > 0$, then*

$$x(\delta(S)) + r(\lceil T(S) \rceil, \lambda) y(\delta(S)) \geq r(\lceil T(S) \rceil, \lambda) \lceil T(S)/\lambda \rceil \tag{4.7}$$

is a facet of $\mathcal{P}^{\mathcal{X}}$ provided $\lceil T(S) \rceil > 1$ or $C(\delta(S)) > 0$ or $|\delta(S)| = 1$.

Proof. To simplify notation, let $E' = \delta(S)$, $T^+ = \lceil T(S)/\lambda \rceil$ and $r^+ = r(\bar{T}, \lambda)$. We will first rewrite (4.7) as

$$x(E') \geq r^+(T^+ - y(E')).$$

For any $p = (x, y, f) \in \mathcal{P}^{\mathcal{X}}$, if $y(E') \geq T^+$ then it is easy to see that (4.7) is valid. On the other hand if $y(E') \leq T^+ - 1$ then (4.2) and (4.3) imply that

$$\begin{aligned} x(E') &\geq \lceil T(S) \rceil - \lambda y(E') \\ &= \lambda \left\lfloor \frac{\lceil T(S) \rceil}{\lambda} \right\rfloor + r^+ - \lambda y(E') \\ &= r^+ + \lambda \left(\left\lfloor \frac{\lceil T(S) \rceil}{\lambda} \right\rfloor - y(E') \right) \\ &\geq r^+ \left(T^+ - y(E') \right). \end{aligned}$$

We will first construct a point in $F = \{(x, y, f) \in \mathcal{P}^{\mathcal{X}} : x(E') = r^+(T^+ - y(E'))\}$ and then we will show that it is a facet.

If $C(E') > 0$, then let $e_0 \in E'$, be an edge such that $C_{e_0} > 0$, otherwise choose an arbitrary edge $e_0 \in E'$ and consider $\bar{p} = (\bar{x}, \bar{y}, \bar{f})$ where

$$\bar{x}_e = \begin{cases} M & e \notin E' \\ 0 & \text{otherwise} \end{cases} \qquad \bar{y}_e = \begin{cases} M & e \notin E' \\ T^+ & e = e_0 \\ 0 & \text{otherwise} \end{cases}$$

and $\bar{f}$ is such that $\bar{f}^k_{ij}, \bar{f}^k_{ji} > 0$ for all $k \in K$ for edges with positive $\bar{y}_{ij}$, and e_0 is not saturated. Obviously $\bar{p} \in F$.

Assume that F is not a facet of $\mathcal{P}^{\mathcal{X}}$, and let

$$\alpha x + \beta y + \gamma f = \pi \tag{4.8}$$

be an equation different from (4.7) satisfied by all points $p = (x, y, f) \in F$.

Using similar arguments as in the proof of Theorem 4.2.2, it is possible to choose a spanning tree $T = (V, E'')$ of G and show that $\alpha_e = \beta_e = 0$ for all $e \notin E'$ and $\alpha_e = (1/r^+)\beta_e$ for all $e \in E'$. Furthermore, we can also show that $\gamma_{ij}^k = 0$ for all $\{i, j\} \in (E \setminus E') \cup \{e_0\}$ and $k \in K$.

If $|E'| \geq 2$, then choose an edge $\{u, v\} = e_1 \in E'$ different from e_0 and consider the point $p' = (x', y', f') \in F$ where

$$x'_e = \begin{cases} M & e \notin E' \\ r^+ - 1 & e = e_0 \\ 1 & e = e_1 \\ 0 & \text{otherwise} \end{cases} \qquad y'_e = \begin{cases} M & e \notin E' \\ T^+ - 1 & e = e_0 \\ 0 & e = e_1 \\ 0 & \text{otherwise} \end{cases}$$

and f' is such that neither e_0 nor e_1 is saturated and flow for all commodities is positive on both directions for e_0 and e_1. Notice that the conditions of the theorem and the choice of e_0 imply that $x'_{e_0} + \lambda y'_{e_0} + C'_{e_0} > 0$.

Now using this point we can find a cycle containing e_0 and e_1 and some of the edges in $E \setminus E'$ and send circulation flows to argue that $\gamma_{uv}^k = 0$ for all $k \in K$. Since e_1 is arbitrary, we conclude that $\gamma = 0$.

Lastly, $p' \in F$ also implies that for some $\bar{\alpha} \in R$, $\alpha_e = \bar{\alpha} = (1/r^+)\beta_e$ for all $e \in E'$. Therefore (4.8) is a multiple of (4.7) (plus a linear combination of flow-balance equalities) and (4.7) defines a facet of $\mathcal{P}^{\mathcal{X}}$. ■

Notice that, given a strong subset S of V, if $\lceil T(S) \rceil = 1$, $C(\delta(S)) = 0$ and $|\delta(S)| > 1$, then all of the points on the face defined by (4.7) satisfy the family of equations,

$$f_{ij}^k - f_{ji}^k = (x_{i,j} + y_{i,j}) \sum_{v \in V \setminus S} t_{kv}$$

for $i, k \in S$, $\{i, j\} \in \delta(S)$, and thus (4.7) is not facet defining. In the next section we introduce some facets of the CEP polytope that include the flow variables as well as the capacity variables, and these facets can be considered as generalizations of cut-set facets.

The model studied in [22] differs from ours primarily in that there is a single commodity (i.e. a single origin-destination node pair for which there is positive demand) and there are *three* types of capacity batches that one can add to any edge. In [22] it is stated that the above cut-set inequalities are facet-defining, as well as a third type of cut-set inequality, which arises by applying the MIR procedure one additional time (to handle the third type of capacity variable). It is shown therein that if there are no flow costs, then under reasonable assumptions on the cost coefficients the linear program containing all cut-set inequalities has some optimal solution that is integral; and they present an efficient algorithm for computing that solution which uses the optimal dual variables.

We note that for the multicommodity case, the cut-set inequalities typically reduce the integrality gap to 30% and they are also helpful in terms of pinpointing "interesting" subset of vertices. Below we consider stronger inequalities which include the cut-set facets as a special case.

4.3 Flow-cut-set Facets

In this section we generalize the cut-set facets to include the flow variables as well. Consider a subset S of V and the cut-set facets (4.4) and (4.7) related with it. After including these facets in the LP-relaxation of CEP, there exists feasible points to the extended formulation which assign an integer amount of total capacity across the cut $\delta(S)$ but allocate this capacity fractionally among the edges in the cut. The flow-cut-set facets exclude some of these points from the feasible region.

Given a subset S of V and a non-empty partition $\{E_1, E_2\}$ of $\delta(S)$, we will denote the edges in E_i directed away from S by A_i (i.e. $A_i = \{(u,v) : u \in S, v \notin S, \{u,v\} \in E_i\}$), and similarly by $\bar{A}_i$ we will denote the edges in E_i directed to S.

Consider a simple instance of CEP where there is a single commodity to be routed from S to $\bar{S}$. Furthermore, assume that the cost of routing flow through A_1 is smaller than that of A_2 but cost of adding capacity on E_1 is bigger. In this case, solutions to the LP-relaxation will assign just enough (fractional) capacity to the E_2 edges, but send all the flow using A_2. When combined with cut-set facets, the flow-cut-set facets force the capacity added to E_2 to be integral. These facets have the following common structure,

$$bx(E_2) \ + \ cy(E_2) \ + \ f^Q(A_1) \ \geq \ d \tag{4.9}$$

where Q is a subset of S and $b, c, d \in R$.

Before proceeding any further, we first prove the following technical lemma, which will help us keep the facet proofs less lengthy. In Lemma 4.3.2 we consider a facet of the form (4.9) and investigate some properties of the equations which are satisfied by all points of this facet.

Definition 4.3.1 *Given two sets S and Q such that $Q \subseteq S \subseteq V$ we define $t(W, V \setminus S) = \sum_{i \in W} \sum_{j \notin S} t_{ij}$, and we call Q a* **"commodity subset "** *of S if $t(q, V \setminus S) > 0$ for all $q \in Q$.*

Lemma 4.3.2 *Given a strong subset S of V, a commodity subset Q of S, a nonempty partition $\{E_1, E_2\}$ of $\delta(S)$ and a face*

$$F = \left\{ (x, y, f) \in \mathcal{P}^{\mathcal{X}} : b \sum_{e \in E_2} x_e + c \sum_{e \in E_2} y_e + \sum_{a \in A_1} \sum_{k \in Q} f_a^k = d \right\}$$

of $\mathcal{P}^{\mathcal{X}}$ where $b, c, d \in R$, assume that the equation $\alpha x + \beta y + \gamma f = \pi$ is satisfied by all points in F. Then, without loss of generality ,

(i) *If F is proper (i.e. $F \neq \emptyset$), then $\alpha_e = \beta_e = 0$ for all $e \in E \setminus E_2$.*

(ii) *If there exists $\bar{p} = (\bar{x}, \bar{y}, \bar{f}) \in F$ such that $\bar{x}(E_2) + \lambda \bar{y}(E_2) + C(E_2) > \bar{f}^K(A_2)$, then $\gamma_a^k = 0$ for all $k \in K, a \notin A_1$ and $k \notin Q, a \in A_1$.*

(*iii*) *If* $\gamma_a^k = 0$ *for* $k \in K, a \notin A_1$ *and there is a point* $\hat{p} = (\hat{x}, \hat{y}, \hat{f}) \in F$ *satisfying* $\hat{x}(E_2) > 0$, *then there exists* $\bar{\alpha} \in R$ *such that* $\alpha_e = \bar{\alpha}$ *for all* $e \in E_2$, *and similarly if* $\hat{y}(E_2) > 0$, *then there exists* $\bar{\beta} \in R$ *such that* $\beta_e = \bar{\beta}$ *for all* $e \in E_2$.

(*iv*) *If* $\gamma_a^k = 0$ *for* $k \in K, a \notin A_1$ *and there is a point* $\tilde{p} = (\tilde{x}, \tilde{y}, \tilde{f}) \in F$ *such that* $\tilde{f}^Q(A_1) > 0$, *then for all* $k \in Q$ *there exists* $\bar{\gamma}^k \in R$ *such that* $\gamma_a^k = \bar{\gamma}^k$ *for all* $a \in A_1$. *Furthermore, if* $\tilde{f}^Q(A_2) > 0$ *as well, then, there exists* $\bar{\gamma} \in R$ *such that* $\bar{\gamma}_a^k = \bar{\gamma}$ *for all* $k \in K, a \in A_1$

Proof.

(i) Given $p = (x, y, f) \in F$ choose a fixed edge $e \in E \setminus E_2$, and let $p' = (x', y', f')$ be identical to p with the exception that $x'_e = \bar{x}_e + 1$. Then $p' \in F$ and consequently $\alpha_e = 0$. Similarly $\beta_e = 0$.

(ii) Given $\bar{p} = (\bar{x}, \bar{y}, \bar{f}) \in F$ satisfying $\bar{x}(E_2) + \lambda\bar{y}(E_2) + C(E_2) > \bar{f}^K(A_2)$, we can assume that $\bar{x}(E_2) + \lambda\bar{y}(E_2) + C(E_2) > \bar{f}^K(\bar{A}_2)$ as well, since it is possible to route some of the flow on $\bar{A}_2$ using edges in $E \setminus E_2$ (after increasing the capacities, if necessary).

Let $e_0 \in E_2$ be such that $C(e_0) = \max_{e \in E_2}\{C(e)\}$ and modify $\bar{p}$ to obtain $p' = (x', y', f') \in F$, where

$$x'_e = \begin{cases} M & e \in E \setminus E_2 \\ \bar{x}(E_2) & e = e_0 \\ 0 & \text{otherwise} \end{cases} \qquad y'_e = \begin{cases} M & e \in E \setminus E_2 \\ \bar{y}(E_2) & e = e_0 \\ 0 & \text{otherwise} \end{cases}$$

and f' is obtained from $\bar{f}$ by rerouting any flow in excess of existing capacities on edges $E_2 \setminus e_0$, to e_0, using the edges in $E \setminus \delta(S)$. Notice that e_0 is not saturated.

Let $T = (V, E'')$ be a spanning tree of V such that $E'' \subset (E \setminus \delta(S)) \cup e_0$. As in the proof of Theorem 4.2.2, we can first argue that $\gamma_{ij}^k = 0$ for all $\{i, j\} \in E''$, $k \in K$ and then by using circulation flows, show that $\gamma_{ij}^k = 0$ for all $k \in K, a \notin A_1$ and $k \notin Q, a \in A_1$.

(iii) Whenever $\hat{x}(E_2) > 0$ and $|E_2| > 1$, it is possible to choose e_0 with $x_{e_0} > 0$ and $e_1 \in E_2 \setminus e_0$ and construct a new point similar to $\hat{p}$ by increasing $\hat{x}_{e_1}$ by $\hat{x}_{e_0}$ and decreasing $\hat{x}_{e_0}$ to zero and then rerouting flow. Therefore, $\alpha_e = \bar{\alpha}$ for $e \in E_2$. Similarly, $\beta_e = \bar{\beta}$ for $e \in E_2$ if $\hat{y}(E_2) > 0$.

(iv) Without loss of generality we can assume that $\tilde{f}$ does not saturate edges in $E \setminus E_2$. For an arbitrary $q \in Q$, if $\tilde{f}^q(A_1) > 0$, then let $a \in A_1$ be a directed edge with $\tilde{f}^q_a > 0$. Whenever $|E_1| > 1$, we can choose $a' \in A_1 \setminus a$ and construct $p'' = (\tilde{x}, \tilde{y}, f'')$ where f'' is obtained from $\tilde{f}$ by routing some flow of commodity q to go through a' instead of a. Therefore, $\gamma^q_a = \bar{\gamma}^q$ for all $a \in A_1$.

On the other hand, if $\tilde{f}^q(A_1) = 0$ then $|Q| > 1$ and $\tilde{f}^q_a > 0$ for some $a \in A_2$. In this case, it is possible to find $q' \in Q \setminus q$ such that $\tilde{f}^{q'}_{a'} > 0$ for some $a' \in A_1$, and construct $p'' = (\tilde{x}, \tilde{y}, f'') \in F$ where f'' is obtained from $\tilde{f}$ by rerouting flow to decrease $\tilde{f}^q_a$ and $\tilde{f}^{q'}_{a'}$ by $\min\{\tilde{f}^q_a, \tilde{f}^{q'}_{a'}\}$ and increase $\tilde{f}^{q'}_a$ and $\tilde{f}^q_{a'}$ by the same amount. This new point has $f''^q(A_1) > 0$, and thus $\gamma^q_a = \bar{\gamma}^q$ for all $a \in A_1$.

