Edge Splitting-off and Network Design Problems

YUNG, Chun Kong

A Thesis Submitted in Partial Fulfilment

of the Requirements for the Degree of

Master of Philosophy

in

Computer Science and Engineering

The Chinese University of Hong Kong

September 2009

Abstract

Edge Splitting-off and Network Design Problems

YUNG Chun Kong

Master of Philosophy

Department of Computer Science and Engineering

The Chinese University of Hong Kong

2009

The work of this thesis is motivated by the *degree bounded network design problem*, which is a topic of much research due to both its theoretical interest and practical applications. The key technical tool we apply is a simple but powerful operation, known as *edge splitting-off*. Apart from network design, this technique is also widely-used in some other edge-connectivity problems. By extending and improving some edge splitting-off results, we obtain new approximation results for several network design problems.

Given a complete undirected graph, a cost function on edges and a degree upper bound B, the *degree bounded network design problem* is to find a minimum cost subgraph with maximum degree B satisfying given connectivity requirements. This problem captures some key problems in combinatorial optimization and it also has practical applications in various areas as well. However, this problem does not admit any polynomial factor approximation algorithm, since the feasibility problem is NP-Hard even for simple connectivity requirement such as finding a spanning tree. In this thesis, we show that when the cost function satisfies triangle inequalities, there are constant factor approximation algorithms for various degree bounded network design problems. These approximation algorithms return solutions with smallest possible maximum degree, and the cost guar-

antee is obtained by comparing to the optimal cost when there are no degree constraints. This demonstrates that degree constraints can be incorporated well into network design problems with metric costs.

The main technical tool in these algorithms is a combinatorial operation, known as *edge splitting-off*. Splitting-off a pair of edges (xu, xv) means deleting these two edges and adding a new edge uv. Lovász and Mader proved that edge splitting-off can be performed to preserve edge-connectivities of a graph. Apart from network design, these splitting-off theorems have applications in various graph problems, including connectivity augmentation, tree packing and graph orientation. Efficient splitting-off algorithms have been developed to give fast algorithms for the above problems. Also, splitting-off theorems have been extended to find new applications in connectivity augmentation problems and network design problems. Most of the efficient algorithms and extensions are developed only in the global edge-connectivity setting, although there are important applications in the more general local edge-connectivity setting.

Extending the techniques we developed, we obtain new edge splitting-off results in the local edge-connectivity setting. First, we develop efficient edge splitting-off algorithms that improve the time complexity of the best known algorithm by a factor of $\tilde{\Omega}(n)$. These algorithms are conceptually very simple and can be extended to different settings. Moreover, we also extend edge splitting-off to incorporate some additional constraints and apply these results to give additive approximation algorithms for several constrained connectivity augmentation problems.

摘要

Edge Splitting-off and Network Design Problems

邊分離及網絡設計問題

翁振剛

香港中文大學

計算機科學與工程學系

哲學碩士

二零零九

本文的主要動機乃針對度約束網絡規劃問題 (degree bounded network design problem) 的研究。由於它在理論上及應用上兩方面的重要性，這題目一直是眾多研究的主題。文中所應用的核心工具是一項名為邊分離 (edge splitting-off) 的技術。這項簡單而強大的工具不單用於網絡設計，同時亦廣泛應用於各種邊連通性 (edge-connectivity) 的問題上。透過擴展及改進現有的邊分離技術，我們在數個網絡設計上提出了新的逼近算法。

在既定的邊成本函數 (edge cost function) 及度上限 (degree upper bound) 下，度約束網絡規劃問題乃是在一個完整的無向圖 (complete undirected graph) 中求取一個合乎特定連通性要求 (connectivity requirement) 的最低成本子圖 (subgraph)。這問題不單包含了數個在組合優化 (combinatorial optimization) 中的重要問題，同時亦可實際應用於在多個領域之中。然而，這問題所對應的可行性問題在簡單的連接要求下已是 NP-難。故此，它没有任何多項式係數近似解 (polynomial factor approximate solution)。本文提出了在邊成本函數滿足三角形不等式 (triagnle inequality) 的前提下用以解決度約束網絡規劃問題的常數因子近似算法 (constant factor approximation

algorithm)。這些算法所求得的解擁有的最大度 (maximum degree) 乃是所有可能情況下的最小值。而且，這些算法的近似比 (approximation ratio) 是和這問題在没有度上限的情況所求得的解作比較。這顯示出在邊成本函數滿足三角形不等式的情況下，網絡設計可將度上限納入考慮之內。

文中所應用的主要工具是一項名為邊分離的組合技術 (combinatorial technique)。分離一對邊 (xu, xv) 即是刪去這兩條邊並增加一條新的邊 uv。Lovász 和 Mader 證明了在進行邊分離時，邊連通性可以被保存。除了網絡設計以外，邊分離還可應用於不同的圖問題，包括了連通擴充 (connectivity augmentation) 、樹壓縮 (tree packing) 及圖定向 (graph orientation)。故此，一些研究提出了邊分離的高效算法。這為上述問題提供了一些高效算法。另一方面，一些研究對邊分離作出各種擴展，並應用於連通擴充及網絡設計等問題上。大部份高效算法及擴展皆是集中於全域邊連通性的設定 (global edge-connectivity)。然而，不少重要的應用卻是在局部邊連通性的設定 (local edge-connectivity) 下。

透過擴展我們先前所研究的技術，我們得到一些關於邊分離在局部邊連通性的設定下的成果。首先，我們提出了一些比已知的算法快上 $\tilde{\Omega}(n)$ 倍的高效算法。這些算法不單概念上非常簡單，並且可以用於其他設定之下。此外，我們亦擴展了邊分離以納入其他限制在考慮之內。而這些成果可用於一些約束連通擴充問題 (constrained connectivity augmentation problems) 的近似算法之中。

Acknowledgements

I am greatly indebted to all the people who have helped me through the years of work that has led to this thesis.

My advisor, Professor Lap Chi LAU is the first among the people who have helped and inspired me during my master study. I would like to express my sincerest and deepest gratitude to Professor LAU, for his endless support, constant motivation and invaluable advice. His guidance helped me in all the time of research and writing of this thesis, from introducing me to this area of combinatorial optimization, to going through this voluminous work repeatedly. I do not believe the work could have been accomplished without his help.

I would also like to show my heartfeld appreciation to all the professors and students in theory group for the fruitful discussions and enjoyable environment. In particualr, I must sincerely thank Professor Leizhen CAI, who spent plenty of time on my work in the years; Tom, who provided important assistance in the writing of this thesis; Isaac and Jesse, who raised numerous interesting questions on different topics.

I am much indebted to the researchers contributed in the topics covered in this thesis. My special thanks goes to Professor András Frank, for his effort on deriving clear and beautiful proofs in edge splitting-off. His work gave me crucial insights into this technique, which is the basis of this thesis.

Finally, I am immensely grateful to my family for their unconditional love, continuing care, tremendous support, and for everything they did for me. I dedicate this thesis to my mother Fung Ying YU.

Contents

Chapter 1

Overview

In this thesis, we present approximation algorithms for network design problem by using edge splitting-off technique. Here we give an outline of the whole thesis. More comprehensive introductions of each topic are given in the corresponding chapters.

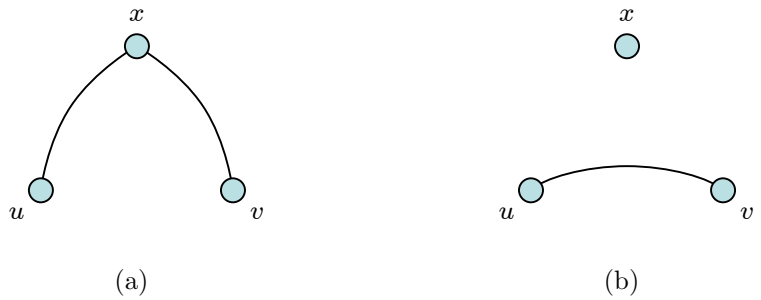
Network design problem is a central topic in combinatorial optimization and approximation algorithms, and it has various applications in computer network and VLSI design. Given connectivity requirements $r(u, v)$, the objective is to find a minimum cost subgraph that has $r(u, v)$ disjoint paths between each two vertices u and v. This is a general problem which captures a number of classical problems as special cases. One example is the minimum spanning tree problem, where the connectivity requirement is one for every pair of vertices. Another example is the minimum Steiner tree problem, where connectivity requirement is one for every pair of terminal vertices. The connectivity setting is said to be *global* in the former example because the requirements are the same throughout the graph; and it is said to be *local* in the latter example because the requirements vary for different vertex pairs. In many applications, the maximum connectivity requirement $r_{\max}$ is very small, compared to the size of graph.

Connectivity augmentation problem is a well-studied special case of the network design

problem. The goal of this problem is to add a minimum number of edges to a given graph so that the resulting graph satisfies the given edge-connectivity requirements. This special case is of particular interest because it can be solved optimally in polynomial-time. In contrast, the general network design problem is NP-hard to solve. The research on network design problem has hence focused on obtaining approximation algorithms.

In recent years, much effort has been put on *constrained network design problem*, a more general class of network design problems that incorporate additional constraints. A well-studied case is the *degree bounded network design problem*. The objective is to find a minimum cost subgraph satisfying the given edge-connectivity requirements as well as an upper bound on the maximum vertex degree. On one hand, this problem generalizes some key problems in combinatorial optimization, including the travelling salesman problem. On another hand, it also has practical applications where the degree constraint acts as a tool to bound the workload of nodes. However, this problem captures the travelling salesman problem as a special case, thus it does not admit any non-trivial approximation algorithm in general. Recent research has thus focused on obtaining bicriteria approximation algorithms, which approximates the cost of subgraph but the vertex degree bound may be slightly violated.

In some network design problems when the cost function satisfies triangle inequalities (metric costs), much stronger algorithmic results are known. For the travelling salesman problem, although there is no non-trivial approximation algorithms in general, it is well-known that there is a 1.5-approximation algorithm assuming triangle inequalities. This motivates us to study the more general *degree bounded network design problems with metric costs*. The assumption of metric costs actually arises from in many applications of network design, such as the design of telecommunication networks. In this thesis, we generalize the result in travelling salesman problem to give the first constant



(a) (b)

Figure 1.1: When splitting-off an edge pair (xu, xv), the two edges xu and xv in (a) are replaced by a new edge uv in (b).

factor approximation algorithms, showing that degree constraints can be incorporated into network design problems with metric costs.

> *There are polynomial-time constant factor approximation algorithms for the degree bounded network design problem with metric cost.*

The main technical tool used is a combinatorial operation, known as *edge splitting-off*. Splitting-off a pair of edges $(e = ux, f = xv)$ at a designated vertex x means deleting these two edges and adding a new edge uv (Figure 1.1). This short-cutting operation does not increase the total edge cost under the assumption of triangle inequality. A fundamental result in graph theory shows that edge splitting-off can be performed to preserve pairwise edge-connectivities, if addition of parallel edges is allowed. Under appropriate conditions, we extend this result to show that edge splitting-off can be performed to preserve pairwise edge-connectivities, even if addition of parallel edges is forbidden. By using this stronger result, edge splitting-off can be performed repeatedly to decrease vertex degree to satisfy the given upper bound. Details of these approximation algorithms and the edge splitting-off result are presented in Chapter 3.

Edge splitting-off is a simple but powerful operation. It plays an important role in many graph problems, both in proving theorems and obtaining polynomial time algorithms. Apart from network design problems, it is also used crucially in tree packing

problems, edge-connectivity augmentation problems and graph orientation problems. Efficient splitting-off algorithms have been developed to give fast algorithms for the above problems. Also, splitting-off theorems have been extended to incorporate additional constraints and found new applications in connectivity augmentation problems and network design problems. Most of the efficient algorithms and extensions were developed only in the global edge-connectivity setting, although there are important applications in the local edge-connectivity setting.

Extending the techniques we developed, we obtain some new edge splitting-off results in the local edge-connectivity setting. First we prove a general structural property about edge splitting-off, showing that most edge pairs are splittable for preserving pairwise edge-connectivities. This extends the classical result which guarantees the existence of splittable edge pairs.

> *When the degree of designated vertex is sufficiently large, most of the edge pairs are splittable for preserving pairwise edge-connectivities.*

This structural property can be applied in proofs of new splitting-off theorems, as well as in design of efficient splitting-off algorithms. In *constrained edge splitting-off*, edge splitting-off is extended to incorporate some additional constraints. The structural property states that large proportion of edge pair is splittable, and hence naturally implies that some of them satisfy the additional constraints as well. These edge splitting-off results can be applied to obtain additive approximation algorithms for several *constrained edge-connectivity augmentation problems.* Details of the structural property and the approximation algorithms are presented in Chapter 4 and Chapter 5.

> *There are polynomial-time additive approximation algorithms for several constrained edge-connectivity augmentation problems.*

Finally we present efficient algorithms to split-off all edges incident to a designated vertex. In the local edge-connectivity setting, we develop a faster deterministic algorithm that improves the time complexity of the best known algorithm by a factor of $\tilde{\Omega}(n)$. Furthermore, when the degree of designated vertex is large, the general structural property implies that a random edge pair is splittable with high probability. Using this observation, we can design a faster randomized algorithm that simultaneously split-off many edge pairs at random when the degree of designated vertex is large.

> *There is a deterministic* $\tilde{O}(m + {r_{\max}}^2 \cdot n^2)$*-time algorithm and a randomized* $\tilde{O}(m + {r_{\max}}^3 \cdot n)$*-time algorithm for edge splitting-off problem in the local edge-connectivity setting.*

These algorithms are conceptually very simple. The efficiency is based on new structural results of edge splitting-off and fast edge-connectivity algorithms by Bhalgat et.al. The details are presented in Chapter 6. In the following chapter, we first review some background materials and previous works, before presenting the main materials in Chapter 3 to Chapter 6.

Chapter 2

Background

In this chapter, we will present some background materials regarding edge-connectivity and edge splitting-off. We start by giving some basic notations and fundamental results for graphs and edge-connectivity in Section 2.1; and for edge splitting-off in Section 2.2. Then we will see applications of edge splitting-off by introducing the two edge-connectivity problems we consider in this thesis in Section 2.3. Finally, we will discuss some previous work in efficient edge splitting-off algorithms in Section 2.4.

2.1 Graphs and Edge-connectivity

In this section, we will first cover some basic notations for graphs. Then, we will present the two well-known definitions of edge-connectivity in Section 2.1.2; and show their equivalence by introducing the famous Menger's theorem in Section 2.1.3.

An *undirected graph* $G = (V, E)$ consists of an n-element set of *vertices* $V = V(G)$ and an m-element set *edges* $E = E(G)$ that each edge is an unordered pair of vertices. An edge $e \in E$ is incident at a vertex $v \in V$ if $v \in e$; such an edge is called a v-edge. The two vertices an edge is incident at are called its *end-vertices*; and we say the edge *connects* its end-vertices. If two edges connect the same pair of vertices, we call those

edges *parallel edges*. A graph without parallel edges is called a *simple graph*. If t parallel edges are represented as t separate edges (appear t distinct times), then the graph is called an *unweighted graph*; if it is represented once, along with its multiplicity t, then the graph is called a *weighted graph*.

A *directed graph* $D = (V, A)$ consists of an n-element set of *vertices* $V = V(D)$ and an m-element set *arcs* $A = A(D)$ that each arc is an ordered pair of vertices. We say that an arc uv *leaves* u; *enters* v; and *goes* from u *to* v. For an arc uv, the first vertex u is its *tail* and the second vertex v is its *head*. Two arcs are in parallel if they have the same head and tail. In the following, we will define some basic notations related to graphs.

Neighbourhood: In an undirected graph $G = (V, E)$, two vertex u, v of G are *adjacent* or *neighbours* of each other if uv is an edge of G. The *neighbour set* (*neighbourhood*) $N_G(v)$ of a vertex v in G is the set of vertices adjacent to v in G. We call a vertex $v \in V(G)$ a *x-neighbour* if $v \in N_G(x)$. The neighbour set in a directed graph $D = (V, A)$ is defined in the same manner. A vertex u is an *in-neighbour* of v in D, or equivalently v is an *out-neighbour* of u in D if uv is an arc of D. This defines the *in-neighbour set* $N_D^+(v)$ and *out-neighbour set* $N_D^-(v)$ of a vertex v in D. We call a vertex $v \in V(G)$ an *x-in-neighbour* if $v \in N_D^+(x)$ and an *x-out-neighbour* if $v \in N_D^-(x)$.

Degree: In an undirected graph $G = (V, E)$, the *degree* $d_G(u, v)$ between two vertices u, v in G is the number of edges $uv \in E$. Generalizing this idea, we define the edge set $\delta_G(X, Y)$ connecting vertex sets X and Y in G as

$$\delta_G(X, Y) := \{uv \in E(G) : |X \cap \{u, v\}| = |Y \cap \{u, v\}| = 1\}.$$

In other words, $\delta_G(X, Y)$ consists of edges with one end-vertex in $X - Y$ and the other in $Y - X$. The *degree* $d_G(X, Y) := |\delta_G(X, Y)|$ of vertex sets X and Y is defined to be the number of edges connecting the two sets in G.

Similarly, the arc sets $\delta_D^-(X,Y)$ going from X to Y; and $\delta_D^+(X,Y)$ going from Y to X in a directed graph $D=(V,A)$ are defined as

$$\delta_D^-(X,Y) := \{uv \in A(D) : u \in X, v \in Y\}.$$

$$\delta_D^+(X,Y) := \{uv \in A(D) : v \in X, u \in Y\}.$$

The *out-degree* $d_D^-(X,Y)$ and *in-degree* $d_D^+(X,Y)$ of X from Y in D are defined to be $|\delta_D^-(X,Y)|$ and $|\delta_D^+(X,Y)|$ respectively. The arc set $\delta_D(X,Y) = \delta_D^+(X,Y) \cup \delta_D^-(X,Y)$ between X and Y consists of the arcs with one end-vertex in X and the other end-vertex in Y in D. So the degree $d_D(X,Y)$ between X and Y is defined to be $|\delta_D(X,Y)|$.

Without ambiguity, we will not distinguish between a one-vertex set $\{v\}$ and its element v; and we denote $f(Z, V-Z)$, $Z \subseteq V$, by $f(Z)$ for any function f regarding two vertex subsets $X, Y \subseteq V$. This simplifies the representation of degree function we just defined (and other set functions we will define as well).

2.1.1 Subgraphs

Sometimes, we may want to study only part of a given graph. We say that a graph $H=(U,F)$ is a *subgraph* of a graph $G=(V,E)$ if $U \subseteq V$ and $F \subseteq E$. The graph $H=(U,F)$ is an *induced subgraph* of $G=(V,E)$ on vertex subset $V' \subseteq V$ if $U=V'$ and $F=\{uv \in E : u,v \in V'\}$; and H is an induced subgraph of G on edge subset $E' \subseteq E$ if $F=E'$ and $U=\{v \in V : v \in e \text{ for some } e \in E'\}$. A graph is called a *complete graph* if the corresponding edge (arc) uv exists for every two vertices u and v. And conversely, a graph is called an *empty graph* if there is no edge (arc) in the graph.

To obtain a smaller graph from a given graph, there are two widely-used operations. Note that the previous notations and the following operations are defined for both undirected and directed graphs.

Deletion: Given a graph $G=(V,E)$, *deleting a subset of edges* $F \subseteq E$ means removing

F from the edge set while keeping vertex set unchanged. The resulting subgraph $H = (V, E - F)$ is denoted by $G - F$. On another hand, *deleting a subset of vertices* $W \subseteq V$ means removing W from the vertex set; together with the removal of the edge set F, which consists of edges that are incident to some vertices of W, from the edge set. The resulting subgraph $H = (V - W, E - F)$ is denoted by $G - W$.

Contraction: Given a graph $G = (V, E)$, *contracting a subset of vertices* $W \subseteq V$ returns a graph H arising from G by adding a new vertex v_W to $G - W$; and adding $d_G(v, W)$ parallel edges between v and v_W in H for every $v \in V(G) - W$. The resulting graph is denoted by G/W. For a vertex subset $Z \subseteq V(G)$ for which either $Z \subseteq V(G) - W$ or $W \subseteq Z \subseteq V$, its *corresponding vertex subset* in G' is $\{v : v \in Z\}$ or $\{v : v \in Z - W\} + v_W$ respectively.

2.1.2 Cut and Edge-Connectivity

First of all, we say two vertices are *connected* in an undirected graph if they are linked by a *path*, where a *path* is defined to be a sequence of distinct vertices $P = \{v_0, v_1, \ldots, v_k\}$ such that $v_i v_{i+1} \in E(G)$ for all $0 \leq i < k$. In a directed graph, a *directed path* (or path for short) from vertices v_0 to v_k is defined to be a to a sequence of distinct vertices $P = \{v_0, v_1, \ldots, v_k\}$ such that $v_i v_{i+1} \in A(D)$ for all $0 \leq i < k$. Two vertices are *strongly connected* (or connected for short) in a directed graph if there exist a path from u to v and a path from v to u. To measure how well two vertices are connected, we use the notations of *edge-connectivity*. Here, we will present the two well-known definitions for edge-connectivity. These two definitions are equivalent, as we will see in Section 2.1.3.

Number of edge-disjoint paths: An intuitive definition of *edge-connectivity* is based on the size of the connection. In an undirected graph, two vertices u and v are *k-edge-connected* if they are linked by k *edge-disjoint paths*, i.e. k paths that do not share an edge.

The *local edge-connectivity* (or edge-connectivity for short) $\lambda_G(u, v)$ of two vertices u and v in G is defined to be the largest integer k for which u and v are k-edge-connected. The definitions are similar in a directed graph that two vertices u and v are *k-arc-connected* if there exist k *arc-disjoint paths* from u to v and k *arc-disjoint paths* from v to u. The *local arc-connectivity* (or arc-connectivity for short) $\lambda_D(u, v)$ of two vertices u and v is hence defined to be the largest integer k for which u and v are k-arc-connected.

Cut size: The second definition of edge-connectivity is based on the robustness of the connection. Two vertices are *k-edge-connected* in an undirected graph G if they are connected in $G - F$ for every edge subset F with $|F| < k$. Similarly, two vertices are *k-arc-connected* in a directed graph D if they are (strongly) connected in $D - F$ for every arc subset F with $|F| < k$. This gives rise to the notation of *cut*. We define a *cut* $(C, V - C)$ of a graph to be a bipartition of the vertex set into two sides C and $V - C$; and the *cut size* of a cut C to be the number of edges connecting C and $V - C$, i.e. $d(C)$. A cut C *separates* two vertices u and v if $u \in C, v \notin C$; and C is called a $u\bar{v}$ set or a u-v cut. We call an edge e a *cut-edge* if there exists a cut consists of e only. Now, we can use the cut size as a measure of robustness of the connection --- we say that two vertices are k-edge-connected in an undirected graph G if the size of any cut separating them are at least k.

A vertex subset $U \in V(G)$ is said to be k-edge-connected if u and v are k-edge-connected for every pair of vertices $u, v \in U$; if $U = V(G)$, we say that the undirected graph G is k-edge-connected. The *global edge-connectivity* (or edge-connectivity for short) $\lambda(G)$ of a graph G is the largest integer k for which the graph is k-edge-connected. Conversely, the *local edge-connectivity* of a graph is the collection of all pairwise local edge-connectivity in the graph. We define $\lambda_{\max} := \max_{u,v \in V(G)} \{\lambda_G(u, v)\}$ to be the maximum edge-connectivity among all vertex pairs. The definition of k-arc-connected

for a vertex subset, *global arc-connectivity* and *local arc-connectivity* in a directed graph follow in the same manner.

2.1.3 Menger's Theorem

Menger [57] proved a min-max relation regarding edge-connectivity --- the maximum number of edge-disjoint paths linking two vertices equals the size of a minimum cut separating them. In other words, the two definition of edge-connectivity covered in Section 2.1.2 are actually equivalent.

Theorem 2.1. ***(Menger's theorem [57])*** *Let $D = (V, A)$ be a directed graph, and $s, t \in V(D)$ be distinct vertices. There are k arc-disjoint paths from s to t if and only if*

$$d_D^+(X) \geq k \text{ for every } s\bar{t} \text{ set } X \subseteq V(D).$$

This min-max relation is one of the cornerstones of graph theory. From this directed version of Menger's theorem, one can obtain the following results.

Proposition 2.2. *Let $D = (V, A)$ be a directed graph, $S \subseteq V(D)$ be a vertex subset and k be a positive integer. The following statements are equivalent:*

(i) There are k arc-disjoint paths from any vertex of S to any other vertex of S.

(ii) $d_D^+(X) \geq k$ for every vertex subset $X \subset V(D)$ separating S.

(iii) S remains strongly connected in D upon removal of $k - 1$ arcs.

One can derive the undirected version of Menger's theorem from the directed version easily. This gives the following results.

Theorem 2.3. ***(Menger's theorem [57])*** *Let $G = (V, E)$ be an undirected graph, and $s, t \in V(G)$ be distinct vertices. There are k edge-disjoint paths linking s and t if and*

only if

$$d_G(X) \geq k \text{ for every } s\bar{t} \text{ set } X \subseteq V(G).$$

Proposition 2.4. *Let $G = (V, E)$ be an undirected graph, $S \subseteq V(G)$ be a vertex subset and k be a positive integer. The following statements are equivalent:*

(i) There are k edge-disjoint paths linking any two vertices of S.

(ii) $d_G(X) \geq k$ for every vertex subset $X \subset V(G)$ separating S.

(iii) S remains connected in G upon removal of $k-1$ edges.

2.2 Edge Splitting-off

In this section, we will first introduce the edge splitting-off operation and the famous results given by Lovász and Mader in undirected graphs. Then, we will present Frank's proof on Mader's result in Section 2.2.1 and Section 2.2.2. This proof includes the main tools used in proving edge splitting-off results and is hence a good starting point for understanding edge splitting-off. Finally, we will discuss edge splitting-off in directed graphs in Section 2.2.3, and extensions of this technique, the so-called constrained edge splitting-off, in Section 2.2.4.

Edge splitting-off is a simple but powerful operation introduced by Lovász. It plays an important role in many basic graph problems, both in proving theorems and obtaining polynomial time algorithms. In an undirected graph $G = (V + x, E)$, splitting-off a pair of edges $e = ux, f = xv$ means deleting these two edges and adding a new edge uv if $u \neq v$. The resulting graph is denoted by G^{ef}. Clearly, edge-connectivity never increases after splitting-off an edge pair. On another hand, Lovász [54, 55] proved that splitting-off some specific edge pairs does not decrease the edge-connectivity in the graph. More

specifically, he showed the existence of an edge pair that keeps the graph k-edge-connected upon splitting-off the pair.

Theorem 2.5. ***(Lovász' undirected splitting-off lemma [54, 55])*** *Let $G = (V + x, E)$ be an undirected graph with $d(x)$ even, and let $k \geq 2$ be an integer that*

$$\lambda_G(w, t) \geq k \text{ for every pair of vertices } w, t \in V.$$

Then there is an edge $f = xv$ for every edge $e = xu$ so that $\lambda_{G^{ef}}(w, t) \geq k$ holds for every pair of vertices $w, t \in V$.

After splitting-off an edge pair specified in Lovász' splitting-off lemma, the conditions in Theorem 2.5 still hold. Therefore, one can apply Lovász' splitting-off lemma repeatedly until all edges incident to x are splitted-off. We call the operation of splitting-off all x-edges a *complete edge splitting-off* at x.

Lovász proved that edge splitting-off operation can preserve global edge-connectivity of a graph. Mader [56] later generalized Lovász's result by showing that the local edge-connectivity (pairwise edge-connectivity between each vertex pair) can also be preserved by edge splitting-off. That is, $\lambda_G(w, t) = \lambda_{G^{ef}}(w, t)$ holds for every $w, t \in V$ after splitting-off an edge pair $(e = xu, f = xv)$. An edge pair satisfying this condition is said to be *admissible*. For an admissible edge pair (e, f), we say that e and f are *admissible partners* of each other.

Theorem 2.6. ***(Mader's theorem [56])*** *Let $G = (V + x, E)$ be a connected undirected graph with $d(x) \neq 3$. If there is no cut-edge incident to x, then there is an admissible pair of edges incident at x.*

The above theorem by Mader guarantees the existence of an admissible edge pair under natural assumptions. It can also be expressed in the following equivalent statement, which states the existence of a complete edge splitting-off at x when $d(x)$ is even.

Theorem 2.7 ([54, 27]). *Let $G = (V + x, E)$ be an undirected graph with $d(x)$ even. If there is no cut-edge incident to x, then the set of edges incident to x can be partitioned into $d(x)/2$ disjoint splittable pairs.*

To show their equivalence, assume first Theorem 2.6 holds with even $d(x)$ and consider an admissible edge pair. Recall that local edge-connectivity is preserved upon splitting-off an admissible pair. Therefore, no cut-edge incident to x will be introduced. We can apply Theorem 2.6 for $d(x)/2$ times to get the disjoint splittable pairs in Theorem 2.7.

Conversely, assume Theorem 2.7 holds. The case for even $d(x)$ is trivial, so we assume $d(x)$ to be odd and hence $d(x) \geq 5$. Let G' denote a graph arising from G by adding a new vertex y with three parallel edges connecting x and y. Since no cut-edge incident to x is introduced, we can apply Theorem 2.7 to get $(d(x) + 3)/2$ disjoint splittable edge pairs. For $d(x) \geq 5$, at least one of the splittable pair consists of original edges only. It is clear that such edge pair is also splittable in G, which proves Theorem 2.6.

In the following, we will present a proof of Mader's theorem by Frank [27] in Section 2.2.1 and Section 2.2.2. Some materials concerning edge splitting-off in directed graph and extension that incorporates additional constraints will be presented in Section 2.2.3 and Section 2.2.4 respectively.