Therefore, if $\tilde{f}^Q(A_1) > 0$ then for all $q \in Q$ and $a \in A_1$, $\gamma^q_a = \bar{\gamma}^q$. Furthermore if $\tilde{f}^Q(A_2) > 0$ as well, then the above argument also implies that $\bar{\gamma}^q = \bar{\gamma}$. ∎

All of the facet defining inequalities presented in this section exploit the following basic idea. Consider the polyhedron

$$P = \text{conv}\left\{x \in Z^+, f \in R^+ : f + ax \geq b\right\}$$

when $a > r(b, a) > 0$ (i.e. $a, b > 0$ and b is not an integer multiple of a), and let CP denote its continuous relaxation. As described in [24], it is easy to observe that all of the points in $CP \setminus P$ violate the inequality $f \geq r(b, a)(\lceil b/a \rceil - x)$ and consequently P can also be expressed as,

$$P = \left\{x, f \in R^+ : f + ax \geq b,\ f \geq r(b, a)(\lceil b/a \rceil - x)\right\}.$$

Also notice that, for an arbitrary polyhedron, if $f + ax \geq b$ is a valid inequality for $x \in Z^+$

and $f \in R^+$ then

$$f \geq r(b,a)(\lceil b/a \rceil - x) \tag{4.10}$$

is a valid (MIR) inequality.

Given two sets S and Q such that $Q \subseteq S \subseteq V$, it is easy to see that the total flow of commodities in Q leaving S should be sufficient to satisfy the total demand in $V \setminus S$. Let $\{E_1, E_2\}$ be a partition of $\delta(S)$, and remember that A_i denotes the edges in E_i oriented from S to $V \setminus S$. Then, we can write

$$f^Q(A_1) + f^Q(A_2) \geq t(Q, V \backslash S)$$

implying

$$f^Q(A_1) + x(E_2) + \lambda y(E_2) + C(E_2) \geq t(Q, V \backslash S)$$

and

$$f^Q(A_1) + x(E_2) + \lambda y(E_2) \geq t(Q, V \backslash S) - C(E_2). \tag{4.11}$$

We now write an inequality of the form (4.10) using the fact that $f^Q(A_1) \in R^+$ and $x(E_2) + \lambda y(E_2) \in Z^+$. For a given subset Q of S, the following theorem develops a lower bound on $f^Q(A_1)$ when $x(E_2) + \lambda y(E_2)$ is less than the minimum integral capacity that can carry the total demand of Q in $V \setminus S$. We also note that (4.12) of Theorem 4.3.3 becomes the cut-set inequality (4.4) when $E_1 = \emptyset$.

Theorem 4.3.3 *Given a strong subset S of V, a commodity subset Q of S and a nonempty partition $\{E_1, E_2\}$ of $\delta(S)$, let $T' = t(Q, V \backslash S) - C(E_2)$, $r' = r(T')$ and $\bar{T} = \lceil T' \rceil$.*
(i) If $1 > r' > 0$ then

$$f^Q(A_1) \geq r' \left(\bar{T} - x(E_2) - \min\{\lambda, \bar{T}\} y(E_2)\right) \tag{4.12}$$

is a facet of $\mathcal{P}^{\mathcal{X}}$ provided $T' > 1$ or $C(E_2) > 0$.

(ii) If $1 > T' > 0$ *and* $C(E_2) = 0$ *then*

$$f^Q(A_1) \geq T'(1 - x(E_2) - y(E_2)) \tag{4.13}$$

is a facet of $\mathcal{P}^{\mathcal{X}}$ *provided* $|Q| = 1$.

Proof. Since (ii) is a special case of (i), we will only show the validity of (i). When $\bar{T} \geq \lambda$, validity of (i) is same as (4.10). If $\lambda > \bar{T}$, then it is easy to see that (4.12) is valid when $y(E_2) > 0$, and if $y(E_2) = 0$ then the above derivation is still valid.

(i) Choose a fixed edge $e_0 \in E_2$ and consider $p^1 = (x^1, y^1, f^1) \in F$, where

$$x_e^1 = \begin{cases} M & e \in E \setminus E_2 \\ \bar{T} & e = e_0 \\ 0 & \text{otherwise} \end{cases} \qquad y_e^1 = \begin{cases} M & e \in E \setminus E_2 \\ 0 & \text{otherwise} \end{cases}$$

and f^1 is a feasible flow vector satisfying $(f^1)^Q(A_2) = t(Q, V \setminus S)$, $(f^1)^Q(A_1) = 0$ and $(f^1)^{K\setminus Q}(A_2) = 0$. Notice that since S is a strong subset and $x^1(E_2) + y^1(E_2) > T'$, it is possible to find a flow vector f^1. Next consider $p^2 = (x^2, y^2, f^1)$ which is identical to p^1 with the exception that $y_{e_0}^2 = 1$ and $x_{e_0}^2 = x_{e_0}^1 - \min\{\lambda, \bar{T}\}$. Clearly $p^2 \in F$.

If F is not a facet, then there is an equation $\alpha x + \beta y + \gamma f = \pi$ different from (4.12) satisfied by all points $(x, y, f) \in F$. Applying Lemma 4.3.2 with p^1 and p^2, we can argue that $\gamma_a^k = 0$ unless $q \in Q$ and $a \in A_1$; $\alpha_e = \beta_e = 0$ for $e \in E \setminus E_2$ and there exist $\bar{\alpha}, \bar{\beta} \in R$ such that, $\alpha_e = \bar{\alpha}$, $\beta_e = \bar{\beta}$ for all $e \in E_2$. Furthermore, $p^1, p^2 \in F$ also imply that $\bar{\beta} = \min\{\lambda, \bar{T}\}\bar{\alpha}$.

Next consider $p^3 = (x^3, y^1, f^3) \in F$ where x^3 is identical to x^1 with the exception that $x_{e_0}^3 = \bar{T} - 1$, and f^3 is a feasible flow vector satisfying $(f^3)^Q(A_2) = x^3(E_2) + y^3(E_2) + C(E_2)$ and $(f^3)^Q(A_1) = r' > 0$. Notice that that $(f^3)^Q(A_2) = t(Q, V \setminus S) - r' > 0$ and using Lemma 4.3.2 we can conclude that $\gamma_a^k = \bar{\gamma}$ for $k \in Q, a \in A_1$, furthermore, $p^1, p^3 \in F$ also imply that $\bar{\alpha} = r'\bar{\gamma}$.

Finally $p^1 \in F$ implies that $\pi = \bar{\alpha}\bar{T}$, and consequently, $\alpha x + \beta y + \gamma f = \pi$ is a multiple of (4.13) plus a linear combination of flow-balance equalities.

(ii) The proof is identical to the first part with the only difference being $(f^3)^Q(A_2) = 0$. But since $|Q| = 1$, Lemma 4.3.2 still implies that $\gamma_a^k = \bar{\gamma}$ for $k \in Q, a \in A_1$. ∎

Example 4.3.4 *Consider the instance of CEP with* $|S| = |\bar{S}| = 1$ $E_1 = e_1$, $E_2 = e_2$ *and assume that* $T' = 6.8$ *and* $\lambda = 4$. *A possible solution to this instance (that is, with appropriate cost coefficients) has* $y(E_2) = 1.7$, $x(E_2) = 0$ *and* $f(A_1) = 0$ *and this fractional solution is cut-off by the flow-cut-set inequality*

$$f(A_1) \geq 0.8\,(7 - 4y(E_2) - x(E_2)) \tag{4.14}$$

since the right hand side is 0.16. *After including (4.14) in the formulation the new solution has* $y(E_2) = 1.75$, $x(E_2) = 0$ *and* $f(A_1) = 0$.

As this example demonstrates, (4.12) and (4.13) are not sufficient to force $y(E_2)$ to be integral when both $x(E_2)$ and $f(A_1)$ are zero. Next we write another inequality of the form (4.10) which implies that if $f^Q(A_1) = x(E_2) = 0$ then $y(E_2)$ can not be less than the minimum integral capacity that can carry $t(Q, V\backslash S) - C(E_2)$.

Theorem 4.3.5 *Given a strong subset* S *of* V, *a commodity subset* Q *of* S *and a nonempty partition* $\{E_1, E_2\}$ *of* $\delta(S)$, *let* $T' = t(Q, V\backslash S) - C(E_2)$, $T^+ = \lceil T'/\lambda \rceil$ *and* $\bar{r} = r(T', \lambda)$. *Then,*

$$f^Q(A_1) + \min\{1, \bar{r}\}x(E_2) \geq \bar{r}\,(T^+ - y(E_2)) \tag{4.15}$$

is a facet of $\mathcal{P}^{\mathcal{X}}$ *provided* $T' > 1$ *and* $\lambda > \bar{r}$.

Proof. To show that (4.15) is a valid for $\mathcal{P}^{\mathcal{X}}$ we first note that it is implied by non-negativity constraints whenever $y(E_2) \geq T^+$ or $\min\{1, \bar{r}\}x(E_2) \geq \bar{r}\,(T^+ - y(E_2))$. So we will concentrate on the case when $y(E_2) \leq T^+ - 1$ and $\min\{1, \bar{r}\}x(E_2) < \bar{r}\,(T^+ - y(E_2))$, and rewrite the lower bound on the total flow of Q-commodities on A_1 edges,

$$f^Q(A_1) \;\; \geq \;\; T' - x(E_2) - \lambda y(E_2)$$

$$= \lambda\,(T^+ - 1) + \bar{r} - x(E_2) - \lambda y(E_2)$$
$$= \lambda\,(T^+ - 1 - y(E_2)) + \bar{r} - x(E_2). \qquad (4.16)$$

Next we consider two cases. When $\bar{r} > 1$ then using $y(E_2) \leq T^+ - 1$ and $\lambda \geq \bar{r}$, (4.16) can be modified as

$$\begin{aligned} f^Q(A_1) &\geq \bar{r}(T^+ - 1 - y(E_2)) + \bar{r} - x(E_2) \\ &= \bar{r}\,(T^+ - y(E_2)) - x(E_2). \end{aligned}$$

On the other hand, if $\bar{r} < 1$, then using $\lambda > 1$ and $x(E_2) < (T^+ - y(E_2))$, we can write

$$\begin{aligned} f^Q(A_1) &\geq \lambda\,(T^+ - 1 - y(E_2) - x(E_2)) + \bar{r} \\ &\geq \bar{r}\,(T^+ - y(E_2) - x(E_2))\,. \end{aligned}$$

and conclude that (4.15) is a valid inequality.

To show that (4.15) is a facet we will construct several points on the related face. Let $e_0 \in E_2$ and consider $p^1 = (x^1, y^1, f^1) \in F$, where

$$x_e^1 = \begin{cases} M & e \in E \setminus E_2 \\ 0 & \text{otherwise} \end{cases} \qquad y_e^1 = \begin{cases} M & e \in E \setminus E_2 \\ T^+ & e = e_0 \\ 0 & \text{otherwise} \end{cases}$$

and f^1 is a feasible flow vector such that it does not saturate e_0 and $(f^1)^Q(A_1) = 0$. Next we construct $p^2 = (x^1, y^2, f^2) \in F$ where y^2 is same as y^1 except $y_{e_0}^2 = T^+ - 1$ and f^2 is a feasible flow vector saturating all the edges in E_2 and satisfying $(f^2)^Q(A_1) = \bar{r}$. Lastly we construct $p^3 = (x^3, y^2, f^3) \in F$ where x_3 is same as x^1 except $x_{e_0}^2 = 1$, and f^3 saturates all the edges in E_2 and satisfies $(f^3)^Q(A_1) = \bar{r} - \min\{1, \bar{r}\}$.

Assume that (4.15) is not a facet and let $\alpha x + \beta y + \gamma f = \pi$ be an equation different from (4.12) satisfied by all points $(x, y, f) \in F$. Notice that if $t(Q, V\backslash S) > \bar{r}$ (i.e. when

$C(E_2) > 0$ or $t(Q, V \backslash S) > \lambda$), then $(f^2)^Q(A_2) > 0$ and if $t(Q, V\backslash S) = \bar{r}$, then $\bar{r} > 1$ and $(f^3)^Q(A_2) > 0$. Therefore, applying Lemma 4.3.2 with p^1, p^2, and p^3 we can show that there exist $\bar{\alpha}, \bar{\beta}, \bar{\gamma} \in R$ satisfying;

$$\alpha_e = \begin{cases} \bar{\alpha} & e \in E_2 \\ 0 & \text{otherwise} \end{cases} \qquad \beta_e = \begin{cases} \bar{\beta} & e \in E_2 \\ 0 & \text{otherwise} \end{cases} \qquad \gamma_a^k = \begin{cases} \bar{\gamma} & a \in A_1, k \in Q \\ 0 & \text{otherwise.} \end{cases}$$

Furthermore, $p^1, p^2, p^3 \in F$ also imply that, $\bar{\gamma} = \bar{\beta}/\bar{r}$, $\bar{\alpha} = \min\{1, \bar{r}\}\bar{\gamma}$, and $\pi = \bar{\beta}T^+$. ■

Example 4.3.4 (continued) *Recall that, after including (4.14) in the formulation, the solution had* $y(E_2) = 1.75$, $x(E_2) = 0$ *and* $f(A_1) = 0$. *As* $T' = 6.8$ *and* $\lambda = 4$, *this solution does not satisfy (4.15) since the right hind side of*

$$f(A_1) + x(E_2) \geq 2.8\,(2 - y(E_2)) \tag{4.17}$$

is positive. After including (4.17) in the formulation, the new solution is $y(E_2) = 1.1\bar{6}$, $x(E_2) = 2.\bar{3}$ *and* $f(A_1) = 0$, *still not an integral solution.*

The last flow-cut-set facet (4.18) can be considered as an extension of (4.10) to three variables, and it states that when $y(E_2)$ is not sufficient to carry all the flow, and $x(E_2)$ is not big enough to carry the remainder, then $f(A_1)$ can not be zero. We also note that (4.18) of Theorem 4.3.6 becomes the cut-set inequality (4.7) when $E_1 = \emptyset$.

Theorem 4.3.6 *Given a strong subset* S *of* V, *a commodity subset* Q *of* S *and a nonempty partition* $\{E_1, E_2\}$ *of* $\delta(S)$, *let* $T' = t(Q, V\backslash S) - C(E_2)$, $r' = r(T')$, $T^+ = \lceil T'/\lambda \rceil$ *and* $r^+ = r(\lceil T' \rceil, \lambda)$. *Then,*

$$f^Q(A_1) \geq r'\left(r^+\left(T^+ - y(E_2)\right) - x(E_2)\right) \tag{4.18}$$

is a facet of $\mathcal{P}^{\mathcal{X}}$ *provided* $T' > 1$ *and* $1 > r'$.

Proof. We first show (4.18) is valid. This can be shown by applying the MIR procedure twice, but we will present a direct proof. For any $p = (x, y, f) \in \mathcal{P}^{\mathcal{X}}$, (4.18) is valid whenever $y(E_2) \geq T^+$ or $x(E_2) \geq r^+ (T^+ - y(E_2))$. Now consider the case when $y(E_2) \leq T^+ - 1$ and $x(E_2) \leq r^+ (T^+ - y(E_2)) - 1$. We know that

$$\begin{aligned} f^Q(A_1) &\geq T' - x(E_2) - \lambda y(E_2) \\ &= \lambda (T^+ - 1) + (r^+ - 1) + r' - x(E_2) - \lambda y(E_2) \\ &= \lambda (T^+ - 1 - y(E_2)) + r^+ - 1 - x(E_2) + r'. \end{aligned}$$

Using $\lambda \geq r^+$ and $1 \geq r'$ we can write

$$\begin{aligned} f^Q(A_1) &\geq r^+ (T^+ - 1 - y(E_2)) + r^+ - 1 - x(E_2) + r' \\ &= r^+ (T^+ - y(E_2)) - 1 - x(E_2) + r' \\ &\geq r' (r^+ (T^+ - y(E_2)) - 1 - x(E_2)) + r' \\ &= r' (r^+ (T^+ - y(E_2)) - x(E_2)). \end{aligned}$$

Therefore, (4.18) is a valid inequality for $\mathcal{P}^{\mathcal{X}}$. To see that it is a facet, we will construct several points on the related face. Let $e_0 \in E_2$ and consider $p^1 = (x^1, y^1, f^1) \in F$, where

$$x^1_e = \begin{cases} M & e \in E \setminus E_2 \\ r^+ & e = e_0 \\ 0 & \text{otherwise} \end{cases} \qquad y^1_e = \begin{cases} M & e \in E \setminus E_2 \\ T^+ - 1 & e = e_0 \\ 0 & \text{otherwise} \end{cases}$$

and f^1 is a feasible flow vector such that it does not saturate e_0 and satisfies $(f^1)^Q(A_1) = 0$. Next we construct $p^2 = (x^2, y^2, f^1) \in F$ where x^2 and y^2 are same as x^1 and y^1 except $x^2_{e_0} = 0$ and $y^2_{e_0} = T^+$. Lastly we construct $p^3 = (x^3, y^1, f^3) \in F$ where x_3 is same as x^1 except $x^2_{e_0} = r^+ - 1$, and f^3 saturates all the edges in E_2 and satisfies $(f^3)^Q(A_1) = r'$.

Using similar arguments as in the proof of Theorem 4.3.5, we can use points p^1, p^2 and p^3 to show that (4.18) is a facet of $\mathcal{P}^{\mathcal{X}}$. ∎

Example 4.3.4 (continued) *When applied to the given instance, (4.18) becomes*

$$f(A_1) \geq 0.8\,(3\,(2 - y(E_2)) - x(E_2)) \tag{4.19}$$

and including (4.19) in the formulation finally results in the integral solution with $y(E_2) = 1$, $x(E_2) = 2$ *and* $f(A_1) = 0.8$.

4.4 Three-partition Facets

When deriving the cut-set or the flow-cut-set facets, the main idea is to find an edge-cut dividing the network into two connected components, and develop lower bounds on the variables related with the edges appearing on this cut. A natural extension of this approach is to consider a multi-cut, partitioning the network into three components, and study the facets related with this multicut.

Let $\Delta \subset E$ be such a multicut and $\{S_1, S_2, S_3\}$ be the related partition of the node set. If each S_i is strong, then it is possible to develop a lower bound on the capacity to be added across this multicut as follows. First we add up the cut-set inequalities (4.4) related with each S_i and then divide both sides of the resulting inequality by two to get the valid (implied) inequality $x(\Delta) + \lambda y(\Delta) \geq (\lceil T(S_1) \rceil + \lceil T(S_2) \rceil + \lceil T(S_3) \rceil)/2$. Notice that if the right hand side is fractional (i.e. $\sum_i \lceil T(S_i) \rceil$ is odd), then it is possible to strengthen the inequality by replacing the right hand side by its ceiling to obtain,

$$x(\Delta) + \lambda y(\Delta) \geq \left\lceil \frac{\lceil T(S_1) \rceil + \lceil T(S_2) \rceil + \lceil T(S_3) \rceil}{2} \right\rceil . \tag{4.20}$$

Although one would expect the strengthened inequality to be a facet of the CEP polytope (similar inequalities are facet defining for NLP, see [23]), in some cases it does not even define a supporting hyperplane. The following example demonstrates one such case.

Example 4.4.1 *Consider the single batch-size version of CEP when* $|V| = 3$, G *is the complete graph* K_3 *and there is no existing capacity. Let* $t_{12} = t_{13} = t_{23} = 5$ *and* $t_{21} =$

$t_{31} = t_{32} = 0$. *For this case (4.20) becomes,* $x_{12} + x_{13} + x_{23} \geq 13$ *as* $T(1) = T(3) = 10$ *and* $T(2) = 5$.

Notice that before reaching its destination, each unit of t_{12}, t_{13} *or* t_{23} *has to go through the directed edges* $(1,2), (1,3)$ *or* $(2,3)$ *at least once. This observation implies that*

$$x_{12} + x_{13} + x_{23} \geq t_{12} + t_{13} + t_{23} = 15$$

is a valid inequality, dominating (4.20).

4.4.1 A Three Node Problem

Next, we study the polyhedral structure of this simplified version of CEP (i.e. when there is a single batch-size and $G = K_3 = (V_3, E_3)$). We denote the integral polyhedron related with this problem by $\mathcal{P}^{\mathcal{X}3}$and its continuous relaxation by $\mathcal{CP}^{\mathcal{X}3}$. For $\mathcal{CP}^{\mathcal{X}3}$, Lemma 4.4.2 establishes the necessary conditions on the capacity variables x, under which one can find a feasible flow vector.

Lemma 4.4.2 *Given* $\bar{x} \in \mathcal{R}^3$*, there exists a flow vector* $\bar{f}$ *such that* $(\bar{x}, \bar{f}) \in \mathcal{CP}^{\mathcal{X}3}$ *if and only if*

$$
\begin{array}{llcll}
(i) & \bar{x}(i) & \geq & T(i) & \text{for all } i \in V_3, \\
(ii) & \bar{x}(E_3) + C(E_3) & \geq & t_{ij} + t_{ik} + t_{kj} & \text{for all permutations } \pi = (i,j,k) \text{ of } V_3, \text{ and} \\
(iii) & \bar{x}_{i,j} & \geq & 0 & \text{for all } \{i,j\} \in E_3.
\end{array}
$$

Proof. The necessity of (i) - (iii) is obvious. To show that they are sufficient, we construct a feasible flow vector $\bar{f}$ which satisfies the following two conditions for every ordered pair of nodes (i,j):

- If $\bar{x}_{i,j} + C_{i,j} \geq t_{ij}$ then t_{ij} is sent directly from node i to node j.
- If $\bar{x}_{i,j} + C_{i,j} < t_{ij}$ then $\bar{x}_{i,j} + C_{i,j}$ flow is routed on (i,j) and $t_{ij} - \bar{x}_{i,j} - C_{i,j}$ via k.

It is easy to check that $\bar{f}$ satisfies the flow-balance equalities, and for all $i \neq j$

$$\sum_v \bar{f}_{ij}^v = \min\{t_{ij}, \bar{x}_{i,j} + C_{i,j}\} + (t_{ik} - \bar{x}_{i,k} - C_{i,k})^+ + (t_{kj} - \bar{x}_{k,j} - C_{k,j})^+.$$

To show that $\bar{f}$ also satisfies the capacity constraints, we consider the following two cases.

For any ordered pair (i,j), if $t_{ij} \geq \bar{x}_{i,j} + C_{i,j}$ then both $(t_{ik} - \bar{x}_{i,k} - C_{i,k})^+$ and $(t_{kj} - \bar{x}_{k,j} - C_{k,j})^+$ are zero due to (i) applied to node i and node j, respectively, and thus $\sum_v f_{ij}^v = \bar{x}_{i,j} + C_{i,j}$.

On the other hand, when $t_{ij} < \bar{x}_{i,j} + C_{i,j}$, then the total flow on (i,j) equals $t_{ij} + (t_{ik} - \bar{x}_{i,k} - C_{i,k})^+ + (t_{kj} - \bar{x}_{k,j} - C_{k,j})^+$. When either the second or the third term is zero, this is at most $\bar{x}_{i,j} + C_{i,j}$ by (i) applied to i or j, respectively. When they are both positive, this is also at most $\bar{x}_{i,j} + C_{i,j}$ by (ii). ∎

In other words, Lemma 4.4.2 states that $\mathcal{CP}^{\mathcal{X}3}$can be projected on the space of x variables by using (i) - (iii). We note that (i) - (ii) of Lemma 4.4.2 belong to a family of inequalities called *"metric inequalities"* (see [15], for example). It is known that these inequalities are sufficient to project the continuous relaxation of a capacitated multicommodity flow polyhedron on the space of the discrete variables. In Lemma 4.4.2 we identify the important metric inequalities for $\mathcal{CP}^{\mathcal{X}3}$. Also notice that if we define

$$\theta = \max_{\pi=(i,j,k)} \{t_{ij} + t_{ik} + t_{kj}\}$$

then (ii) can be replaced by a single inequality $\bar{x}(E_3) + C(E_3) \geq \theta$.

Corollary 4.4.3 *Given $\bar{x} \in Proj_x(\mathcal{CP}^{\mathcal{X}3})$, if $\bar{x}$ satisfies (i) with strict inequality for all nodes, and if $\bar{x}(E_3) + C(E_3) > \theta$, then it is possible to find a feasible flow vector $\bar{f}$ such that $(\bar{x}, \bar{f}) \in \mathcal{CP}^{\mathcal{X}3}$ and $\bar{f}$ does not saturate edge $e \in E_3$ if $x_e + C_e > 0$.*

Corollary 4.4.4 *Given an integral vector $\bar{x} \in Proj_x(\mathcal{CP}^{\mathcal{X}3})$, $\bar{x} + C > 0$, if $\bar{x}$ satisfies (i) with strict inequality for all nodes, and if $\bar{x}(E_3) + C(E_3) > \theta$, then it is possible to find a feasible flow vector $\bar{f}$ such that $(\bar{x}, \bar{f}) \in \mathcal{P}^{\mathcal{X}3}$ and $\bar{f}$ does not saturate any edge $e \in E_3$.*

Using Lemma 4.4.2, we next show that the projection of $\mathcal{P}^{\mathcal{X}3}$on the space of the discrete variables can be obtained by strengthening (i) and (ii). Lemma 4.4.5 can be considered as a generalization of the result by Magnanti, Mirchandani and Vachani. In [23] Magnanti et al. study a similar three-node network design problem (called NLP) where it is assumed that there is a single batch size and there is no existing capacity on the edges. Furthermore, the capacity constraints are different from the ones we study here, and consequently they can assume that there are only two source nodes with positive supply nodes.

Lemma 4.4.5

$$proj_x(\mathcal{P}^{\mathcal{X}3}) = \Big\{ \quad x \in \mathcal{R}^3 :$$

$$x(i) \geq \lceil T(i) \rceil \quad \textit{for all } i \in V_3 \tag{4.21}$$

$$\sum_{i>j} x_{i,j} \geq \max\left\{ \left\lceil \frac{\sum_i \lceil T(i) \rceil}{2} \right\rceil, \lceil \theta - C(E_3) \rceil \right\} \tag{4.22}$$

$$x_{i,j} \geq 0 \tag{4.23}$$

$$\Big\}$$

Proof. Let Q be the polyhedron defined by (4.21) - (4.23) and notice that $proj_x(\mathcal{P}^{\mathcal{X}3}) \subseteq Q \subseteq proj_x(\mathcal{C}\mathcal{P}^{\mathcal{X}3})$.

Consider any extreme point $\bar{x}$ of Q. If the inequalities defining $\bar{x}$ include (4.22) or one of (4.23), it is easy to see that $\bar{x}$ is integral. The remaining case occurs when $\bar{x}$ is defined by inequalities (4.21) alone. In this case $\bar{x}_{i,j} = (\lceil T(i) \rceil + \lceil T(j) \rceil - \lceil T(k) \rceil)/2$ implying $\bar{x}_{1,2} + \bar{x}_{1,3} + \bar{x}_{2,3} = (\lceil T(i) \rceil + \lceil T(j) \rceil + \lceil T(k) \rceil)/2$. Since $\bar{x}$ must also satisfy (4.22), it follows that

$$\frac{\sum_i \lceil T(i) \rceil}{2} \geq \left\lceil \frac{\sum_i \lceil T(i) \rceil}{2} \right\rceil$$

implying $\sum_i \lceil T(i) \rceil / 2$, and thus $\bar{x}$ is integral. ∎

We also note that (4.21) - (4.23) provide a non-redundant description of $proj_x(\mathcal{P}^{\mathcal{X}3})$ when

$$\max\left\{\left\lceil\frac{\sum_i \lceil T(i)\rceil}{2}\right\rceil, \lceil\theta - C(E_3)\rceil\right\} > \frac{\sum_i \lceil T(i)\rceil}{2} \tag{4.24}$$

and (4.22) is redundant when (4.24) holds as an equality.

4.4.2 Main Three-Partition Facets

In the remainder of this section, we will work with three-partitions of V and using the obvious relationship between three-partitions and K_3, we describe some facets of $\mathcal{P}^{\mathcal{X}}$ using (4.22) of Lemma 4.4.5 and its extensions. Given a partition $\Pi = \{S_1, S_2, S_3\}$ of V, we use $\delta(i,j)$ to denote $\delta(S_i) \cap \delta(S_j)$ and Δ to denote $\delta(1,2) \cup \delta(1,3) \cup \delta(2,3)$. For typographical ease, we use $x(i,j), y(i,j)$ and $C(i,j)$ in place of $x(\delta(i,j)), y(\delta(i,j))$ and $C(\delta(i,j))$ respectively.

Given a three-partition of V, for the generalization of (4.22) of Lemma 4.4.5 to define a facet of $\mathcal{P}^{\mathcal{X}}$, the partition has to satisfy certain properties. We next state these properties.

Definition 4.4.6 *Given a capacitated network $G = (V, E)$ and related traffic demands, a three-partition $\{S_1, S_2, S_3\}$ of V is called a* "**critical partition** " *of V if every S_i is a strong subset of V, $\lceil T(S_i)\rceil > T(S_i)$ for $i = 1, 2, 3$ and*

$$\lceil T(S_i)\rceil < \lceil T(S_j)\rceil + \lceil T(S_k)\rceil$$

for any permutation (i, j, k) of $\{1, 2, 3\}$.

As in Section 4.3, we first consider a generic three-partition facet and investigate some properties of the equations which are satisfied by all points of this facet.

Lemma 4.4.7 *Given a critical partition $\Pi = \{S_1, S_2, S_3\}$ of V and a proper face*

$$F = \left\{(x, y, f) \in \mathcal{P}^{\mathcal{X}} : \sum_{j>i} a_{i,j} \sum_{e\in\delta(i,j)} x_e + \sum_{j>i} b_{i,j} \sum_{e\in\delta(i,j)} y_e = c\right\}$$

of $\mathcal{P}^{\mathcal{X}}$*, where* $a_{i,j}, b_{i,j} \in R$*,* $j > i$*, and* $c \in R$*, assume that equation* $\alpha x + \beta y + \gamma f = \pi$ *is satisfied by all points in* F*.*

If there exists $\bar{p} = (\bar{x}, \bar{y}, \bar{f}) \in F$ *such that* $\bar{x}(\Delta) + \lambda \bar{y}(\Delta) > \max\{\sum_i T(i)/2, \theta - C(\Delta)\}$ *and* $\bar{x}(i,j) + \bar{y}(i,j) + C(i,j) > 0$ *for all* $j > i$ *then without loss of generality ,*

(i) $\alpha_e = \beta_e = 0$ *for all* $e \in E \setminus \Delta$,

(ii) $\gamma = 0$,

(iii) *for any* $j > i$*, if* $\bar{x}(i,j) > 0$ *then there exists* $\bar{\alpha}_{i,j} \in R$ *such that* $\alpha_e = \bar{\alpha}_{i,j}$*, for all* $e \in \delta(i,j)$*, and*

(iv) *if* $\bar{y}(i,j) > 0$*, then there exists* $\bar{\beta}_{i,j} \in R$ *such that* $\beta_e = \bar{\beta}_{i,j}$ *for all* $e \in \delta(i,j)$*.*

Proof.