2.2.1 The Basics

As shown in Menger's theorem, the edge-connectivity between two vertices equals the size of the minimum cut separating them. Therefore, Mader's theorem can be proved by showing that edge splitting-off preserves the size of pairwise minimum cuts. The size of cuts has very nice property of submodular functions. Frank [27] used this property to give an elegant proof for Mader's theorem. Here, we would like to cover the basics of these functions before presenting Frank's proof in the next section.

2.2.1.1 Supermodular and Submodular Set Functions

Let V be a finite ground set and $f : 2^V \to \mathbb{R}$ be a real-valued function defined on the subsets of V. We say that two sets $X, Y \subseteq V$ are *intersecting* if $X - Y, Y - X$ and $X \cap Y$ are all non-empty, and they are *crossing* if they are intersecting and $X \cup Y \subset V$, i.e. $X \cup Y \neq V$. A set function f is called *fully supermodular* (or *supermodular* for short) if the following inequality holds for any two subsets $X, Y \subseteq V$:

$$f(X) + f(Y) \leq f(X \cup Y) + f(X \cap Y)$$

On another hand, a set function f is called *fully submodular* (or *submodular* for short) if the function $f' = -f$ is supermodular, i.e. the following inequality holds for any two subsets $X, Y \subseteq V$:

$$f(X) + f(Y) \geq f(X \cup Y) + f(X \cap Y)$$

For a more general class of functions, we say that a set function f is *skew supermodular* if at least one of the following inequalities holds for any two subsets $X, Y \subseteq V$:

$$f(X) + f(Y) \leq f(X \cap Y) + f(X \cup Y) \tag{2.1a}$$

$$f(X) + f(Y) \leq f(X - Y) + f(Y - X) \tag{2.1b}$$

Similarly, we say that a set function f is *skew submodular* if $f' = -f$ is skew supermodular. Now, we would like to define a skew supermodular function in an undirected graph $G = (V, E)$. This function is useful not only in the proof of Mader's theorem but in edge-connectivity augmentation algorithms (Section 2.3.2.1) as well. Let $f(x, y)$ be a symmetric non-negative function, that is $f(x, y) = f(y, x) \geq 0$ for every $x, y \in V$. Define a skew supermodular set function F by f as follows:

$$F(X) := \begin{cases} 0 & \text{if } X = \emptyset \text{ or } X = V, \\ \max\{f(x, y) : x \in X, y \in V - X\} & \text{if } \emptyset \subset X \subset V. \end{cases} \tag{2.2}$$

Lemma 2.8. *The set function F is skew supermodular.*

Proof. First, note that we can transform between (2.1a) and (2.1b) by replacing Y by $V - Y$. Let (z, z') be a pair that maximizes $f(z, z')$ over all pairs that are separated by at least one of X and Y. By exchanging z and z' if necessary, we can assume $z \in X$ and $z' \in V - X$. And by replacing Y by $V - Y$ if necessary, we can assume $z \notin Y$.

If $z' \in Y$, then we have $f(z, z') = F(X) = F(Y) = F(X - Y) = F(Y - X)$. Thus, inequality (2.1b) holds with equality. If $z' \notin Y$, then we have $f(z, z') = F(X) = F(X \cup Y) = F(X - Y)$. Clearly, $F(Y) \leq F(X \cap Y)$ or $F(Y) \leq F(Y - X)$. Inequalities (2.1a) or (2.1b) holds accordingly. □

2.2.1.2 Set Functions regarding Edge-Connectivity

Here, we introduce some useful set functions regarding edge-connectivity. In an undirected graph $G = (V + x, E)$, we have defined $d_G(X, Y) = |\{uv \in E(G) : |X \cap \{u, v\}| = |Y \cap \{u, v\}| = 1\}|$ to be the number of edges between $X - Y$ and $Y - X$. Here, we define $\bar{d}_G(X, Y) := d_G(X \cap Y, V - (X \cup Y))$ to be the number of edges connecting the intersection of the sets and outside of their union. The following proposition concerns the degree function with two vertex sets. It can be proved by observing that each edge has the same contribution to the two sides of the identities.

Proposition 2.9. *For arbitrary $X, Y \subseteq V + x$ both of the following equalities hold:*

$$d(X) + d(Y) = d(X \cap Y) + d(X \cup Y) + 2d(X, Y) \tag{2.3a}$$

$$d(X) + d(Y) = d(X - Y) + d(Y - X) + 2\bar{d}(X, Y) \tag{2.3b}$$

Now, we define set function $r_G(X)$ to be the maximum edge-connectivity of any vertex pair separated by $X \subseteq V$.

$$r_G(X) := \max\{\lambda_G(u, v) : u \in X, v \in V - X\}$$

For $X = \emptyset$ and $X = V$, we define $r_G(X)$ to be 0. We can see the definition of r_G satisfies the condition of (2.2). This gives the following proposition.

Proposition 2.10. *For arbitrary $X, Y \subseteq V$ at least one of the following inequalities holds:*

$$r(X) + r(Y) \leq r(X \cap Y) + r(X \cup Y) \tag{2.4a}$$

$$r(X) + r(Y) \leq r(X - Y) + r(Y - X) \tag{2.4b}$$

Now, we define $s(X) := d(X) - r(X)$ to be the *surplus* of X. By the last two propositions, we can obtain the following directly.

Proposition 2.11. *For arbitrary $X, Y \subseteq V$ at least one of the following inequalities holds:*

$$s(X) + s(Y) \geq s(X \cap Y) + s(X \cup Y) + 2d(X, Y) \tag{2.5a}$$

$$s(X) + s(Y) \geq s(X - Y) + s(Y - X) + 2\bar{d}(X, Y) \tag{2.5b}$$

By the min-max relation of Menger's theorem (Theorem 2.3), $d(X) \geq r(X) = r(V - X)$ and hence $s(X) \geq 0$ before splitting-off any edge pair. After splitting-off an edge pair (e, f), the edge-connectivity is preserved in G^{ef} if $d_{G^{ef}}(X) \geq r_G(X)$ for every vertex subset $X \subseteq V$. Therefore, we can identify admissible pairs by checking the value of these set functions. This gives rise to the idea of dangerous set, which we will discuss in the next section.

2.2.1.3 Dangerous and Tight Sets

Let $G = (V + x, E)$ be an undirected graph with no cut-edge incident at x. We call a vertex subset $X \subseteq V$ *dangerous* if $d(X) \leq r(X) + 1$ (or equivalently $s(X) \leq 1$); and *tight* if $d(X) = r(X)$ (or equivalently $s(X) = 0$). Dangerous sets are of particular interest

because they actually define all the non-admissible edge pairs. The idea is stated formally in the following proposition.

Proposition 2.12 ([27]). *An edge pair $(e = xu, f = xv)$ is not admissible if and only if u, v are contained in a dangerous set.*

Proof. The existence of such a dangerous set X clearly makes (e, f) non-admissible because $d_{G^{ef}}(X) \leq d_G(X) - 2 \leq r(X) - 1$. To show the other direction, suppose that (e, f) is non-admissible. Then, there is a vertex pair (w, t) such that $\lambda_{G^{ef}}(w, t) < \lambda_G(x, y)$. This implies the existence of a vertex subset X separating w and t for which $d_{G^{ef}}(X) = \lambda_{G^{ef}}(w, t)$. Hence, $d_{G^{ef}}(X) < d_G(X)$, which requires $u, v \in X$. Therefore, $d_G(X) - 2 = d_{G^{ef}}(X) = \lambda_{G^{ef}}(w, t) < \lambda_G(w, t) \leq r_G(X)$. This gives $d_G(X) - 2 < r_G(X)$ and hence $d_G(X) \leq r_G(X) + 1$. In other words, X is a dangerous set containing u and v. □

Apart from defining non-admissible pairs, tight sets have the special property that contraction of a tight set into a single vertex (singleton) does not make any non-admissible pairs become admissible (Lemma 2.13). When we are looking for an admissible pair, we can hence reduce the graph into an instance with all tight sets being singletons. This gives additional structural properties in the graph and significantly simplifies proofs of edge splitting-off theorems. We call a vertex set $T \subseteq V$ *non-trivial* if $|X| \geq 2$ and $|V + x - T| \geq 2$.

Lemma 2.13 ([56, 27]). *Let T be a non-trivial tight set and t be the corresponding vertex in G/T. For an x-neighbour w in G/T, if $w \neq t$, let w' be the corresponding vertex in G; if $w = t$, let w' be any x-neighbour in T in G. Suppose (xu, xv) is an admissible pair in G/T, then (xu', xv') is an admissible pair in G.*

Proof. Suppose, by way of contradiction, that (xu', xv') is not an admissible pair in G. Then there exists a maximal dangerous set D containing u', v' in G, i.e. $s_G(D) \leq 1$. If

either $D \cap T = \emptyset$ or $T \subseteq D$, then

$$s_{G/T}(D - T + t) = d_{G/T}(D - T + t) - r_{G/T}(D - T + t) = d_G(D) - r_G(D) = s_G(D) \leq 1.$$

Hence $D-T+t$ is also a dangerous set in G/T, contradicting the assumption that (xu, xv) is an admissible pair in G/T. Therefore $D - T \neq \emptyset$ and $T - D \neq \emptyset$. Inequality (2.5a) cannot hold for D and T. For otherwise, $D \cup T$ is also a dangerous set in G, since

$$1 + 0 = s_G(D) + s_G(T) \geq s_G(D \cup T) + s_G(D \cap T),$$

contradicting the maximality of D since $T - D \neq \emptyset$. Therefore inequality (2.5b) must hold for D and T in G, and hence

$$1 + 0 = s_G(D) + s_G(T) \geq s_G(D - T) + s_G(T - D) + 2\bar{d}_G(D, T).$$

It follows that $(D - T)$ is a dangerous set and $xu, xv \in \delta_G(D - T)$ (since $\bar{d}_G(D, T) = 0$), which contradicts that (xu, xv) is an admissible pair in G/T. Therefore, such D does not exist, proving the lemma. □

2.2.2 Proof of Mader's Theorem

We will now present Frank's proof of Mader's theorem (Theorem 2.7). It is an analysis on the structure of non-admissible pairs (equivalently, structure of dangerous sets) using the property of submodular functions. Note that the given assumptions of Mader's theorem ($d(x)$ even and no cut-edge incident to x) still hold after splitting-off an admissible pair. It is sufficient to prove the theorem by showing the existence of one admissible edge pair under the assumptions.

Let $G = (V+x, E)$ be a counter-example with a minimum number of vertices. In other words, we assume no admissible edge pair exists in G but the theorem holds for every graph with fewer vertices. By Claim 2.13, a non-admissible pair remains non-admissible

upon contraction of tight sets. Therefore, we may assume that every tight set consists of a single vertex only. Let $t \in N(x)$ be the vertex with minimum degree among all x-neighbours. We can show that xt is a cut-edge if it has no admissible partner by the following claims. This contradicts the assumption that no cut-edge is incident to x and proves the theorem.

Claim 2.14. *$\lambda(w,t) = \min\{d(w), d(t)\}$ for every $w, t \in V$ if every tight set is a singleton, i.e. consists of one vertex only.*

Proof. For any tight set $X \subseteq V$ separating w and t, it consists of one vertex only, i.e. either w or t. Therefore, the edge-connectivity $\lambda(w,t)$ is determined by the minimum of $d(w)$ and $d(t)$. □

Claim 2.15. *$r(X-t) \geq r(X)$ for every vertex subset $X \subseteq V$ with $t \in X$ and $|N(x) \cap X| \geq 2$.*

Proof. Let $u \in N(x) \cap (X - t)$, we have $d(u) \geq d(t)$ by the choice of t. By the definition of $r(X)$, we have $r(X) = \lambda(z, z')$ for some $z \in X, z' \in V - X$. If $z \neq t$, then $r(X - t) \geq \lambda(z, z') = r(X)$. Otherwise,

$$r(X) = \lambda(t, z') = \min\{d(t), d(z')\} \leq \min\{d(u), d(z')\} = \lambda(u, z') \leq r(X - t)$$

according to Claim 2.14. □

Claim 2.16. *$d(x, X) \leq d(x, V - X)$ for dangerous set $X \subset V$.*

Proof. Let $\alpha := d(x, X)$ and $\beta := d(x, V - X)$. Then, we can derive $\alpha \leq \beta + 1$ from the following.

$$r(V - X) = r(X) \geq d(X) - 1 = d(V - X) - \beta + \alpha - 1$$

$$\geq r(V - X) - \beta + \alpha - 1$$

Since $d(x)$ is even, the inequality $\alpha \leq \beta+1$ cannot hold with equality. Therefore, $\alpha < \beta+1$ and hence $\alpha \leq \beta$. □

Since G is a counter-example to the theorem, it is clear that xt has no admissible partner. Therefore, every x-neighbour belongs to a dangerous set containing t according to Proposition 2.12. Let $\mathcal{L}$ be a smallest collection of dangerous sets containing t that $N(x) \subseteq \cup_{X\in\mathcal{L}} X$.

Claim 2.17. $|\mathcal{L}| \geq 3$.

Proof. According to Claim 2.16, $d(x, X) \leq d(x, V - X)$ for any dangerous set X. This implies $|\mathcal{L}| \geq 2$. Assume $|\mathcal{L}| = 2$ that $\mathcal{L} = \{X, Y\}$. By Claim 2.16 again, we have.

$$d(x, X) \leq d(x, V - X) < d(x, (V - X) + t) \leq d(x, Y)$$

By symmetry, we also have $d(x, Y) < d(x, X)$ which implies $d(x, X) < d(x, X)$, a contradiction. □

Claim 2.18. *Inequality (2.5b) holds for each two members $X, Y \in \mathcal{L}$.*

Proof. Suppose that inequality (2.5a) holds instead of inequality (2.5b). The minimality of $\mathcal{L}$ implies $s(X \cup Y) \geq 2$ and hence $s(X \cap Y) = 0$ by the following.

$$1 + 1 \geq s(X) + s(Y) \geq s(X \cup Y) + s(X \cap Y) \geq 2 + 0$$

Therefore, $X \cap Y$ is a tight set and it must be a singleton, i.e. $\{t\}$, according to our hypothesis. Then we have $X - Y = X - t$ and $Y - X = Y - t$. These give $r(X) \leq r(X - Y)$ and $r(Y) \leq r(Y - X)$ according to Claim 2.15. By equality (2.3b), $s(X) + s(Y) \geq s(X - Y) + s(Y - X) + 2\bar{d}(X, Y)$, i.e. inequality (2.5b) holds. □

Claim 2.19. $|X - Y| = |Y - X| = 1$ *and* $\bar{d}(X, Y) = 1$ *for each two members $X, Y \in \mathcal{L}$.*

Proof. By Claim 2.18, we have

$$1 + 1 \geq s(X) + s(Y) \geq s(X - Y) + s(Y - X) + 2\bar{d}(X, Y) \geq 0 + 0 + 2.$$

This implies $\bar{d}(X, Y) = 1$ and both $X - Y$ and $Y - X$ are tight. They must be singletons, i.e. $|X - Y| = |Y - X| = 1$, according to our hypothesis. □

To complete the proof of the theorem, let X_1, X_2, X_3 be three elements in $\mathcal{L}$ and let $M = X_1 \cap X_2 \cap X_3$. By Claim 2.19 and the minimality of $\mathcal{L}$, we have $X_i = M + x_i$ and $\bar{d}(X_i, X_j) = 1$ for $1 \leq i < j \leq 3$ that $x_i \neq x_j$ for $i \neq j$. This implies the edge xt is the only edge leaving M. In other words, xt is a cut-edge, contradicting the hypothesis of the theorem.

2.2.3 Global Arc-Connectivity Setting

In this subsection, we will discuss edge splitting-off in directed graphs. The following directed version of Lovász' splitting-off lemma is given by Mader [56]. It states that edge splitting-off can preserve global arc-connectivity when in-degree and out-degree of x are equal.

Theorem 2.20. ***(Mader's directed splitting-off lemma [56])*** *Let $D = (V + x, A)$ be a directed graph that is k-arc-connected in V with $d^+(x) = d^-(x)$, There is an edge $f = ux$ for every edge $e = xt$ for which the directed graph D^{ef} arising from splitting-off (e, f) is k-arc-connected in V.*

By the min-max relation in Menger's theorem (Theorem 2.1), this splitting-off theorem can also be proved by showing that edge splitting-off preserves the size of minimum cut in the graph. A vertex set V is k-arc-connected in a directed graph $D = (V + x, A)$ if

$$d^+_D(X), d^-_D(X) \geq k \text{ for every vertex subset } X \subset V. \tag{2.6}$$

We call a vertex subset $X \subseteq V$ *in-tight* if $d^+(X) = k$ and *out-tight* if $d^-(X) = k$. A vertex subset $X \subseteq V$ is called *tight* if it is either in-tight or out-tight. It is clear that an edge pair $(e = xu, f = xv)$ is admissible if and only if there is no tight set X containing u and v. The following proposition characterize some properties of the degree function. It can be proved by counting the contribution of each involved arc.

Proposition 2.21. *For arbitrary $X, Y \subseteq V$ both of the following equalities hold:*

$$d^+(X) + d^+(Y) = d^+(X \cap Y) + d^+(X \cup Y) + d(X, Y) \tag{2.7a}$$

$$\begin{aligned} d^+(X) + d^+(Y) = {} & d^+(X - Y) + d^+(Y - X) + \bar{d}(X, Y) \\ & + d^+(X \cap Y) - d^-(X \cap Y) \end{aligned} \tag{2.7b}$$

With the above proposition, we can show that the union of two tight sets is also tight if they contain a common x-neighbour. In other words, each x-neighbour is contained in at most one (unique) maximal tight set. This gives the key statement in the proof of the directed splitting-off lemma.

Claim 2.22. *$X \cup Y$ is tight if X and Y are tight sets containing an x-neighbour t.*

Proof. Define $\bar{Z} := V + x - Z$ to be the complement of a vertex subset $Z \in V$. The claim is trivial for $X \subseteq Y$ or $Y \subseteq X$ so we assume this is not the case. If $d^+(X) = k$ and $d^-(Y) = k$, then by equality (2.7a), we have

$$\begin{aligned} k + k = d^+(X) + d^+(\bar{Y}) &= d^+(X \cap \bar{Y}) + d^+(X \cup \bar{Y}) + d(X, \bar{Y}) \\ &\geq k + k + |zt| \geq 2k + 1 \end{aligned}$$

Therefore, it is impossible that one of them is in-tight and the other one is out-tight.

Suppose that both X and Y are out-tight. If $X \cup Y = V$, we have $\bar{X} \cap \bar{Y} = x$ and so $d^+(\bar{X} \cap \bar{Y}) = d^-(\bar{X} \cap \bar{Y})$. Applying equality (2.7b) to $\bar{X}$ and $\bar{Y}$, we have,

$$\begin{aligned} k + k = d^+(\bar{X}) + d^+(\bar{Y}) &= d^+(\bar{X} - \bar{Y}) + d^+(\bar{Y} - \bar{X}) + \bar{d}(\bar{X}, \bar{Y}) \\ &\geq k + k + |zt| \geq 2k + 1. \end{aligned}$$

This implies $X \cup Y \subset V$. Now, replace d^+ by d^- in equality (2.7a), we have

$$k + k = d^-(X) + d^-(Y) \geq d^-(X \cap Y) + d^-(X \cup Y) \geq k + k.$$

This restricts $d^-(X \cap Y) = d^-(X \cup Y) = k$ and hence $X \cup Y$ is a tight set. The proof for the case that both X and Y are in-tight is basically the same, so the proof of this claim is completed here. □

Proof of Theorem 2.20: If there is no tight set containing t, then xt is an admissible partner to every edge entering x. So we assume this is not the case, i.e. t is contained in some tight sets. By Claim 2.22, there exists a unique maximal tight set T containing t. We can complete the proof by showing the existence of an edge entering x that does not leave from T.

Suppose to the contrary that every in-edge of x leaves from T. If T is an in-tight set, i.e. $d^+(T) = k$, then we have

$$d^-(V - T) = d^+(T + x) \leq d^+(T) - 1 = k - 1.$$

On the other hand, if T is an out-tight set, i.e. $d^-(T) = k$, then we have

$$d^+(V - T) = d^-(T + x) \leq (d^-(T) - d^-(x)) + (d^+(x) - 1) \leq d^-(T) - 1 = k - 1.$$

Both contradict premise (2.6) for V being k-arc-connected. □

2.2.3.1 Local Arc-Connectivity Setting

In the previous sections, we have discussed some general edge splitting-off theorems in the local edge-connectivity and global arc-connectivity settings. However, no similar edge splitting-off theorem is known in the local arc-connectivity setting. The key reason may be that the surplus functions do not have the nice property of submodular functions, as we will show in the following.

Suppose that the arc-connectivity requirements are symmetric, i.e. $r^+(u,v) = r^-(u,v)$ and hence $r(u,v) = r(v,u)$. We define $s^+(X) = d^+(X) - r(X)$ and $s^-(X) = d^-(X) - r(X)$ to be the *surplus* functions in the local arc-connectivity setting. Combining Proposition 2.10 and Proposition 2.21, we can guarantee at least one of the following inequalities holds.

$$s^+(X) + s^+(Y) \geq s^+(X \cap Y) + s^+(X \cup Y) + d(X,Y) \tag{2.8a}$$

$$\begin{aligned} s^+(X) + s^+(Y) \geq\ & s^+(X - Y) + s^+(Y - X) + \bar{d}(X,Y) \\ & + d^+(X \cap Y) - d^-(X \cap Y) \end{aligned} \tag{2.8b}$$

When inequality (2.8b) holds and $\bar{d}(X,Y) + d^+(X \cap Y) < d^-(X \cap Y)$, we may have

$$s^+(X) + s^+(Y) \leq s^+(X - Y) + s^+(Y - X).$$

Therefore, we cannot ensure at least one of inequality (2.1a) or inequality (2.1b) holds, and thus the surplus function in a directed graph is not skew submodular in general. Note that the supermodular (submodular) function is the cornerstone supporting edge splitting-off theorems. Therefore, we cannot extend the directed splitting-off lemma to the local arc-connectivity setting unless we can support the splitting-off theorems by something else. The local arc-connectivity setting will not be covered anymore in the rest of this thesis.

2.2.4 Incorporating Additional Properties

When reducing a graph by edge splitting-off operations, we may want to preserve some extra properties of the graph, such as graph simplicity, in addition to the edge-connectivity. This raises the interest of extending edge splitting-off operation to *constrained edge splitting-off* operation, which is a generalization that not only edge-connectivity must be maintained but the split edge has to satisfy some additional properties as well.

An edge pair is called *legal* if the specified properties are preserved upon splitting-off the pair. In other words, we look for *legal admissible edge pairs*. The extra requirements for split edges to satisfy are called additional *constraints*.

There are several established results on constrained edges splitting-off [4, 3, 44, 37, 59, 5]. Like Mader's theorem, these results are proved by analyses on the structure of non-admissible pairs (or equivalently, dangerous sets) which use the submodular functions discussed in Section 2.2.1.1. Note that in general it is NP-hard to decide whether all edges incident to a designated vertex can be splitted-off by constrained edge splitting-off [43]. Therefore the constrained edge splitting-off theorems usually make stronger assumption to guarantee the existence of a legal admissible edge pair. In other words, it may be impossible to split-off all edges incident to a designated vertex. Some of these results will be covered later in Chapter 4. Here, we would like to highlight the *non-admissibility graph* method developed by Jordán [45]. He gave a more general method to prove constrained edge splitting-off results in the global edge-connectivity setting.

2.2.4.1 Non-Admissibility Graph Method

To show the existence of an admissible edge pair, Frank studied the structure of non-admissible edge pairs in the graph. Therefore, a straightforward approach to show the existence of a legal admissible edge pair is to study the structure of non-admissible pairs under a given constraint. Taking a different approach, Jordán considered admissibility and the additional constraint separately [45]. Note that the structure of non-admissible pairs (dangerous sets) remains unchanged regardless of the type of additional constraint. Therefore, the structural properties identified can be used in different additional constraints and hence contribute to a general method.

Consider an undirected graph $G = (V + x, E)$ that G is k-edge-connected for some $k \geq 2$, and a constraint C that G has to satisfy. Jordán defined a *non-admissibility*

graph $B(x) = (N(x), E(B(x)))$ of G with respect to the vertex x. The graph $B(x)$ takes the neighbour set of x as the vertex set; two vertices $u, v \in N(x)$ are adjacent in $B(x)$ if and only if the corresponding edge pair $(e = xu, f = xv)$ is non-admissible in G. Therefore, there exists an admissible edge pair in G if and only if there exists two non-adjacent vertices u and v in $B(x)$, i.e. $B(x)$ is not a complete graph. Under the global edge-connectivity setting, Jordán identified some useful structural properties of the non-admissibility graph $B(x)$.

Then, Jordán defined a constraint graph $H_C(x) = (N(x), E(H_C(x)))$ of G with respect to the vertex x and additional constraint C. The graph $H_C(x)$ also takes the neighbour set of x as the vertex set; two vertices $u, v \in H_C(x)$ are adjacent if and only if G^{ef} satisfies the constraint C upon splitting-off the corresponding edge pair $(e = xu, f = xv)$. For any legal pair $(e = xu, f = xv)$ in G, we have $uv \in E(H_C(x))$ and $uv \notin E(B(x))$, i.e. $uv \in E(H_C(x)) \cap E(\overline{B(x)})$. In words, there exists a legal admissible edge pair if there is an edge in the graph $(N(x), E(H_C(x)) \cap E(\overline{B(x)}))$, or equivalently, the graph $(N(x), E(\overline{H_C(x)}) \cap E(B(x)))$ is not a complete graph.

Therefore, one can prove constrained edge splitting-off by studying the non-admissibility graph and constraint graph separately. Note that the structural properties of a non-admissibility graph is not affected by the additional constraint. Therefore, the structure identified by Jordán can be applied in various additional constraints and so gives rise to a general method for solving problems of this type. The exact structures characterized by Jordán in the global edge-connectivity is quite irrelevant to this thesis since we are studying the settings of local edge-connectivity and global arc-connectivity. The details are hence not covered here. Interested readers are referred to Jordán's paper [45].

2.3 Edge-Connectivity Problems

Edge-connectivity problems include several classical graph problems, like minimum spanning tree, and are the topic of much research. A problem we are particularly interested in is the *network design problem*. The objective is to find a minimum cost subgraph that satisfies certain *edge-connectivity requirements*. In this thesis, we consider network design problem with the following setting:

Definition 2.23. ***(Network Design Problem)*** *Consider a graph $G = (V, E)$, with edge-connectivity requirement function $r : V \times V \to \mathbb{Z}$ on each vertex pairs, and cost function $c : E \to \mathbb{Q}$ on each edge. Find a minimum cost subgraph $H = (V, F) \subseteq G$ that $\lambda_H(u, v) \geq r(u, v)$ for each two vertices $u, v \in V$.*

We assume that $\lambda_G(u, v) \geq r(u, v)$ to guarantee the existence of a solution. Note that a number of classical graph problems in combinatorial optimization can be expressed in this manner. For example, the minimum k-edge-connected subgraph problem asks for a minimum cost subgraph that each two vertices u, v are k-edge-connected, i.e. $\lambda(u, v) \geq k$; and the Steiner tree problem asks for a minimum cost subgraph that each two terminal vertices u, v are connected. The network design problem is hence a generalization of some classical graph problems in combinatorial optimization.

Another problem we are also interested in is the edge-connectivity augmentation problem. The goal of this problem is to improve the edge-connectivity of a given graph to specified extent by adding a minimum set of new edges.

Definition 2.24. ***(Edge-Connectivity Augmentation Problem)*** *Consider a graph $G = (V, E)$ with edge-connectivity requirement function $r : V \times V \to \mathbb{Z}$ on each vertex pairs, and a graph $H = (V, F)$ with cost function $c : F \to \mathbb{Q}$ on each edge. Find a minimum cost edge subset $F' \subseteq F$ that $\lambda_{G'}(u, v) \geq r(u, v)$ for each two vertices $u, v \in V$ in the augmented graph $G' = (V, E + F')$.*

These two mentioned edge-connectivity problems are actually equivalent. A network design problem can be reformulated as an edge-connectivity augmentation problem that augments an edge-less input graph. Suppose that an edge-connectivity augmentation problem is tractable for certain connectivity requirements. Then the corresponding network design problem is also tractable by this reduction. Conversely, an edge-connectivity augmentation problem can also be reformulated as a network design problem that has a set of zero cost initially assigned edges. The tractability of a network design problem for certain connectivity requirements, hence, implies the tractability of the corresponding edge-connectivity augmentation problem.

We would like to distinguish between the edge-connectivity augmentation problem and network design problem to avoid considering these equivalent problems twice. In the rest of this thesis, we restrict our attention of edge-connectivity augmentation problem to the so-call *free augmentation* version. That is, we can add any number of copies of any possible edge connecting the vertices of the input graph, and the cost of each new edge is identical. The free augmentation problem is of special interest because it can be solved in polynomial-time [26][1]. This is in contrast to the network design problem, which is NP-hard to solve in general. For example, finding a minimum cardinality 2-edge-connected spanning subgraph, i.e. Hamiltonian cycle, in a graph is well-known to be NP-complete but the free augmentation problem that augments a graph to be 2-edge-connected is polynomial-time solvable.

2.3.1 Degree Bounded Network Design Problems

Network design is a central topic in combinatorial optimization and approximation algorithms. In recent years, much research has been done on the design of approximation algorithms for *degree bounded network design problem*. That is, the problem of

[1]In directed graph, only some special cases, e.g. $\lambda(u, v) \equiv k$ for all vertex pair u, v, can be solved.

finding a minimum cost subgraph with maximum vertex degree at most B in a weighted undirected graph that satisfies certain edge-connectivity requirements. This problem generalizes some key problems in combinatorial optimization, such as the travelling salesman problem, and also have applications in various areas including VLSI design and communication networks [20, 73, 6, 68, 67]. In these applications, vertex degree constraints act as a tool to model the workload of nodes. For example, in the applications to multicasting, vertex degree constraint corresponds to a bound on the number of multicast copies a switch can make in the network.

In general this problem does not admit any non-trivial approximation algorithm, since the feasibility problem, which captures the Hamiltonian cycle problem as a special case, is already NP-hard. Recent research has thus focused on obtaining bicriteria approximation algorithms for degree bounded network design problems [38, 51, 70, 52]. An $(\alpha, +\beta)$ bicriteria approximation algorithm is an algorithm which returns a solution with cost at most α times minimum cost and vertex degree at most $B_v + \beta$ for all v, where B_v is the given degree upper bound on vertex v. A polyhedral approach is applied successfully to obtain bicriteria approximation algorithms with only an additive violation on the degree: there is a $(1, +1)$-approximation algorithm for the minimum bounded degree spanning tree problem [70], a $(2, +O(k))$-approximation algorithm for the minimum bounded degree k-edge-connected subgraph problem [52], and a $(2, +O(r_{\max}))$-approximation algorithm for the minimum bounded degree Steiner network problem [52], while the maximum degree of the solutions is at most $2B + 3$ [70].

2.3.1.1 Metric Cost Assumption

In some network design problems, stronger algorithmic results are known when the cost function satisfies triangle inequality [48, 17, 1]. Take the travelling salesman problem as an example. Although there is no polynomial factor approximation algorithm in

general, it is well-known that there is a 1.5-approximation algorithm assuming triangle inequality [66]. This motivates us to study more general degree bounded network design problems with metric costs. In this thesis, we restrict our attention for network design problem to the version with vertex degree bound and metric cost.

Definition 2.25. ***(Degree Bounded Network Design Problem with Metric Costs)*** *Given a graph $G = (V, E)$, an edge-connectivity requirement function $r : V \times V \rightarrow \mathbb{Z}$ on each vertex pair, a cost function $c : E \rightarrow \mathbb{Q}$ on each edge satisfying triangle inequality ($c(uv) + c(vw) \geq c(uw)$ for all u, v, w), and a degree upper bound B on each vertex v. Find a minimum cost subgraph $H = (V, F) \subseteq G$ that $\lambda_H(u, v) \geq r(u, v)$ for each two vertices $u, v \in V$, and the degree of each vertex in H is at most B.*

For *degree bounded network design problem with metric costs*, there are approximation algorithms to construct a k-edge-connected subgraph with maximum degree k [31, 33], if parallel edges are allowed. For local edge-connectivity, there is some known result [32][2], but no constant factor approximation algorithm is known even if parallel edges are allowed.

Note that edge splitting-off reduce vertex degree of the designated vertex by 2 while satisfying pairwise edge-connectivity requirements. Furthermore, this operation does not increase the cost of the subgraph when cost function satisfies triangle inequality. Given any subgraph satisfying the edge-connectivity requirements, we can hence apply edge splitting-off to reduce vertex degree while maintaining edge-connectivity and keeping the total cost of the subgraph. Actually, this approach is used in some of the results we just mentioned above [31, 32]. In Chapter 3, we will show how to solve the degree bounded network design problem with metric costs in general, even when graph simplicity is taken into account.

[2]Graph is assumed to be 2-edge-connected in [32].

2.3.2 Edge-Connectivity Augmentation Problems

To distinguish from network design problem, we restrict our attention of edge-connectivity augmentation problem to the free augmentation version in this thesis. That is, we can add any number of copies of any possible edge connecting the vertices of the initial graph, and the cost of each new edge is identical. We define the problem in free augmentation version as follows. Note that this version is sometimes called the *cardinality case*.

Definition 2.26. ***(Edge-Connectivity Augmentation Problem)*** *Consider a graph $G = (V, E)$ with edge-connectivity requirement function $r : V \times V \to \mathbb{Z}$ on each vertex pair. Find a minimum set of edges F that in the augmented graph $G' = (V, E + F)$, $\lambda_{G'}(u, v) \geq r(u, v)$ for each two vertices $u, v \in V$.*

The edge-connectivity augmentation problem is a very well-studied topic. In the global edge-connectivity setting, Eswaran and Tarjan [23] gave a polynomial-time algorithm to augment a graph to 2-edge-connected with minimum number of new edges. Generalizing their idea, Watanabe and Nakamura [74], Naor, Gusfield and Martel [65] augmented a graph to k-edge-connected in polynomial-time for arbitrary $k \geq 2$. Cai and Sun [13] also solved the same problem, and in addition, they provided a min-max relation for this problem. Note that the application of edge splitting-off technique in augmentation problem was first introduced in this result. By using this approach, Frank solved the problem in the local edge-connectivity setting in strongly polynomial time [26].

Regarding the global arc-connectivity setting, Eswaran and Tarjan [23] showed how to make a directed graph strongly connected, i.e. $r(u, v) \equiv 1$, by adding a minimum number of arcs. For the general setting $r(u, v) \equiv k$, Fulkerson and Shapley [34] solved the problem when the initial graph has no arc. Kajitani and Ueno [46] solved the problem in a less restricted case that the initial graph is an arborescence. Applying edge splitting-off technique, Frank [26] solved the problem for any initial graph.

Frank's result provided a general framework to solve problems of this kind by edge splitting-off technique. This framework is later applied with constrained edge splitting-off operations to solve the so-called constrained edge-connectivity augmentation problems [4, 3, 44, 37, 59, 5]. Some materials of constrained edge-connectivity augmentation problems will be covered in Section 2.3.2.2. Here, we will first study Frank's framework in the local edge-connectivity setting. Note that Frank's algorithm works in both local edge-connectivity and global arc-connectivity settings, but not local arc-connectivity setting. This is because of the absence of strong edge splitting-off lemma regarding local arc-connectivity, as we explained in Section 2.2.3.1.

2.3.2.1 Frank's Framework

To solve edge-connectivity augmentation problems, Frank [26] developed a framework based on the edge splitting-off results. This framework can be applied in the global edge-connectivity and local edge-connectivity settings to give polynomial time algorithms for the augmentation problem. Consider an undirected graph $G = (V, E)$ with edge-connectivity requirement $r(u, v)$ between every two vertices $u, v \in V$. Frank's framework consists of the following two phases.

(1) External Augmentation Subroutine: In the first subroutine, an extra vertex x and a minimum set of x-edges are added to the graph to satisfy the edge-connectivity requirements. Benczúr and Karger [7] called the problem of finding such a minimal edge set an *external augmentation problem*. The degree of x is kept to be even by adding at most one extra x-edge. It can be proved that the optimal value of the external augmentation problem is at most twice the optimal value of the original augmentation problem.

(2) Complete Edge Splitting-off Subroutine: In the second subroutine, we perform

a complete edge splitting-off at x, i.e. split-off all edges incident to x. This is feasible according to Mader's theorem because the first subroutine guarantee the degree of x to be even[3]. Furthermore, the set of split edges is an optimal solution for the edge-connectivity augmentation problem since the first subroutine also guarantee the number of x-edges is twice the optimal value.

To show the correctness of Frank's algorithm, the *deficiency* function q_G is used here. In order to satisfy edge-connectivity requirements in the augmented graph G', we have to make sure $d_{G'}(X) \geq r(X)$ for every vertex subset $X \subset V$. Hence, we define $q_G(X) := r(X) - d_G(X)$ to be the deficiency of a vertex subset $X \subseteq V$. In words, we have to add at least $q_G(X)$ new edges incident to X in order to satisfy the edge-connectivity requirements. This gives a lower bound on the size of an optimal solution, $\gamma(G)$:

$$\sum_{1 \leq i \leq t} q_G(X_i) \leq 2\gamma(G) \text{ for every sub-partition } \{X_1, X_2, \ldots X_t\} \text{ of } V. \tag{2.9}$$

Now, we can show that the size of solution returned by the external augmentation subroutine is at most twice of $\gamma(G)$ by considering the deficiency function (Claim 2.27). Since each external edge pair is replaced by (at most) one split edge in the complete edge splitting-off subroutine, the size of the solution returned by Frank's algorithm is hence bounded by $\gamma(G)$. This shows the optimality of the algorithm.

Claim 2.27. *The size of solution returned by the external augmentation subroutine is at most twice of $\gamma(G)$.*

Proof. Before adding the extra edge to make $d(x)$ even, the set of x-edge is minimal to satisfy the given edge-connectivity requirements. Therefore, there exists a tight set $X \subset V$ containing v for each x-neighbour $v \in N(x)$ so that removal of xv will violate some edge-connectivity requirements. Let $\mathcal{F} := \{X_1, X_2, \ldots, X_t\}$ be a family of tight sets

[3]An extra procedure is used to ensure there is no cut-edge incident to x.

covering the set of x-neighbours $N(x)$ so that t is minimal, and $\sum |X_i|$ is minimal for a given t. If $\mathcal{F}$ is a subpartition of V, then the proof can be completed by applying the lower bound of (2.9) directly. Therefore, we are going to show that X_i's are disjoint.

Apply Proposition 2.11 to any two tight sets X and Y. If inequality (2.5a) holds, then both $X \cap Y$ and $X \cup Y$ are tight sets since

$$0 + 0 = s(X) + s(Y) \geq s(X \cap Y) + s(X \cup Y).$$

If inequality (2.5b) holds, then both $X - Y$ and $Y - X$ are tight sets with $\bar{d}(X, Y) = 0$ since

$$0 + 0 = s(X) + s(Y) = s(X - Y) + s(Y - X) + 2\bar{d}(X, Y).$$

Now, consider $X, Y \in \mathcal{F}$ with $X \cap Y$ nonempty. Their union $X \cup Y$ cannot be tight, for otherwise, we can replace them by their union and this contradicts the minimality of t. Therefore, we know that both $X - Y$ and $Y - X$ are tight with $\bar{d}(X, Y) = 0$. We can get another family of tight sets covering $N(x)$ by replacing X and Y by $X - Y$ and $Y - X$. Note that such a family gives smaller $\sum |X_i|$ and contradicts the choice of $\mathcal{F}$. Therefore, we can conclude that $X \cap Y = \emptyset$ for any $X, Y \in \mathcal{F}$ and hence X_i's are disjoint. □

Finally, we remark that the framework by Frank actually handles the so-called *marginal components*. These components may introduce cut-edges incident to the external vertices x and violate the assumption for Mader's theorem. Therefore, Frank used an extra procedure to handle such components so as to guarantee the existence of a complete edge splitting-off operation in the second subroutine. Interested readers are referred to Frank's paper [26].

2.3.2.2 Constrained Edge-Connectivity Augmentation Problems

In an edge-connectivity augmentation problem, the goal is to find a minimum set of edges whose addition satisfies the given edge-connectivity requirements. Sometimes,

we may want the resulting graph (or equivalently the set of split edges) to satisfy some extra properties, in addition to the connectivity requirement. Given a bipartite graph, it is natural to require the resulting graph to stay bipartite. This raises the interest of studying *constrained edge-connectivity augmentation problem*, which are extensions of the original problem that take additional constraints into account. Note that the constrained edge-connectivity augmentation problem is NP-complete in general [43]. Therefore, we usually aim at obtaining approximation algorithms.

Definition 2.28. ***(Constrained Edge-Connectivity Augmentation Problem)*** *Consider a graph $G = (V, E)$ with edge-connectivity requirement function $r : V \times V \to \mathbb{Z}$ on each vertex pairs. Find a minimum subset of edges F satisfying certain property P and that in the augmented graph $G' = (V, E + F)$, $\lambda_{G'}(u, v) \geq r(u, v)$ for each two vertices $u, v \in V$.*

Frank [26] developed a framework solve edge-connectivity augmentation problem using edge splitting-off operations. It is hence a natural attempt to approximate a constrained edge-connectivity augmentation problem by applying constrained edge splitting-off operations in Frank's framework. In other words, we try to apply the framework that uses constrained edge splitting-off in the second subroutine. However, such attempt may give a very bad approximation ratio. Consider a bipartite graph $G = (U + V, E)$ such that an edge uv exists only if $u \in U$ and $v \in V$. Suppose that the external augmentation subroutine returns a solution with all edges connecting between x and U. Then it is impossible to split-off any edge pair while preserving the bipartiteness of the graph. Therefore, we are stuck with with arbitrarily large $d(x)$ in the second subroutine.

Note that the properties of the set of split edges are unavoidably dependent on the properties of the set of x-edges. To handle the issue we addressed above, we solve a *constrained external augmentation problem* in the first subroutine so that the set of x-

edges also satisfies certain properties. Like the unconstrained case, the number of x-edges is usually bounded by twice of the optimal value of the given constrained edge-connectivity augmentation problem.

As mentioned in Section 2.2.4, a complete constrained edge splitting-off operation may not exist. Therefore, usually we can only find an approximate solution in the second subroutine. Hence, we need an extra procedure to handle the remaining x-edges. This may incur extra cost, say by adding extra edges, and makes the solution to be an approximate one. As a conclusion, the revised framework for solving constrained edge-connectivity augmentation problem consists of the following 3 subroutines.

(1) Constrained External Augmentation Subroutine: Add an extra vertex x and a minimum set of x-edges to the graph in order to satisfy edge-connectivity requirements, where each x-edge satisfies certain additional constraint. An extra x-edge may be added to make the degree of x to be even. The number of x-edges is usually bounded by twice the optimal value of the (original) constrained edge-connectivity augmentation problem.

(2) Constrained Complete Edge Splitting-off Subroutine: In the second subroutine, we try to split-off all x-edges by constrained edge splitting-off operations. Since a complete edge splitting-off operation may not exist in such problem, the subroutine returns an approximate solution with some remaining x-edges.

(3) Reconciliation Subroutine: In this final subroutine, we isolate and remove x from the graph. This can be accomplished by replacing the remaining x-edges by some other x-edges while satisfying both edge-connectivity requirements and the additional constraints. Such operations usually incur extra cost and make the solution to be an approximate one.

2.3.3 Edge Splitting-off Problems

Edge splitting-off operation is an important subroutine in various edge-connectivity algorithms. In the previous section (Section 2.3.2), we showed an application in edge-connectivity augmentation problem. Here, we define the problem of finding an admissible edge pair formally.

Definition 2.29. ***(Edge Splitting-off Problem)*** *Consider a graph $G = (V, E)$ with a designated vertex x and edge-connectivity requirement function $r : V \times V \to \mathbb{Z}$ on each vertex pair, find a pair of edges $e, f \in \delta(x)$ such that $\lambda_{G^{ef}}(u, v) \geq r(u, v)$ for every two vertices $u, v \in V$.*

As shown by Mader's theorem (Theorem 2.7), edge splitting-off can be applied repeatedly and this gives a sequence of admissible edge pairs. Such sequence of edge pair is called an *edge splitting-off sequence*; and is called a *complete edge splitting-off sequence* if it is a partition of edges incident to the designated vertex. This gives rise to the *complete edge splitting-off problem*. Note that a complete sequence may not always exist, e.g. when $d(x)$ is odd. Therefore, the problem finds an edge splitting-off sequence that contains the maximum number (copies) of x-edges instead.

Definition 2.30. ***(Complete Edge Splitting-off Problem)*** *Consider a graph $G = (V, E)$ with a designated vertex x and edge-connectivity requirement function $r : V \times V \to \mathbb{Z}$ on each vertex pair, find a sequence of admissible edge pairs with the maximum number of x-edges such that in the graph G' arising from G by splitting-off all the pairs in the sequence, $\lambda_{G'}(u, v) \geq r(u, v)$ for every two vertices $u, v \in V$.*

Similarly, we can define *constrained edge splitting-off problem* and *constrained complete edge splitting-off problem* accordingly. The *edge splitting-off sequence* and *complete edge splitting-off sequence* are referred to the sequences of legal admissible edge pairs in these problems.

Definition 2.31. ***(Constrained Complete Edge Splitting-off Problem)*** *Consider a graph $G = (V, E)$ with a designated vertex x, edge-connectivity requirement function $r : V \times V \to \mathbb{Z}$ on each vertex pairs and an additional constraint C, find a sequence of legal admissible edge pairs with maximum number of x-edges such that in the graph G' arising from G by splitting-off all pairs in the sequence, (i) $\lambda_{G'}(u, v) \geq r(u, v)$ for every two vertices $u, v \in V$; and (ii) the additional constraint C is satisfied.*

The edge splitting-off we consider here takes the edge-connectivity requirements of the designated vertex x into account. It is slightly more general than what we discussed in Mader's theorem in Section 2.2.2. In the rest of this thesis, we will study edge splitting-off in this more general setting. As a remark, Frank's proof can be modified slightly to prove Mader's result in this setting.

2.4 Edge Splitting-off Algorithms

Lovász [54, 55] and Mader [56] proved that edge splitting-off can be performed to preserve edge-connectivity of a graph. This combinatorial operation is used as an important subroutine in several algorithms for connectivity problems, including connectivity augmentation, network design, tree packing and graph orientation. An efficient construction of an edge splitting-off sequence can hence give fast algorithms for these problems. Most of the efficient algorithms are developed only for the global edge-connectivity setting, although there are important applications in the more general local edge-connectivity setting, such as Steiner tree packing. Here, we will first cover some results of the best algorithms (Section 2.4.1) and present an intuitive approach for solving the problem (Section 2.4.2). Then, we will see some approaches in the global edge-connectivity setting that are more efficient than the intuitive one (Section 2.4.3); and explain why it is difficult to extend these approaches to the local edge-connectivity setting (Section 2.4.4).

For the sake of convenience, we assume the degree of the designated vertex $d(x)$ to be even to make sure a complete splitting-off sequence exist.

2.4.1 Fastest Algorithms

In the global edge-connectivity setting, i.e. $r(u, v) \equiv k$, Gabow [36] found a complete splitting-off sequence in $O(n^2 m \log (n^2/m))$ time for a weighted graph; and in $O(k^2 n + m)$ time for an unweighted graph. Bhalgat, Hariharan, Kavitha and Panigrahi [10] improved Gabow's result that they found complete splitting-off sequences for each vertex of a specified vertex subset in the same time bound for an unweighted graph. As the best deterministic algorithm in weighted graph, Nagamochi [58] solved the problem in $O(nm + n^2 \log n)$ time, improving the $\tilde{O}(nm)$-time algorithm by Nagamochi and Ibaraki [63]. Benczúr and Karger [7] improved the time bound to $\tilde{O}(n^2)$ by randomization techniques. In the global arc-connectivity setting, Gabow [36] found a complete splitting-off sequence in $O(km)$ time for an unweighted graph. Bhalgat, Hariharan, Kavitha and Panigrahi [10] again improved Gabow's result that they found complete splitting-off sequences for each vertex of a specified vertex subset in the same time bound.

This problem is, however, not very well-studied in the local edge-connectivity setting. The current best time bounds are due to Gabow [36], who finds a complete splitting-off sequence in $O(n^3 m \log (n^2/m))$ time for a weighted graph; and in $O(kn^2 m)$ time for an unweighted graph with maximum edge-connectivity requirement $r_{\max} = k$. These results remain to be the fastest algorithms since 1994. It seems the design of efficient algorithm becomes more difficult when we move from the global edge-connectivity setting to the local edge-connectivity setting. This may be because nice properties, such as those regarding minimum cuts, vanish when we consider local edge-connectivity. We will have a deeper discussion regarding this in Section 2.4.4.

2.4.2 An Intuitive Approach

A straightforward approach to construct a complete splitting-off sequence is to split-off admissible edge pairs iteratively --- split-off an admissible edge pair and reduce the graph. According to Mader's theorem (Theorem 2.6), we can always find an admissible edge pair by this approach. However, finding an admissible edge pair is not a trivial task. Gabow [36] used a nice approach to find an admissible edge pair in the local edge-connectivity setting by using $O(n)$ maximum flow computations. Since each maximum flow computation takes considerable amount of time, this imposes a bound on the efficiency of his edge splitting-off algorithm.

Instead of splitting-off an admissible edge pair, we may also split-off any edge pair $(e = xu, f = xv)$ without knowing the admissibility and then un-split it if any edge-connectivity requirement is violated. Note that checking the violation is equivalent to finding a minimum cut in the global edge-connectivity setting; and taking wild guesses may result in $O(|N(x)|^2)$ splitting-off attempts. Therefore, the time complexity is by no means reduced by using this approach.

2.4.3 Global Connectivity Settings

To obtain an algorithm that runs faster than finding admissible edge pairs one by one, a natural attempt is to consider the admissibility of several edge pairs together. We may either split-off multiple edge pairs at once and then fix the decreased edge-connectivity; or find multiple admissible pairs simultaneously. In both cases, the cornerstones are the properties of minimum cuts.

2.4.3.1 Legal Ordering

The edge splitting-off problem in the global edge-connectivity setting is closely related to the minimum cut problem. A traditional approach to find a global minimum cut uses Menger's theorem. That is, we use the min-max relation of cut and flow, and compute the size of minimum cut by solving maximum flow problems. Nagamochi and Ibaraki [62, 64] developed a revolutionary algorithm for minimum cut problem that does not use Menger's theorem and flows. A key idea of their algorithm is an ordering of vertices called *legal ordering*. Note that a legal ordering is unrelated to the legal edge pair we defined before.

Consider an undirected graph $G = (V, E)$. The algorithm of Nagamochi and Ibaraki [62, 64] is based on the following observation: for any two vertices $u, v \in V(G)$ that $d(u) = \lambda(u, v)$,

(a) if there exists a global minimum cut of G separating the two vertices, then $\lambda(G) = \lambda(u, v) = d(u)$ and this gives a global minimum cut;

(b) if there is no global minimum cut of G separating the two vertices, global minimum cuts are preserved upon contraction of u and v into a single vertex, i.e. a cut is a global minimum cut in G if and only if it is a global minimum cut in $G/\{u, v\}$.

If there always exist the desired vertices u and v, then the global minimum cut can be found by an iterative approach --- check $d(u)$ and then contract u, v into a single vertex. It is clear that the size of a cut will never decrease upon contraction of vertex subsets, i.e. no new minimum cut would be created during this iterative algorithm. Therefore, the algorithm can find the global minimum cut after $n - 1$ iterations.

The existence of vertices $u, v \in V(G)$ with $d(u) = \lambda(u, v)$ is guaranteed by an application of a legal ordering. An ordering $v_1, v_2, \ldots, v_n$ of all vertices in $V(G)$ is called *legal* if

$$d(\{v_1, v_2, \ldots, v_i\}, v_{i+1}) \geq d(\{v_1, v_2, \ldots, v_i\}, v_j) \text{ for any } 1 \leq i < j \leq n.$$

A legal ordering [61, 60, 71] satisfies the need that $\lambda(v_n, v_{n-1}) = d(v_n)$ always holds, and such ordering can be found efficiently in $\tilde{O}(m)$ time. The algorithm of Nagamochi and Ibaraki hence consists of $(n-1)$ computations of legal ordering and has a time bound of $\tilde{O}(nm)$.

Nagamochi and Ibaraki [63] used this algorithm as a subroutine to design an iterative algorithm for complete edge splitting-off problem. In each iteration, they split-off all edges incident to x without knowing admissibility. Then, they execute the iterative algorithm described above. Given an integer k, the iterative algorithm can find cuts with size smaller than k easily. With the identified small cut, they used some clever operations to un-split a minimum subset of edge pairs to restore the edge-connectivity of the graph to k. Since the subset of edge pairs they un-split is minimal, each of the edge pair is non-admissible. The admissibility information obtained can be used to reduce the number of splitting-off attempts in later iterations. This bounds the total number of iteration to $O(\log |N(x)|)$ and the total runtime complexity is hence $\tilde{O}(nm)$.

2.4.3.2 Edmonds' Arborescences

We define a directionless r-spanning tree to be a spanning tree that edges incident to r must be directed away from r while other edges can be in any direction. Also, we define *r-connectivity* to be the maximum number of arc-disjoint paths from a vertex r to every other vertex. A classical result of Edmonds [22] shows that a directed graph has r-connectivity of k if and only if it has k edge-disjoint directionless r-spanning trees that r has in-degree 0 and every other vertex has in-degree k over all the trees. Gabow [35] gave construction algorithm for Edmonds' result that constructs k edge-disjoint directionless r-spanning trees one by one for a directed graph with r-connectivity k. The algorithm terminates when it finds a cut $(C, V-C)$ separating $r \in C$ from some other vertices with $d^-(C) = k$. This implies that the construction of the $(k+1)$th tree is impossible.

Bhalgat, Hariharan, Kavitha and Panigraji [10] used Gabow's algorithm to detect and then fixed small cuts in their edge splitting-off algorithm in the global arc-connectivity setting. Their algorithm solves complete edge splitting-off problems for each vertex in any specified vertex subset. First, they split-off all the pairs incident to any vertex from the specified vertex subset. Then they applied Gabow's algorithm to construct edge-disjoint directionless r-spanning trees for any arbitrary vertex r. Whenever Gabow's algorithm halts with a cut with size less than k, they re-pair some edge pairs to increase the cut size back to k and then resume the algorithm. After constructing k trees, the graph is k-arc-connected and hence the edge pairs splitted-off are all admissible.

For any undirected graph G, we define its *directed version* to be the directed graph D obtained by replacing each edge of G to arcs of both directions. Clearly, the directed version of a k-edge-connected graph is k-arc-connected and hence has k arc-disjoint arborescences rooted at any vertex. By considering the directed version of an undirected graph, they solved complete edge splitting-off problems for each vertex in any specified vertex subset in the global edge-connectivity setting. Their algorithm has the same time bound as Gabow's construction algorithm, which is $\tilde{O}(km)$ for unweighted directed graph and $\tilde{O}(k^2n + m)$ for unweighted undirected graph. Note that Gabow's earlier result [36] also used the edge-disjoint directionless spanning tree for solving complete edge splitting-off problem. Bhalgat, Hariharan, Kavitha and Panigraji improved Gabow's result in a sense that they solved the problem for each vertex from any specified vertex subset while Gabow solved the problem for one specified vertex.

2.4.4 Local Edge-Connectivity Setting

In the global edge-connectivity setting, the complete edge splitting-off problem can be solved efficiently by splitting-off multiple edge pairs at once. However, it seems these

approaches cannot easily be extended to the local edge-connectivity setting. Recall that the key ideas supporting these approaches are all regarding to the global minimum cut. Instead of global minimum cuts, what is of concern in the local edge-connectivity setting are pairwise minimum cuts, i.e. the collection of minimum cuts separating each vertex pair. Local edge-connectivity for some highly connected vertex pairs may decrease even if the size of global minimum cut is preserved. It seems not hopeful, or at least not trivial, to use the properties regarding global minimum cut in the local edge-connectivity setting.

In the local edge-connectivity setting, the runtime of complete edge splitting-off algorithm is boosted either by finding an admissible edge pair more efficiently or by bounding the length of the splitting-off sequence. Based on these two ideas, Gabow [36] solved the complete edge splitting-off problem in $O(n^3 m \log(n^2/m))$ for a weighted graph and $O(kn^2 m)$ for a graph with maximum edge-connectivity requirement $r_{\max} = k$. The former algorithm bounds the length of a splitting-off sequence by a wise grouping of x-neighbours, followed by an analysis using submodular functions. The latter algorithm finds an admissible edge pair efficiently by using Gomory-Hu tree, a compact representation of pairwise minimum cuts. These results remain to be the best algorithms since 1994.

Chapter 3

Degree Bounded Network Design Problem with Metric Cost

The results in this chapter are based on joint work with Yuk Hei CHAN Wai Shing FUNG and Lap Chi LAU [14].

Degree bounded network design problems are interesting problems in combinatorial optimization, which also have applications in various areas such as VLSI design, vehicle routing and communication networks [20, 73, 6, 68, 67], where degree constraints are useful to bound the workload on nodes. Consider the problem of finding a minimum cost k-edge-connected subgraph with maximum degree at most B in a weighted undirected graph. This is a generalization of the travelling salesman problem when $k = B = 2$, and the minimum bounded degree spanning tree problem when $k = 1$.

Since this problem captures the travelling salesman problem as a special case, we can see that it does not admit any polynomial factor approximation algorithm as the feasible problem (Hamiltonian cycle problem) is NP-Hard. Recent research has thus focused on obtaining bicriteria approximation algorithms for degree bounded network design problems [38, 51, 70, 52].

In some network design problems the cost function satisfies triangle inequalities, and stronger algorithmic results are known [48, 17, 1]. For the travelling salesman problem, although there is no polynomial factor approximation algorithms in general, it is well-known that there is a 1.5-approximation algorithm assuming triangle inequalities [18]. This motivates us to study more general degree bounded network design problems with metric costs.

Definition 2.25. ***(Degree Bounded Network Design Problem with Metric Costs)*** *Given a graph $G = (V, E)$, an edge-connectivity requirement function $r : V \times V \rightarrow \mathbb{Z}$ on each vertex pair, a cost function $c : E \rightarrow \mathbb{Q}$ on each edge satisfying triangle inequality ($c(uv) + c(vw) \geq c(uw)$ for all u, v, w), and a degree upper bound B on each vertex v. Find a minimum cost subgraph $H = (V, F) \subseteq G$ that $\lambda_H(u, v) \geq r(u, v)$ for each two vertices $u, v \in V$, and the degree of each vertex in H is at most B.*

Like the problem in general cost, only bicriteria approximation algorithms were known for this problem. Here, we give the first constant factor approximation algorithms for various degree bounded network design problems with metric costs. In addition, some of these algorithms return solutions with smallest possible maximum degree (e.g. k-connected subgraphs with maximum degree k) and the cost is within constant time the optimal cost when there are no degree constraints. This demonstrates that degree constraints can be incorporated into network design problems with metric costs.

Theorem 3.1. *Given a complete graph with metric costs, there is a polynomial time $(2 + 1/k)$-approximation algorithm for the minimum bounded degree k-edge-connected subgraph problem.*

Theorem 3.2. *Given a complete graph with metric costs, there is a polynomial time $(4, +1)$-approximation algorithm for the minimum bounded degree Steiner network prob-*

lem. For $r_{\max}$ even, there is a polynomial time 6-approximation algorithm for the minimum bounded degree Steiner network problem[1].

This chapter is organized as follows. First, we will outline the approximation algorithm in Section 3.1. Then, we will present the technical detail of the algorithm in Section 3.3 and Section 3.2. Finally, we will show the time complexity in both the global and local edge-connectivity settings in Section 3.4.