(i) Given $e \in E \setminus \Delta$, let $p' = (x', y', f')$ be identical to p with the exception that $x'_e = \bar{x}_e + 1$. Then $p' \in F$ and consequently $\alpha_e = 0$. Similarly $= \beta_e = 0$.

(ii) Choose edges $e_{i,j} = (u_i, v_j) \in \delta(i,j)$ for all $j > i$ such that $C(e_{i,j}) = \max_{e \in \delta(i,j)}\{C_e\}$, and using $\bar{p}$ construct $p' = (x', y', f') \in F$ where

$$y'_e = \begin{cases} M & e \in E \setminus \Delta \\ \bar{y}(i,j) & e = e_{i,j} \\ 0 & \text{otherwise} \end{cases} \qquad x'_e = \begin{cases} M & e \in E \setminus \Delta \\ \bar{x}(i,j) & e = e_{i,j} \\ 0 & \text{otherwise} \end{cases}$$

and f' is a feasible flow vector. Since $\bar{p} \in F$, p' is also in F and using Corollary 4.4.4 we can assume that $x_{e_{i,j}} + \lambda y_{e_{i,j}} + C(e_{i,j}) > \max\{f^K_{u_i,v_j}, f^K_{v_j,u_i}\}$ for all $j > i$.

Let $T = (V, E'')$ be a spanning tree of V such that $E'' \subset \{e_{1,2}, e_{1,3}\} \cup (E \setminus \Delta)$. As in the proof of Theorem 4.2.2, we can first argue that $\gamma^k_{ij} = 0$ for $\{i,j\} \in E''$, $k \in K$ and then by using circulation flows, show that $\gamma^k_{ij} = 0$ for $\{i,j\} \in E \setminus E''$, $k \in K$ as well.

(iii) If $|\delta(i,j)| \geq 2$ then let $e_0 \in \delta(i,j)$ be an edge with $\bar{x}_{e_0} > 0$. To obtain a new point in F, we first choose an edge $e_1 \in \delta(i,j)$ different from e_0, and perturb $\bar{p}$ by decreasing x_{e_0} to zero and increasing x_{e_1} by $\bar{x}_{e_0}$ (if necessary increase x_e for $e \notin \Delta$) and rerouting some of the flow on e_0 to go through e_1 so that flows on e_0 and e_1 do not exceed capacity. This new point together with $\bar{p}$ imply that $\alpha_{e_0} = \alpha_{e_1}$ and since $e_1 \in \delta(i,j)$ is arbitrary, we can conclude that $\alpha_e = \bar{\alpha}_{i,j}$ for all $e \in \delta(i,j)$.

(iv) The proof is identical to part (iii). ∎

Given a three-partition $\Pi = \{S_1, S_2, S_3\}$ of V, we denote $\lceil T(S_i) \rceil$ by $\bar{T}(i)$ and $\sum_{u \in S_i} \sum_{v \in S_j} t_{ij}$ by $T(i,j)$.

We use θ for $\max_\pi \{T(i,j) + T(i,k) + T(k,j)\}$ and $\bar{\theta}$ for $\lceil \theta - C(\Delta) \rceil$. Lastly we define Θ to be

$$\Theta = \max \left\{ \left\lceil \frac{\sum_i \bar{T}(i)}{2} \right\rceil, \bar{\theta} \right\}.$$

The following is a straight forward extension of Lemma 4.4.5 to three-partitions of V.

Theorem 4.4.8 *Given a critical partition* $\Pi = \{S_1, S_2, S_3\}$ *of* V*, if* $\Theta - \bar{T}(i) \geq \lambda$ *for* $i = 1, 2, 3$*, then,*

$$x(\Delta) + \lambda y(\Delta) \geq \Theta \tag{4.25}$$

is a facet of $\mathcal{P}^{\mathcal{X}}$ *provided* $\Theta > \max\{\sum_i \bar{T}_i/2, \theta - C(E)\}$.

Proof. Validity of (4.25) is due to Lemma 4.4.5. As before, we will start with constructing a point on the related face $F = \{(x,y,f) \in \mathcal{P}^{\mathcal{X}} : x(\Delta) + \lambda y(\Delta) = \Theta\}$.

Choose a fixed $e_{i,j} \in \delta(i,j)$ for all $j > i$, and let $\bar{T}(3) \geq \bar{T}(2) \geq \bar{T}(1)$. Consider the point $p^1 = (x^1, y^1, f^1)$, where

$$x_e^1 = \begin{cases} M & e \notin \Delta \\ \lceil \sum_i \bar{T}(i)/2 \rceil - \bar{T}(3) & e = e_{1,2} \\ \lceil \sum_i \bar{T}(i)/2 \rceil - \bar{T}(2) & e = e_{1,3} \\ \lfloor \sum_i \bar{T}(i)/2 \rfloor - \bar{T}(1) + (\Theta - \lceil \sum_i \bar{T}(i)/2 \rceil) & e = e_{2,3} \\ 0 & \text{otherwise} \end{cases} \qquad y_e^1 = \begin{cases} M & e \notin \Delta \\ 0 & \text{otherwise} \end{cases}$$

and f^1 is some feasible flow vector which exists by Corollary 4.4.4. Observe that $p^1 \in F$.

Notice that, since Π is a critical partition ,

$$\left\lceil \frac{\sum_i \bar{T}(i)}{2} \right\rceil - \bar{T}(j) \geq \left\lceil \frac{\sum_i \bar{T}(i)}{2} \right\rceil - \bar{T}(3) = \left\lceil \frac{\bar{T}(1) + \bar{T}(2) - \bar{T}(3)}{2} \right\rceil \geq 1$$

for all j. It is also true that $\lfloor \sum_i \bar{T}(i)/2 \rfloor - \bar{T}(j) \geq 0$ for $j = 1, 2, 3$. Therefore $x^1 \geq 0$ and $x^1(1,2), x^1(1,3) > 0$. To use Lemma 4.4.7, we also need to show that $x^1(2,3) > 0$. Assume that $x^1(2,3) = 0$, and therefore $\Theta = \lceil \sum_i \bar{T}(i)/2 \rceil$, and $\lfloor \sum_i \bar{T}(i)/2 \rfloor = \bar{T}(1)$. In this case

$$0 = \left\lfloor \sum_i \bar{T}(i)/2 \right\rfloor - \bar{T}(1) \geq \frac{\bar{T}(2) + \bar{T}(3) - \bar{T}(1) - 1}{2}$$

together with the choice of $\bar{T}(1)$ implies that $\bar{T}(1) = \bar{T}(2) = \bar{T}(3) = 1$. Consequently, $\Theta = 2$ is less than $\lambda + \bar{T}(1)$, a contradiction. Therefore, $x^1(2,3)$ is also positive.

Assuming F is not a facet, let

$$\alpha x + \beta y + \gamma f = \pi \tag{4.26}$$

be an equation different from (4.25) satisfied by all points $p = (x, y, f) \in F$.

Using Lemma 4.4.7 with p^1 it is easy to see that $\gamma = 0$ and $\alpha_e = \beta_e = 0$ for $e \in E \setminus \Delta$. It is also true that, for $j > i$, if $e \in \delta(i,j)$, then $\alpha_e = \bar{\alpha}_{i,j}$. To show that there exists $\bar{\alpha} \in R$ such that, $\alpha_e = \bar{\alpha}$ for all $e \in \Delta$, we consider two cases. If $\sum_i \bar{T}(i)$ is odd, then it is possible to perturb p^1 by increasing $x^1_{e_{2,3}}$ and decreasing $x^1_{e_{1,2}}$ or $x^1_{e_{1,3}}$. On the other hand, if $\sum_i \bar{T}(i)$ is even then, $\Theta - \sum_i \bar{T}(i) > 0$ and we can perturb p^1 by decreasing $x^1_{e_{2,3}}$ and increasing $x^1_{e_{1,2}}$ or $x^1_{e_{1,3}}$. In both cases the resulting capacities satisfy the feasibility conditions by Lemma 4.4.5 so that it is possible to find a related flow vector.

Now consider a different point $p^2 = (x^2, y^1, f^2)$ where

$$x^2_e = \begin{cases} \lceil \sum_i \bar{T}(i)/2 \rceil - \bar{T}(3) + (\Theta - \lceil \sum_i \bar{T}(i)/2 \rceil) & e = e_{1,2} \\ \lfloor \sum_i \bar{T}(i)/2 \rfloor - \bar{T}(1) & e = e_{2,3} \\ x^1_e & \text{otherwise} \end{cases}$$

and f^2 is some feasible flow vector. Notice that ${x^2}_{1,2} \geq \lambda$ so that by perturbing p^2, we can show that $\alpha_{e_{1,2}} = (1/\lambda)\beta_{e_{1,2}}$. Constructing different points with $x_{e_{2,3}} \geq \lambda$ and $x_{e_{1,3}} \geq \lambda$ we can conclude that $\alpha_e = \bar{\alpha} = (1/\lambda)\beta_e$ for all $e \in \Delta$. Therefore, (4.26) is a multiple of (4.25) (plus a linear combination of flow-balance equalities), and thus F is a facet of $\mathcal{P}^{\mathcal{X}}$. ∎

Next we consider the case when given a critical partition $\{S_1, S_2, S_3\}$ of V, $\Theta - \bar{T}(i) \geq \lambda$ does not hold for all S_i. Let $\bar{T}(3) \geq \bar{T}(2) \geq \bar{T}(1)$. If $\lambda > \Theta - \bar{T}(3)$, then

$$\Theta - \bar{T}(3) \geq \left(\frac{\sum_i \bar{T}(i)}{2} + \frac{1}{2} \right) - \bar{T}(3) \geq \frac{\bar{T}(1) + \bar{T}(2) - \bar{T}(3) + 1}{2}$$

implies that $2\lambda + \bar{T}(3) > \bar{T}(1) + \bar{T}(2) + 1 \geq \bar{T}(3)$. Therefore, this case arises when $2\lambda - 1 > \bar{T}(1) + \bar{T}(2) - \bar{T}(3)$, or, in other words, when the sum $\bar{T}(1) + \bar{T}(2)$ is not very big when compared to $\bar{T}(3)$.

Theorem 4.4.9 *Given a critical partition* $\Pi = \{S_1, S_2, S_3\}$ *of* V*, let* $\bar{T}(3) \geq \bar{T}(2) \geq \bar{T}(1)$*. If* $\Theta - \bar{T}(3) < \lambda$ *and* $\bar{T}(3) \geq \lambda$*, then*

$$x(\Delta) + (\Theta - \bar{T}(3))y(1,2) + \lambda y(1,3) + \lambda y(2,3) \geq \Theta \tag{4.27}$$

is a facet of $\mathcal{P}^{\mathcal{X}}$ *provided* $\Theta > \sum_i \bar{T}_i/2$.

Proof. For any point $p = (x, y, f) \in \mathcal{P}^{\mathcal{X}}$ inequality (4.27) is clearly valid when $y(1,2) = 0$. On the other hand, if $y(1,2) \geq 1$, then notice that

$$\begin{aligned} x(\Delta) + (\Theta - \bar{T}(3))y(1,2) &+ \lambda y(1,3) + \lambda y(2,3) \\ &\geq x(1,3) + x(2,3) + (\Theta - \bar{T}(3)) + \lambda y(1,3) + \lambda y(2,3) \\ &= x(S_3) + \lambda y(S_3) + (\Theta - \bar{T}(3)) \\ &\geq \bar{T}(3) + (\Theta - \bar{T}(3)) = \Theta. \end{aligned}$$

Let F be the related face, assuming that it is not a facet, let $\alpha x + \beta y + \gamma f = \pi$ be an equation different from (4.27) satisfied by all points $p = (x, y, f) \in F$.

Notice that $\bar{T}(1) + \bar{T}(2) > \bar{T}(3) \geq \lambda$ implies that $\sum_i \bar{T}(i) > 2\lambda$ so that $\Theta > \lambda \geq 2$. Therefore, we can use point p^1 of Theorem 4.4.8 to show that $\gamma = 0$, $\alpha_e = \beta_e = 0$ for all $e \in E \setminus \Delta$, and $\alpha_e = \bar{\alpha}$ for all $e \in \Delta$.

Next, using p^2 of Theorem 4.4.8 we construct $p^3 = (x^3, y^3, f^2) \in F$ which is identical to p^2 with the exception that $x^3_{e_{1,2}} = 0$ and $y^3_{e_{1,2}} = 1$. Existence of $p^2 \in F$ together with $p^3 \in F$ imply that $\beta_{e_{1,2}} = (\Theta - T(3))\alpha_{e_{1,2}}$.

Lastly, we construct $p^4 = (x^4, y^3, f^4)$ where x^4 is identical to x^3 with the exception that $x_{e_{2,3}} = (T(2) - \lambda)^+$ and $x_{e_{1,3}} = T(3) - (T(2) - \lambda)^+$. Notice that $x(\Delta) + \lambda y(\Delta) \geq x(\Delta) + (\Theta - \bar{T}(3))y(1,2) = \Theta$, and $x_{e_{1,3}} = \min\{T(3), T(3) - T(2) \ +\lambda\} \geq \lambda$.

To show that $p^4 \in \mathcal{P}^{\mathcal{X}}$ and thus $p^4 \in F$, we first note that $x^4(S_3) = \bar{T}(3)$ and $x^4(S_2) + \lambda y^3(S_2) = \lambda + (\bar{T}(2) - \lambda)^+ \geq (\bar{T}(2)$. Furthermore if $(\bar{T}(2) - \lambda)^+ = 0$ then $\tilde{x}_{e_{1,3}} = \bar{T}(3) \geq \bar{T}(1)$, on the other hand, if $\bar{T}(2) > \lambda$ then

$$\begin{aligned} x^4_{e_{1,3}} + \lambda y^4_{e_{1,2}} &= \bar{T}(3) - \bar{T}(2) + 2\lambda \\ &> \bar{T}(3) - \bar{T}(2) + 2\Theta - 2\bar{T}(3) \\ &= 2\Theta - \bar{T}(2) - \bar{T}(3) \\ &> \bar{T}(1). \end{aligned}$$

Therefore $p^4 \in F$ and since it is possible to perturb p^4 by decreasing $x^4_{e_{1,3}}$ by λ and increasing $y^4_{e_{1,3}}$ by 1, we can conclude that $\beta_{e_{1,3}} = \lambda\alpha_{e_{1,3}}$. Constructing a similar point with $x_{e_{2,3}} \geq \lambda$ we can also argue that $\beta_{e_{2,3}} = \lambda\alpha_{e_{2,3}}$ and complete the proof. ∎

4.4.3 Other Three-Partition Facets

Next we study facets of the CEP polytope which primarily exclude points with $y(\Delta) = \Theta/\lambda$ from the feasible region when Θ/λ is fractional. We basically consider two cases depending on which one of the two terms dominates in determining Θ. But, before proceeding any further, we need some more notation. Given a partition $\Pi = \{S_1, S_2, S_3\}$ of V, we define $r^+(i)$ to denote $r(\bar{T}(i), \lambda)$ and $T^+(i)$ to denote $\lceil \bar{T}(i)/\lambda \rceil$. Notice that $\bar{T}(i) = \lambda\,(T^+(i) - 1) + r^+(i)$ for all $S_i \in \Pi$. We further define $r_{\max} = \max\{r^+(i)\}$, $r_{\min} = \min\{r^+(i)\}$ and $r_{\text{med}} = \sum_i r^+(i) - r_{\min} - r_{\max}$.