3.1 Christofides'-like Algorithm

In this section, we will present an outline of the algorithm that the technical details are deferred to the next sections. Our algorithms can be seen as a generalization of Christofides' algorithm for metric TSP [18]. Christofides' algorithm first constructs a minimum spanning tree, then adds a minimum perfect matching between odd degree vertices, and finally short-cuts the high degree vertices without increasing the cost. Our approach is similar. We illustrate it in the global edge-connectivity setting. First we construct a k-edge-connected subgraph H (without degree constraints) by using an existing 2-approximation algorithm for the minimum cost k-edge-connected subgraph problem [47]. Then we apply a short-cutting procedure to transform H into a k-edge-connected subgraph H' of maximum degree $k+1$ without increasing the cost. Finally we add a minimum cost perfect matching to vertices with degree $k+1$ in H', and then apply the short-cutting procedure once again to transform it to a k-edge-connected subgraph H'' of maximum degree k.

To short-cut the high degree vertex x, we use the edge splitting-off operation, which we discussed in the previous section. With the metric cost assumption, this operation does not increase total edge cost, and so it can be used to decrease the degree of x by 2 without

[1]When $r_{\max}$ is odd, each connected component may have one vertex with degree $r_{\max}+1$.

increasing the cost. We are concerned with the simplicity of the solutions, and so require a *simplicity-preserving* edge splitting-off operation that maintains edge-connectivity and does not introduce new parallel edges.

The strategy for local edge-connectivity is similar. We remark that the procedure of reducing maximum degree by edge splitting-off operation was first used by Bienstock, Brickell and Monma [11], where they proved a similar result to Theorem 3.11(1) when parallel edges are allowed. Note that the edge splitting-off we consider in the rest of this thesis takes the edge-connectivity requirements of the designated vertex x into account. It is slightly more general than what we discussed in Mader's theorem in Section 2.2.2.

3.2 Simplicity-Preserving Edge Splitting-Off

To short-cut high degree vertices in our network design problem, we consider *simplicity-preserving* edge splitting off that does not introduce new parallel edges, that is, we do not allow splitting off xu, xv if the edge uv already exists. This was first studied by Bang-Jensen and Jordán [4], where they proved that if $d(x) = \Omega(k^2)$, then there exists a splittable edge pair; if $d(x) = \Omega(k^4)$, then there is a complete splitting-off on x that maintains k-edge-connectivity in the remaining graph. For degree bounded network design problems, there is no need for a complete edge splitting-off. The following theorem provides sufficient conditions that guarantee the existence of a simplicity-preserving edge splitting-off operation that maintains local edge-connectivity requirements (include those regarding x). As a remark, we will present an efficient procedure for this edge splitting-off operation in Section 6.6.

Theorem 3.3. *Suppose $N(x)$ is not a clique and $|N(x)| \geq r_{\max} + 2$. If $d(x) \neq 3$ and there is no cut-edge incident to x, then there is a simplicity-preserving edge splitting-off operation on x that maintains the local edge-connectivity for any pair of vertices $u, v \in V$.*

Before proving Theorem 3.3, we state some of the important notations and propositions as a recap. Two edges xu, xv form an *admissible pair* if the graph after splitting-off xu, xv does not violate $s(X) \geq 0$ for all $X \subset V$; and they form an *legal pair* if no new parallel edge is formed after the pair is splitted-off. In other words, we split-off *legal admissible pairs* in our algorithms. When we consider a pair of edges, they are assumed to be incident to x unless otherwise specified. For a vertex set $X \subseteq V - x$, X is called *tight* if $s(X) = 0$ and *dangerous* if $s(X) \leq 1$. A pair xu, xv is not admissible if and only if u, v are both contained in some dangerous set (Proposition 2.12). Finally, we mention the following proposition, which is useful in the proof of edge splitting-off result.

Proposition 3.4 ([4])**.** *Suppose there is no cut-edge incident to x. For any disjoint vertex sets S_1, S_2 with $d(S_1, S_2) = 0$ and $d(x, S_1) \geq 1$ and $d(x, S_2) \geq 1$, then $S_1 \cup S_2$ is not a dangerous set.*

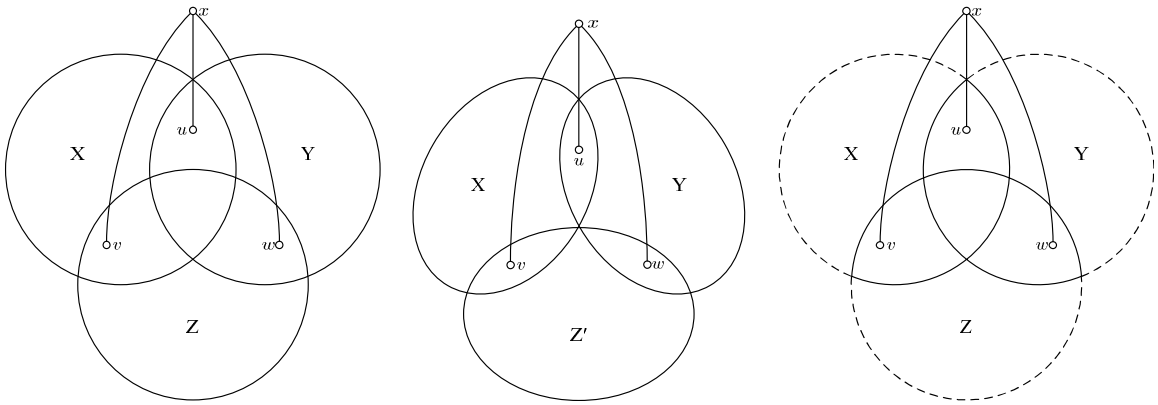
Proof. For any disjoint vertex sets S_1 and S_2 with $d(S_1, S_2) = 0$, we have

$$\begin{aligned} s(S_1 \cup S_2) &= d(S_1 \cup S_2) - r(S_1 \cup S_2) \\ &\geq d(S_1) + d(S_2) - \max\{r(S_1), r(S_2)\} \\ &\geq \min\{d(S_1), d(S_2)\}. \end{aligned}$$

Since $d(x, S_1) \geq 1$ and $d(x, S_2) \geq 1$, it follows that either $d(S_1) = 1$ or $d(S_2) = 1$, which implies a cut-edge incident to x, violating the assumption. □

3.2.1 Proof of Theorem 3.3

Suppose, by way of contradiction, that all the conditions in Theorem 3.3 are satisfied, but there is no legal admissible pair on x. We will prove in Lemma 3.7 that a certain 3-dangerous-set structure exists, see Figure 3.1(a) and Definition 3.5. Then we will prove



(a) (b) (c)

Figure 3.1: The 3-dangerous-set structures.

in Lemma 3.8 that such a 3-dangerous-set structure would imply that either $d(x) = 3$ or there is a cut-edge incident to x, violating the conditions in Theorem 3.3.

Definition 3.5. *A 3-dangerous-set structure characterize maximal dangerous sets X, Y, Z such that $u, v, w \in N(x)$ and $u \in X \cap Y$, $v \in X \cap Z$, $w \in Y \cap Z$ and $u, v, w \notin X \cap Y \cap Z$.*

First we need the following claim to establish the 3-dangerous-set structure.

Claim 3.6. *Suppose $|N(x)| \geq r_{\max} + 2$. Then for any dangerous set D, there exists a vertex $w \in N(x) \cap (V - D)$ with $d(w, D) = 0$.*

Proof. Suppose to the contrary that $d(v, D) \geq 1$ for each $v \in (V - D) \cap N(x)$.

$$\begin{aligned} d(D) &\geq d(x, N(x) \cap D) + d(D, N(x) \cap (V - D)) \\ &\geq |N(x) \cap D| + |N(x) \cap (V - D)| \\ &= |N(x)| \geq r_{\max} + 2, \end{aligned}$$

□

which contradicts the assumption that D is dangerous. Therefore there exists a vertex $w \in N(x) \cap (V - D)$ with $d(w, D) = 0$.

The following lemma shows that a certain 3-dangerous-set structure as shown in Figure 3.1(a) must exist, which is a crucial step in the proof.

Lemma 3.7. *Suppose $N(x)$ is not a clique and $|N(x)| \geq r_{\max} + 2$. If there is no legal admissible pair on x, then there exist maximal dangerous sets X, Y, Z and $u, v, w \in N(x)$ such that $u \in X \cap Y$, $v \in X \cap Z$, $w \in Y \cap Z$ and $u, v, w \notin X \cap Y \cap Z$.*

Proof. Since $N(x)$ is not a clique, there exist $u', v' \in N(x)$ with $u'v' \notin E$. (xu', xv') must be non-admissible because there is no legal admissible pair on x. By Proposition 2.12, there exists a dangerous set that contains both u' and v'. Let X be a maximal dangerous set containing u', v' such that $X \cap N(x)$ is not a proper subset of $D \cap N(x)$ for any dangerous set D.

By Claim 3.6, there exist $w' \in N(x) \cap (V - X)$ such that $u'w', v'w' \notin E$. As there is no legal admissible pair on x, both (xu', xw') and (xv', xw') must be non-admissible. By Proposition 2.12, there exist a dangerous set containing (u', w') and a dangerous set containing (v', w'). If there exist maximal dangerous sets Y and Z that $u', w' \in Y$, $v' \notin Y$ and $v', w' \in Z$, $u' \notin Z$, then we get the desired 3-dangerous-set structure.

Otherwise, there must exist a maximal dangerous set Y that $u', v', w' \in Y$. Since $\bar{d}(X, Y) \geq d(u' + v', x) \geq 2$, we have $s(X) + s(Y) \leq 1 + 1 < 2\bar{d}(X, Y)$. So inequality (2.5b) cannot hold for (X, Y), and thus inequality (2.5a) must hold for (X, Y). As X and Y are maximally dangerous, $X \cup Y$ cannot be dangerous and thus $s(X \cup Y) \geq 2$. Therefore, by inequality (2.5a), $s(X \cap Y) = d(X, Y) = 0$. By the definition of X, $X \cap N(x)$ is not a proper subset of $Y \cap N(x)$. Since $w' \in N(x) \cap Y$, there must exist $t' \in N(x) \cap X$ and $t' \notin Y$. From $d(X, Y) = 0$, we have $t'w' \notin E$, i.e. the edge pair is legal. Since (xt', xw') is not legal admissible there exists a maximal dangerous set Z containing both w' and t' for the pair to be non-admissible. We will show that both u' and v' are not in Z. By using this, we can define $u = u'$, $w = w'$, $v = t'$ and get the desired 3-dangerous-set structure.

We now complete the proof by showing that both u' and v' are not in Z. Suppose,

by way of contradiction, that $u' \in Z$. Then since $\bar{d}(Y,Z) \geq d(u'+w',x) \geq 2$ and $s(Y)+s(Z) \leq 1+1=2$, inequality (2.5b) does not hold for (Y,Z) and thus inequality (2.5a) must hold for (Y,Z). As Y and Z are maximal dangerous sets, $Y \cup Z$ cannot be a dangerous set and $s(Y \cup Z) \geq 2$. By inequality (2.5a) for (Y,Z), this implies that $d(Y,Z) = s(Y \cap Z) = 0$. Consider $Y \cap Z$ and X. Note that $\bar{d}(Y \cap Z, X) \geq d(u',x) \geq 1$ and $s(Y \cap Z) + s(X) \leq 0+1=1$. So inequality (2.5b) does not hold for $(Y \cap Z, X)$, and thus inequality (2.5a) must hold for $(Y \cap Z, X)$. Therefore we have $s((Y \cap Z) \cup X) \leq s(Y \cap Z) + s(X) = 1$, which implies that $(Y \cap Z) \cup X$ is a dangerous set. Since $w \in (Y \cap Z) - X$, this contradicts the maximality of X and completes the proof. □

The following lemma shows that the 3-dangerous-set structure in Lemma 3.7 (Figure 3.1(a)) would contradict with the conditions of Theorem 3.3[2]. This completes the proof of Theorem 3.3.

Lemma 3.8. *If there are maximal dangerous sets X, Y, Z and $u,v,w \in N(x)$ such that $u \in X \cap Y$, $v \in X \cap Z$, $w \in Y \cap Z$ and $u,v,w \notin X \cap Y \cap Z$, then either $d(x)=3$ or there is a cut-edge incident to x.*

Proof. The proof is divided into 2 cases: **Case 1:** Inequality (2.5a) holds for at least one of $(X,Y),(X,Z),(Y,Z)$. Without loss of generality, assume inequality (2.5a) holds for (X,Y). Since $w \notin Y-X$, by the maximality of X, $s(X \cup Y) \geq 2$. By inequality (2.5a) for (X,Y), this implies that $s(X \cap Y) = d(X,Y) = 0$ and $s(X \cup Y) = 2$. Consider $X \cap Y$ and Z. Suppose inequality (2.5a) holds for $(X \cap Y, Z)$, then $(X \cap Y) \cup Z$ will be dangerous, but this contradicts the maximality of Z since $u \in (X \cap Y) - Z$. Therefore, inequality (2.5b) must hold for $(X \cap Y, Z)$. Thus, $s(Z-(X \cap Y)) \leq s(X \cap Y) + s(Z) \leq 0+1=1$. Note that $Z-(X \cap Y)$ is non-empty since $v,w \in Z-(X \cap Y)$. This implies that $Z-(X \cap Y)$

[2]Similar structures also appear in [5, 8].

is dangerous.

Define $Z' = Z - (X \cap Y)$; hence $X \cap Y \cap Z' = \emptyset$, see Figure 3.1(b). Consider $X \cup Y$ and Z'. Note that $\bar{d}(X \cup Y, Z') \geq d(v + w, x) \geq 2$ and $s(X \cup Y) + s(Z') \leq 2 + 1 = 3$. So inequality (2.5b) does not hold for $(X \cup Y, Z')$, and thus inequality (2.5a) must hold. Since $w \in Z' - X$, by the maximality of X, $X \cup Y \cup Z'$ cannot be dangerous, and hence $s(X \cup Y \cup Z') \geq 2$. By inequality (2.5a) for $(X \cup Y, Z')$, this implies that $s((X \cup Y) \cap Z') = s((X \cap Z') \cup (Y \cap Z')) \leq 1$. Note that $d(X, Y) = 0$ implies that $d(X \cap Z', Y \cap Z') = 0$. Applying Proposition 3.4 with $S_1 := X \cap Z'$ and $S_2 := Y \cap Z'$, this shows that either xv or xw is a cut-edge, completing the proof of Case 1.

Case 2: Inequality (2.5a) does not hold for any pair $(X, Y), (X, Z), (Y, Z)$. In other words, inequality (2.5b) holds in these three pairs. Consider X and Y, $\bar{d}(X, Y) \geq d(u, x) \geq 1$ and $s(X) + s(Y) \leq 1 + 1 = 2$. By inequality (2.5b) for (X, Y), this implies that $s(X - Y) = s(Y - X) = 0$. Consider $X - Y$ and Z, $\bar{d}(X - Y, Z) \geq d(v, x) \geq 1$ and $s(X - Y) + s(Z) \leq 0 + 1 = 1$, and so inequality (2.5b) does not hold for $(X - Y, Z)$. Thus inequality (2.5a) must hold for $(X - Y, Z)$, and so $d((X - Y) \cup Z) \leq s(X - Y) + s(Z) \leq 0 + 1 = 1$. Therefore, $(X - Y) \cup Z$ is dangerous. By the maximality of Z, $X - Y - Z$ must be empty. Using similar argument, $Y - X - Z$ and $Z - X - Y$ are also empty, see Figure 3.1(c).

Since inequality (2.5b) holds for $(X, Y), (X, Z), (Y, Z)$ and X, Y, Z are all dangerous, $\bar{d}(X, Y) = d(u, x) = 1, \bar{d}(X, Z) = d(v, x) = 1, \bar{d}(Y, Z) = d(w, x) = 1$. Therefore $d(X \cup Y \cup Z, V - (X \cup Y \cup Z) - x) = 0$. Suppose $d(x) \neq 3$. Consider another x-neighbour t, then $t \in V - X \cup Y \cup Z$. From $\bar{d}(X, Y) = 1$, $ut \notin E$, i.e. the edge pair is legal. Since (xu, xt) is not legal admissible, there exists a dangerous set D containing u and t for the pair to be non-admissible. Applying Proposition 3.4 with $S_1 := D - (X \cup Y \cup Z)$ and

$S_2 := D \cap (X \cup Y \cup Z)$, this shows that there is a cut-edge incident to x. Therefore, either $d(x) = 3$ or there is a cut-edge incident to x. This completes the proof of Case 2, and thus the Lemma. □

3.3 Approximation Algorithms for Network Design Problems

With the simplicity preserving edge splitting-off operation, we can short-cut high degree vertices while maintaining connectivity requirements and preserving simplicity. Here, we present the approximation algorithms for degree bounded network design problems with metric costs. Our algorithms can be seen as a generalization of Christofides' algorithm on metric TSP. The following is an overview of the algorithm for the case of local edge-connectivity.

First we use Jain's algorithm [42] to compute a simple Steiner network whose cost is no more than twice the optimal cost. Note that there may be vertices with degree larger than $r_{\max}$. We plan to use the simplicity preserving edge splitting-off operation to short-cut those vertices. To do so we need to make sure that the conditions in Theorem 3.3 are satisfied. If $r_{\max} = 1$, there is a simple 2-approximation algorithm for the minimum bounded degree Steiner network problem. Hence we assume $r_{\max} \geq 2$, and thus $d(v) \neq 3$ when $|N(v)| \geq r_{\max} + 2$. We also augment the Steiner network so that each connected component is 2-edge-connected, with a double in the cost (see Section 3.4.2), and thus there is no cut-edge in the Steiner network. In Section 3.3.1, we show that if $|N(v)| \geq r_{\max} + 2$ and $N(v)$ is a clique, then we can remove redundant edges without violating any connectivity requirements and without introducing cut-edges.

With all the conditions satisfied, we can apply Theorem 3.3 on a vertex v with

$|N(v)| \geq r_{\max}+2$. Call a vertex $u \in V$ *r-even* if $d(u)$ has the same parity as $r_{\max}$, and call u *r-odd* if $d(u)$ has different parity than $r_{\max}$. For every r-even vertex u, by repeatedly applying Theorem 3.3, its degree can be reduced to at most $r_{\max}$ since $d(u) = |N(u)|$ in a simple graph. Similarly, for every r-odd vertex, its degree can be reduced to at most $r_{\max} + 1$ by repeatedly applying Theorem 3.3. Since the cost function satisfies triangle inequalities, the cost of the resulting Steiner network is no more than the cost of the initial Steiner network. This gives approximation algorithms with maximum vertex degree $r_{\max} + 1$.

To further reduce the maximum degree from $r_{\max} + 1$ to $r_{\max}$, we adds a matching between vertices like Christofides' algorithm does. Assume for simplicity that $r_{\max}$ is even, and thus the number of r-odd vertices is even. We add a minimum cost perfect matching on r-odd vertices to make them r-even, and so all the vertices with degree larger than $r_{\max}$ are of degree $r_{\max}+2$. Note that parallel edges may be created when we add a matching. In Section 3.3.3, we prove that the simplicity-preserving edge splitting-off operation can be performed on those vertices with degree $r_{\max}+2$ to maintain connectivity and *restore simplicity* again, so that the resulting graph is simple and has maximum degree $r_{\max}$. Now let us present the details for different settings.

3.3.1 Removing Redundant Edges

The following claim shows that whenever the neighbours of a vertex x form a clique and degree of x is high, we can always remove some edges without violating edge-connectivity requirements.

Claim 3.9. *If $d(x) \geq r_{\max} + 2$ and $N(x)$ is a clique, then for any $u, v \in N(x)$, we can remove uv, xu and xv without violating that $s(X) \geq 0$ for all $X \subset V$.*

Proof. Suppose a set $D \subset V$ with $d(D) < r(D)$ after removing the edges uv, xu, xv. By

symmetry, assume $x \in D$. For $d(D) < r(D)$, at least one of $u, v \in V - D$. Without loss of generality, assume $u \in V - D$. We have

$$d(D) \geq d(x, N(x) \cap (V - D)) + d(x, N(x) \cap D) = d(x) \geq r_{\max} \geq r(D).$$

This leads to a contradiction and shows removal of uv, xu and xv will not violate the condition. □

Note that in removing the three edges, no cut-edges is introduced. Furthermore, the parities of the degrees of x, u, v remain the same. Henceforth, we assume that whenever $d(x) \geq r_{\max} + 2$, then $N(x)$ is not a clique.

3.3.2 Perfect Matching

In the global edge-connectivity setting, we say a vertex is a *k-odd* vertex if it has degree of different parity than k. The following claim bounds the cost of a minimum cost perfect matching between k-odd vertices.

Claim 3.10. *The cost of a minimum cost perfect matching between k-odd vertices in a graph G is at most $EC_k(G)/k$, where $EC_k(G)$ denotes the optimal cost of a k-edge-connected subgraph.*

Proof. Let the set of k-odd vertices be T. First, assume that $|T|$ is even. When the cost function satisfies triangle inequalities, the cost of a minimum cost perfect matching between T is equal to the cost of a minimum T-join, where a T-join is a subgraph in which T is equal to the set of vertices with odd degree. Let H be a k-edge-connected subgraph with minimum cost. Since H is k-edge-connected, by setting $x_e = 1/k$ for each edge $e \in H$, it is a feasible solution to the up hull of the T-join polytope [69]. Since the T-join polytope is integral, this implies that the cost of a minimum cost perfect matching between T is at most $EC_k(G)/k$.

When $|T|$ is odd, then k is odd, and there is no k-regular subgraph. Given a specific vertex v, if $v \in T$, we set $T' := T - \{v\}$; if $v \notin T$, we set $T' := T \cup \{v\}$. Then we can add a minimum cost perfect matching between T', so that v is the only vertex with degree $k+1$ in the resulting graph. By the same argument, the cost of this matching is at most $EC_k(G)/k$. □

3.3.3 Edge Splitting-Off Restoring Simplicity

Suppose we are given a simple Steiner network with maximum degree $r_{\max}+1$. Suppose further that there is no cut-edge and number of r-odd vertices is even. We need to further reduce its maximum degree to $r_{\max}$. First we add a minimum cost perfect matching between the r-odd vertices. Note that the resulting Steiner network may then have parallel edges. We plan to apply edge splitting-off operations to reduce the maximum degree to $r_{\max}$ and furthermore restore the simplicity of the Steiner network.

Consider a vertex x with degree $r_{\max}+2$. We can assume that $d(x) \neq 3$ and $N(x)$ is not a clique by Claim 3.9. If there are no parallel edges incident to x, then we can apply Theorem 3.3 to reduce the degree of x to $r_{\max}$ without increasing the cost, while maintaining local edge-connectivity without introducing new parallel edges. Now consider the case when there are parallel edges incident to x. Let v be the unique neighbour of x so that there are two parallel edges between x and v. If x and v have at least $r_{\max}$ common neighbours (which includes the case that $N(x)$ is a clique), then there are $r_{\max}$ edge-disjoint paths between x and v, and so both parallel edges between x and v can be removed while keeping local edge-connectivity requirement for all pairs. So assume that x and v have at most $r_{\max}-1$ common neighbours. If there exists u so that $xu \in E$ and $vu \notin E$ and xu, xv are admissible, then this is a simplicity-preserving edge splitting-off operation to reduce the degree of x to $r_{\max}$ and there is no more parallel edges incident

to x. By repeatedly applying this operation, we can reduce the degree of every vertex to $r_{\max}$ while keeping connectivity requirements and restoring simplicity. It remains to prove that such a u must exist.

Suppose, by way of contradiction, that x has no neighbour u that (xu, xv) is a legal admissible pair. Let x, v share $r_{\max} - l$ common neighbours with $l \geq 1$. Denote by $\{u_1, u_2, \ldots, u_l\}$ the set of neighbours of x that is not adjacent to v. Since xu_i, xv are not admissible for all u_i, there exists a dangerous set D_i such that $u_i, v \in D_i, x \notin D_i$ for $1 \leq i \leq l$. Since one parallel edge between xv is added in the matching, this implies that D_i is tight before the addition of the matching for all i. Consider the Steiner network G before the addition of the matching. Since $\bar{d}_G(D_i, D_j) \geq d(x, v) = 1$, inequality (2.5b) cannot hold for (D_i, D_j), and thus inequality (2.5a) must hold for (D_i, D_j). This implies that the union of these tight sets is tight in G. Therefore, there exists a tight set T in G such that $u_i, v \in T, x \notin T$ for $1 \leq i \leq l$, and thus $d_G(x, T) \geq l + 1$. In addition, the $r_{\max} - l$ common neighbours of x and v provide $r_{\max} - l$ edge-disjoint paths between x and v in G, Therefore, $d_G(T) \geq r_{\max} + 1$, which contradicts that T is a tight set in G. This shows that such a u must exist, and thus the simplicity-preserving edge splitting-off operation can be applied to obtain a simple Steiner network with maximum degree $r_{\max}$.

3.4 Results in Different Settings

The Christofides'-like Algorithm gives constant factor approximation algorithms for various degree bounded network design problems with metric costs. In addition, these algorithms return solutions with smallest possible maximum degree (e.g. k-connected subgraphs with maximum degree k) and the cost is within a constant time the optimal cost when there are no degree constraints. This demonstrates that degree constraints can be incorporated into network design problems with metric costs.

3.4.1 Global Edge-Connectivity

We first consider the problem of finding a minimum cost k-edge-connected simple subgraph with metric costs. The main procedure is to transform any k-edge-connected simple subgraph into a k-edge-connected simple subgraph with maximum degree k, with only a small increase in the cost.

Theorem 3.11. *Given a complete graph $G = (V, E)$ with metric costs and any simple k-edge-connected subgraph H of G, there is a polynomial time algorithm to construct:*

(1) A simple k-edge-connected subgraph H' with maximum degree at most $k + 1$ and $cost(H') \leq cost(H)$.

(2) A simple k-edge-connected subgraph H'' with maximum degree k and $cost(H'') \leq cost(H) + EC_k(G)/k$, where $EC_k(G)$ is the cost of a minimum k-edge-connected subgraph of G[3]*.*

Proof. Given any k-edge-connected graph with $k \geq 2$, after removing redundant edges as in Section 3.3.1, we can apply Theorem 3.3 repeatedly to obtain a simple k-edge-connected graph with maximum degree $k + 1$, without increasing the cost. This proves (1). By Claim 3.10, we can add a perfect matching between k-odd vertices with cost at most $EC_k(G)/k$. Then, as in Section 3.3.3, we can apply the simplicity-preserving edge splitting-off operation once again to obtain a simple k-edge-connected subgraph with maximum degree k, without increasing the cost. This proves (2). □

Applying Khuller's 2-approximation algorithm [47], we can obtain a simple k-edge-connected subgraph as the initial graph for Theorem 3.11. This gives the result of The-

[3]When both k and $|V|$ are odd numbers, then it is impossible to have a k-regular-subgraph. In that case our algorithm can choose any vertex v in the graph, and returns a solution with v having degree $k + 1$ while all other vertices having degree k, which is best possible.

orem 3.1, the first constant factor approximation algorithm for the minimum bounded degree k-edge-connected subgraph problem.

We remark that if parallel edges are allowed in the solutions, then a similar statement as Theorem 3.11(1) is proved by Bienstock, Brickell, Monma [11]. However, for degree bounded network design problems, there are capacity constraints on edges and so their result can not be directly applied[4].

3.4.2 Local Edge-Connectivity

Theorem 3.11 can be extended to general edge-connectivity requirements. In the following let $r_{\max} := \max_{u,v} r(u,v)$ be the maximum edge-connectivity requirement, and call a subgraph H satisfying all connectivity requirements a Steiner network.

Theorem 3.12. *Given a complete graph $G = (V, E)$ with metric costs and any simple Steiner network H of G, there is a polynomial time algorithm to construct:*

(1) A simple Steiner network H' with maximum degree at most $r_{\max}+1$ and $cost(H') \leq 2 \cdot cost(H)$.

(2) A simple Steiner network H'' with maximum degree at most $r_{\max}$ and $cost(H'') \leq 3 \cdot cost(H)$, when $r_{\max}$ is even.[3]

Proof. Suppose we are given a Steiner network H. In order to apply Theorem 3.3 to short-cut the high degree vertices, we first augment the Steiner network so that each connected component is 2-edge-connected, and thus there is no cut-edge in the resulting Steiner network. One way to augment the graph is as follows: Double H to obtain H' so that each connected component of H is 2-edge-connected, then short-cut and remove redundant edges to make H' simple and each connected component 2-edge-connected.

[4]Incidentally, if parallel edges are allowed, then there is a simple constant factor approximation algorithm by taking $k/2$ copies of an approximate solution of metric TSP.

The cost of H' is at most twice the cost of H. We can then apply Theorem 3.3 to obtain a simple Steiner network which has maximum degree $r_{\max}+1$, without increasing the cost. This proves (1). In the following we assume $r_{\max}$ is even. We add a minimum cost perfect matching on r-odd vertices in each component of the current Steiner network. Then we apply Theorem 3.3 once again to obtain a simple Steiner network with maximum degree $r_{\max}$ as in Section 3.3.3. Note that the cost of the matching is at most the cost of H, which can be proved by standard doubling and short-cutting argument. Therefore, the cost of the resulting Steiner network is at most 3 times the cost of the initial Steiner network H, which proves (2). □

Applying Jain's algorithm [42], we can obtain a simple Steiner network, which has cost at most 2OPT, as the initial Steiner network H in Theorem 3.12. This proves Theorem 3.2, the first constant factor approximation algorithm for the minimum bounded degree Steiner network problem with metric costs.

Chapter 4

Constrained Edge Splitting-off

The results in this chapter are based on joint work with Lap Chi LAU [53].

In an edge splitting-off operation, we replace an edge pair (xu, xv) by a split edge uv while maintaining edge-connectivity. Mader's theorem (Theorem 2.6) shows this is feasible under natural assumptions. However, we may sometimes want the graph, arising from edge splitting-off operations, to satisfy some additional constraints as well. For example, it is very natural to require a bipartite graph to remain bipartite upon edge splitting-off operations. In this chapter, we will discuss constrained edge splitting-off problems that satisfy additional constraints.

Constrained edge splitting-off is a generalization of the edge splitting-off that not only edge-connectivity must be maintained but also the split edges have to satisfy some additional properties as well [4, 3, 44, 37, 59, 5]. Examples of this type include the simplicity-preserving edge splitting-off problem we studied in the previous chapter. Here, we consider complete constrained edge splitting-off problem that split-off the maximum number of edges. Note that the complete constrained edge splitting-off problem is NP-hard in general [43]. Therefore, we sometimes aim at approximate solutions instead, i.e. splitting-off most of the x-edges.

Mader's theorem guarantees a splittable edge pair always exists under natural assumption and this implies a polynomial time algorithm for complete edge splitting-off problem. Here, we look for sufficient conditions for the existence of a splittable edge pair in constrained edge splitting-off. And similarly, these give polynomial time approximation algorithms for the complete constrained edge splitting-off problem. These sufficient conditions and algorithms can be applied in the corresponding constrained edge-connectivity augmentation problems to give additive approximation algorithms. These will be discussed in detail in the next chapter.

Our constrained edge splitting-off results are based on a structural property of non-admissible edge pairs, which states that each x-edge is included in at most $2r_{\max}$ non-admissible edge pairs (Theorem 4.1). In other words, a high proportion of edge pairs are admissible when $d(x)$ is sufficiently large. It is intuitive that some of these admissible edge pairs also satisfy the additional constraints and these edge pairs are the qualified pairs in constrained edge splitting-off problems. Also, this simple structural property can be used in the design of efficient randomized procedure to split-off edges, as we will later show in Chapter 6. And as a remark, this property can be viewed as an extension of the classical result by Mader [56], which states that every x-edge is contained in at least one admissible edge pair (Theorem 2.7); and later results by Bang-Jensen et.al. [4] and Szigeti [72].

This chapter is organized as follows. First, we will cover some basic definitions and related works in Section 4.1. Then, we will present a short proof for the general constrained edge splitting-off lemma (Theorem 4.2) in Section 4.2. The lemma is based (solely) on structural properties of non-admissible edge pairs. This structural properties is a key result in this thesis that they are also used to speedup complete edge splitting-off algorithm in Chapter 6. We will cover the details in Section 4.3. After that, we will apply

our sufficient condition in some constrained edge splitting-off problems in Section 4.4. Finally, we end this chapter by some similar results in the global arc-connectivity setting in Section 4.5.

4.1 Preliminaries

Consider a constrained edge splitting-off problem. We say that an edge pair is *admissible* if the edge-connectivity requirements are satisfied upon splitting-off the pair; and it is *legal* if the addition of its corresponding split edge does not violate the additional constraint. Given an edge pair (xu, xv), the edges xu and xv are *non-admissible partner* to each other if the edge pair is non-admissible; and they are *illegal partner* to each other if the edge pair is illegal. The goal of this problem is to find an edge pair that satisfies both edge-connectivity requirements and additional constraint upon splitting-off. In order words, we look for *legal admissible* edge pairs. For some other definitions and notation, please refer to Section 2.2.

The constrained edge splitting-off problem is well-studied under several specific constraints [4, 3, 44, 37, 59, 5]. Some of these results were later generalized by Jordán, who presented a general framework by investigating the structural properties of non-admissible pairs [45]. This framework uses the so-called non-admissibility graph and constraint graph and is briefly described in Section 2.2.4.1. However, most of the work are done only for global edge-connectivity setting. The only established result in the local edge-connectivity setting is sufficient conditions in simplicity-preserving edge splitting-off problem by Bang-Jensen and Jordán [4]. Here, we focus on the following 3 specific constraints.

Vertex partition constraint: Consider a graph $G = (V, E)$ with a designated vertex x and a vertex partition $\mathcal{P} = \{P_1, P_2, \ldots, P_l\}$ on $V - x$. In partition-preserving edge

splitting-off problem, we have to make sure that no split edge is included in any vertex partition while satisfying edge-connectivity requirements. In the global edge-connectivity setting, Bang-Jensen, Gabow, Jordán and Szigeti [3] characterized the structure for which no complete edge splitting-off sequence exists. With this structure, they showed that there exists a splitting-off sequence with length at least $d(x)/2-2$ under natural assumptions; and there exists a complete edge splitting-off sequence by adding two extra x-edges.

Graph simplicity constraint: Consider a graph $G=(V,E)$ with a designated vertex x. In simplicity-preserving edge splitting-off problem, we have to make sure that the split edges are not in parallel with any other edges while satisfying edge-connectivity requirements. In the global edge-connectivity setting and under natural assumptions, Bang-Jensen and Jordán [4] showed that a legal admissible pair exists if $d(x) \geq k(k+1)$; and a complete edge splitting-off sequence exists if $d(x) \geq 3k^4$ and $d(x)$ is even. These results are also extended to the local edge-connectivity setting in the same paper, and these are the only results for constrained edge splitting-off problem in the local edge-connectivity setting.

Simultaneous graph constraint: Consider c graphs $G_i=(V,E_i+E')$ with a designated vertex x for $i=1,\ldots,c$ that $E'=\delta(x)$. In simultaneous-graph edge splitting-off problem, we have to make sure that the split edges satisfy edge-connectivity requirements in all the c graphs. Jordán considered this problem in the global edge-connectivity setting with $c=2$ [44]. He characterized the structure for which no complete edge splitting-off sequence exists. With this structure, he showed that there exists a splitting-off sequence with length at least $d(x)/2-2$ under natural assumptions; and there exists a complete edge splitting-off sequence by adding two extra x-edges.

As a remark, the structural properties of non-admissible edge pairs under local edge-connectivity setting are also studied by Bang-Jensen et.al. [4], Szigeti [72] and Bernáth

et.al. [8]. Interested reader can take a look at their papers.

4.2 General Constrained Edge Splitting-off Lemma

In this section, we will present a sufficient condition for the existence of a legal admissible edge pair in a general constrained edge splitting-off problem. To show the existence of a legal admissible edge pair, we can consider admissibility and legality separately. We prove a legal admissible edge pair exists by showing that the numbers of non-admissible edge pair and illegal edge pair are both bounded. With the 3-dangerous-set structure studied in the previous chapter, we get useful structural properties on any collection of maximal dangerous sets containing any specific x-neighbour $v \in N(x)$. These properties allow us to apply an inductive argument to show that each x-edge is included in at most $2r_{\max} - 2$ non-admissible pairs. This result is stated formally in the following theorem. The proof and the related structural properties will be covered in detail in the Section 4.3.

Theorem 4.1. *Consider a graph $G = (V, E)$ with a designated vertex x. The number of non-admissible partners for any given x-edge is at most $\max\{2r_{\max} - 2, r_{\max}\}$ if there is no cut-edge incident to x and $d(x) \neq 3$.*

Since each x-edge is included in $d(x) - 1$ edge pairs and at most $\max\{2r_{\max} - 2, r_{\max}\}$ of them are non-admissible, each x-edge is included in a number of admissible edge pairs

$$(d(x) - 1) - \max\{2r_{\max} - 2, r_{\max}\} = \min\{d(x) - 2r_{\max} + 1, d(x) - 1 - r_{\max}\}.$$

Therefore, there exists a legal admissible edge pair if there exists an x-edge that is included in at most than $\min\{d(x) - 2r_{\max}, d(x) - 3\}$ illegal edge pairs. This gives the following theorem, a sufficient condition in the general constrained edge splitting-off problem.

Theorem 4.2. *Consider an undirected graph $G = (V, E)$ with a designated vertex x. There exists a legal admissible pair in constrained edge splitting-off problem if (i) $d(x) \geq$*

$\max\{2r_{\max}, 4\}$, *(ii) there is no cut-edge incident to* x *and (iii) there exists an* x*-edge which is included in at most* $\min\{d(x) - 2r_{\max}, d(x) - 3\}$ *illegal edge pairs.*

We will present some constraints that satisfy the assumption in the above theorem in Section 4.4. This gives sufficient conditions for the corresponding constrained edge splitting-off problems.

4.3 Structural Properties of Non-Admissible Pairs

In this section, we will show that each edge has at most $\max\{2r_{\max} - 2, r_{\max}\}$ non-admissible partners. Given an edge pair (xv, xw), if it is a non-admissible pair, then there is a dangerous set D with $\{xv, xw\} \subseteq \delta(D)$ by Proposition 2.12, and we say such a dangerous set D covers xv and xw. Let P be the set of non-admissible partners of xv in the initial graph. Our goal is to show that $|P| \leq \max\{2r_{\max} - 2, r_{\max}\}$.

We first present an outline of the proof. Let $\mathcal{D}_P$ be a minimal set of maximal dangerous sets such that (i) each set $D \in \mathcal{D}_P$ covers the edge xv and (ii) each edge in P is covered by some set $D \in \mathcal{D}_P$. Since $d(D) \leq r_{\max} + 1$ and D covers xv for each $D \in \mathcal{D}_P$, each set in $\mathcal{D}_P$ can cover at most $r_{\max}$ non-admissible partners of xv. So the theorem follows immediately if $|\mathcal{D}_P| = 1$.

For $|\mathcal{D}_P| \geq 2$, we have $d(x, D) < d(D) \leq r_{\max} + 1$ for every $D \in \mathcal{D}_P$; for otherwise, either D or every other $D' \in \mathcal{D}_P$ is not a maximal dangerous set according to Proposition 4.4. Since D covers xv for every $D \in \mathcal{D}_P$, D covers at most $r_{\max} - 1$ other x-edges. So the theorem follows immediately if $|\mathcal{D}_P| = 2$.

The next step is to show that $|\mathcal{D}_P| \leq r_{\max} - 1$, where the proof use very similar ideas as in [4, 72]. When $|\mathcal{D}_P| \geq 3$, we show in Lemma 4.5 that inequality (2.5a) must hold for each pair of dangerous sets in $\mathcal{D}_P$. Since each dangerous set is connected by Proposition 3.4, this allows us to conclude in Lemma 4.7 that $|\mathcal{D}_P| \leq r_{\max} - 1$. By the

previous argument, this implies that $|P| \leq (r_{\max} - 1)^2$.

To improve this bound, we use a new inductive argument to show that $|P| \leq r_{\max} - 1 + |\mathcal{D}_P| \leq 2r_{\max} - 2$ for $r_{\max} \geq 2$ (the case is trivial for $r_{\max} = 1$ since since $|\mathcal{D}_P| = 1$ by Lemma 4.7). First we prove in Lemma 4.8 that there is an admissible pair (xa, xb) in P (so by definition $a, b \neq v$). By splitting-off (xa, xb), let $P' = P - \{xa, xb\}$ with $|P'| = |P| - 2$. In the resulting graph, we prove in Lemma 4.9 that $|\mathcal{D}_{P'}| \leq |\mathcal{D}_P| - 2$. Hence, by repeating this reduction, we can show that after splitting-off $\lfloor |\mathcal{D}_P|/2 \rfloor$ pairs of edges in P, the remaining edges in P is covered by one dangerous set if $|\mathcal{D}_P|$ is odd or by one tight set if $|\mathcal{D}_P|$ is even. Therefore, we can conclude that $|P| \leq r_{\max} - 1 + |\mathcal{D}_P| \leq 2r_{\max} - 2$, completing the proof.

In the following, we will first present some useful lemmas in edge splitting-off in Section 4.3.1. Then, we will prove the upper bound on $|\mathcal{D}_P|$ in Section 4.3.2 and provide the details of the inductive argument in the Section 4.3.3.

4.3.1 Some Useful Lemmas

Here, we state two simple propositions that will be used later.

Proposition 4.3 ([4])**.** *If $d(x, v) \geq 2$, (2.5a) holds for two dangerous sets X, Y containing v.*

Proof. Suppose to the contrary that inequality (2.5a) does not hold. Then inequality (2.5b) must hold for X and Y, which is impossible since

$$\begin{aligned} 1 + 1 &\geq s(X) + s(Y) \geq s(X - Y) + s(Y - X) + 2\bar{d}(X, Y) \\ &\geq s(X - Y) + s(Y - X) + 2d(x, v) \\ &\geq s(X - Y) + s(Y - X) + 2 \cdot 2 \end{aligned}$$

□

Proposition 4.4. *Suppose $d(x, D) = d(D)$ for a vertex subset D, then for any other maximal dangerous set D', either $D \cap N(x) \subseteq D'$ or $D \cap N(x) \subseteq V - D'$.*

Proof. Suppose to the contrary that there exists $u \in N(x) \cap D \cap D'$ and $v \in N(x) \cap (D - D')$. Since $d(x, D) = d(D)$, we have $d(D' \cap D, D' - D) = 0$. Let $S_1 = D' \cap D$ and $S_2 = D' - D$. By Proposition 3.4, $D' = S_1 \cup S_2$ is not a dangerous set, contradicting the definition of D'. □

4.3.2 An Upper Bound on $|\mathcal{D}_P|$

By contracting non-trivial tight sets, each edge in P is still a non-admissible partner of xv by Lemma 2.13. Henceforth, we will assume that all tight sets in G are singletons. Also we assume there is no cut-edge incident to x and $d(x) \neq 3$ as required in Theorem 4.2. Recall that $\mathcal{D}_P$ is a minimal collection of maximal dangerous sets such that (i) each set $D \in \mathcal{D}_P$ covers the edge xv and (ii) each edge in P is covered by some set $D \in \mathcal{D}_P$. Here, we will first give some characterizations for on the collection of $\mathcal{D}_P$ for $|\mathcal{D}_P|$ in Lemma 4.5 and Proposition 4.6. Then, we will use these characterizations to derive an upper bound on $|\mathcal{D}_P|$.

Lemma 4.5 ([4])**.** *If $|\mathcal{D}_P| \geq 3$, then inequality (2.5a) holds for every $X, Y \in \mathcal{D}_P$.*

Proof. Suppose, by way of contradiction, that inequality (2.5b) holds for X and Y. Then

$$\begin{aligned} 1 + 1 &\geq s(X) + s(Y) \geq s(X - Y) + s(Y - X) + 2\bar{d}(X, Y) \\ &\geq s(X - Y) + s(Y - X) + 2(d(x, v) + d(V - x - (X \cup Y), X \cap Y)) \\ &\geq s(X - Y) + s(Y - X) + 2 + 2d(V - x - (X \cup Y), X \cap Y). \end{aligned}$$

It implies that both $X - Y$ and $Y - X$ are tight and $d(V - x - (X \cup Y), X \cap Y) = 0$. Since tight sets are singletons, we let $X - Y = \{a\}$ and $Y - X = \{b\}$. By the minimality

of $\mathcal{D}_P$, a and b are both x-neighbours and any dangerous set $Z \in \mathcal{D}_P - X - Y$ contains none of a and b. Since $d(V - x - (X \cup Y), X \cap Y) = 0$ and $a, b \notin Z$, it follows that

$$d(Z - (X \cap Y \cap Z), X \cap Y \cap Z) = 0.$$

Let $S_1 = X \cap Y \cap Z$ and $S_2 = Z - X \cap Y \cap Z$, and thus $d(S_1, S_2) = 0$. Since $v \in X \cap Y \cap Z$, we have $d(x, S_1) \geq 1$. By the minimality of $\mathcal{D}_P$, there is an x-neighbour in $S_2 = Z - (X \cap Y \cap Z)$ and thus $d(x, S_2) \geq 1$. By Proposition 3.4, $Z = S_1 \cup S_2$ is not a dangerous set, contradicting the definition of $\mathcal{D}_P$. □

Proposition 4.6 ([72])**.** *If $|\mathcal{D}_P| \geq 3$, then $X \cap Y = \{v\}$ is a tight set for any $X, Y \in \mathcal{D}_P$.*

Proof. Since $X, Y \in \mathcal{D}_P$ are maximal dangerous sets, $X \cup Y$ is not a dangerous set. By Lemma 4.5, inequality (2.5a) holds for X and Y, and it follows that $X \cap Y$ is a tight set. Since each tight set is a singleton and $v \in X \cap Y$, the proposition follows. □

Lemma 4.7. $|\mathcal{D}_P| \leq r_{\max} - 1$ *if* $r_{\max} \geq 2$*, and* $|\mathcal{D}_P| = 1$ *for otherwise.*

Proof. First, suppose to the contrary that $|\mathcal{D}_P| \geq 2$ and $r_{\max} = 1$. Then every $X, Y \in \mathcal{D}_P$ share a common x-neighbour v, and there exist $u \in N(x) \cap (X - Y), w \in N(x) \cap (Y - X)$ by the minimality of $\mathcal{D}_P$, we have $d(x, X) < d(X)$. For otherwise, the vertex subset Y cannot be dangerous according to Proposition 4.4. Since each dangerous set in $\mathcal{D}_P$ covers at least 2 x-edges, this leads to a contradiction that

$$2 \leq d(x, X) < d(X) \leq 2.$$

Now, we consider the case $r_{\max} \geq 2$. If $|\mathcal{D}_P| \leq 2$, then the lemma holds since we have $r_{\max} \geq 2$. So we assume $|\mathcal{D}_P| \geq 3$. Then, by Proposition 4.6, we have $X \cap Y = \{v\}$ for any $X, Y \in \mathcal{D}_P$. For each set $X \in \mathcal{D}_P$, we have $d(x, v) \geq 1$ and $d(x, X - v) \geq 1$ by the minimality of $\mathcal{D}_P$. Therefore, we must have $d(v, X - v) \geq 1$; otherwise, by Proposition 3.4, it follows that X is not a dangerous set, contradicting the definition of

$\mathcal{D}_P$. By Proposition 4.6, $X - v$ and $Y - v$ are disjoint for each pair $X, Y \in \mathcal{D}_P$. Since $d(v, X - v) \geq 1$ for each $X \in \mathcal{D}_P$, it follows that $|\mathcal{D}_P| \leq d(v)$. By Proposition 4.6, $\{v\}$ is a tight set and thus $|\mathcal{D}_P| \leq d(v) - d(x, v) \leq r_{\max} - 1$. □

4.3.3 An Inductive Argument

The goal is to prove that $|P| \leq r_{\max} - 1 + |\mathcal{D}_P|$ if $r_{\max} \geq 2$. By Lemma 4.7, this holds if $d(x, X - v) = 1$ for every dangerous set $X \in \mathcal{D}_P$. Hence we assume that there is a dangerous set $A \in \mathcal{D}_P$ with $d(x, A - v) \geq 2$; this property will only be used at the very end of the proof. By Lemma 4.5, inequality (2.5a) holds for A and B for every $B \in \mathcal{D}_P$. By the minimality of $\mathcal{D}_P$, there exists an x-neighbour $a \in A$ which is not contained in any other set in $\mathcal{D}_P$. Similarly, there exists $b \in B$ which is not contained in any other set in $\mathcal{D}_P$. The following lemma shows that the edge pair (xa, xb) is admissible (proof is deferred to the end of this section).

Lemma 4.8. *For any $A, B \in \mathcal{D}_P$ satisfying inequality (2.5a), an edge pair (xa, xb) is admissible if $a \in A - B$ and $b \in B - A$.*

After splitting-off (xa, xb), let the resulting graph be G' and let $P' = P - \{xa, xb\}$. Clearly, since each edge in P' is a non-admissible partner of xv in G, every edge in P' is still a non-admissible partner of xv in G'. Furthermore, by contracting non-trivial tight sets in G', each edge in P' is still a non-admissible partner of xv by Lemma 2.13. Hence we assume all tight sets in G' are singletons. Let $\mathcal{D}_{P'}$ be a minimal set of maximal dangerous sets such that (i) each set $D \in \mathcal{D}_{P'}$ covers the edge xv and (ii) each edge in P' is covered by some set $D \in \mathcal{D}_{P'}$. The following lemma shows that there exists $\mathcal{D}_{P'}$ with $|\mathcal{D}_{P'}| \leq |\mathcal{D}_P| - 2$ (proof is deferred to the end of this section).

Lemma 4.9. *When $|\mathcal{D}_P| \geq 3$, the edges in P' can be covered by a set $\mathcal{D}_{P'}$ of maximal dangerous sets in G' such that (i) each set in $\mathcal{D}_{P'}$ covers xv, and (ii) each edge in P' is*

covered by some set in $\mathcal{D}_{P'}$, and (iii) $|\mathcal{D}_{P'}| \leq |\mathcal{D}_P| - 2$.

Recall that we chose A with $d(x, A-v) \geq 2$, and hence $d(x, v) \geq 2$ after the splitting-off and contraction of tight sets. Therefore, by Proposition 4.3, inequality (2.5a) holds for every two maximal dangerous sets in $\mathcal{D}_{P'}$. By induction, when $|\mathcal{D}_P| \geq 3$, we have $|P| = |P'| + 2 \leq r_{\max} - 1 + |\mathcal{D}_{P'}| + 2 \leq r_{\max} - 1 + |\mathcal{D}_P|$. In the base case when $|\mathcal{D}_P| = 2$ and $A, B \in \mathcal{D}_P$ satisfy (2.5a), the same argument in Lemma 4.9 can be used to show that the edges in P' is covered by one tight set after splitting-off (xa, xb), and thus $|P| = |P'| + 2 \leq r_{\max} - 1 + 2 \leq r_{\max} - 1 + |\mathcal{D}_P|$. This completes the proof that $|P| \leq r_{\max} - 1 + |\mathcal{D}_P|$, proving the theorem.

Proof of Lemma 4.8: Suppose, by way of contradiction, that (xa, xb) is non-admissible. Then, by Proposition 2.12, there exists a maximal dangerous set C containing a and b. We claim that $v \in C$; otherwise there exists a 3-dangerous-set structure, contradicting Lemma 3.8. Then $d(x, A \cap C) \geq d(x, \{v, a\}) \geq 2$, and so inequality (2.5b) cannot hold for A and C, since

$$1 + 1 \geq s(A) + s(C) \geq s(A - C) + s(C - A) + 2\bar{d}(A, C) \geq 0 + 0 + 2 \cdot 2.$$

Therefore, inequality (2.5a) must hold for A and C. Since A and C are maximal dangerous sets, $A \cup C$ cannot be a dangerous set, and thus

$$1 + 1 \geq s(A) + s(C) \geq s(A \cup C) + s(A \cap C) + 2d(A, C) \geq 2 + s(A \cap C) + 0,$$

which implies that $A \cap C$ is a tight set, but this contradicts the assumption that each tight set is a singleton as $\{v, a\} \subseteq A \cap C$. □

Proof of Lemma 4.9: We will use the dangerous sets in $\mathcal{D}_P$ to construct $\mathcal{D}_{P'}$. Since each pair of sets in $\mathcal{D}_P$ satisfies inequality (2.5a), we have $s(A \cup D) = 2$ before splitting-off (xa, xb) for each $D \in \mathcal{D}_P$. Also, before splitting-off (xa, xb), for $A, B, C \in \mathcal{D}_P$,

inequality (2.5b) cannot hold for $A \cup B$ and C because

$$2+1 = s(A\cup B)+s(C) \geq s((A\cup B)-C)+s(C-(A\cup B))+2\bar{d}(A\cup B, C) \geq 2+0+2\cdot 1,$$

where the last inequality follows since $v \in A \cap B \cap C$ and $(A \cup B) - C$ is not dangerous (as it covers the admissible edge pair (xa, xb)). Therefore inequality (2.5a) must hold for $A \cup B$ and C, which implies that $s(A \cup B \cup C) \leq 3$ since

$$2 + 1 = s(A \cup B) + s(C) \geq s((A \cup B) \cup C) + s((A \cup B) \cap C).$$

For A and B as defined before Lemma 4.8, since $s(A\cup B) = 2$ before splitting-off (xa, xb), $A \cup B$ becomes a tight set after splitting-off (xa, xb). For any other set $C \in \mathcal{D}_P - A - B$, since $s(A \cup B \cup C) \leq 3$ before splitting-off (xa, xb), $A \cup B \cup C$ becomes a dangerous set after splitting-off (xa, xb). Hence, after splitting-off (xa, xb) and contracting the tight set $A \cup B$ into v, each set in $\mathcal{D}_P - A - B$ becomes a dangerous set. Then $\mathcal{D}_{P'} = \mathcal{D}_P - A - B$ is a set of dangerous sets covering each edge in P', satisfying properties (i)-(iii). By replacing a dangerous set $C \in \mathcal{D}_{P'}$ by a maximal dangerous set $C' \supseteq C$ and removing redundant dangerous sets in $\mathcal{D}_{P'}$ so that it minimally covers P', we have found $\mathcal{D}_{P'}$ as required by the lemma. □

4.4 Non-Admissibility Graph and Constraint Graph

In this section, we will construct non-admissibility graph and constraint graphs similar to the ones by Jordán [45] to give an illustration for our sufficient condition. Note that these graphs take $N(x)$ the neighbourhood of x in the input graph as the vertex set in Jordán's work. Since our sufficient condition is on $d(x)$ but not $|N(x)|$, these graphs take $\delta(x)$ the set of x-edges as vertex set in our case.

Let the non-admissibility graph $B = (U, F)$ be a graph that takes the set of x-edges $\delta(x)$ to be the vertex set. If there exists l copies of edge xu, then U contains $u_1, u_2, \ldots, u_l$.

Two vertices are adjacent in B if and only if the corresponding edge pair is non-admissible in G. Since U is the set of x-edges and F corresponds to the non-admissible edge pairs, we have $d(u) \leq 2r_{\max} - 2$ for any $u \in B(U)$ according to Theorem 4.1.

Let the constraint graph $C = (U, F')$ be a graph that takes the same vertex set as B. Two vertices u_i, v_j are adjacent in C if and only if the corresponding split edge uv violates the given constraint. Therefore, a legal admissible edge pair exists if and only if there exist two vertices w_i, t_i that are non-adjacent to each other in both non-admissibility and constraint graphs. Since vertex degree is at most $2r_{\max} - 2$ in the non-admissibility graph, such vertex pair exists if the given constraint does not introduce too many edges in C.

We will demonstrate how to obtain sufficient conditions in some constrained edge splitting-off problems by this idea. Actually, direct application of Theorem 4.2 also gives the same result but the idea of constraint graph seems more illustrative. Since the admissibility and legality are considered separately in this general approach, it is not surprising that the sufficient conditions obtained are not best possible (tight). However, these sufficient conditions are still highly comparable to the tight ones that they only differ by constant coefficients. We will mention the tight sufficient conditions in Section 4.4.4.

4.4.1 Vertex Set Partition Constraint

Consider an undirected graph $G = (V, E)$ with a designated vertex x and vertex partition $\mathcal{P} = \{P_1, P_2, \ldots, P_l\}$ of $V - x$ and edge-connectivity requirements $r(u, v)$ for each two vertices $u, v \in V$. In partition-preserving edge splitting-off problem, we have to make sure that no split edge is included in any vertex partition, i.e. an edge pair (xu, xv) is illegal if and only if $u, v \in P_i$ for some i.

Lemma 4.10. *Consider an undirected graph $G = (V, E)$ with a designated vertex x and a vertex partition $\mathcal{P} = \{P_1, P_2, \ldots, P_l\}$ on $V - x$ that $d(x, P_i) \geq d(x, P_j)$ for $i \leq j$. There*

exists a legal admissible pair including an x-edge from P_1 in the partition-constrained edge splitting-off problem if $d(x) \geq 4r_{\max}$ and $P_1 \leq d(x)/2$ and there is no cut-edge incident to x.

Proof. First, we construct the constraint graph that u_i, v_j are connected if and only if the corresponding vertices $u, v \in G(V)$ are in the same partition. Since $d(x, P_i) \leq d(x)/2$, vertex degree is strictly less than $d(x)/2$ in the constraint graph $C = (U, F')$. On another hand, vertex degree is at most $\max\{2r_{\max} - 2, r_{\max}\}$ in the non-admissible graph $B = (U, F)$ since each x-edge is included in at most $\max\{2r_{\max} - 2, r_{\max}\}$ non-admissible edge pairs according to Theorem 4.1. Vertex degree in $(U, F \cup F')$ is strictly less than

$$\max\{d(x)/2 + 2r_{\max} - 2, d(x)/2 + r_{\max}\} \leq d(x) - 1.$$

Therefore, each vertex is non-adjacent to at least one other vertex in both graphs. This implies that each x-edge is included in at least one legal admissible edge pair. □

4.4.2 Graph Simplicity Constraint

Consider an undirected graph $G = (V, E)$ with a designated vertex x. In simplicity-preserving edge splitting-off problem, we have to make sure that the split edges are not in parallel with any other edges, i.e. an edge pair (xu, xv) is illegal if and only if $uv \in E$.

Lemma 4.11. *Consider an undirected graph $G = (V, E)$ with a designated vertex x. There exists a legal admissible pair in the simplicity-preserving edge splitting-off problem, if $d(x) \geq {r_{\max}}^2 + 2r_{\max} + 1$ and there is no cut-edge incident to x.*

Proof. First, we may assume that that $d(u, v) \leq r_{\max}$ for any $u, v \in N(x)$ since the extra copies of an edge are redundant for satisfying edge-connectivity requirements. On another hand, we can show that removal of (one copy of) an edge pair (xu, xv) does not violate edge-connectivity requirements if $|N(u) \cap N(x)|, |N(v) \cap N(x)| \geq r_{\max}$. Therefore, we

may assume that $|N(u) \cap N(x)| < r_{\max}$ holds for at least $|N(x)| - 1$ of all the x-neighbours $u \in N(x)$. Each of these corresponding x-edges is hence included in at most ${r_{\max}}^2$ illegal edge pairs, and this implies the vertex degree at most ${r_{\max}}^2$ for a number of vertices in the constraint graph. By Theorem 4.1, vertex degree is at most $2r_{\max} - 1$ in the non-admissible graph since each x-edge is included in at most $\max\{2r_{\max} - 2, r_{\max}\}$ non-admissible edge pairs. Vertex degree in $(U, F \cup F')$ is hence bounded by ${r_{\max}}^2 + 2r_{\max} - 1$. Therefore, there exist two vertices that are non-adjacent to each other in both graphs. They represent a legal admissible edge pair in the input graph. □

4.4.3 Simultaneous Graph Constraint

Consider c undirected graphs $G_i = (V, E_i + E')$ with a designated vertex x for $i = 1, \ldots, c$ that $E' = \delta(x)$. In simultaneous-graph edge splitting-off problem, we have to make sure that the split edges satisfy edge-connectivity requirements of all the c graphs, i.e. an edge pair is legal if and only if it is admissible in all c graphs. Let $r_{\max}$ be the maximum edge-connectivity among all the c graphs.