Notice that if $\bar{T}(3) \geq \bar{T}(2) \geq \bar{T}(1)$, then $T^+(3) \geq T^+(2) \geq T^+(1)$. Furthermore, when $\bar{T}(3) < \bar{T}(1) + \bar{T}(2)$, $T^+(3)$ is no more than $T^+(2) + T^+(1)$, and

$$\left\lfloor \frac{\sum_i T^+(i)}{2} \right\rfloor - T^+(i) \geq \left\lfloor \frac{\sum_i T^+(i)}{2} \right\rfloor - T^+(3) = \left\lfloor \frac{T^+(1) + T^+(2) - T^+(3)}{2} \right\rfloor \geq 0.$$

The following theorem has the same spirit as Theorem 4.2.4 and (4.28) is a MIR inequality. Remember that

$$\Theta = \max\left\{ \left\lceil \frac{\sum_i \bar{T}(i)}{2} \right\rceil, \bar{\theta} \right\}$$

and we note that the required conditions can hold only if the second term strictly dominates the first. In other words, (4.28) is a facet only if $\Theta > \lceil \sum_i \bar{T}(i)/2 \rceil$.

Theorem 4.4.10 *Given a critical partition $\Pi = \{S_1, S_2, S_3\}$ of V, let $\bar{T}(3) = \max\{\bar{T}(i)\}$ and $r^+(1) \geq r^+(2)$. If $\lambda > r(\Theta, \lambda)$, then*

$$x(\Delta) + r(\Theta, \lambda) y(\Delta) \geq r(\Theta, \lambda) \lceil \Theta/\lambda \rceil \tag{4.28}$$

is a facet of $\mathcal{P}^{\mathcal{X}}$ *provided one of the following conditions is true.*

(i) $\lceil\Theta/\lambda\rceil - 1 \geq \lceil\sum_i T^+(i)/2\rceil$.

(ii) $\lceil\Theta/\lambda\rceil = \lceil\sum_i T^+(i)/2\rceil > \sum_i T^+(i)/2$, $r(\Theta,\lambda) \geq r^+(2)$ *and* $\bar{T}(2) > 1$.

(iii) $\lceil\Theta/\lambda\rceil = \lceil\sum_i T^+(i)/2\rceil = \sum_i T^+(i)/2$, $r(\Theta,\lambda) \geq r_{\max}$, $r(\Theta,\lambda) > r_{\min}$ *and*

$T^+(1) + T^+(2) > T^+(3)$.

Proof. Validity of (4.28) should be clear. To show that $F = \{(x,y,f) \in \mathcal{P}^{\mathcal{X}} : x(\Delta) + r(\Theta,\lambda)y(\Delta) = r(\Theta,\lambda)\lceil\Theta/\lambda\rceil\}$ is a facet, we first analyze cases (i) and (ii).
$(i),(ii)$ Choose a fixed edge $e_{i,j} \in \delta(i,j)$ for all $j > i$, and consider $p^1 = (x^1, y^1, f^1)$ where

$$y_e^1 = \begin{cases} M & e \notin \Delta \\ \lfloor\sum_i T^+(i)/2\rfloor - T^+(3) & e = e_{1,2} \\ \lceil\sum_i T^+(i)/2\rceil - T^+(2) & e = e_{1,3} \\ \lceil\Theta/\lambda\rceil - T^+(1) - 1 & e = e_{2,3} \\ 0 & \text{otherwise} \end{cases} \qquad x_e^1 = \begin{cases} M & e \notin \Delta \\ 1 & e = e_{1,2} \\ 0 & e = e_{1,3} \\ r(\Theta,\lambda) - 1 & e = e_{2,3} \\ 0 & \text{otherwise} \end{cases}$$

and f^1 is a feasible flow vector. Notice that $\lceil\Theta/\lambda\rceil - T^+(1) - 1 \geq \lceil T^+(2)/2\rceil - 1 \geq 0$, and therefore $x^1, y^1 \geq 0$. To see that it is possible to find a feasible flow vector f^1, first note that $x^1(\Delta) + \lambda y^1(\Delta) = \Theta$ and p^1 satisfies cut-set inequalities for S_1 and S_3. Next, observe that the capacity across the cut $\delta(S_2)$ is

$$\begin{aligned} x^1(\delta(S_2)) + \lambda y^1(\delta(S_2)) &\geq \lambda\left(\left\lfloor\frac{\sum_i T^+(i)}{2}\right\rfloor + \left\lceil\frac{\Theta}{\lambda}\right\rceil - 1 - T^+(1) - T^+(3)\right) + r(\Theta,\lambda) \\ &\geq \lambda\left(\left\lfloor\frac{\sum_i T^+(i)}{2}\right\rfloor + \left\lceil\frac{\sum_i T^+(i)}{2}\right\rceil - T^+(1) - T^+(3) - 1\right) + r(\Theta,\lambda) \\ &\geq \lambda\left(T^+(2) - 1\right) + r(\Theta,\lambda) \end{aligned}$$

$$\geq \quad T^+(2)$$

(where the last inequality follows from $r(\Theta, \lambda) \geq r^+(2)$) so that p^1 satisfies the cut-set inequality for S_2 as well. Therefore, using Lemma 4.4.2, $p^1 \in \mathcal{P}^{\mathcal{X}}$ and thus $p^1 \in F$.

Assuming F is not a facet, let $\alpha x + \beta y + \gamma f = \pi$ be an equation different from (4.28) satisfied by all points $p = (x, y, f) \in F$. Observe that $x^1(1,2) > 0$ and $y^1(1,3) = \lceil \sum_i T^+(i)/2 \rceil - T^+(2) \geq (T^+(3) + T^+(1) - T^+(2))/2 \geq T^+(1)/2 > 0$. Lastly, if $y^1(2,3) = 0$ then,

$$y^1(2,3) = 0 = \left\lceil \frac{\Theta}{\lambda} \right\rceil - T^+(1) - 1 \geq \frac{\sum_i T^+(i) + 1}{2} - T^+(1) - 1 \geq \frac{T^+(2) - 1}{2} \geq 0$$

implies that $\lceil \Theta/\lambda \rceil = \lceil \sum_i T^+(i)/2 \rceil$ and $T^+(2) = 1$. In this case $x^1(2,3) = r(\Theta, \lambda) - 1 \geq r^+(2) - 1 = \bar{T}(2) - 1 > 0$. Therefore, we can conclude that $x^1(2,3) + \lambda y^1(2,3) > 0$.

Using Lemma 4.4.7, we can now argue that $\gamma = 0$ and $\alpha_e = \beta_e = 0$ for all $e \in E \setminus \Delta$. Moreover, it is possible to perturb p^1 by decreasing $x_{e_{1,3}}$ and increasing $x_{e_{1,2}}$ or $x_{e_{2,3}}$, implying that for some $\bar{\alpha} \in R$, $\alpha_e = \bar{\alpha}$ whenever $e \in \Delta$.

We next construct a point $p^2 = (x^2, y^2, f^2) \in F$ where $x^2 = 0$, y^2 is identical to y^1 with the exception that $y^2_{e_{2,3}} = \lceil \Theta/\lambda \rceil - T^+(1)$, and f^2 is some feasible flow vector which exists by Lemma 4.4.2. Perturbing p^2 by increasing $y^2_{e_{1,2}}$ and decreasing $y^2_{e_{1,3}}$ or $y^2_{e_{2,3}}$, we conclude that there exists $\bar{\beta} \in R$, such that $\beta_e = \bar{\beta}$, for all $e \in \Delta$. Furthermore, $p^1, p^2 \in F$ implies that $\bar{\beta} = r(\Theta, \lambda)\bar{\alpha}$ and thus F is indeed a facet.

(iii) Choose a fixed edge $e_{i,j} \in \delta(i,j)$ for all $j > i$, and consider $p^3 = (x^3, y^3, f^3) \in F$ where

$$y^3_e = \begin{cases} M & e \notin \Delta \\ (T^+(1) + T^+(2) - T^+(3))/2 & e = e_{1,2} \\ (T^+(1) + T^+(3) - T^+(2))/2 & e = e_{1,3} \\ (T^+(2) + T^+(3) - T^+(1))/2 & e = e_{2,3} \\ 0 & \text{otherwise} \end{cases} \qquad x^3_e = \begin{cases} M & e \notin \Delta \\ 0 & e \in \Delta \end{cases}$$

and f^3 is a feasible flow vector. Since $y^3(i,j) > 0$ for all $j > i$, we can apply Lemma 4.4.7 with p^3 and show that $\gamma = 0$, $\alpha_e = \beta_e = 0$ for all $e \in E \setminus \Delta$, and for all $j > i$, there exists $\bar{\beta}_{i,j} \in R$ such that $\beta_e = \bar{\beta}_{i,j}$ for all $e \in \delta(i,j)$.

Next for each $e_{i,j}$ we perturb p^3 by decreasing $y_{e_{i,j}}$ by 1 and increasing $x_{e_{i,j}}$ by $r(\Theta,\lambda)$ to obtain new points in F. Using these points together with p^3, we conclude that for all $j > i$, if $e \in \delta(i,j)$ then, $\alpha_e = r(\Theta,\lambda)\bar{\beta}_{i,j}$.

Lastly, let $\{a,b,c\}$ be a permutation of $\{1,2,3\}$ so that $r^+(a) \geq r^+(b) \geq r^+(c)$. Since $r(\Theta,\lambda) > r_{\min} = r^+(c)$, it is possible to permute p^3 by decreasing $y_{e_{b,c}}$ by 1, increasing $x_{e_{b,c}}$ by $r(\Theta,\lambda) - 1$ and increasing $x_{e_{a,b}}$ by 1. Similarly, it is possible to permute p^3 by decreasing $y_{e_{a,c}}$ by 1, increasing $x_{e_{a,c}}$ by $r(\Theta,\lambda) - 1$ and increasing $x_{e_{a,b}}$ by 1. These new points are in F, and thus $\bar{\beta}_{a,b} = \bar{\beta}_{a,c} = \bar{\beta}_{b,c}$, implying that F is a facet of $\mathcal{P}^{\mathcal{X}}$. ∎

We also note that it is possible to relax the condition $\bar{T}(2) > 1$ from (ii) of Theorem 4.4.10, but in this case $C(2,3)$ has to be positive whenever $\bar{T}(2) = 1$. To avoid complicating the proof any further, we chose to skip this.

In the remainder of this section, we consider the case when for a critical partition $\{S_1, S_2, S_3\}$ of V, Θ is equal to $\lceil \sum_i \bar{T}(i)/2 \rceil$, and we identify facets of $\mathcal{P}^{\mathcal{X}}$ that exclude some of the fractional points from the feasible region when $y(\Delta)$ is less than $\lceil \Theta/\lambda \rceil$. Before that we will make an observation concerning the identity $\bar{T}(i) = \lambda\,(T^+(i) - 1) + r^+(i)$ and the cut-set inequalities. First note that

$$\sum_i \bar{T}(i) = \lambda \left(\sum_i T^+(i) - 3 \right) + \sum_i r^+(i)$$

implying

$$\frac{\sum_i \bar{T}(i)}{2} = \lambda \left(\frac{\sum_i T^+(i)}{2} - \frac{3}{2} \right) + \frac{\sum_i r^+(i)}{2}.$$

Therefore, depending on $\sum_i T^+(i)$, we can write

$$\left\lceil \frac{\sum_i \bar{T}(i)}{2} \right\rceil = \begin{cases} \lambda \left(\left\lceil \frac{\sum_i T^+(i)}{2} \right\rceil - 2 \right) + \left\lceil \frac{\sum_i r^+(i)}{2} \right\rceil & \text{if } \sum_i T^+(i) \text{ is odd} \\ \lambda \left(\left\lceil \frac{\sum_i T^+(i)}{2} \right\rceil - 2 \right) + \left\lceil \frac{\lambda + \sum_i r^+(i)}{2} \right\rceil & \text{if } \sum_i T^+(i) \text{ is even.} \end{cases}$$

Next, note that when $x(\Delta) = 0$, the cut set inequalities imply that $y(\Delta) \geq \lceil \sum_i T^+(i)/2 \rceil$ and when $y(\Delta) = \lceil \sum_i T^+(i)/2 \rceil - 1$, then

$$x(\Delta) \geq \begin{cases} r_{\min} & \text{if } \sum_i T^+(i) \text{ is odd} \\ r_{\text{med}} & \text{if } \sum_i T^+(i) \text{ is even.} \end{cases}$$

This is easy to see as $y(\Delta) = \lceil \sum_i T^+(i)/2 \rceil - 1$ implies that $y(\delta(S_i)) \leq T^+(i) - 1$ holds for some $i \in \{1, 2, 3\}$ and using the cut-set inequality (4.7), $x(\delta(S_i)) \geq r^+(i)$. Furthermore if $\sum_i T^+(i)$ is even, then either $y(\delta(S_i)) \leq T^+(i) - 2$ for some $i \in \{1, 2, 3\}$ and $x(\delta(S_i)) \geq r^+(i) + \lambda$, or $y(\delta(S_i)) \leq T^+(i) - 1$ and thus $x(\delta(S_i)) \geq r^+(i)$ holds for two separate subsets.

Next, we study the case when $\sum_i T^+(i)$ is odd more closely. For a given $p = (x, y, f) \in \mathcal{P}^{\mathcal{X}}$, let k denote $(\lceil \sum_i T^+(i)/2 \rceil - y(\Delta))^+$. Using the previous observations and the three-partition inequality (4.25) we can write,

$$x(\Delta) \geq \begin{cases} 0 & \text{if } k = 0 \\ r_{\min} & \text{if } k = 1 \\ \lceil \sum_i r^+(i)/2 \rceil + \lambda(k-2) & \text{if } k \geq 2. \end{cases} \tag{4.29}$$

As seen in Figure 4.1, it is possible to write valid inequalities stronger than $x(\Delta) + \lambda y(\Delta) \geq \lceil \sum_i \bar{T}(i)/2 \rceil$ when $x(\Delta) < \lceil \sum_i r^+(i)/2 \rceil$. We note that $(1/\lambda) \lceil \sum_i \bar{T}(i)/2 \rceil$, the value $y(\Delta)$ assumes when $x(\Delta) + \lambda y(\Delta) = \lceil \sum_i \bar{T}(i)/2 \rceil$ and $x(\Delta) = 0$ is not necessarily integral and it is strictly less than $\lceil \sum_i T^+(i)/2 \rceil$. Depending on the value of $\lceil \sum_i r^+(i)/2 \rceil$, $(1/\lambda) \lceil \sum_i \bar{T}(i)/2 \rceil$ can be larger or smaller than $\lceil \sum_i T^+(i)/2 \rceil - 1$, but in either case point p_2 lies above the line $x(\Delta) + \lambda y(\Delta) = \lceil \sum_i \bar{T}(i)/2 \rceil$. In other words, $r_{\min} + \lambda (\lceil \sum_i T^+(i)/2 \rceil - 1) \geq$

$\lceil \sum_i \bar{T}(i)/2 \rceil$.