Lemma 4.12. *Consider c undirected graphs $G_i = (V, E_i + E')$ with a designated vertex x for $i = 1, \ldots, c$ that $E' = \delta(x)$. there exists a legal pair in the simultaneous-graph edge splitting-off problem if $d(x) \geq 2cr_{\max}$ and there is no cut-edge incident to x.*

Proof. Consider the non-admissibility graphs $B_i = (U, F_i)$ for each of G_i, $i = 1, \ldots, c$. By Theorem 4.2, vertex degree in each of these graphs is at most $\max\{2r_{\max} - 2, r_{\max}\}$. Vertex degree in the constraint graph $C = (U, \cup_{i=1,\ldots,c} F_i)$ is at most $\max\{2cr_{\max} - 2c, cr_{\max}\} \leq d(x)$. Therefore, each x-edge is included in a number of legal admissible edge pairs. □

4.4.4 Tight Sufficient Conditions

By consider the non-admissibility and legality together, we can get structural properties to tighten the sufficient conditions. The proof just involves some technical work and hence and covered here.

Lemma 4.13. *Consider an undirected graph $G = (V, E)$ with a designated vertex x and vertex partition $\mathcal{P} = \{P_1, P_2, \ldots, P_l\}$ that $d(x, P_i) \geq d(x, P_j)$ for $i \leq j$. There exists a legal admissible pair including an x-edge from P_1 in the partition-constrained edge splitting-off problem if $d(x) \geq \frac{4}{3}r_{\max} + 2 \geq 6$, $P_1 \leq d(x)/2$ and no cut-edge incident to x.*

Lemma 4.14. *Consider an undirected graph $G = (V, E)$ with a designated vertex x. there exists a legal admissible pair in simplicity-preserving edge splitting-off problem if no cut-edge incident to x, $d(x) > r_{\max}(r_{\max} + 1)/4$ and $d(x) \neq 3$.*

Note that the sufficient conditions in Lemma 4.13 and Lemma 4.14 are tight as there are examples achieving them (for arbitrary $r_{\max} > 1$). For the simultaneous-graph edge splitting-off problem, we cannot find any better structural property even if we consider non-admissibility and legality together. We hence do not have any sufficient condition better than the one in Lemma 4.12. On another hand, we also cannot construct any example to show that the sufficient condition is tight.

4.5 Global Arc-Connectivity Setting

The constrained edge splitting-off problem is also not very well-studied in the global arc-connectivity setting. The only established result is a sufficient condition in partition-preserving edge splitting-off problem done by Gabow and Jordán [37]. In this section, we will present a sufficient condition in the general constrained edge splitting-off problem; and then give results in the 3 specified constrained edge splitting-off problems.

Applying the approach used in the local edge-connectivity setting, we consider the non-admissibility and legality separately. By investigating the structural property of non-admissible edge pairs, we can show that the total number of non-admissible edge pair in the whole graph is bounded. This bound is stated formally in Lemma 4.15 and the proof is deferred to Section 4.5.1.

Lemma 4.15. *Consider a directed graph $D = (V, A)$ with a designated vertex x that $V - x$ is k-edge-connected and $d^+(x) = d^-(x)$. There are at most $d(x)^2/8$ non-admissible edge pairs in the graph. Furthermore, there are strictly less than $d(x)^2/8$ non-admissible edge pairs in the graph when $d(x) > 4k$.*

Since the total number of non-admissible edge pairs is bounded, there exists a legal admissible edge pair if the total number of illegal edge pairs is also bounded. This gives the following sufficient condition for the general constrained edge splitting-off problem in the global arc-connectivity setting.

Theorem 4.16. *Consider a directed graph $D = (V, A)$ with a designated vertex x that $V - x$ is k-edge-connected and $d^+(x) = d^-(x) > 2k$. There exists a legal admissible edge pair for constrained edge splitting-off operation if the constraint introduce at most $d(x)^2/8$ illegal edge pairs in the whole graph.*

Applying this general constrained edge splitting-off lemma, we get sufficient conditions in the following constrained edge splitting-off problems.

Lemma 4.17 (citegj01)**.** ***(Also in [37])*** *Consider a directed graph $D = (V, A)$ with a designated vertex x that $V-x$ is k-arc-connected, and vertex partition $\mathcal{P} = \{P_1, P_2, \ldots, P_l\}$ that $d(x, P_i) \leq d(x, P_j)$ for $i \leq j$. There exists a legal admissible pair including an x-edge from P_1 in partition-preserving edge splitting-off problem if $P_1 \leq d(x)/2$ and $d^+(x) = d^-(x) > 2k$.*

Lemma 4.18. *Consider a k-arc-connected directed graph $D = (V, A)$ with a designated vertex x that $V - x$ is k-arc-connected. There exists a legal admissible pair in simplicity-preserving edge splitting-off problem if $|N^+(x)|, |N^-(x)| > 2k$ and $\delta^+(x) = \delta^-(x)$. This sufficient condition can be tighten to $|N^+(x)|, |N^-(x)| > k$ and $d^+(x) = d^-(x)$ by a deeper analysis.*

Lemma 4.19. *Consider two directed graph $D_1 = (V, E_1 + E')$, $G_2 = (V, E_2 + E')$ and a designated vertex x that $E' = \delta(x)$ and D_1, D_2 are k-arc-connected and l-arc-connected respectively. There exists a legal admissible pair in simultaneous-graph edge splitting-off problem if $\delta^+(x) = \delta^-(x) > 2k$, where $k \geq l$.*

4.5.1 Proof of Lemma 4.15

By Claim 2.22, the union of tight sets containing $t \in N(x)$ is also tight. For each $v \in N(x)$, there exists at most one maximal tight set containing v. Let $\mathcal{T}$ be the collection of these tight sets and define $\mathcal{T}' = \{T \cap N(x) : T \in \mathcal{T}\}$. Each x-neighbours exists in at most one subset from $\mathcal{T}$, so $\mathcal{T}'$ is a subpartition of $N(x)$. Note that an edge pair (ux, xv) is not admissible if and only if u and v are in the same subset from $\mathcal{T}'$. In other words, the total number of non-admissible edge pairs equals

$$\sum_{T \in \mathcal{T}'} d^+(x, T) \cdot d^-(x, T).$$

With the definition of $\mathcal{T}'$, we can now show that the total number of non-admissible edge pairs is at most $d(x)^2/8$. Suppose that the number is at least $d(x)^2/8 = d^+(x)/2$. Then, there exists $u \in N^+(x)$ that ux has more than $\delta^-(x)/2$ non-admissible partners. This implies the existence of $U \in \mathcal{T}'$ that $\delta^-(x, U) \geq \delta^-(x)/2 = \delta(x)/4$. By the following simple proposition, we have $\delta^+(x, U) \leq \delta(x)/4$.

Proposition 4.20. *For any tight set, $\delta^+(x, T) + \delta^-(x, T) \leq \delta(x)/2$.*

We can now show the upper on the total number of non-admissible edge pairs as follows:

$$\begin{aligned}
&d^+(x,U)\cdot d^-(x,U) + \sum_{T\in\mathcal{T}'-U} d^+(x,T')\cdot d^-(x,T') \\
&\leq d^+(x,U)\cdot d^-(x,U) + d^+(x,V-U)\cdot d^-(x,V-U) \\
&\leq d^+(x)/2\cdot d^-(x)/2 + d^+(x)/2\cdot d^-(x)/2 \\
&= d(x)^2/8
\end{aligned}$$

Furthermore, the inequality holds with equality only if $d^+(x,U), d^+(x,V-U) = d^+(x)/2$ and $d^-(x,U), d^-(x,V-U) = d^-(x)/2$. These conditions can be satisfied only if $d(x) \leq 4k$; for otherwise, U cannot be tight. This proves the second part of the lemma.

Chapter 5

Constrained Edge-Connectivity Augmentation Problem

The results in this chapter are based on joint work with Lap Chi LAU [53].

In edge-connectivity augmentation problem, the goal is to find a minimum set of new edges so that the given edge-connectivity requirement is satisfied after adding the edge set. This optimization problem is very well-studied [34, 23, 74, 13, 65, 26] and there are a number of efficient algorithms for solving this problem [26, 36, 63, 7]. Apart from the edge-connectivity requirement, we may also want the resulting graph to preserve certain properties. Given a simple input graph, it is natural to require the resulting graph to remain simple. This gives rise to the study of constrained edge-connectivity augmentation problems, which are different extensions of the original problem that consider additional constraints together with the given edge-connectivity requirements.

Definition 2.28. ***(Constrained Edge-Connectivity Augmentation Problem)*** *Consider a graph $G = (V, E)$ with edge-connectivity requirement function $r : V \times V \to \mathbb{Z}$ on each vertex pairs. Find a minimum subset of edges F satisfying certain property P and that in the augmented graph $G' = (V, E + F)$, $\lambda_{G'}(u, v) \geq r(u, v)$ for each two vertices*

$u, v \in V$.

To solve edge-connectivity augmentation problems, Frank developed a general framework based on the edge splitting-off results. His framework was later modified to work with constrained edge splitting-off operations for solving constrained edge-connectivity augmentation problem in several settings [4, 3, 44, 37, 59]. Here, we will use this approach with the constrained edge splitting-off results obtained in the previous chapter. This gives additive approximation algorithms for some unsolved constrained edge-connectivity augmentation problems.

This chapter is organized as follows. First, we will cover some basic definitions and related works in Section 5.1. Then, we will present the additive approximation algorithms for some constrained edge-connectivity augmentation problems in the local edge-connectivity setting in Section 5.2. Finally, we end this chapter by some similar results in the global arc-connectivity setting in Section 5.3.

5.1 Preliminaries

Consider a constrained edge-connectivity augmentation problem with additional constraint C in an undirected graph $G = (V, E)$. The goal is to add a minimum set of new edges F to G in order to satisfy edge-connectivity, where the set of new edge F satisfies the given constraint C. We denote the optimal value, i.e. size of an optimal solution, of this constrained edge-connectivity augmentation problem by $\gamma_C(G)$; and denote the optimal value of the corresponding unconstrained edge-connectivity augmentation problem by $\gamma_S(G)$.

In edge-connectivity augmentation problem, Frank [26] developed a framework based on Mader's result [56] that a splittable edge pair always exists. This framework was later modified to solve constrained edge-connectivity augmentation problem by applying

constrained edge splitting-off results. The modified framework consists of the following 3 subroutines.

(1) **Constrained External Augmentation Subroutine:** Add an external vertex x and a minimum set of x-edges to satisfy the edge-connectivity requirements and some additional properties.

(2) **Constrained Complete Edge Splitting-off Subroutine:** Split-off as many x-edges by constrained edge splitting-off operations as possible.

(3) **Reconciliation Subroutine:** Replace any remaining x-edge by other edges while satisfying edge-connectivity requirements and the given constraint.

By applying appropriate constrained edge splitting-off operation (and edge replacement operation), the set of new edges are guaranteed to satisfy the given constraint. Note that the edge replacement operation may add extra edges and the algorithm may hence return an approximate solution. It is important to guarantee most of x-edges can be splitted-off in the second subroutine in order to reduce the number of extra edges added in the last subroutine. This is the reason for imposing an additional requirement in the first subroutine. Interested readers are referred to Section 2.3.2.1 and Section 2.3.2.2 for more details about both Frank's original framework and the modified framework. This modified framework is applied to solve the constrained edge-connectivity augmentation problem in several settings. Here, we focus on the following 3 specific constraints.

Vertex partition constraint: Consider an undirected graph $G = (V, E)$ with vertex partition $\mathcal{P} = \{P_1, P_2, \ldots, P_l\}$. In partition-preserving edge-connectivity augmentation problem, the objective is to add a minimum set of new edges F to G in order to satisfy edge-connectivity requirements, where every new edge has end-vertices from two different partitions. It is quite natural to require a graph to remain l-partite while in-

creasing edge-connectivity. Also, this augmentation problem has application in the graph rigidity [12]. In the global edge-connectivity setting, Bang-Jensen, Gabow, Jordán and Szigeti [3] solved the problem efficiently in $O((nm + n^2 \log n) \cdot \log n)$ time. In addition, they showed that the optimal value increases by at most one when we take vertex partition constraint into account. That is, $\gamma_C(G) \leq \gamma_S(G)+1$, where $\gamma_C(G)$ and $\gamma_S(G)$ are the size of an optimal solutions with and without vertex partition constraint. In the global arc-connectivity setting, Gabow and Jordán [37] gave an approximate solution with an additive cost of $O(k)$.

Graph simplicity constraint: Consider an undirected graph $G = (V, E)$. In simplicity-preserving edge-connectivity augmentation, the objective is to add a minimum set of new edges in order to satisfy edge-connectivity requirements, where every new edge is not in parallel with any other edge. Like the network design problem we discussed in Chapter 3, it is of practical interest to study the edge-connectivity augmentation problem that returns a simple graph. In the global edge-connectivity setting, Bang-Jensen and Jordán [4] gave an approximate solution with an additive cost of $O(k^2)$; and an exact solution when the optimal value is at least $O(k^4)$. In addition, they showed that the optimal value increases by at most $O(k^2)$ when we take simplicity constraint into account. That is, $\gamma_C(G) \leq \gamma_S(G) + O(k^2)$, where $\gamma_C(G)$ and $\gamma_S(G)$ are the size of an optimal solutions with and without simplicity constraint. They also extended these results to the local edge-connectivity setting in the same paper.

Simultaneous graph constraint: Consider two undirected graphs $G_i = (V, E_i)$ for $i = 1, 2$. In simultaneous-graph edge-connectivity augmentation problem, the objective is to add a minimum set of new edges in order to satisfy edge-connectivity requirements in both graphs, where every new edge is common to both graphs. In the global edge-connectivity setting, i.e. $r_{G_1}(u, v) \equiv k$, $r_{G_2}(u, v) \equiv l$, Jordán [44] gave an approximate

solution with an additive cost of 1; and an exact solution when k and l are both even.

5.2 Additive Approximation Algorithms

In this section, we will show how to modify Frank's framework to give additive approximation algorithms in 3 specific constrained edge-connectivity augmentation problems under local edge-connectivity setting. These algorithms are very similar to the established ones in the global edge-connectivity setting [3, 4, 44]. Our main efforts are (i) new constrained edge splitting-off results applied in second subroutine and (ii) edge replacement operations applied in the third subroutine. For the sake of simplicity, we assume the input graphs here to be connected to avoid the trouble of handling the so-called *marginal components* (Section 2.3.2.1).

5.2.1 Edge-Connectivity Augmentation Preserving Vertex Set Partition

Consider an undirected graph $G = (V, E)$ with vertex partition $\mathcal{P} = \{P_1, P_2, \ldots, P_l\}$. In partition-preserving edge-connectivity augmentation problem, the objective is to add a minimum set of new edges F to G in order to satisfy edge-connectivity requirements, where every new edge has end-vertices from two different partitions. By applying our sufficient condition in partition-preserving edge splitting-off problem obtained in the previous chapter, we get an additive approximation algorithm in the local edge-connectivity setting.

Lemma 5.1. *Under local edge-connectivity setting, there is a polynomial time algorithm for edge-connectivity augmentation problem with vertex set partition constraint that gives an approximate solution with an additive cost of at most* $\lceil \frac{1}{3} r_{\max} \rceil$.

(1) Constrained External Augmentation Subroutine: This subroutine is basically the same as the one used in the global edge-connectivity setting [3]. The objective is to add a small set of edges connecting to an external vertex x to satisfy the edge-connectivity requirements with the additional constraint that $\max_i \{d(x, P_i)\} \leq d(x)/2$. For the sake of convenience, we assume $d(x, P_i) \geq d(x, P_j)$ for $i \geq j$. To make sure $d(x, P_1) \leq d(x)/2$, we solve an (unconstrained) external augmentation problem, and perform two sets of operations if $d(x, P_1) > d(x)/2$. First, we try to swap x-edges to reduce the difference. That is, we remove x-edge incident to $u \in P_1$ and add x-edge incident to $v \notin P_1$ while keeping edge-connectivity requirements (Claim 5.2). Second, we add extra x-edges if no swapping can be performed. More precisely, $d(x, P_1) - d(x)/2$ copies of x-edges are added to $v \notin P_1$. It can be shown (as in [3]) that the number of x-edges is at most twice the optimal value of the partition-preserving edge-connectivity augmentation problem (Claim 5.3). This gives a solution for the constrained external augmentation subroutine.

(2) Constrained Complete Edge Splitting-off Subroutine: By Lemma 4.10, there exists a legal edge pair containing x-edge from P_1 if (i) $d(x) \geq 4r_{\max}$, (ii) $d(x, P_1) \leq d(x)/2$ and (iii) there is no cut-edge incident to x. First, Condition (ii) is satisfied in the solution returned by the first phase. And we assume the initial graph to be connected to satisfy Condition (iii). Therefore, we can apply constrained edge splitting-off that keeps $\max_{P_i \in \mathcal{P}} d(x, P_i) \leq d(x)/2$ as long as $d(x) \geq 4r_{\max}$.

(3) Reconciliation Subroutine: For the remaining x-edges, ignore the vertex partition constraint and use a complete splitting-off algorithm to split-off all of them. Pair-up these split edges that each pair consists of edge from different partitions. Then we replace each pair by three other edges to satisfy both edge-connectivity requirements and partition constraints (Claim 5.4). Note that this pairing is feasible since $\max_{P_i \in \mathcal{P}} d(x, P_i) \leq d(x)/2$. Since constrained edge splitting-off operation is applicable as long as $d(x) \geq 4r_{\max}$ in the

previous subroutine, there are less than $2r_{\max}$ illegal split edges. Therefore, less than $r_{\max}$ extra edges are added in this subroutine and this implies an approximate solution with an additive cost of $r_{\max}$. Note that we can get a smaller additive cost of $\lceil \frac{1}{3} r_{\max} \rceil$ if we apply a tighter sufficient condition (Lemma 4.13) in constrained complete edge splitting-off subroutine.

Claim 5.2. *Consider any two x-neighbour $u, v \in N(x)$ that u and v are in the same minimal tight set. Swapping one copy of xu to xv maintain the edge-connectivity requirements.*

Proof. Suppose to the contrary that some edge-connectivity requirement are violated after removing one copy of xu and adding one copy of xv. This implies the existence of a tight set T containing u but not v before the swapping. Consider T together with the minimal tight set T' containing u and v. Since $u \in T \cap T'$, inequality (2.5b) cannot hold for (T', T_u); for otherwise,

$$0 + 0 = s(T) + s(T') \geq s(T - T') + s(T' - T) + 2\bar{d}(T, T') \geq 0 + 0 + 2 \cdot 1.$$

Hence, inequality (2.5a) holds and this implies that $T' \cap T_u$ is tight since

$$0 + 0 = s(T) + s(T') \geq s(T \cup T') + s(T \cap T') + 2d(T, T') \geq 0 + s(T \cap T') + 2 \cdot 0.$$

With $v \in T - T'$, we have $T \cap T' \subset T'$. Therefore, $T \cap T'$ is a smaller tight set, contradicting the definition of T'. □

Claim 5.3. *After adding $d(x, P_1) - d(x)/2 \geq 0$ copies of x-edges, the number of x-edges is at most twice the optimal value of the partition-preserving edge-connectivity augmentation problem.*

Proof. Denote the collection of minimal tight sets containing x-neighbours from P_1 by $\mathcal{T}$. By Claim 5.2, each tight set in $\mathcal{T}$ is a subset of P_1 when no swapping can be done. Therefore, there exists a subpartition $\mathcal{C}$ of P_1 with total deficiency equals $d(x, P_1)$.

Now, suppose to the contrary that there exists a solution S to the partition-preserving edge-connectivity augmentation problem with size strictly less than $d(x, P_1)$. As explained in Section 2.3.2.1, $d_S(X)$ is not less than the deficiency for any vertex subset $X \in V$ in order to satisfy the edge-connectivity requirement. This implies $\sum_{X \in \mathcal{C}} d_S(X) \geq d(x, P_1)$. Therefore, some edges in S are connecting between vertices in P_i since $|S| < d(x, P_1)$ by the definition of S. This violates the vertex partition constraint and contradicts that S is a solution. We can now conclude that the size of any solution is not less than $d(x, P_1)$ and hence $d(x)$ is at most twice the optimal value after adding $d(x, P_1) - d(x)/2 \geq 0$ copies of x-edges. □

Claim 5.4. *For any two edges uv and wt that $u, v \in P_i, w, t \in V - P_i$, we can replace them by three other edges to satisfy both edge-connectivity requirements and vertex partition constraint.*

Proof. Add an extra vertex x and un-split-off uv, wt, i.e. add xu, xv, xw, xt and remove uv, wt. It is clear that (xu, xv) and (xw, xt) are admissible edge pairs. We claim that one of (xu, xw) and (xu, xt) is admissible after adding an extra copy of xu.

Suppose to the contrary that none of them is admissible. Then there exists dangerous sets D_1 that contains (u, w) and D_2 that contains (u, t). Inequality (2.5b) cannot hold for D_1 and D_2, for otherwise,

$$1 + 1 \geq s(D_1) + s(D_2) \geq s(D_1 - D_2) + s(D_2 - D_1) + 2\bar{d}(D_1, D_2) \geq 0 + 0 + 2 \cdot 2.$$

Since an extra copy of xu is added, $D_1 \cap D_2$, which contains u, cannot be tight. This implies that $D_1 \cup D_2$ is a dangerous set by considering inequality (2.5a).

$$1 + 1 \geq s(D_1) + s(D_2) \geq s(D_1 \cup D_2) + s(D_1 \cap D_2) + 2d(D_1, D_2) \geq s(D_1 \cup D_2) + 1 + 2 \cdot 0.$$

Since $w, t \in D_1 \cup D_2$, the edge pair (xw, xt) is hence non-admissible and leads to a contradiction. Therefore, one of (xu, xw) and (xu, xt) is admissible after adding an extra

copy of xu. We split-off the admissible pair among them, say (xu, xw), and then add an extra copy of xt. By the same argument, we can show that one of (xu, xt) and (xv, xt) is admissible, which actually implies a complete splitting-off sequence $\{(xu, xt), (xv, xt)\}$. The edges ut, vt and uw are the three edges that satisfy both edge-connectivity requirements and vertex partition constraint. □

5.2.2 Edge-Connectivity Augmentation Preserving Simplicity

Consider an undirected graph $G = (V, E)$. In simplicity-preserving edge-connectivity augmentation, the objective is to add a minimum set of new edges in order to satisfy edge-connectivity requirements, where every new edge is not in parallel with any other edge. By applying our sufficient condition in simplicity-preserving edge splitting-off problem obtained in previous chapter, we get an additive approximation algorithm in the local edge-connectivity setting.

Lemma 5.5. *Under local edge-connectivity setting, there is a polynomial time algorithm for simultaneous edge-connectivity augmentation problem that gives an approximate solution with an additive cost of at most* $r_{\max}(r_{\max} + 1)/8$.

Here, the approximate error is compared with the optimal solution in unconstrained edge-connectivity augmentation problem. In other words, this means that $\gamma_C(G) \leq \gamma_S(G) + r_{\max}(r_{\max} + 1)/8$, where $\gamma_C(G)$ and $\gamma_S(G)$ are the optimal values in augmentation problems with and without simplicity constraint respectively.

(1) External Augmentation Subroutine: Since the approximate solution is compared with the optimal value in unconstrained edge-connectivity augmentation problem, no extra requirements can be imposed on the set of x-edges. In other words, this is an (unconstrained) external augmentation subroutine that the objective is to add a small set of edges connecting to an external vertex x to satisfy the edge-connectivity requirements.

(2) Constrained Complete Edge Splitting-off Subroutine: By Lemma 4.11, there exists a legal edge pair if (i) $d(x) \geq {r_{\max}}^2 + 2r_{\max} + 1$ and (ii) there is no cut-edge incident to x. By assuming the initial graph to be connected to satisfy Condition (ii), we can apply constrained edge splitting-off as long as $d(x) \geq {r_{\max}}^2 + 2r_{\max} + 1$.

(3) Reconciliation Subroutine: For the remaining x-edges, ignore the graph simplicity constraint and use a complete splitting-off algorithm to split-off all of them. Then, we replace each of these illegal split edges by two other edges to satisfy both edge-connectivity requirements and simplicity constraints (Claim 5.6). Since constrained edge splitting-off operation is applicable as long as ${r_{\max}}^2 + 2r_{\max} + 1$, there are less than $O({r_{\max}}^2)$ illegal split edges. Therefore, less than $O({r_{\max}}^2)$ extra edges are added in this phase and this implies an approximate solution with an additive cost of $O({r_{\max}}^2)$. Note that we can get a smaller additive cost of $r_{\max}(r_{\max} + 1)/8$ if we apply a tighter sufficient condition (Lemma 4.14) in the constrained complete edge splitting-off subroutine.

Claim 5.6. *If there are parallel edges between $u, v \in V$, we can remove one copy of uv and add at most 2 other edges so that edge-connectivity requirements are preserved and no new parallel edge is formed.*

Proof. If either the removal of uv does not violate connectivity requirements or there exists $w \in V$ that $uw, vw \notin E$, it is clear we have to add at most 2 edges to maintain connectivity requirements. Thus, suppose neither of the above conditions holds. No desired vertex w exists implies $N(u) \cup N(v) = V$. Let $N'(v) = N(v) - (N(u) + u) = \{v_1, v_2, \dots\}$ and $N'(u) = N(u) - (N(v) + v) = \{u_1, u_2, \dots\}$. We claim that there exist vertices $v^* \in N'(v), u^* \in N'(u)$ that adding edge uv^* and u^*v maintains the connectivity requirements after removing one copy of uv.

Suppose to the contrary that no such v^*, u^* exists. This implies for every pair $v_i \in N'(v), u_j \in N'(u)$, there exists a maximal tight set $T_{i,j}$ that $u, v_i \in T_{i,j}$, $v, u_j \notin T_{i,j}$. We

claim that inequality (2.5b) cannot hold for any two tight sets X, Y if $X \cup Y \subset V$ and they share an x-neighbour. For otherwise,

$$0 + 0 = s(X) + s(Y) \geq s(X - Y) + s(Y - X) + 2\bar{d}(X, Y) \geq 0 + 0 + 2 \cdot 1.$$

Inequality (2.5a) hence holds for X and Y, which implies that $X \cup Y$ is a tight set:

$$0 + 0 = s(X) + s(Y) \geq s(X \cup Y) + s(X \cap Y) + 2d(X, Y) \geq s(X \cup Y) + 0 + 2 \cdot 0.$$

Since $u \in \cap T_{i,j}$ and $v \notin \cup T_{i,j}$, we can union all $T_{i,j}$ to give a tight set T'. Clearly, $\{u, N'(v)\} \subseteq T'$ and $\{v, N'(u)\} \subseteq V - T'$. Now, notice that

$$\begin{aligned} d(T') &\geq d(u, v) + d(v, N'(v)) + d(u, N'(u)) + |N(u) \cap N(v)| \\ &\geq d(u, v) + |N(v) \cup N(u) - \{u, v\}| \\ &\geq 2 + (|V| - 2) \\ &\geq r_{\max} + 1. \end{aligned}$$

This contradicts the definition of tight set and shows the desired u^*, v^* must exist. □

5.2.3 Simultaneous-Graph Edge-Connectivity Augmentation

Consider two undirected graphs $G_i = (V, E_i)$ for $i = 1, 2$. In simultaneous-graph edge-connectivity augmentation problem, the objective is to add a minimum set of new edges in order to satisfy edge-connectivity requirements in both graphs, where every new edge is common to both graphs. By applying our sufficient condition in simultaneous-graph edge splitting-off problem obtained in previous chapter, we get an additive approximation algorithm in the local edge-connectivity setting.

Lemma 5.7. *Under local edge-connectivity setting, there is a polynomial time algorithm for simultaneous edge-connectivity augmentation problem that gives an approximate solution with an additive cost of at most* $2r_{\max}$.

(1) Constrained External Augmentation Subroutine: This subroutine is basically the same as the one used in the global edge-connectivity setting [44]. The objective is to add a small set of edges connecting to an external vertex x to satisfy the edge-connectivity requirements in both graph that every new edge is common to the 2 graphs. Let $z(A)$ be the number of parallel edges between $A \subseteq V$ and x after adding x-edges. As shown in Section 2.3.2.1, $z(A)$ is not less than the deficiency of the vertex set A for every $A \subseteq V$ to satisfy edge-connectivity requirements. Frank [26] showed that the set of feasible z defines a contra-polymatroid C. Finding a minimum set of x-edges to satisfy edge-connectivity requirements is equivalent to finding an integer-valued vector of C with $z(V)$ minimum. Similarly, finding a minimum set of x-edges to satisfy edge-connectivity requirements of both G_1 and G_2 is equivalent to finding an integer-valued vector common to C_{G_1} and C_{G_2} with minimum $z(V)$. By Frank's g-polymatroid intersection theorem [25, 30], the system of vectors common to C_{G_1} and C_{G_2} is a submodular flow system. Therefore, we can find a minimum integer-valued vector in polynomial time by submodular flow algorithms.

(2) Constrained Complete Edge Splitting-off Subroutine: By Lemma 4.12, there exists a legal edge pair if (i) $d(x) \geq 4r_{\max}$ and (ii) there is no cut-edge incident to x. By assuming the initial graph to be connected to satisfy Condition (ii), we can apply constrained edge splitting-off as long as $d(x) \geq 4r_{\max}$.