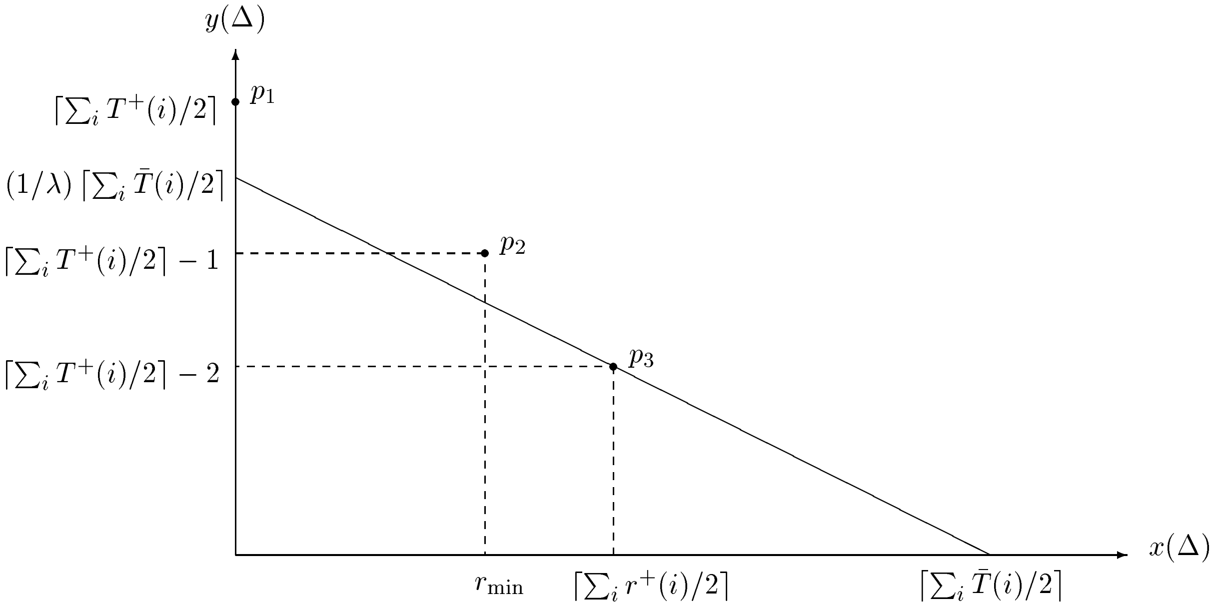

Figure 4.1: Finding new cuts using cut-set and 3-partition inequalities ($\sum_i T^+(i)$ is odd).

We first consider the case when p_2 lies above the line joining p_1 and p_3.

Theorem 4.4.11 *Given a critical partition* $\Pi = \{S_1, S_2, S_3\}$ *of* V*, if* $\Theta = \left\lceil \frac{\sum_i \bar{T}(i)}{2} \right\rceil$ *and* $r_{\min} \geq \frac{1}{2} \left\lceil \frac{\sum_i r^+(i)}{2} \right\rceil$*, then,*

$$x(\Delta) \geq \frac{1}{2} \left\lceil \frac{\sum_i r^+(i)}{2} \right\rceil \left(\left\lceil \frac{\sum_i T^+(i)}{2} \right\rceil - y(\Delta) \right) \tag{4.30}$$

is a facet of $\mathcal{P}^{\mathcal{X}}$ *provided* $\Theta > \max\{2, \theta - C(\Delta)\}$ *and both* $\sum_i T^+(i)$ *and* $\sum_i r^+(i)$ *are odd.*

Proof. Validity of (4.30) is due to (4.29). To see that it is a facet, let $\bar{T}(3) \geq \bar{T}(2) \geq \bar{T}(1)$, and F be the face of $\mathcal{P}^{\mathcal{X}}$ implied by (4.30). Choose a fixed edge $e_{i,j} \in \delta(i,j)$ for all $j > i$, and consider $p^1 = (x^1, y^1, f^1)$ where

$$y^1_e = \begin{cases} M & e \notin \Delta \\ \lfloor \sum_i T^+(i)/2 \rfloor - T^+(3) & e = e_{1,2} \\ \lfloor \sum_i T^+(i)/2 \rfloor - T^+(2) & e = e_{1,3} \\ \lfloor \sum_i T^+(i)/2 \rfloor - T^+(1) & e = e_{2,3} \\ 0 & \text{otherwise} \end{cases} \qquad x^1_e = \begin{cases} M & e \notin \Delta \\ \lceil \sum_i r^+(i)/2 \rceil - r^+(3) & e = e_{1,2} \\ \lceil \sum_i r^+(i)/2 \rceil - r^+(2) & e = e_{1,3} \\ \lfloor \sum_i r^+(i)/2 \rfloor - r^+(1) & e = e_{2,3} \\ 0 & \text{otherwise} \end{cases}$$

and f^1 is a feasible flow vector. Clearly $y^1 \geq 0$, $\lambda y^1(\Delta) + x^1(\Delta) = \Theta$ and p^1 satisfies the cut-set inequalities for all $S_i \in \Pi$. As we show next, $x^1 \geq 0$ and thus $p^1 \in F$.

To see that $x^1 \geq 0$, note that when $\sum_i r^+(i)$ is odd, $r_{\min} \geq \frac{1}{2} \left\lceil \frac{\sum_i r^+(i)}{2} \right\rceil$ implies

$$\begin{aligned} r_{\text{med}} + r_{\min} &\geq \frac{\sum_i r^+(i)}{2} + \frac{1}{2} \\ \frac{1}{2}(r_{\text{med}} + r_{\min}) &\geq \frac{r_{\max}}{2} + \frac{1}{2} \\ r_{\text{med}} + r_{\min} &\geq r_{\max} + 1 \end{aligned}$$

so that $x^1 \geq 0$ and $x^1(1,2), x^1(1,3) > 0$. To see that $x^1(2,3) + \lambda y^1(2,3) > 0$, notice that if $y^1(2,3) = 0$ then,

$$0 = \frac{T^+(3) + T^+(2) - T^+(1) - 1}{2} \geq \frac{T^+(3) - 1}{2}$$

implies that $T^+(3) = T^+(2) = T^+(1) = 1$. If at the same time $x^1(2,3) = 0$ then

$$0 = \frac{r^+(3) + r^+(2) - r^+(1) - 1}{2} = \frac{\bar{T}(3) + \bar{T}(2) - \bar{T}(1) - 1}{2} \geq \frac{\bar{T}(3) - 1}{2}$$

implying $\lceil \sum_i \bar{T}(i)/2 \rceil = 2$, a contradiction.

If we let $\alpha x + \beta y + \gamma f = \pi$ be an equation satisfied by all $p = (x, y, f) \in F$, then by applying Lemma 4.4.7 with p^1, we can show that $\gamma = 0$ and $\alpha_e = \beta_e = 0$ for all $e \in E \setminus \Delta$.

It is possible to modify p^1 by decreasing $x_{e_{1,2}}$ or $x_{e_{1,3}}$ by 1 and increasing $x_{e_{2,3}}$ by 1 (and modifying flow) to obtain new points in F, which implies that there is an $\bar{\alpha} \in R$ such that $\alpha_e = \bar{\alpha}$ for all $e \in \Delta$.

Lastly, we construct $p^2 = (x^2, y^2, f^2) \in F$ where $x^2 = 0$, f^2 is a feasible flow vector and y^2 is identical to y^1 with the exception that $y^2_{e_{1,2}} = y^1_{e_{1,2}} + 1$ and $y^2_{e_{1,3}} = y^1_{e_{1,3}} + 1$. Since it is possible to find new points by modifying p^2 by decreasing $y_{e_{1,2}}$ or $y_{e_{1,3}}$ by 1 and increasing $y_{e_{2,3}}$ by 1, we can argue that for some $\bar{\beta} \in R$, $\beta_e = \bar{\beta}$ for all $e \in \Delta$. Finally $p^1, p^2 \in F$ implies that $\bar{\beta} = \frac{1}{2}\left\lceil \frac{\sum_i r^+(i)}{2} \right\rceil \bar{\alpha}$, which completes the proof. ∎

We next consider the case when p_2 lies below the line joining p_1 and p_3. Notice that when $r_{\min} < (1/2)\lceil \sum_i r^+(i)/2 \rceil$, (4.30) of Theorem 4.4.11 is not valid, but in this case we can write two new valid inequalities using p_1 and p_2 or p_2 and p_3. These inequalities are,

$$x(\Delta) \geq r_{\min}\left(\left\lceil \sum_i T^+(i)/2 \right\rceil - y(\Delta)\right) \tag{4.31}$$

and

$$x(\Delta) \geq r_{\min} + \left(\left\lceil \sum_i r^+(i)/2 \right\rceil - r_{\min}\right)\left(\left\lceil \sum_i T^+(i)/2 \right\rceil - y(\Delta) - 1\right). \tag{4.32}$$

Unfortunately, these inequalities do not define facets of $\mathcal{P}^{\mathcal{X}}$ since any point on the faces related with (4.31) and (4.32) also satisfies $y(a) = T^+(a) - 1$ where $a = \text{argmin}\{\bar{T}(i)\}$. We next present a facet of $\mathcal{P}^{\mathcal{X}}$ which combines (4.31) and (4.32).

Theorem 4.4.12 *Given a critical partition* $\Pi = \{S_1, S_2, S_3\}$ *of* V*, let* $r^+(3) \geq r^+(2) \geq r^+(1)$. *If* $\Theta = \lceil \sum_i \bar{T}(i)/2 \rceil$, *and* $r^+(1) \leq \frac{1}{2}\lceil \sum_i r^+(i)/2 \rceil$, *then*

$$\frac{x(1,2) + x(1,3)}{r^+(1)} + \frac{x(2,3)}{\min\{r^+(2), \lceil \sum_i r^+(i)/2 \rceil - r^+(1)\}} \geq \left(\left\lceil \frac{\sum_i T^+(i)}{2} \right\rceil - y(\Delta)\right) \tag{4.33}$$

defines a facet of $\mathcal{P}^{\mathcal{X}}$ *provided* $\max_i\{T^+(i)\} > 1$ *and* $\sum_i T^+(i)$ *is odd.*

Proof. Before showing that (4.33) is a valid inequality, we first define $x(1)$ to denote $x(1,2) + x(2,3)$, α to denote $1/r^+(1)$ and β to denote the coefficient of $x(2,3)$ in (4.33). We further let $g(x)$ to denote the left hand side of (4.33) so that $g(x) = \alpha x(1) + \beta x(2,3)$.

Notice that $\lceil \sum_i r^+(i)/2 \rceil \geq 2r^+(1)$ together with $r^+(2) \geq r^+(1)$ implies that $\alpha \geq \beta$ and $\beta r^+(2) \geq 1$.

First note that (4.33) is valid for any $p = (x, y, f) \in \mathcal{P}^{\mathcal{X}}$ whenever $y(\Delta) \geq \lceil \sum_i T^+(i)/2 \rceil$. Next, consider the case when $y(\Delta) \geq \lceil \sum_i T^+(i)/2 \rceil - 1$, so that, there exists an index $i \in \{1, 2, 3\}$ with $y(\delta(S_i)) < T^+(i)$. If $y(\delta(S_1)) < T^+(1)$ then, the cut-set inequality for S_1 implies $x(1) \geq r^+(1)$ and thus $g(x) \geq 1$. On the other hand, if $y(\delta(S_1)) \geq T^+(1)$ then, using the cut-set inequalities for S_2 or S_3 we have $x(\Delta) \geq r^+(2)$ and $g(x) \geq \beta x(\Delta) \geq \beta r^+(2) \geq 1$.

The last case we consider is when $y(\Delta)$ is at most $\lceil \sum_i T^+(i)/2 \rceil - 2$. Let $k(i) = (T^+(i) - y(\delta(S_i)))^+$ and $K = \lceil \sum_i T^+(i)/2 \rceil - y(\Delta) \geq 2$ and note that $\sum_i k(i) \geq 2K - 1$. If $k(1) \geq 1$ then, to find a lower bound on $g(x)$, we look at the optimal value of the following linear program.

$$\begin{aligned}
\min \quad z = \quad & \alpha x(1) + \beta x(2,3) \\
\text{st.} \qquad\qquad x(1) \quad & \geq \quad r^+(1)k(1) \\
x(1) + x(2,3) \quad & \geq \quad \lambda(K-2) + \lceil \textstyle\sum_i r^+(i)/2 \rceil \\
x(1), x(2,3) \quad & \geq \quad 0
\end{aligned}$$

where, we minimize $g(x)$ subject to some valid inequalities. It is easy to see that the optimal solution has $x(1) = r^+(1)k(1)$ and $x(2,3) = \lambda(K-2) + \lceil \sum_i r^+(i)/2 \rceil - r^+(1)k(1)$ yielding,

$$\begin{aligned}
z &= \frac{r^+(1)k(1)}{r^+(1)} + \frac{\lambda(K-2) + \lceil \sum_i r^+(i)/2 \rceil - r^+(1)k(1)}{\min\{r^+(2), \lceil \sum_i r^+(i)/2 \rceil - r^+(1)\}} \\
&\geq k(1) + (K-2) + \frac{\lceil \sum_i r^+(i)/2 \rceil - r^+(1)k(1)}{\min\{r^+(2), \lceil \sum_i r^+(i)/2 \rceil - r^+(1)\}} \\
&\geq k(1) + (K-2) + 1 - \frac{r^+(1)\,(k(1)-1)}{\min\{r^+(2), \lceil \sum_i r^+(i)/2 \rceil - r^+(1)\}} \\
&\geq k(1) + 1 + (K-2) - k(1) + 1
\end{aligned}$$

so that $g(x) \geq z \geq K$.

On the other hand, if $k(1) = 0$, then $k(2) + k(3) \geq 2K - 1$ and $\max\{k(1), k(2)\} \geq K$. Writing the cut-set inequalities for S_2 and S_3 we have, $x(\Delta) \geq \max\{x(1,2)+x(2,3), x(1,2)+x(2,3)\} \geq \max\{r^+(2)k(2), r^+(3)k(3)\} \geq r^+(2)K$. Therefore $g(x) \geq \beta x(\Delta) \geq \beta r^+(2)K \geq K$ and (4.33) is satisfied by all $p = (x, y, f) \in \mathcal{P}^{\mathcal{X}}$.

To show that F, the face related with (4.33), is a facet of $\mathcal{P}^{\mathcal{X}}$, we first choose a fixed edge $e_{i,j} \in \delta(i,j)$ for all $j > i$, and let (a, b, c) be a permutation of $\{1, 2, 3\}$ such that $T^+(a) \geq T^+(b) \geq T^+(c)$. Then, we construct a point $p^1 = (x^1, y^1, f^1)$ where

$$y_e^1 = \begin{cases} M & e \notin \Delta \\ \lceil \sum_i T^+(i)/2 \rceil - T^+(a) & e = e_{b,c} \\ \lceil \sum_i T^+(i)/2 \rceil - T^+(b) & e = e_{a,c} \\ \lfloor \sum_i T^+(i)/2 \rfloor - T^+(c) & e = e_{a,b} \\ 0 & \text{otherwise} \end{cases}$$

$x = 0$ and f^1 is a feasible flow vector. Notice that p^1 satisfies the feasibility conditions of Lemma 4.4.2. Assume that $\alpha x + \beta y + \gamma f = \pi$ is an equation satisfied by all points $p = (x, y, f) \in F$, and note that $y^1(e_{a,c}), y^1(e_{b,c}) > 0$ since $\sum_i T^+(i)$ is odd, and $y^1(e_{a,b}) > 0$ by $T^+(a) > 1$. Using the fact that $y(\Delta) > \Theta$, we can applying Lemma 4.4.7 with p^1 to show that $\gamma = 0$ and $\alpha_e = \beta_e = 0$ for all $e \in E \setminus \Delta$. Furthermore, since it is possible to modify p^1 by decreasing $y(e_{a,c})$ or $y(e_{b,c})$ by 1 and increasing $y(e_{a,b})$ by 1 (and modifying flow) to obtain new points in F, we can argue that there exists $\bar{\beta} \in R$ such that $\beta_e = \bar{\beta}$ for all $e \in \Delta$.

Next, we construct $p^2 = (x^2, y^2, f^2) \in F$ where

$$y^2_e = \begin{cases} M & e \notin \Delta \\ \lfloor \sum_i T^+(i)/2 \rfloor - T^+(3) & e = e_{1,2} \\ \lfloor \sum_i T^+(i)/2 \rfloor - T^+(2) & e = e_{1,3} \\ \lceil \sum_i T^+(i)/2 \rceil - T^+(1) & e = e_{2,3} \\ 0 & \text{otherwise} \end{cases} \qquad x^2_e = \begin{cases} M & e \notin \Delta \\ r^+(1) & e = e_{1,2} \\ 0 & \text{otherwise} \end{cases}$$

and f^2 is a feasible flow vector. Applying Lemma 4.4.7 with p^2 and using the fact that $p^1, p^2 \in F$, we can argue that $\alpha_e = \bar{\beta}/r^+(1)$ for all $e \in \delta(1,2)$. Constructing a similar point with $x_{e_{1,2}} = 0$ and $x_{e_{1,3}} = r^+(1)$ we can also show that $\alpha_e = \bar{\beta}/r^+(1)$ for all $e \in \delta(1,3)$.

Lastly, if $r^+(2) > \lceil \sum_i r^+(i)/2 \rceil - r^+(1)$, then we construct $p^3 = (x^3, y^3, f^3) \in F$ where

$$y^3_e = \begin{cases} M & e \notin \Delta \\ \lfloor \sum_i T^+(i)/2 \rfloor - T^+(3) & e = e_{1,2} \\ \lfloor \sum_i T^+(i)/2 \rfloor - T^+(2) & e = e_{1,3} \\ \lfloor \sum_i T^+(i)/2 \rfloor - T^+(1) & e = e_{2,3} \\ 0 & \text{otherwise} \end{cases} \qquad x^3_e = \begin{cases} M & e \notin \Delta \\ \lceil \sum_i r^+(i)/2 \rceil - r^+(3) & e = e_{1,2} \\ \lfloor \sum_i r^+(i)/2 \rfloor - r^+(2) & e = e_{1,3} \\ \lceil \sum_i r^+(i)/2 \rceil - r^+(1) & e = e_{2,3} \\ 0 & \text{otherwise} \end{cases}$$

and f^3 is a feasible flow vector. Note that $r^+(2) > \lceil \sum_i r^+(i)/2 \rceil - r^+(1)$ implies that $r^+(1) + r^+(2) \geq r^+(3)$ so that $x^3, y^3 \geq 0$ and p^3 satisfies the feasibility conditions of Lemma 4.4.2. Since $x^3_{2,3} > 0$ and $p^1, p^3 \in F$, we can conclude that $\alpha_e = \bar{\beta}/(\lceil \sum_i r^+(i)/2 \rceil - r^+(1))$ for all $e \in \delta(2,3)$ and thus F is a facet of $\mathcal{P}^{\mathcal{X}}$.

On the other hand, when $r^+(2) = \min\{r^+(2), \lceil \sum_i r^+(i)/2 \rceil - r^+(1)\}$, in order to show that $\alpha_e = \bar{\beta}/r^+(2)$ for all $e \in \delta(2,3)$, it suffices to construct a point $p^4 = (x^4, y^4, f^4)$ where x^4 is identical to x^2 with the exception that $x^4_{e_{2,3}} = r^+(2)$, $x^4_{e_{1,2}} = 0$ and y^4 is identical to y^2 with the exception that $y^4_{e_{1,3}} = y^2_{e_{1,3}} + 1$, $y^4_{e_{2,3}} = y^2_{e_{2,3}} - 1$. ∎

In Theorems 4.4.11 and 4.4.12, we considered the case when $\Theta = \lceil \sum_i \bar{T}(i)/2 \rceil$ and $\sum_i T^+(i)$ is odd. If $\sum_i T^+(i)$ is even, then for any $p = (x, y, f) \in \mathcal{P}^{\mathcal{X}}$ we can bound $x(\Delta)$ from below by

$$x(\Delta) \geq \begin{cases} 0 & \text{if } k = 0 \\ r_{\text{med}} & \text{if } k = 1 \\ \left\lceil \frac{\lambda + \sum_i r^+(i)}{2} \right\rceil + \lambda(k-2) & \text{if } k \geq 2. \end{cases} \tag{4.34}$$

where $k = (\lceil \sum_i T^+(i)/2 \rceil - y(\Delta))^+$. Using (4.34), it is possible to develop valid inequalities of the form (4.30) , (4.31) or (4.32), but these inequalities are not facet defining.

4.5 Computational Results

In this section, we present the results of our computational experience with a cutting-plane algorithm. Here we remind the reader that the problem we are studying, CEP, is part of a much larger problem. Our objective in this study is to strengthen the formulation of CEP. The computational experiments test how well our inequalities perform in this regard. The separation routines described herein are not extremely sophisticated, although they appear to be reasonably fast.

4.5.1 The Cutting-Plane Algorithm

We developed an iterative algorithm which uses the facet defining inequalities as cutting-planes and includes them in the formulation whenever they are valid (and violated but not necessarily facet defining). The algorithm has three modules, one for each class of facets we presented in Sections 4.2 - 4.4. We used these modules in a hierarchical manner, and for a given iteration, executed a module only if no violated cuts are found by the previous modules. For each module, there is an upper limit on the number of cuts that can be introduced to the extended formulation in a single iteration. Limiting the number of cuts helps to keep the size of the LP reasonable. During the course of our study, we observed that it is better to use these modules in the following order: the cut-set module, the three-partition module and the flow-cut-set module.

When implementing the cutting plane algorithm, we used CPLEX, Version 2.1 as the linear programming optimizer and after obtaining the extended formulations, we applied CPLEX mixed integer optimizer to find optimal (integral) solutions.

We used two sets of real-life data, which arise, as described before, as part of ATM network design problems. The traffic demand matrices are fully dense and it is not practical to use the disaggregated formulation (i.e. defining a commodity for every source-destination pair) for these problems. As we explain later, we also made some modifications on the data to generate additional test problems while disturbing the underlying structure in a minimal way. The first data set is of a network with 15 nodes and 22 edges (see Figure 4.2). The traffic demands are fairly large when compared with the existing capacity on the edges and there is a cost related with flow variables as well as the capacity variables. The second network (see Figure 4.3) is much denser when compared with the first one and it has 16 nodes and 49 edges. In this data set, traffic demands are quite small and there is no existing capacity. Further, there is no cost related with the flow variables. In both of the test problems cost of adding capacity on an edge has a fixed component (related with the switches on both ends of the edge) and a variable component proportional to the actual length of the edge. The unit-batches correspond to so-called OC-3 facilities and and λ-batches correspond to OC-12 facilities and thus, λ is 4. The cost related with these facilities is such that cost of an OC-12 facility is more than the cost of one OC-3 facility but it is less than that of two OC-3 facilities and, therefore, in the optimal solution x variables are either 0 or 1. We included these bounds for the x variables in the original formulation but did not modify the valid inequalities using this information.

For each of the three modules of the algorithm, there is an exponential number of related facets and to implement the algorithm we need to find a practical way to choose violated inequalities, or, in other words, we need to find a way to solve the separation problem. The main purpose of our computational study is to see how closely we can approximate the the CEP polytope using the facet defining inequalities presented in the previous sections. Therefore, little effort was spent on the separation problem and it is likely that our cutting plane algorithm can be substantially speeded up by developing more efficient separation modules. As the networks related with the data sets are quite different and the valid inequalities basically depend on the underlying network, we postpone addressing the separation problem and look at the data sets more closely.

4.5.2 Data Set 1

For every strong subset of the node set, there two related cut-set facets and even when the number of nodes is small (15 in this case), there are potentially $2^{|V|}$ subsets to be considered. Notice that the number of strong subsets of a graph is closely related with the density of the graph and as seen in Figure 4.2, the network related with this instance is fairly sparse.

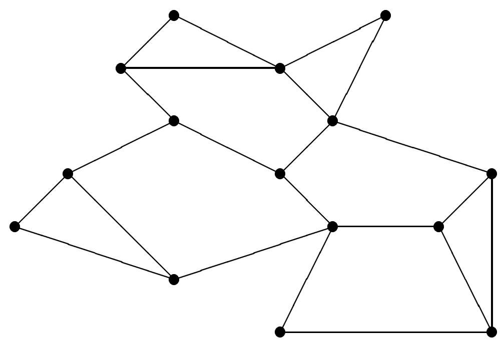

Figure 4.2: Network 1, $n = 15$ and $m = 22$

For this example network, there are only 190 strong subsets and it is feasible to check all of them to see if the related cut-set facets are violated or not. Recall that for a subset to qualify as a strong subset , both the subset and its complement have to be connected. For example, for a tree with n vertices and $n-1$ edges, only $2(n-1)$ of the 2^n subsets are strong subsets .

Similarly, the number of critical partitions of Network 1 is not very large (close to one thousand) and it is possible to check all of them in each iteration to see if they are violated or not. Lastly, we need to consider a number of flow-cut-set facets for each strong subset S, each commodity subset Q of S and each nonempty partition $\{E_1, E_2\}$ of $\delta(S)$. In our experiments we noticed that these cuts are more effective when (i) Q is "compact", i.e. small and connected, (ii) $|E_2|$ is small and (iii) edges in E_2 are "close" to Q, mostly when they are incident with nodes in Q. Using these observations, for each strong subset S, we generated sets Q such that $Q = \{v\}$ for all $v \in S$ and $Q = \{u, v\}$ for all $u, v \in S$ and $\{u, v\} \in E$. For choosing E_1 and E_2, we looked at the partitions that consist of no more than three edges in E_2.

We first run the algorithm without a time limit and generated an extended formulation by including all of the violated cuts in the original LP-relaxation. The optimal integral solution to this problem has a cost of 2231 and the lower bound generated by the extended LP is 2222, only 0.4% away from the optimum. This run took approximately 30 minutes on a SPARC10 - 40 machine and the statistics of this run are presented in Table 4.1. We define the "scaled gap" to be the difference between the value of the extended formulation and the optimal (integral) solution divided by the difference between the value of the LP-relaxation and the optimal (integral) solution.

As seen in Table 4.1, after 21 seconds the algorithm produces an extended formulation that reduces the scaled gap to under 3% and after one minute of run time the scaled gap is under 2%. After iteration 9 it takes almost half an hour to cut the scaled gap from 1.9% to 1.3%.

When we applied (CPLEX) branch and bound using the resulting extended formulation, the (integral) optimum was found in a few seconds. To balance the run-time between the cutting-plane algorithm and branch and bound, we next limited the use of flow-cut-set facets and stopped the algorithm after 70 seconds. After this modification, total run-time (i.e. generating the extended formulation and running branch and bound) is reduced to under two minutes. When we applied the branch and bound without any cuts, it took more than an hour to find the integral optimal solution.

Next, we modified the original data ('Cap1') to generate new problem instances and

iteration number	number of cuts	cut type	LP value	gap (%)	scaled gap(%)	time (sec)
0	0	-	1534	31.0	100	.42
1	45	c-s	1971	11.4	36.8	1.62
2	7	c-s	1998	10.0	32.3	2.08
3	37	3-p	2156	3.4	11.0	2.95
4	3	c-s	2156	3.4	11.0	4.17
5	7	3-p	2203	1.2	3.9	4.67
6	2	3-p	2204	1.2	3.9	5.45
7	57	f-c	2210	0.9	2.9	21.13
8	58	f-c	2215	0.7	2.3	51.32
9	56	f-c	2218	0.6	1.9	93.58
10	57	f-c	2220	0.5	1.6	239.93
11	30	f-c	2221	0.4	1.3	488.80
12	1	3-p	2221	0.4	1.3	490.62
13	24	f-c	2222	0.4	1.3	746.68
14	12	f-c	2222	0.4	1.3	1000.73
15	2	f-c	2222	0.4	1.3	1251.58
16	1	f-c	2222	0.4	1.3	1502.82
17	0	-	2222	0.4	1.3	1755.48

Table 4.1: Example run of the algorithm on Data Set 1 (no time limit).

to test the performance of the algorithm when applied to instances with different nature. Keeping the underlying network the same, we generated four more instances by changing the data as follows: Second data set is same as the first one, but the existing capacities are assumed to be zero, the third set is obtained by doubling the traffic demands and the last two sets are generated by respectively increasing and decreasing the flow costs. Tables 4.2 and 4.3 summarize the results of these runs.

We run the cutting-plane algorithm on a SPARC10 Model 40 and the branch and bound on a SPARC10 Model 51 (both using CPLEX 2.1) and the run-times are presented in Tables 4.2 and 4.3 are CPU-times on these machines. We note that for all of the test problems, the total CPU-time needed to find the optimal solution is under two minutes and the algorithm is not effected by the changes in the input as long as the underlying

problem	z(LP)	z(ELP)	z(IP)	gap(%)	sc. gap(%)	time(sec)
1: Cap1	1534	2218	2231	0.58	1.87	74
2: NoXcap1	4075	4576	4607	0.67	5.83	89
3: 2traf1	5608	6339	6354	0.23	2.01	74
4: NoFC1	949	1623	1631	0.49	1.17	83
5: 5FC1	2411	3105	3132	0.86	3.74	81

Table 4.2: Example problems generated using Data Set 1

problem	# of cuts	B&B time	Pure B&B time
1: Cap1	299	10sec	1hour
2: NoXcap1	336	15sec	12mins
3: 2traf1	124	11sec	hours
4: NoFC1	282	8sec	hours
5: 5FC1	307	9sec	30mins

Table 4.3: B&B times for Data Set 1

network stays the same. As seen in the Tables 4.2 and 4.3, when we apply branch and bound without any cutting-planes, the run-times vary from 12 minutes to several hours. For Problems 3 and 4 when we terminated the run after more than 3 hours of CPU-time, the branch and bound tree had more than 20,000 nodes and the gap between the upper and lower bounds was still large. This is very encouraging as it is a measure of how robust the algorithm is.

4.5.3 Data Set 2

As seen in Figure 4.3, the network related with this data set is dense and consequently the number of strong subsets is quite large. There are more than 25,000 strong subsets related with this network and although it is still feasible to consider all of the cut-set facets, it is not possible to do the same for all of the three-partition or flow-cut-set facets.