(3) Reconciliation Subroutine: For the remaining x-edges, duplicate each of them so that each of G_1 and G_2 has a separate copy of x-edges. This reduces the problem to two (unconstrained) edge splitting-off problems that we can apply a complete splitting-off algorithm to split-off all of them. Since constrained edge splitting-off operation is applicable as long as $d(x) \geq 4r_{\max}$ in the previous subroutine, there are less than $2r_{\max}$ extra split edges. This implies an approximate solution with an additive cost of $2r_{\max}$.

5.3 Global Arc-Connectivity Setting

In this section, we will study the 3 constrained edge-connectivity augmentation problems under the global arc-connectivity setting. With the constrained edge splitting-off results obtained in the previous chapter (Section 4.5), we can modify Frank's framework to give additive approximation algorithms for these problems easily.

5.3.1 Edge-Connectivity Augmentation Preserving Vertex Set Partition

Consider a directed graph $D = (V, A)$ with vertex partition $\mathcal{P} = \{P_1, P_2, \ldots, P_l\}$ and global arc-connectivity requirement k. In partition-preserving edge-connectivity augmentation problem, the objective is to add a minimum set of new edges F to G in order to satisfy edge-connectivity requirements, where every new edge has end-vertices from two different partitions. By applying our sufficient condition in partition-preserving edge splitting-off problem obtained in previous chapter, we get an additive approximation algorithm in the global arc-connectivity setting. Note that the same result was obtained by Gabow and Jordán [37].

Lemma 5.8 ([37])**.** *Under global arc-connectivity setting, there is a polynomial time algorithm for edge-connectivity augmentation problem with vertex set partition constraint that gives an approximate solution with an additive cost of at most k.*

(1) Constrained External Augmentation Subroutine: This subroutine is basically the same as the one used in the local edge-connectivity setting we discussed in Section 5.2.1. The objective is to add a small set of edges connecting to an external vertex x to satisfy the edge-connectivity requirements with the additional constraint that $\max_i \{d(x, P_i)\} \leq d(x)/2$ and $d^+(x) = d^-(x)$. For the sake of convenience, we assume

$d(x, P_i) \geq d(x, P_j)$ for $i \geq j$. To make sure $d(x, P_1) \leq d(x)/2$, we solve an (unconstrained) external augmentation problem, and perform two sets of operations if $d(x, P_1) > d(x)/2$. First, we try to swap x-edges to reduce the difference. That is, we remove x-edge incident to $u \in P_1$ and add x-edge incident to $v \notin P_1$ while keeping edge-connectivity requirements. Second, we add extra x-arcs if no swapping can be performed (while minimizing $|d^+(x) - d^-(x)|$). More precisely, $d(x, P_1) - d(x)/2$ copies of x-arcs are added to $v \notin P_1$. It can be shown that the number of x-edges is at most twice the optimal value of the partition-preserving edge-connectivity augmentation problem. Finally, some extra x-edges are added to make $d^+(x) = d^-(x)$. This gives a solution for the constrained external augmentation subroutine.

(2) Constrained Complete Edge Splitting-off Subroutine: By Lemma 4.17, there exists a legal admissible edge pair containing an x-edge from P_1 if (i) $d^+(x), d^-(x) > 2k$ (ii) $P_1 \leq d(x)/2$ and (iii) $d^+(x) = d^-(x)$. Both the second and the third conditions are satisfied in the solution returned by the first phase. Therefore, we can apply constrained edge splitting-off that keeps $\max_{P_i \in \mathcal{P}} d(x, P_i) \leq d(x)/2$ as long as $d^+(x), d^-(x) > 2k$.

(3) Reconciliation Subroutine: For the remaining x-arcs, ignore the vertex partition constraint and use a complete splitting-off algorithm to split-off all of them. Then, we replace each two of these illegal split edges by at most three other edges to satisfy both edge-connectivity requirements and partition constraints. Since constrained edge splitting-off operation is applicable as long as $d(x) > 4k$, there are at most than $2k$ illegal split edges. Therefore, at most k extra edges are added in this phase and this implies an approximate solution with an additive cost of at most k.

5.3.2 Edge-Connectivity Augmentation Preserving Simplicity

Consider a directed graph $D = (V, A)$ with global arc-connectivity requirement k. In simplicity-preserving edge-connectivity augmentation, the objective is to add a minimum set of new edges in order to satisfy edge-connectivity requirements, where every new edge is not in parallel with any other edge. By applying our sufficient condition in simplicity-preserving edge splitting-off problem obtained in previous chapter, we get an additive approximation algorithm in the global arc-connectivity setting.

Lemma 5.9. *Under global arc-connectivity setting, there is a polynomial time algorithm for edge-connectivity augmentation problem with simplicity constraint that gives an approximate solution with an additive cost of at most* $k^2/2$.

Here, the approximate error is compared with the optimal solution in unconstrained edge-connectivity augmentation problem. In other words, this also implies that $\gamma_C(D) \leq \gamma_S(D) + k^2/2$, where $\gamma_C(D)$ and $\gamma_S(D)$ are the optimal values in augmentation problems with and without simplicity constraint respectively. Note that the additive error can be reduce to around $k^2/8$ by a deeper analysis.

(1) External Augmentation Subroutine: Since the approximate solution is compared with the optimal value in unconstrained edge-connectivity augmentation problem, no extra requirements can be imposed no the set of x-edges. In other words, this is an (unconstrained) external augmentation subroutine that the objective is to add a small set of edges connecting to an external vertex x to satisfy the edge-connectivity requirements, where $d^+(x) = d^-(x)$.

(2) Constrained Complete Edge Splitting-off Subroutine: By Lemma 4.18, there exists a legal admissible edge pair if (i) $|N^+(x)|, |N^-(x)| > k$ and (ii) $d^+(x) = d^-(x)$. The second condition is satisfied in the solution returned by the first phase. Therefore,

we can apply constrained edge splitting-off as long as $|N^+(x)|, |N^-(x)| > k$, which can be implied by $d^+(x), d^-(x) > k^2$.

(3) Reconciliation Subroutine: For the remaining x-arcs, ignore the graph simplicity constraint and use a complete splitting-off algorithm to split-off all of them. Then, we replace each of these illegal split edges by two other edges to satisfy both edge-connectivity requirements and simplicity constraints. Since constrained edge splitting-off operation is applicable as long as $d^+(x), d^-(x) > k^2$, there are at most k^2 illegal split edges. Therefore, at most $k^2/2$ extra edges are added in this phase and this implies an approximate solution with an additive cost of at most $k^2/2$.

5.3.3 Simultaneous Edge-Connectivity Augmentation

Consider two directed graphs $D_i = (V, A_i)$ for $i = 1, 2$ with global arc-connectivity requirement k and l respectively (assume $k \geq l$). In simultaneous-graph edge-connectivity augmentation problem, the objective is to add a minimum set of new edges in order to satisfy edge-connectivity requirements in both graphs, where every new edge is common to both graphs. By applying our sufficient condition in simultaneous-graph edge splitting-off problem obtained in previous chapter, we get an additive approximation algorithm in the global arc-connectivity setting.

Lemma 5.10. *Under global arc-connectivity setting, there is a polynomial time algorithm for simultaneous edge-connectivity augmentation problem that gives an approximate solution with an additive cost of* $2k$.

(1) Constrained External Augmentation Subroutine: The objective is to add a small set of edges connecting to an external vertex x to satisfy the arc-connectivity requirements in both graph that every new edge is common to the 2 graphs. This subroutine is very similar to the one used in the local edge-connectivity setting that we use

two submodular flow procedures separately to find a minimum set of x-in-edges and a minimum set of x-out-edges to satisfy the given arc-connectivity requirements. Finally, some extra x-edges are added to make $d^+(x) = d^-(x)$.

(2) Constrained Complete Edge Splitting-off Subroutine: By Lemma 4.19, there exists a legal admissible edge pair if (i) $d(x) > 4k$ and (ii) $d^+(x) = d^-(x)$. The second condition is satisfied in the solution returned by the first phase. Therefore, we can apply constrained edge splitting-off as long as $d(x) > 4k$.

(3) Reconciliation Subroutine: For the remaining x-arcs, duplicate each of them so that each of D_1 and D_2 has a separate copy of x-edges. This reduces the problem to two (unconstrained) edge splitting-off problems that we can apply a complete splitting-off algorithm to split-off all of them. Since constrained edge splitting-off operation is applicable as long as $d(x) > 4k$ in the previous subroutine, there are less than $2k$ extra split edges. This implies an approximate solution with an additive cost of $2k$.

Chapter 6

Efficient Edge Splitting-off Algorithm

The results in this chapter are based on joint work with Lap Chi LAU [53].

In the previous chapters, we have covered some applications of edge splitting-off technique. These results are basically sufficient conditions for the existence of splittable edge pairs. In this chapter, we will study how to find such splittable edge pairs efficiently in the local edge-connectivity setting. The main technical tools here are efficient algorithms by Bhalgat, Hariharan, Kavitha, and Panigrahi [10] and structural properties of non-admissible pairs.

Construction of a complete edge splitting-off sequence has been the subject of much research [36, 63, 7, 10] and has found many applications in graph connectivity problems, e.g. connectivity augmentation [74, 26, 16], graph orientation [29, 28], Steiner tree packing [2, 49, 50, 10, 15], etc. A straightforward approach to construct a complete edge splitting-off sequence is to split-off any edge pair $(e = xu, f = xv)$ without knowing the admissibility and then un-split it if any edge-connectivity requirement is violated. Note that taking wild guesses may result in $O(|N(x)|^2)$ splitting-off attempts. Algorithms

taking this approach may run very slowly.

We consider the complete splitting-off problem in the local edge-connectivity setting for unweighted graphs. The fastest algorithm in the local edge-connectivity setting an $O({r_{\max}}^2 \cdot n^3)$-time algorithm by Gabow [36]. We will present two algorithms that improves the running time of Gabow's algorithm by a factor of $\tilde{\Omega}(n)$. In many applications, it is a valid assumption that $r_{\max}$ is small, and the improvement of the randomized algorithm is more significant.

Theorem 6.1. *In the local edge-connectivity setting, there is a deterministic* $\tilde{O}(m+{r_{\max}}^2 \cdot n^2)$*-time algorithm and a randomized* $\tilde{O}(m + {r_{\max}}^3 \cdot n)$*-time algorithm for the complete splitting-off problem in unweighted graph.*

Our algorithms are conceptually very simple. They are based on the same framework that finds a complete edge splitting-off sequence by using at most $O(|N(x)|)$ splitting-off attempts (instead of $O(|N(x)|^2)$ attempts by the straightforward algorithm). The time complexity hence depends on the efficiency of the subroutine for splitting-off attempt. This framework gives a deterministic $\tilde{O}(m + {r_{\max}}^2 \cdot n^2)$-time algorithm by using the deterministic tree packing algorithm by Bhalgat et.al. [9]; and gives a randomized $\tilde{O}(m + {r_{\max}}^3 \cdot n)$-time algorithm by using the randomized Gomory-Hu tree algorithm by Bhalgat et.al. [9] (together with some further structural properties of non-admissible pairs).

This chapter is organized as follows. We will first cover some basic background and related works in Section 6.1, and then present the framework that finds a complete edge splitting-off sequence by using $O(|N(x)|)$ splitting-off attempts in Section 6.2. Then, in Section 6.3, we show how to efficiently perform one edge splitting-off attempt, by using some preprocessing steps and applying some fast algorithms to check edge-connectivities. Combining these two steps yields an $\tilde{O}({r_{\max}}^2 \cdot n^2)$ randomized algorithm for the complete splitting-off problem. After that, we will show how to speedup the algorithm by using

further structural properties of non-admissible pairs in a randomized splitting-off procedure, obtaining an $\tilde{O}({r_{\max}}^3 \cdot n)$ randomized algorithm for the problem in Section 6.4. Then, we will show how to modify some steps in the algorithm presented in Section 6.3 to obtain an $\tilde{O}({r_{\max}}^2 \cdot n^2)$ deterministic algorithm for the problem in Section 6.5. Finally, we will end this chapter by mentioning some results by applying of our framework in some other settings in Section 6.6. These include efficient edge splitting-off procedures for the network design problem and constrained edge splitting-off problems we dicussed in previous chapters.

6.1 Preliminaries

We first state some notations used in the algorithms. Since an edge pair (xu, xv) is determined by their end-vertices u and v, we can consider vertex pairs instead of edge pairs in a splitting-off problem. A vertex subset is called a *non-admissible set* if every vertex pair inside the set is non-admissible. We define the *capacity* of an edge pair to be the number of copies of the edge pair that can be splitted-off while satisfying edge-connectivity requirements. In our algorithms we will always split-off an edge pair to its capacity (which could be zero), and only attempt at most $O(|N(x)|)$ pairs. Following the definition of Gabow [36], we say that a splitting-off operation *voids* a vertex u if $d(x, u) = 0$ after the splitting-off.

Throughout the complete splitting-off algorithm, we assume that there is no cut-edge incident to x. This holds at the beginning by our assumption, and so the local edge-connectivity between x and v is at least two for each x-neighbour v. Therefore, we can reset the connectivity requirement between u and v as $\max\{r(u, v), 2\}$. Since pairwise edge-connectivity requirements are preserved by edge splitting-off operations, the set of x-neighbours remains 2-edge-connected, implying that there is no cut-edge incident to x

at each step.

Several efficient algorithms are proposed for the complete edge splitting-off problem [63, 7, 10], but only Gabow's algorithm [36] can be used in the local edge-connectivity setting. He found an admissible edge pair by $O(n)$ maximum flow computation. By splitting-off admissible pairs iteratively, he solved the complete edge splitting-off problem in $O({r_{\max}}^2 \cdot n^3)$ time in unweighted graph and in $\tilde{O}(n^3 \cdot m)$ time in weighted graph. In the global edge-connectivity setting, multiple edge pairs are splitted-off simultaneously to get much faster algorithms [63, 7, 10]. The key properties supporting these algorithms are all regarding to minimum cuts, and hence cannot be applied (directly) in the local edge-connectivity setting. Interested readers are referred to Section 2.4 for more details.

6.1.1 Efficient Tools for Edge-Connectivity Problems

In the following, we state some useful tools in obtaining the fast algorithms. Nagamochi and Ibaraki [61] gave a fast algorithm to find a sparse subgraph that satisfies edge-connectivity requirements, which will be used in Section 6.3 as a preprocessing step.

Theorem 6.2 ([61])**.** *There is an $O(m)$-time algorithm to construct a subgraph with $O(r_{\max} \cdot n)$ edges that satisfies all the connectivity requirements.*

As a key tool in checking local edge-connectivities, we need to construct a Gomory-Hu tree, which is a compact representation of all pairwise min-cuts of an undirected graph. Let $G = (V, E)$ be an undirected graph, a *Gomory-Hu tree* is a weighted tree $T = (V, F)$ with the following property. Consider any $s, t \in V$, the unique s-t path P in T, an edge $e = uv$ on P with minimum weight, and any component K of $T - e$. Then the local edge-connectivity between s and t in G is equal to the weight of e in T, and $\delta(K)$ is a minimum s-t cut in G. To check whether the edge-connectivity requirements are satisfied, we only need to check the pairs with $\lambda(u, v) < r_{\max}$. A *partial Gomory-Hu tree* T_k of G

is obtained from a Gomory-Hu tree T of G by contracting all edges with weight at least k. Therefore, each node in T_k represents a subset of vertices S in G, where the local edge-connectivity between each pair of vertices in S is at least k. For vertices $u, v \in G$ in different nodes of T_k, their local edge-connectivity (which is less than k) is determined in the same way as in an ordinary Gomory-Hu tree. Bhalgat et.al. [9] gave a fast randomized algorithm to construct a partial Gomory-Hu tree. We will use the following theorem by setting $k = r_{\max}$. The following result can be obtained by using the algorithm in [41], together with the fast tree packing algorithm in [9].

Theorem 6.3 ([41, 9])**.** *A partial Gomory-Hu tree T_k can be constructed in time $\tilde{O}(km)$ with high probability.*

6.1.2 An Alternative Proof of Mader's Theorem

Here, we present an alternative proof of Mader's theorem. Our framework of finding complete edge splitting-off sequence can be viewed as a generalization of this proof. The following lemma about non-admissible sets can be used directly to derive Mader's theorem.

Lemma 6.4. *Suppose there is no 3-dangerous set structure. Then, for any non-admissible set $U \subseteq N(x)$ with $|U| \geq 2$, there is a dangerous set containing U.*

We will now prove the equivalent statement of Mader's theorem (Theorem 2.7) that there is an admissible pair on a vertex $x \in V$ when $d(x)$ is even and there is no cut-edge incident to it. By Lemma 3.8, there is no 3-dangerous set structure in G. Suppose that there is no admissible pair incident to x. Then, by Lemma 6.4, there is a dangerous set D containing all vertices in $N(x)$. But this is impossible since

$$r(V - D - x) = r(D) \geq d(D) - 1 = d(V - D - x) + d(x) - 1 \geq d(V - D - x) + 1,$$

contradicting that the connectivity requirements are satisfied in G. This gives an alternative proof for Mader's theorem.

Proof of Lemma 6.4: We prove the lemma by a simple induction. The statement holds trivially for $|U| = 2$ by Proposition 2.12. Consider $U = \{u_1, u_2, \ldots, u_{k+1}\} \subseteq N(x)$ where every pair (u_i, u_j) is non-admissible. By induction, since every pair (u_i, u_j) is non-admissible, there are maximal dangerous sets X, Y such that $\{u_1, ..., u_{k-1}, u_k\} \subseteq X$ and $\{u_1, ..., u_{k-1}, u_{k+1}\} \subseteq Y$. Since (u_k, u_{k+1}) is non-admissible, by Proposition 2.12, there is a dangerous set Z containing u_k and u_{k+1}. If there is some $u_i \notin Z$, then X, Y and Z form a 3-dangerous-set structure with $u = u_i$, $v = u_k$, $w = u_{k+1}$. Hence we must have $U = \{u_1, \ldots, u_{k+1}\} \subseteq Z$, proving the lemma. □

6.2 Framework for Complete Edge Splitting-off

In this section, we will present an iterative approach that finds a complete edge splitting-off sequence by using at most $O(|N(x)|)$ splitting-off attempts (to split-off to capacity). In the algorithm, we maintain a non-admissible set C; initially $C = \emptyset$. In each iteration, the algorithm will apply one of the following three operations guaranteed by the following key lemma. The proof of this lemma will be presented in Section 6.5.

Lemma 6.5. *Suppose that C is a non-admissible set and there is a vertex $u \in N(x) - C$. Then, using at most three splitting-off attempts, at least one of the following operations can be applied:*

(1) Splitting-off an edge pair to capacity that voids an x-neighbour.

(2) Deducing that every pair in $C \cup \{u\}$ is non-admissible, and add u to C.

(3) Contracting a tight set T containing at least two x-neighbours.

We claim that the algorithm maintains the property that C is a non-admissible set, which holds at the beginning of the algorithm when $C = \emptyset$. It is clear that in case (2) the set C remains non-admissible. In case (1), by splitting-off an admissible pair, every pair of vertices in C remains non-admissible. Also, in case (3), by contracting a tight set, every pair of vertices in C remains non-admissible by Lemma 2.13.

The algorithm terminates when there is no vertex in $N(x) - C$. At that time, if $C = \emptyset$, then we have found a complete splitting-off sequence; if $C \neq \emptyset$, then by Mader's theorem, this only happens if $d(x) = 3$ and $d(x)$ is odd at the beginning. In any case, the longest splitting-off sequence is found and the given complete edge splitting-off problem is solved.

Finally, we claim that the total number of splitting-off attempts in the whole algorithm is at most $O(|N(x)|)$. To see this, we claim that each of the operations in Lemma 6.5 will be performed at most $|N(x)|$ times. Indeed, case (1) and (3) will be applied at most $|N(x)|$ times since each application reduces the number of x-neighbours by at least one, and case (2) will be applied at most $|N(x)|$ times since each application reduces the number of x-neighbours in $N(x) - C$ by one. This proves the following theorem.

Theorem 6.6. *A complete edge splitting-off sequence can be computed using at most $O(|N(x)|)$ splitting-off attempts.*

6.2.1 Proof of Lemma 6.5

We consider three cases based on the size of C. When $|C| = 0$, we split-off (u, u) to capacity (i.e. remove the maximum even number of copies of xu while satisfying the connectivity requirements of the graph). Either case (1) applies if u becomes void or case (2) applies in the resulting graph after (u, u) is splitted-off to capacity. When $|C| = 1$, pick the vertex $v \in C$, and split-off (u, v) to capacity. Either case (1) applies when one

of u and v becomes void, or case (2) applies in the resulting graph after (u, v) is splitted-off to capacity. Hence, when $|C| \leq 1$, either case (1) or case (2) applies after only one splitting-off attempt.

The interesting case is when $|C| \geq 2$ for which we consider $v_1, v_2 \in C$. Since C is a non-admissible set, by Lemma 6.4, there is a maximal dangerous set D containing C. First, we split-off (u, v_1) and (u, v_2) to capacity. If case (1) applies then we are done, so we assume that none of the three x-neighbours voids, implying that (u, v_1) and (u, v_2) are non-admissible in the resulting graph G' after splitting-off these edge pairs to capacity. Note that the edge pair (v_1, v_2) is also non-admissible since non-admissible edge pair in G remains non-admissible in G'. By Lemma 6.4 again, there exists a maximal dangerous set D' covering the non-admissible set $\{u, v_1, v_2\}$. Then inequality (2.5b) cannot hold for D and D', since

$$\begin{aligned} 1 + 1 = s(D) + s(D') &\geq s(D - D') + s(D' - D) + 2\bar{d}(D, D') \\ &\geq 0 + 0 + 2d(x, \{v_1, v_2\}) \\ &\geq 2 \cdot 2. \end{aligned}$$

Therefore inequality (2.5a) must hold for D and D', hence

$$1 + 1 = s(D) + s(D') \geq s(D \cap D') + s(D \cup D').$$

This implies that either $D \cup D'$ is a dangerous set for which case (2) applies, since $C \cup \{u\}$ is contained in a dangerous set and hence every pair is a non-admissible pair by Proposition 2.12, or $D \cap D'$ is a tight set for which case (3) applies since v_1 and v_2 are x-neighbours. Note that v_1, v_2 are contained in a tight set if and only if after splitting-off one copy of (xv_1, xv_2) the connectivity requirement of some pair is violated by two (see Section 6.3). Hence this can be checked by one splitting-off attempt, and thus we can distinguish between case (2) and case (3), and in case (3) we can find such a tight set

efficiently (see Section 6.3). Therefore, by making at most three splitting-off attempts $((xu, xv_1), (xu, xv_2), (xv_1, xv_2))$, one of the three operations can be applied, proving the lemma.

6.3 Efficient Splitting-off Attempt

In this section, we will show how to perform a splitting-off attempt efficiently. Our goal is to preserve the local edge-connectivity for each pair u, v with $\lambda(u, v) \leq r_{\max}$. The main tool is a fast Gomory-Hu tree construction algorithm in [9], which allows us to check the local edge-connectivities quickly.

The following is an outline of the whole algorithm for the complete splitting-off problem. First we use the $O(m)$-time algorithm in Theorem 6.2 by Nagamochi and Ibaraki [61] to construct a subgraph of G with $O(r_{\max} \cdot n)$ edges. To find a complete splitting-off sequence, we can thus restrict our attention to preserving the local edge-connectivities in this subgraph. In the next preprocessing step, we will reduce the problem further to an instance where there is a particular *indicator vertex* $t \neq x$, with the property that for any pair of vertices $u, v \in V - x$ with $\lambda(u, v) \leq r_{\max}$, then it holds that $\lambda(u, v) = \min\{\lambda(u, t), \lambda(v, t)\}$. With this indicator vertex, to preserve the local edge-connectivity for all pairs with $\lambda(u, v) \leq r_{\max}$, we only need to preserve the local edge-connectivities from t to every vertex v with $\lambda(v, t) \leq r_{\max}$. This allows us to make only $O(n)$ queries (instead of $O(n^2)$ queries) to check the local edge-connectivities. This reduction step can be done by computing a partial Gomory-Hu tree and contracting appropriate tight sets; the details will be presented in Section 6.3.1. The total preprocessing time is at most $\tilde{O}(m + {r_{\max}}^2 \cdot n)$, by using the fast Gomory-Hu tree algorithm in Theorem 6.3.

After these two preprocessing steps, we can perform a splitting-off attempt (split-off

a pair to capacity) efficiently. For a vertex pair (u, v), we replace $\min\{d(x, u), d(x, v)\}$ copies of xu and xv by copies of uv, and then determine the maximum violation of connectivity requirements by constructing a partial Gomory-Hu tree and check the local edge-connectivities from the indicator vertex t to every other vertex; the details will be discussed in Section 6.3.2. Therefore, using Theorem 6.3, one splitting-off attempt can be performed in $\tilde{O}(r_{\max} \cdot m + n) = \tilde{O}({r_{\max}}^2 \cdot n)$ time. By Theorem 6.6, the complete splitting-off problem can be solved by at most $O(|N(x)|) = O(n)$ splitting-off attempts. Hence we obtain the following result.

Lemma 6.7. *The complete splitting-off problem can be solved in $\tilde{O}(m + {r_{\max}}^2 \cdot n^2)$ time with high probability.*

6.3.1 Indicator Vertex

We show how to reduce the problem of maintaining pairwise edge-connectivity requirement into an instance of preserving rooted edge-connectivity. More specifically, we reduce the problem into an instance with a particular indicator vertex $t \neq x$, with the property that if $\lambda(u, v) \leq r_{\max}$ for $u, v \neq x$, then $\lambda(u, v) = \min\{\lambda(u, t), \lambda(v, t)\}$. Hence if we could preserve the local edge-connectivity from t to v for every $v \in V - x$ with $\lambda(v, t) \leq r_{\max}$, then the connectivity requirements for every pair in $V - x$ will be satisfied. Furthermore, by splitting-off an admissible pair, the indicator vertex t will remain to be an indicator vertex, and therefore this procedure needs to be executed only once. Without loss of generality, we assume that the connectivity requirement for each pair of vertices $u, v \in V - x$ is equal to $\min\{\lambda(u, v), r_{\max}\}$, and $r(x, v) = 0$ for every $v \in V - x$.

First we compute a partial Gomory-Hu tree $T_{r_{\max}}$ in $\tilde{O}(r_{\max} \cdot m)$ time by Theorem 6.3, which is $\tilde{O}({r_{\max}}^2 \cdot n)$ after applying the sparsifying algorithm in Theorem 6.2. Recall that each node in $T_{r_{\max}}$ represents a subset of vertices in G. In the following we will use

a capital letter (say U) to denote both a node in $T_{r_{\max}}$ and the corresponding subset of vertices in G. If $T_{r_{\max}}$ has only one node, then this means that the local edge-connectivity between every pair of vertices in G is at least $r_{\max}$. In this case, any vertex $t \neq x$ is an indicator vertex. So assume that $T_{r_{\max}}$ has at least two nodes. Let X be the node in $T_{r_{\max}}$ that contains x in G, and $U_1, \ldots, U_p$ be the nodes adjacent to X in $T_{r_{\max}}$, and let XU_1 be the edge in $T_{r_{\max}}$ with largest weight among XU_i for $1 \leq i \leq p$. See Figure 6.1 for an illustration.

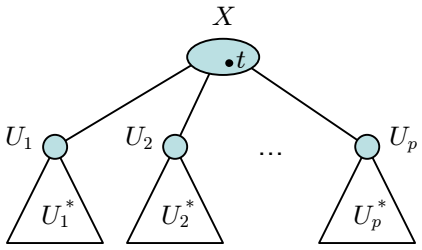


Figure 6.1: The case when X contains a vertex $t \neq x$. In this case each U_i^* is a tight set. After contracting each U_i^* into a single vertex, the vertex t becomes an indicator vertex.

Suppose X contains a vertex $t \neq x$ in G. The idea is to contract tight sets so that t becomes an indicator vertex in the resulting graph. For any edge XU_i in $T_{r_{\max}}$, let T_i' be the component of $T_{r_{\max}}$ that contains U_i when XU_i is removed from $T_{r_{\max}}$. We claim that each $U_i^* := \cup_{U \in T_i'} U$ is a tight set in G; see Figure 6.1. By the definition of a Gomory-Hu tree, the local edge-connectivity between any vertex $u_i \in U_i$ and t is equal to the edge weight of XU_i in $T_{r_{\max}}$. Also, by the definition of a Gomory-Hu tree, $d(U_i^*)$ is equal to the weight of edge XU_i in $T_{r_{\max}}$. Therefore, U_i^* is a tight set in G, because $r(u_i, t) = \lambda(u_i, t) = d(U_i^*)$ for some pair $u_i, t \in V - x$. By Proposition 2.12, we can contract each U_i^* into a single vertex u_i for $1 \leq i \leq p$ without losing any information about admissible pairs in G. Since each U_i^* becomes a single vertex, the vertex t becomes an indicator vertex in the resulting graph.

Suppose X contains only x in G. Then U_1^* may not be a tight set, since there may not exist a pair $u, v \in V - x$ with $r(u, v) = \lambda(u, v) = d(U_1^*)$ (note that there is a vertex v with $\lambda(x, v) = d(U_1^*)$, but $r(x, v) = 0$ for every vertex v). In this case, we contract appropriate tight sets so that any vertex in U_1 becomes an indicator vertex. Let $W_1 \neq X, \ldots, W_q \neq X$ be the nodes (if any) adjacent to U_1 in $T_{r_{\max}}$; see Figure 6.2. By using similar arguments as before, it can be shown that each U_i^* is a tight set for $2 \leq i \leq p$ (through $u_i \in U_i$ and $u_1 \in U_1$). Therefore we can contract each U_i^* into a single vertex u_i for $2 \leq i \leq p$. Similarly, we can argue that each W_j^* (defined analogously as U_i^*) is a tight set, and hence we can contract each W_j^* into a single vertex w_j for $1 \leq j \leq q$. We can see that any vertex $t \in U_1$ is an indicator vertex in the resulting graph, because $\lambda(t, v) \geq \min\{\lambda(w, v), r_{\max}\}$ for any pair of vertices v, w.