For this instance, we modified the algorithm and defined flags related with each strong subset . When executing the cut-set module, we marked a strong subset if the related cut-set inequalities are violated or when the slacks related with the cuts are less than 10% of the right hand side. Using these flags, we only considered the three-partitions which are formed by these subsets. Similarly, we only used the flow-cut-set facets related with the chosen subsets. The number of "important" strong subsets , selected as above, was under 100 and in terms of finding a good lower bound, they were as effective as the whole list. We also note that, in this case the flow-cut-set facets were not very effective as traffic demands are small and the flow costs are zero.

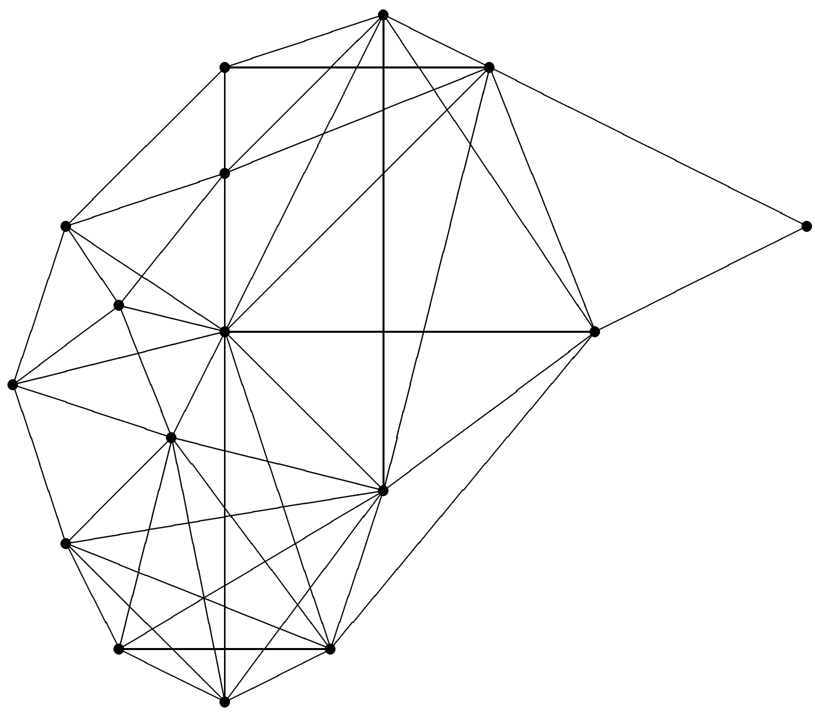

Figure 4.3: Network 2, $n = 16$ and $m = 49$

The LP-relaxation related with this data set has a value of 1,950 and the corresponding optimal integral solution has a value of 10,704 (as we learned later). The best lower bound we obtained by applying the cutting plane algorithm, with the modifications describes above, was 8,491. In other words, this lower bound is 20% less than the optimal value and the scaled gap is more than 25%. When we applied the branch and bound using the resulting extended formulation, the gap between the upper and lower bounds generated by CPLEX was more than 10% (after a few hours) and the program run out of memory (after many hours of CPU-time and) before it found an integral optimal. When we applied branch and bound using the LP-relaxation, the program was not able to find a reasonable lower bound before running out of memory. This is easy to see as the LP-relaxation is very loose.

We next studied the fractional optimal solution to the extended formulation and realized that the overall capacity added to the whole network (that is, $x(E)+y(E)$) was quite small. This is basically because the traffic demands are small. Even though the cut-set facets and the three-partition facets force the degree (i.e. $x(\delta(S)) + y(\delta(S))$) of a strong subset to be at least one, these cuts are myopic, and they do not force a lower bound on the overall capacity. Since there is no existing capacity for this problem and as the resulting network has to be connected, the optimal solution should add capacity on at least 15 (= # of nodes-1) edges so that the optimal solution would contain enough edges to form a connected network.

Using this observation, we then added a new module to the algorithm that checks whether or not some simple valid spanning tree cuts are satisfied by fractional solutions. In this module we have two kinds of valid inequalities. The first one of these can be obtained by shrinking a subset of nodes and requiring the resulting network to have enough edges to form a tree. The second one basically states that after deleting some edges (that is shrinking pairs of nodes) the solution to the design problem should have enough number of edges to form a spanning tree together with the deleted edges. We used the list of strong subsets for the first family of the spanning tree cuts and shrank the edges with $x_e + y_e > 1$ for the second one.

After including this module in the cutting plane algorithm, the lower bound generated by the extended formulation went up to 10,339 or, only 3.4% off the optimal solution. Using the resulting formulation, branch and bound was able to find an optimal solution, with the entire procedure taking under half an hour. To study the effect of this new module more closely, we ran the algorithm by disabling all other modules and the resulting lower bound was 9,071, or less than 10% away from the optimum. However, the resulting extended formulation was very ineffective for the branch and bound.

Lastly, we generated a larger extended formulation by first applying the algorithm and then setting some of the design variables to zero and then applying this procedure iteratively until we generate an integral solution. This way we generated many valid inequalities and using this formulation, branch and bound was able to find a solution more easily. This procedure took around 15 minutes, or reduced the run-time to a half. What this procedure essentially does is to imitate a branch-and-cut algorithm and form an extended formulation which includes some cuts that will help the branch and bound after some variables are set to zero.

We also generated two more problems related with this data set by increasing the traffic demands and by changing the objective function coefficients of the flow variables. In Tables 4.4 and 4.5, we summarize the statistics related with these of the problems.

problem	z(LP)	z(ELP)	z(IP)	gap(%)	sc. gap(%)	time(sec)
1: Cap2	1950	10339	10704	3.4	4.2	160
2: 2traf2	3901	10792	11789	8.5	12.7	175
3: 1FC2 (a)	4092	12779	14384	10.9	15.3	177
4: 1FC2 (b)	4092	13379	14384	7.0	9.8	5hrs

Table 4.4: Example problems generated using Data Set 2

In Table 4.4, problems 3 and 4 correspond to the same problem for different lengths of run-time.

As it can be seen in Tables 4.4 and 4.5, the algorithm is not as successful for the second and third problems (these are the ones we generated by modifying the original data).

For the second problem, the scaled gap is more than 10% after the first phase and

problem	# of cuts	B&B time	Pure B&B time
1: Cap2	247	15min	unsolved
2: 2traf2	258	3hrs	unsolved
3: 1FC2 (a)	508	10hrs*	unsolved

Table 4.5: B&B times for Data Set 2

the branch and bound takes just under three hours. When applying the algorithm to this data set, we limited the use of flow-cut-set facets (to keep the size of the extended-LP small). Since these inequalities play a more important role when the volume of traffic is high, this change results in a larger gap and thus a much longer branch and bound time. Nonetheless, in terms of application, we want to note that the solution time for this problem is reasonable.

For the third problem (1FC2 (a)), we should say that the extended formulation generated by the cutting-plane algorithm is not strong enough and we could not solve the problem to optimality using CPLEX (sequential) branch and bound. The run time and the optimal value reported in Table 4.5 were obtained by J. Eckstein by running his parallel branch-and-bound code CMMIP on a 64 processor CM-5 machine [9]. Starting with the extended formulation, the code took approximately 10 hours to solve this problem to optimality, generating a B-B tree with 2.4 million nodes. This negative result shows that the facet defining inequalities that we have presented in this chapter are not sufficient to solve hard problems (i.e. dense graphs, dense traffic matrices with flow costs) and more work needs to be done on the polyhedral structure of CEP.

4.5.4 Reconstructing Valid Inequalities

As a further test of the strength of our inequalities, we performed the following experiments. Suppose we have generated valid inequalities for a problem instance, and the demand data were to change in a small way. Then the inequalities would generally become invalid.

However, we can recompute the coefficients in the inequalities so that they become valid once again, in a small fraction of the time it took to compute the original inequalities. Note that the resulting inequalities are probably not facet-defining. Nevertheless, how strong are they? This question has great practical significance, since we will usually solve many problems that differ slightly from one another in the demand amounts. To test this, for selected problems we (a) generated an extended formulation as described above, and then (b) randomly perturbed each demand by 10 % and 20 %. Table 4.6 describes the results of these tests. Here LP is the LP-relaxation of the perturbed problem, RELP is

	Data Set 2		Data Set 1	
Perturbation	10%	20%	10%	20%
z(LP)	1955.83	1967.58	1423.44	1401.54
z(RELP)	10315.51	10316.61	2118.73	2112.92
z(ELP)	10315.51	10321.59	2160.97	2157.63
z(IP)	10704.00	10704.00	2182.37	2164.57
B&B time	430sec	382sec	9sec	9sec
Gap	3.6%	3.6%	2.9%	2.4%
Sc. Gap	4.4%	4.4%	8.3%	6.8%

Table 4.6: Perturb & Reconstruct

the *reconstructed* extended formulation, ELP is the extended formulation for the perturbed problem (obtained in the normal way) and IP is the perturbed mixed-integer program. As we can see, the strategy of recomputing cuts appears quite effective. In a certain sense, this shows that our inequalities are "stable" and more "combinatorial" than driven by the demand amounts.

4.6 Concluding Remarks

There are several areas that we plan to explore in the future. The cutsets we described above involve families of subsets of nodes. Roughly speaking, our algorithms maintained a list of "active" subsets. It is easy to decide when a subset is no longer active, but all the approaches we can think of for generating new active sets involve problems similar to the maximum-cut problem.

Another issue is that of generating strong inequalities involving partitions of the node set into more than three classes. Early work on our part appears to show that the structure of the "better" facets is quite complex (they strongly depend on the demand amounts – one can easily generate interesting-looking combinatorial facets that never come into play). Instead, we are developing an approach for automatically computing $face$-defining violated inequalities. Roughly, this approach involves recursively solving problems of type CEP that have a simpler structure.

A simple change to our formulation is that of replacing each edge by three parallel edges, one for existing capacity, one for x-capacity and one for y-capacity, and similarly splitting the flows in the edge into a sum of three values. This will merely increase the number of continuous variables by a factor of three, but the benefit is that we will have a richer family of "flow cut-set" inequalities. As a preliminary step in this direction, we are improving our separation procedure for these inequalities. We note that there are other ways of tightening the split formulation.

A different kind of reformulation involves using path variables instead of flow variables. However, the integral variables remain the same, and potentially the resulting problem is just as difficult as the original one (although there are more ways of strengthening the path formulation).

References

[1] R. Aharoni, P. Erdös and N. Linial, Dual Integer Linear Programs and the Relationship Between Their Optima, *Proc. 17'th ACM Symp. on Th. of Comp.* (1985) 476-483.

[2] A. Balakrishnan, T.L. Magnanti and R. Wong, A dual-ascent procedure for large-scale uncapacitated network design, *Oper. Research* **37**, 716 - 740 (1989).

[3] D. Bienstock and O. Günlük, A Degree Sequence Problem Related to Network Design, *Networks*, **24**, (1994), 195-205.

[4] D. Bienstock and O. Günlük, Computational experience with a difficult multicommodity flow problem, *Math. Programming*, **68**, (1995), 213-238.

[5] D. Bienstock and O. Günlük, Capacitated Network Design - Polyhedral Structure and Computation, *ORSA J. on Computing*, in process (1995).

[6] S. Ceria, personal communication.

[7] S. Chopra, Polyhedra of the equivalent subgraph and some edge connectivity problems, *SIAM J. Disc. Math.* **5**, 321-337 (1992).

[8] W. Cook, personal communication.

[9] J. Eckstein, personal communication.

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

[11] M.D. Grigoriadis and L.G. Khachiyan, Fast approximation schemes for convex programs with many blocks and coupling constrains, Report DCS-TR-273, Dept. of Comp. Sci, Rutgers Univ. (1991).

[12] M. Grötschel, L. Lovász and A. Schrijver, The Ellipsoid Method and Its Consequences in Combinatorial Optimization, *Combinatorica 1* (1980) 169-197.

[13] M. Grötschel and C.L. Monma, Integer polyhedra associated with certain network design problems with connectivity constraints, *SIAM J. Disc. Math.* **3**, 502-523 (1990).

[14] M. Grötschel and W.R. Pulleyblank, Weakly Bipartite Graphs and the Max-Cut Problem, *O.R. Letters 1* (1981) 23-27.

[15] M. Iri, On an extension of the max-flow min-cut theorem to multicommodity flows, *Journal of the Oper. Res. Soc. of Japan* **13**, 129-135 (1971).

[16] J.-F. Labourdette, Rearrangeability techniques for multihop lightwave networks and application to distributed ATM switching systems, Tech. Report CU/CTR/TR 244-91-25, Center for Telecommunications Research, Columbia University (1991).

[17] J.-F. Labourdette, and A.S. Acampora, Partially reconfigurable multihop lightwave networks, *IEEE Trans. Comm* **39**, 1223-1230 (1991).

[18] J.-F. Labourdette and A.S. Acampora, Reconfiguration Algorithms for Rearrangeable Lightwave Networks, *Proc. Infocom 92,* to appear (1992).

[19] F.T. Leighton, F. Makedon, S. Plotkin, C. Stein, E. Tardos and S. Tragoudas, Fast approximation algorithms for multicommodity flow problems, in *Proc. 23rd ACM Symp. on Theory of Computing*, 101-111 (May 1991).

[20] T. Leong, P. Shor and C. Stein, Implementation of a combinatorial multicommodity flow algorithms, in *Proc. DIMACS Implementation Challenge Workshop: Network Flows and Matching* (October 1991).

[21] I. Lustig, personal communication.

[22] T.L. Magnanti and P. Mirchandani, Shortest paths, single origin-destination network design and associated polyhedra, *Networks* **23**, 103-121 (1993).

[23] T.L. Magnanti, P. Mirchandani and R. Vachani, The convex hull of two core capacitated network design problems, *Math. Programming* **60**, 233-250 (1993).

[24] G.L. Nemhauser and L.A. Wolsey, *Integer and Combinatorial Optimization*, Wiley, New York (1988).

[25] M.W. Padberg, T.J. Van Roy and L.A. Wolsey, Valid linear inequalities for fixed charge problems, *Operations Research* **33**, 842-861 (1985).

[26] Y. Pochet and L.A. Wolsey, Network design with divisible capacities: Aggregated flow and knapsack subproblems, *Proceedings IPCO2*, 150-164, Pittsburgh, (1992) .

[27] Y. Pochet and L.A. Wolsey, Lot-sizing with constant batches: Formulation and valid inequalities, *Math. of Oper. Res.*,**18**, 767-785 (1993).

[28] R.L. Rardin and L.A. Wolsey, Valid inequalities and projecting the multicommodity extended formulation for uncapacitated fixed charge network flow problems, manuscript (1991).

[29] F. Shahrokhi and D.W. Matula, The maximum concurrent flow problem, *Journal of the ACM* **37**, 318-334 (1990).

[30] M. Stoer and G. Dahl, A Polyhedral Approach for Multicommodity Survivable Network Design, ZIB-Berlin (1993).

[31] T.J. Van Roy and L.A. Wolsey, Valid inequalities and separation for uncapacitated fixed charge networks, *Operations Research Letters* **4**, 105-112 (1985).

[32] L.A. Wolsey, personal communication.