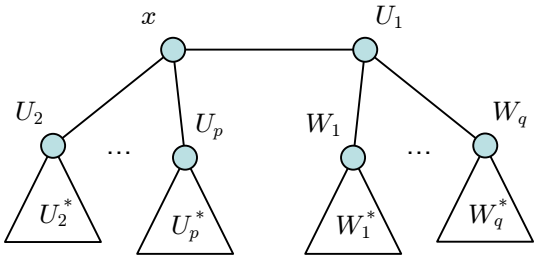


Figure 6.2: The case when X contains only x. In this case each U_i^* is a tight set for $2 \leq i \leq p$, and each W_j^* is a tight set for $1 \leq j \leq q$. After contracting each U_i^* for $2 \leq i \leq p$ and each W_j^* for $1 \leq j \leq q$ into a single vertex, any vertex $t \in U_1$ becomes an indicator vertex.

Henceforth we can consider this resulting graph instead of G for the purpose of computing a complete splitting-off sequence, and using t as the indicator vertex to check connectivities. The running time of this procedure is dominated by the partial Gomory-Hu tree computation, which is at most $\tilde{O}({r_{\max}}^2 \cdot n)$.

6.3.2 Splitting-off to Capacity

Here, we will show how to split-off a pair u, v of x-neighbours to capacity efficiently. Let G' be the graph obtained from G by splitting-off $\min\{d(x,u), d(x,v)\}$ copies of the edge pair (xu, xv). Let t be the indicator vertex as defined in Section 6.3.1. By the definition of t, if the edge-connectivity requirement from t to every vertex $v \in V - x$ (which is equal to $\min\{\lambda(v,t), r_{\max}\}$) is satisfied, then the local edge-connectivity for every pair $u, v \in V - x$ is satisfied. Let $q = \max_{w \in V-x} \{r(w,t) - \lambda_{G'}(w,t)\}$ be the maximum violation of the edge-connectivity requirement from t in G'. Let G'' be the graph obtained from G by splitting-off $\min\{d(x,u), d(x,v)\} - \lceil q/2 \rceil$ copies of the edge pair (xu, xv). We will show in the following claim that the edge-connectivity requirement from t to every vertex $v \in V - x$ is satisfied in G''. This implies that exactly $\min\{d(x,u), d(x,v)\} - \lceil q/2 \rceil$ copies of the edge pair (xu, xv) are admissible in G.

Claim 6.8. *For any vertex $s \in V - x$, it holds that $\lambda_{G''}(s,t) \geq r(s,t)$ in G''.*

Proof. By Menger's theorem, we only need to check that $d_{G''}(X) \geq r(s,t)$ for any X separating s and t. The edge-connectivity requirements are satisfied in G, and so we have $d_G(X) \geq r(s,t)$. The splitting-off operation decreases the degree of X if and only if $u, v \in X$. By definition, $q \geq r(s,t) - \lambda_{G'}(s,t) \geq r(s,t) - d_{G'}(X)$. On the other hand, we have $d_{G''}(X) \geq d_{G'}(X) + q$, since $\lceil q/2 \rceil$ copies of the edge pair are un-splitted. Therefore $d_{G''}(X) \geq r(s,t)$, proving the claim. □

To check the maximum violation, we compute a partial Gomory-Hu tree $T_{r_{\max}}$ in $\tilde{O}({r_{\max}}^2 \cdot n)$ time. Then we can easily compute $q = \max_{v \in V-x} \{r(v,t) - \lambda_{G'}(v,t)\}$ in $O(n)$ time. Furthermore, one can identify the violating cut easily from the Gomory-Hu tree. Therefore, splitting-off a pair to capacity can be implemented in $\tilde{O}({r_{\max}}^2 \cdot n)$ time. Finally, we remark that the same technique can be used to check whether two vertices are contained in a tight set, as needed in Lemma 6.5.

6.4 Randomized and Parallelized Edge Splitting-off Algorithm

In this section we will present a randomized edge splitting-off procedure to speedup the algorithm. By Theorem 4.1, when the degree of x is much larger than $2r_{\max}$, even a random edge pair will be admissible with high probability. Using this observation, we show how to reduce $d(x)$ to $O(r_{\max})$ in $\tilde{O}({r_{\max}}^3 \cdot n)$ time. Then, by Lemma 6.7, the remaining edges can be splitted-off in $\tilde{O}({r_{\max}}^2 \cdot d(x) \cdot n) = \tilde{O}({r_{\max}}^3 \cdot n)$ time. So the total running time of the complete splitting-off algorithm is improved to $\tilde{O}(m + {r_{\max}}^3 \cdot n)$, proving Theorem 6.1.

The idea is to split-off many random edge pairs in parallel, before checking if some connectivity requirement is violated. Suppose that $2^{l+q-1} < d(x) \leq 2^{l+q}$ and $2^{l-1} < r_{\max} \leq 2^l$ for some positive integers l and q. To reduce $d(x)$ to 2^{l+q-1} (i.e. to reduce $d(x)$ by half, perhaps except for the first round), we need to split-off at most 2^{l+q-1} x-edges. Since each x-edge has at most $2r_{\max}$ non-admissible partners by Theorem 4.1, the probability that a random edge pair is admissible is at least $\frac{(d(x)-1)-2r_{\max}}{d(x)-1} \geq \frac{2^{l+q-1}-2^{l+1}}{2^{l+q-1}} = \frac{2^{q-2}-1}{2^{q-2}}$. Now, consider a random splitting-off operation that split-off at most 2^{q-2} edge pairs at random in parallel. The operation is successful if all the edge pairs are admissible. The probability for the operation to succeed is at least $(\frac{2^{q-2}-1}{2^{q-2}})^{2^{q-2}} = O(1)$. After each operation, we run the checking algorithm as in Section 6.3 to determine whether this operation is successful or not. Consider an iteration that consists of $c \cdot \log n$ operations for some constant c. The iteration is successful if it finds a set of 2^{q-2} admissible pairs, i.e. any of its operations succeeds. The probability for an iteration to fail is hence at most $1/n^c$ for $q \geq 3$. The time complexity of an iteration is $\tilde{O}({r_{\max}}^2 \cdot n)$.

Since each iteration reduces the degree of x by 2^{q-2}, with at most $2^{l+1} \approx r_{\max}$ successful iterations, we can then reduce $d(x)$ to 2^{l+q-1} (i.e. reduce $d(x)$ by half, perhaps except for

the first round). This procedure is applicable as long as $q \geq 3$. Therefore, we can reduce $d(x)$ to 2^{l+2} by using this procedure for $O(\log n)$ times. The total running time is thus $\tilde{O}(2^l \cdot \log n \cdot {r_{\max}}^2 \cdot n) = \tilde{O}({r_{\max}}^3 \cdot n)$. Note that there are at most $\tilde{O}(r_{\max})$ iterations and the failure probability of each iteration is at most $1/n^c$. By the union bound, the probability for above randomized algorithm to fail is at most $1/n^{c-1}$. Therefore, with high probability, the algorithm succeeds in $\tilde{O}({r_{\max}}^3 \cdot n)$ time to reduce $d(x)$ to $O(r_{\max})$.

6.5 Deterministic Edge Splitting-off Algorithm

Here, we will show how to modify the randomized algorithm giving Lemma 6.7 to obtain a deterministic algorithm. The randomized Gomory-Hu tree construction is used in two places. First it is used in finding an indicator vertex in Section 6.3.1, and for this purpose it is executed only once. Here we can replace it by a slower deterministic partial Gomory-Hu tree construction algorithm. It is well-known that a Gomory-Hu tree can be computed using at most $n-1$ max-flow computations [40, 39]. By using the Ford-Fulkerson maximum flow algorithm [24], one can obtain an $O({r_{\max}}^2 \cdot n^2)$-time deterministic algorithm to construct a partial Gomory-Hu tree $T_{r_{\max}}$. The randomized partial Gomory-Hu construction is also used many times in the algorithm (Section 6.3) to check whether the connectivity requirements are satisfied. With the indicator vertex t, this task reduces to checking the local edge-connectivities from t to other vertices. It turns out that there is a fast deterministic algorithm for this task. We will need the following result by Bhalgat et.al. [9], which is obtained by a deterministic tree packing algorithm, building on and improving the work by Cole and Hariharan [19].

Theorem 6.9 ([9]). *Given an undirected graph G and a vertex t, there is an $\tilde{O}(r_{\max} \cdot m)$-time deterministic algorithm to compute $\min\{\lambda_G(t,v), r_{\max}\}$ for all vertices $v \in G$.*

Therefore, we can replace the randomized partial Gomory-Hu tree algorithm by this

algorithm, and thus Lemma 6.7 still holds deterministically. Hence there is a deterministic $\tilde{O}({r_{\max}}^2 \cdot n^2)$-time algorithm for the complete splitting-off problem.

6.6 Algorithms in Other Settings

In this section, we will show how to apply the framework (Section 6.2) and the randomized procedure (Section 6.4) in other settings to obtain efficient edge splitting-off algorithms. We will see applications in the network design problem discussed in Chapter 3, and the constrained edge splitting-off problems discussed in Chapter 4.

6.6.1 Edge Splitting-off in Network Design Problems

Here, we will sketch an efficient edge splitting-off procedure for the network design problem we discussed in Chapter 4. We will apply the framework to reduce the degree of a vertex x by at most $O(|N(x)|)$ splitting-off attempts; and hence we have to show the operations to maintain a non-admissible set C.

If $st \in E$ for every pair of $s \in C$ and $t \in N(x) - C$, then it is clear that there is no legal admissible pair contains any of $s \in C$. We can ignore those x-neighbours in the non-admissible set and consider the rest of x-neighbours. Therefore, suppose this is not the case, i.e. there exists $u \in C$ and $w \in N(x) - C$ that $uw \notin E$. Consider also another vertex $v \in C - u$. If either both $uw, vw \notin E$, or both (xu, xw) and (xv, xw) are non-admissible, then we can apply the analysis in Lemma 6.5.

Therefore, the remaining case is that (xu, xw) is non-admissible and (xv, xw) is admissible but $vw \in E$. By Proposition 2.12, there exist a maximal dangerous sets X containing C, and Y containing u and w. Since $vw \in E$, we have $d(X, Y) \geq 1$. Inequality (2.5a) cannot hold; for otherwise, $X \cup Y$ will be a dangerous set, contradicting the

admissibility of (xv, xw):

$$1 + 1 \geq s(X) + s(Y) \geq s(X \cap Y) + s(X \cup Y) + 2d(X, Y) \geq 0 + s(X \cup Y) + 2 \cdot 1.$$

Therefore, inequality (2.5b) must hold. This implies that $X - Y$ and $Y - X$ are both tight, and $\bar{d}(X, Y) = 1$:

$$1 + 1 \geq s(X) + s(Y) \geq s(X - Y) + s(X - Y) + 2\bar{d}(X, Y) \geq 0 + 0 + 2\bar{d}(X, Y).$$

By the contra-positive of Lemma 4.5, there exist two maximal dangerous sets X' and Y' covers all the x-edges that are non-admissible to xu. We can assume that $C \subseteq X'$ and $w \in Y'$, and then show that $X' - Y'$ and $Y' - X'$ are both tight by the above argument. Now, one of the following 3 situations holds for any other x-neighbour $t \in N(x) - C - w$ that $ut \notin E$.

(i) (xu, xt) is an admissible pair.

(ii) (xv, xt) is a non-admissible pair and there exists a tight set, namely $X' - Y'$ containing t and C.

(iii) (xw, xt) is a non-admissible pair and there exists a tight set, namely $Y' - X'$ containing t and w.

If there does not exist $t \in N(x) - C - w$ that $ut \notin E$, then u is not included in any legal admissible edge pair. Hence, we can ignore this x-neighbour in the rest of the algorithm. This shows that we can always make progress by a few edge splitting-off attempts.

6.6.2 Constrained Edge Splitting-off

Here, we will present $\tilde{O}(m + {r_{\max}}^3 n)$-time randomized algorithms for the some constrained edge splitting-off problems. These randomized algorithms are basically the same

as the randomized algorithm for complete edge splitting-off problem in Section 6.4. Recall that the algorithm in Section 6.4 is based on the property of bounded number of non-admissible partners. The same approach can hence be applied in constrained edge splitting-off problems when the number of non-admissible or illegal partners are bounded. Suppose that $2^{l+q-1} < d(x) \leq 2^{l+q}$ and $2^{l-1} < r_{\max} \leq 2^l$ for some positive integers l and q.

Vertex partition constraint: Consider an undirected graph $G = (V, E)$ with a designated vertex x and a vertex partition $\mathcal{P} = \{P_1, P_2, \ldots, P_l\}$ on $V - x$. In partition-preserving edge splitting-off problem, we have to make sure that no split edge is included in any vertex partition while satisfying edge-connectivity requirements. In the randomized algorithm, we pair up x-edges from different partitions randomly so that none of the pairs is illegal. The probability that a random edge pair is a legal admissible pair is at least

$$\frac{d(x)/2 - 2r_{\max}}{d(x)/2} \geq \frac{2^{l+q-2} - 2^{l+1}}{2^{l+q-2}} = \frac{2^{q-3} - 1}{2^{q-3}}.$$

Therefore, we can reduce $d(x)$ to $\tilde{O}(r_{\max})$ in $O(m + {r_{\max}}^3 n)$ time by the randomized approach used in Section 6.4. This gives an approximate solution with additive error of $O(r_{\max})$.

Graph simplicity constraint: Consider an undirected graph $G = (V, E)$ with a designated vertex x. In simplicity-preserving edge splitting-off problem, we have to make sure that the split edges are not in parallel with any other edges while satisfying edge-connectivity requirements. In the randomized algorithm, we pair up x-edges randomly. We have shown that each x-edge is included in at most ${r_{\max}}^2$ illegal edge pairs. The probability that a random edge pair is a legal admissible pair is hence at least

$$\frac{d(x) - {r_{\max}}^2 - 2r_{\max}}{d(x)} \geq \frac{2^{l+q-1} - 2^{2l+2} - 2^{l+1}}{2^{l+q-1}} \geq \frac{2^{q-l-3} - 2}{2^{q-l-3}}.$$

Therefore, we can reduce $d(x)$ to $\tilde{O}({r_{\max}}^2)$ in $O(m + {r_{\max}}^3 n)$ time by the randomized

approach used in Section 6.4. This gives an approximate solution with additive error of $O({r_{\max}}^2)$.

Simultaneous graph constraint: Consider c undirected graphs $G_i = (V, E_i + E')$ for $i = 1, \ldots, c$ with a designated vertex x that $E' = \delta(x)$. In simultaneous-graph edge splitting-off problem, we have to make sure that the split edges satisfy edge-connectivity requirements in all the c graphs. In the randomized algorithm, we pair up x-edges randomly. Recall that an edge pair is legal if it is admissible in all the c graphs. The probability that a random edge pair is a legal admissible pair is at least

$$\frac{d(x) - c \cdot 2r_{\max}}{d(x)} \geq \frac{2^{l+q-1} - c \cdot 2^{l+1}}{2^{l+q-1}} \geq \frac{2^{q-2} - c}{2^{q-2}}.$$

Therefore, we can reduce $d(x)$ to $\tilde{O}(c \cdot r_{\max})$ in $O(m + {r_{\max}}^3 n)$ time by the randomized approach used in Section 6.4. This gives an approximate solution with additive error of $O(c \cdot r_{\max})$.

Chapter 7

Concluding Remarks

In this thesis, we studied edge splitting-off techniques in some edge-connectivity problems. As a main result, we obtained the first constant factor approximation algorithms for various degree bounded network design problem when the cost function satisfies triangle inequalities. Using the insights developed, we obtained further edge splitting-off results in two directions. First, we developed a framework for splitting-off edges efficiently. This framework is conceptually simple and substantially improves the best known algorithm (in specific setting). Second, we identified some structural properties of non-splittable edge pairs. These properties not only provide a short proof for a classical edge splitting-off result, but also have applications in efficient randomized edge splitting-off procedures and constrained edge splitting-off problems.

As a direction for future work, I am most interested in finding some other applications of our structural properties. The results of this thesis are based on extensions and generalizations of a structural properties, named as 3-dangerous-set structure. This simple structure seems to be very useful in deriving edge splitting-off results. It would be nice to see this structure used in other edge-connectivity problems. In particular, I would like to study the compact representation of pairwise minimum cuts. There are well-known

results for capturing all global minimum cuts [21] and for capturing the size of all pairwise minimum cuts [39]. Is it possible to apply our structural properties for finding a representation of all (or a certain portion) of pairwise minimum cuts?

Regarding the efficient edge splitting-off algorithms, a straightforward direction for future work is to investigate their applications in other edge-connectivity problems. Edge splitting-off operation is used as an important subroutine in algorithms for several connectivity problems, including connectivity augmentation, network design, tree packing and graph orientation. It is of practical interests to study how our edge splitting-off procedure speedup these algorithms. As other direction, it is interesting to study the derandomization of our randomized procedure used to edge splitting-off algorithms (Section 6.4). This procedure pair-up edges randomly until it finds a "good" pairing. Actually, "bad" pairings also give important information of non-admissibility. It may be possible to use the non-admissibility information, together with some structural properties, to find a "good" pairing deterministically.

Bibliography

[1] C. Artur and L. Andrzej. *Approximation Schemes for Minimum-cost k-Connectivity Problems in Geometric Graphs*, chapter 51. Chapman & Hall/CRC, Boca Raton, FL, 2007.

[2] J. Bang-Jensen, A. Frank, and B. Jackson. Preserving and increasing local edge-connectivity in mixed graphs. *SIAM Journal on Discrete Mathematics*, 8(2):155--178, 1995.

[3] J. Bang-Jensen, H. Gabow, T. Jordán, and Z. Szigeti. Edge-Connectivity Augmentation with Partition Constraints. *SIAM Journal on Discrete Mathematics*, 12:160, 1999.

[4] J. Bang-Jensen and T. Jordán. Edge-connectivity augmentation preserving simplicity. *SIAM J. Discret. Math.*, 11(4):603--623, 1998.

[5] J. Bang-Jensen and T. Jordán. Splitting off Edges between Two Subsets Preserving the Edge-connectivity of the Graph. *Journal of Combinatorial Optimization*, 276(1--3):5--28, 2004.

[6] F. Bauer and A. Varma. Degree-constrained multicasting in point-to-point networks. In *INFOCOM '95: Proceedings of the Fourteenth Annual Joint Conference of the*

IEEE Computer and Communication Societies (Vol. 1)- Volume, page 369, Washington, DC, USA, 1995. IEEE Computer Society.

[7] A. A. Benczúr and D. R. Karger. Augmenting undirected edge connectivity in $\tilde{O}(n^2)$ time. *Journal of Algorithms*, 37(1):2--36, 2000.

[8] A. Bernáth and T. Király. A New Approach to Splitting-Off. In *Proceedings of Integer Programming and Combinatorial Optimization, 13th International Conference*, pages 401--415. Bertinoro, Italy, 2008.

[9] A. Bhalgat, R. Hariharan, T. Kavitha, and D. Panigrahi. An $\tilde{O}(mn)$ gomory-hu tree construction algorithm for unweighted graphs. In *STOC '07: Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 605--614, New York, NY, USA, 2007. ACM.

[10] A. Bhalgat, R. Hariharan, T. Kavitha, and D. Panigrahi. Fast edge splitting and edmonds' arborescence construction for unweighted graphs. In *SODA '08: Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 455--464, Philadelphia, PA, USA, 2008. Society for Industrial and Applied Mathematics.

[11] D. Bienstock, E. Brickell, and C. Monma. On the Structure of Minimum-Weight$k$-Connected Spanning Networks. *SIAM Journal on Discrete Mathematics*, 3:320, 1990.

[12] E. Bolker and H. Crapo. How to brace a one-story building. *Environ. Plan. B*, 4:125--152, 1977.

[13] G.-R. Cai and Y.-G. Sun. *The minimum augmentation of any graph to a k-edge-connected graph*, pages 151--172. A Wiley Company, 1989.

[14] Y. H. Chan, W. S. Fung, L. C. Lau, and C. K. Yung. Degree bounded network design with metric costs. In *FOCS '08: Proceedings of the 2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 125--134, Washington, DC, USA, 2008. IEEE Computer Society.

[15] C. Chekuri and F. Shepherd. Approximate Integer Decompositions for Undirected Network Design Problems. *SIAM Journal on Discrete Mathematics*, 23:163, 2008.

[16] E. Cheng and T. Jordán. Successive edge-connectivity augmentation problems. *Mathematical Programming*, 84(3):577--593, 1999.

[17] J. Cheriyan and A. Vetta. Approximation algorithms for network design with metric costs. In *STOC '05: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 167--175, New York, NY, USA, 2005. ACM.

[18] N. Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. 1976.

[19] R. Cole and R. Hariharan. A fast algorithm for computing steiner edge connectivity. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 167--176. ACM New York, NY, USA, 2003.

[20] N. Deo and S. Hakimi. The shortest generalized hamiltonian tree. In *Proceedings of the 6th Annual Allerton Conference*, pages 879--888, 1968.

[21] E. A. Dinits, A. V. Karzanov, and M. V. Lomonosov. On the structure of a family of minimal weighted cuts in a graph. *Studies in Discrete Optimization*, pages 290--306, 1976.

[22] J. Edmonds. Edge-disjoint branchings. *Combinatorial Algorithms*, pages 91--96, 1972.

[23] K. P. Eswaran and R. E. Tarjan. Augmentation problems. *SIAM J. Computing*, 5(4):653--665, 1976.

[24] L. Ford and D. Fulkerson. Maximal flow through a network. *Canadian Journal of Mathematics*, 8(3):399--404, 1956.

[25] A. Frank. *Generalized Polymatroids*, pages 285--294. Elsevier Science Publishing Company, Amsterdam, The Netherlands, 1984.

[26] A. Frank. Augmenting graphs to meet edge-connectivity requirements. *SIAM Journal on Discrete Mathematics*, 5(1):25--53, 1992.

[27] A. Frank. On a theorem of Mader. *Annals of Discrete Mathematics*, 101(1-3):49--57, 1992.

[28] A. Frank and T. Király. Combined connectivity augmentation and orientation problems. *Discrete Applied Mathematics*, 131(2):401--419, 2003.

[29] A. Frank and Z. Kiraly. Graph orientations with edge-connection and parity constraints. *Combinatorica*, 22(1):47--70, 2002.

[30] A. Frank and É. Tardos. Generalized polymatroids and submodular flows. *Mathematical Programming*, 42(1):489--563, 1988.

[31] T. Fukunaga and H. Nagamochi. Approximating a generalization of metric tsp. *IEICE - Trans. Inf. Syst.*, E90-D(2):432--439, 2007.

[32] T. Fukunaga and H. Nagamochi. Approximating minimum cost multigraphs of specified edge-connectivity under degree bounds. *Journal of the Operations Research Society of Japan-Keiei Kagaku*, 50(4):339--349, 2007.

[33] T. Fukunaga and H. Nagamochi. *Network design with edge connectivity and degree constraints*, pages 188--201. Springer Berlin / Heidelberg, 2007.

[34] D. R. Fulkerson and L. S. Shapley. Minimal k-arc-connected graphs. *Networks*, 1:91--98, 1971.

[35] H. N. Gabow. A matroid approach to finding edge connectivity and packing arborescences. In *STOC '91: Proceedings of the twenty-third annual ACM symposium on Theory of computing*, pages 112--122, New York, NY, USA, 1991. ACM.

[36] H. N. Gabow. Efficient splitting off algorithms for graphs. In *STOC '94: Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, pages 696--705, New York, NY, USA, 1994. ACM.

[37] H. N. Gabow and T. Jordán. Bipartition constrained edge-splitting in directed graphs. *Discrete Appl. Math.*, 115(1-3):49--62, 2001.

[38] M. X. Goemans. Minimum bounded degree spanning trees. In *FOCS '06: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 273--282, Washington, DC, USA, 2006. IEEE Computer Society.

[39] R. Gomory and T. Hu. Multi-terminal network flows. *Journal of the Society for Industrial and Applied Mathematics*, pages 551--570, 1961.

[40] D. Gusfield. Very simple methods for all pairs network flow analysis. *SIAM Journal on Computing*, 19:143, 1990.

[41] R. Hariharan, T. Kavitha, and D. Panigrahi. Efficient algorithms for computing all low st edge connectivities and related problems. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 127--136. Society for Industrial and Applied Mathematics Philadelphia, PA, USA, 2007.

[42] K. Jain. A factor 2 approximation algorithm for the generalized Steiner network problem. *Combinatorica*, 21(1):39--60, 2001.

[43] T. Jordan. *Two NP-complete augmentation problems*. Odense Universitet.

[44] T. Jordán. Edge-Splitting Problems with Demands. In *Proceedings of the 7th International IPCO Conference on Integer Programming and Combinatorial Optimization*, pages 273--288. Springer-Verlag London, UK, 1999.

[45] T. Jordán. Constrained edge-splitting problems. *SIAM J. Discret. Math.*, 17(1): 88--102, 2004.

[46] Y. Kajitani and S. Ueno. The minimum augmentation of a directed tree to ak-edge-connected directed graph. *Networks*, 16(2), 1986.

[47] S. Khuller and U. Vishkin. Biconnectivity approximations and graph carvings. *Journal of the ACM*, 41(2):214--235, 1994.

[48] G. Kortsarz and Z. Nutov. Approximating node connectivity problems via set covers. In *APPROX '00: Proceedings of the Third International Workshop on Approximation Algorithms for Combinatorial Optimization*, pages 194--205, London, UK, 2000. Springer Berlin / Heidelberg.

[49] M. Kriesell. Edge-disjoint trees containing some given vertices in a graph. *J. Comb. Theory Ser. B*, 88(1):53--65, 2003.

[50] L. C. Lau. An Approximate Max-Steiner-Tree-Packing Min-Steiner-Cut Theorem*. *Combinatorica*, 27(1):71--90, 2007.

[51] L. C. Lau, J. S. Naor, M. R. Salavatipour, and M. Singh. Survivable network design with degree or order constraints. In *STOC '07: Proceedings of the thirty-ninth*

annual ACM symposium on Theory of computing, pages 651--660, New York, NY, USA, 2007. ACM.

[52] L. C. Lau and M. Singh. Additive approximation for bounded degree survivable network design. In *STOC '08: Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 759--768, New York, NY, USA, 2008. ACM.

[53] L. C. Lau and C. K. Yung. Efficient Edge Splitting and Constrained Edge Splitting. Manuscript, 2009.

[54] L. Lova'sz. Lecture. Conference of Graph Theory, Prague, 1974.

[55] L. Lova'sz. *Combinatorial problems and exercises*. Elsevier, Budapest, 1979.

[56] W. Mader. A reduction method for edge-connectivity in graphs. *Annals of Discrete Mathematics*, 3:145--164, 1978.

[57] K. Menger. *Zur Allgemeinen Kurventheorie*, pages 95--115. 1927.

[58] H. Nagamochi. A Fast Edge-Splitting Algorithm in Edge-Weighted Graphs. *IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences*, pages 1263--1268, 2006.

[59] H. Nagamochi and P. Eades. An edge-splitting algorithm in planar graphs. *Journal of Combinatorial Optimization*, 7(2):137--159, 2003.

[60] H. Nagamochi and T. Ibaraki. Computing edge-connectivity in multigraphs and capacitated graphs. *SIAM J. Discrete Math.*, 5(1):54--66, 1992.

[61] H. Nagamochi and T. Ibaraki. A linear-time algorithm for finding a sparse k-connected spanning subgraph of a k-connected graph. *Algorithmica*, 7(5--6):583--596, 1992.

[62] H. Nagamochi and T. Ibaraki. A faster edge splitting algorithm in multigraphs and its application to the edge-connectivity augmentation problem. In *Proceedings of the 4th International IPCO Conference on Integer Programming and Combinatorial Optimization*, pages 403--413, London, UK, 1995. Springer-Verlag.

[63] H. Nagamochi and T. Ibaraki. Deterministic $\tilde{O}(nm)$ time edge-splitting in undirected graphs. *Journal of Combinatorial Optimization*, 1(1):5--46, 1997.

[64] H. Nagamochi, K. Nishimura, and T. Ibaraki. Computing all small cuts in an undirected network. *SIAM J. Discret. Math.*, 10(3):469--481, 1997.

[65] D. Naor, D. Gusfield, and C. Martel. A fast algorithm for optimally increasing the edge connectivity. *SIAM J. Comput.*, 26(4):1139--1165, 1997.

[66] C. Nicos. Worst-case analysis of a new heuristic for the traveling salesman problem. Report 388, Graduate School of Industrial Administration, Carnegie Mellon University, 1976.

[67] C. Oliveira and P. Pardalos. A survey of combinatorial optimization problems in multicast routing. *Computers and Operations Research*, 32(8):1953--1982, 2005.

[68] R. Ravi, M. Marathe, S. Ravi, D. Rosenkrantz, and H. Hunt III. Approximation algorithms for degree-constrained minimum-cost network-design problems. *Algorithmica*, 31(1):58--78, 2001.

[69] A. Schrijver. *Combinatorial optimization: polyhedra and efficiency.* Springer, 2003.

[70] M. Singh and L. C. Lau. Approximating minimum bounded degree spanning trees to within one of optimal. In *STOC '07: Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 661--670, New York, NY, USA, 2007. ACM.

[71] M. Stoer and F. Wagner. A simple min-cut algorithm. *J. ACM*, 44(4):585--591, 1997.

[72] Z. Szigeti. Edge-splittings preserving local edge-connectivity of graphs. *Discrete Appl. Math.*, 156(7):1011--1018, 2008.

[73] S. Voß. Problems with generalized steiner problems. *Algorithmica*, 7(1):333--335, 1992.

[74] T. Watanabe and A. Nakamura. Edge-connectivity augmentation problems. *J. Comput. Syst. Sci.*, 35(1):96--144, 1987.