SOME PRIMAL-DUAL THEOREMS IN GRAPH THEORY - A STUDY

A THESIS
Submitted by

VENKATA HARISH IMMADI

In partial fulfillment for the award of the degree of

MASTER OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING (INFORMATION SECURITY)

Under the guidance of
Dr. K. Murali Krishnan

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY CALICUT
NIT CAMPUS PO, CALICUT
KERALA, INDIA 673601

May, 2012

ACKNOWLEDGEMENTS

First of all, I would like to express my gratitude to my guide Dr.K Murali Krishnan, Department of Computer science and Engineering, National Institute of Technology, Calicut for his inspiring guidance, motivation, support and encouragement. His strenuous efforts and valuable pieces of advice were available to me throughout my endeavor without which it would have been impossible for me to complete this work in time.

I would like to express my sincere thanks to the project coordinator Mr. G Gopa Kumar for allowing me to do this project.

I take this opportunity to express my sincere thanks to, Dr.M. N. Bandyopadhyay, Director, National Institute of Technology, Calicut for providing me with the facilities to carry out the work.

I am thankful to the teaching and non teaching staff for their support and encouragement throughout my work.

I Venkata Harish

DECLARATION

"I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text".

Place: Calicut
Date:

Signature :
Name : I Venkata Harish
Reg.No: M100170CS

CERTIFICATE

*This is to certify that the thesis entitled: "***SOME PRIMAL-DUAL THEOREMS IN GRAPH THEORY - A STUDY***" submitted by Sri* **VENKATA HARISH IMMADI** *to National Institute of Technology Calicut towards partial fulfillment of the requirements for the award of Degree of Master of Technology in Computer Science Engineering is a bona fide record of the work carried out by him under my supervision and guidance.*

Signed by Thesis Supervisor with name and date

Place: Calicut
Date:

Signature of Head of the Department

Office Seal

Contents

Chapter

Abstract

The objective of this report is to present some central min-max theorems in graph theory from the view point of linear programming. In the introductory chapter, we review the well known *Max-flow Min-cut* theorem and discuss the standard proof using Ford-Fulkerson algorithm. The next chapter will discuss some graph theoretic consequences of the *Max-flow Min-cut* theorem in combinatorial optimization. In the third chapter we present the proof of the *Max-flow Min-cut* theorem and *Konig's* theorem using the properties of total unimodular matrices in linear programming. In the fourth chapter, we discuss the problem of Concurrent Multi-commodity Flow(CMFP) and present a linear programming formulation.

Figures

Figure

Chapter 1

Introduction

In this report, we study some min-max theorems in combinatorial optimization. One of the earliest and well recognised result is the *Max-flow Min-cut* theorem by Ford and Fulkerson [5] in 1956. This theorem states the equivalence between the maximum flow and minimum cut in a network. In this chapter we review the *Max-flow Min-cut* theorem and its proof using Ford-Fulkerson algorithm. This theorem had several implications in combinatorial optimization, several other theorems can be viewed as a consequence of this theorem [11]. In second chapter, we presen *Konig's* and *Menger's* theorems as consequences of *Max-flow Min-cut* theorem. The Menger's theorem provide bounds on the connectivity of a graph. There are two versions of Menger's theorem [1], edge version and vertex version. This theorem is a special case of the Maximum flow problem and can be proved by using the *Max-flow Min-cut* theorem. Konig's theorem [13, 9] states the equality between maximum matching and minimum vertex cover in bipartite graph.

All the above theorems are min-max theorems, that is, if one of the problem is a maximization problem then the other is a minimization problem. These problems can be encoded as linear programs. A Chandra Babu et al.[2] has given new proofs for *Max-flow Min-cut* theorem and *Konig's* theorem using total unimodularity of the coefficient matrix in their linear programming formulations. The third chapter is a study of the proofs given by A Chandra Babu et al.[2]. We have filled a few missig links while presenting this material.

The Multi-commodity flow problem is a more generalized network flow problem. The problem of finding a maximum flow in a multi-commodity network arises in many network instances. In a Multi-commodity flow problem, there exists $k \geq 1$ commodities each having its own source and sink. In the fourth chapter, we give an introduction to the Concurrent Multi-commodity Flow problem(CMFP) [15] [1] and present the linear programming formulation for the problem and it's dual. In CMFP, every commodity is assigned a demand D_i, our objective is to assign flows to the commodities so as to maximize a fraction λ such that the flow for any commodity is at least λD_i units.

1.1 The Max-flow Min-cut Theorem

1.1.1 Preliminaries

Definition 1.1.1. *(Network) A Network is a directed graph $G(V, E, c, s, t)$ with vertex set V and an arc set E in which every directed edge $(i, j) \in E$, has a non negative capacity $c(i, j) \geq 0$, $c : V \times V \longrightarrow \mathbf{R}^+$. The vertices $s \in V, t \in V$ are the source vertex and the sink vertex of the network.*

Definition 1.1.2. *(s,t-cut) An s, t-cut is a partitioning of the vertices of a Network into two sets, say A and B, such that $s \in A$ and $t \in B$.*

The capacity of a cut (A, B) is given as

$$C(A, B) = \sum_{u \in A, v \in B} c(u, v)$$

Definition 1.1.3. *(Flow) A flow is a mapping $f : E \longrightarrow R$, denoted by f_{uv} or $f(u, v)$, subject to the following constraints:*

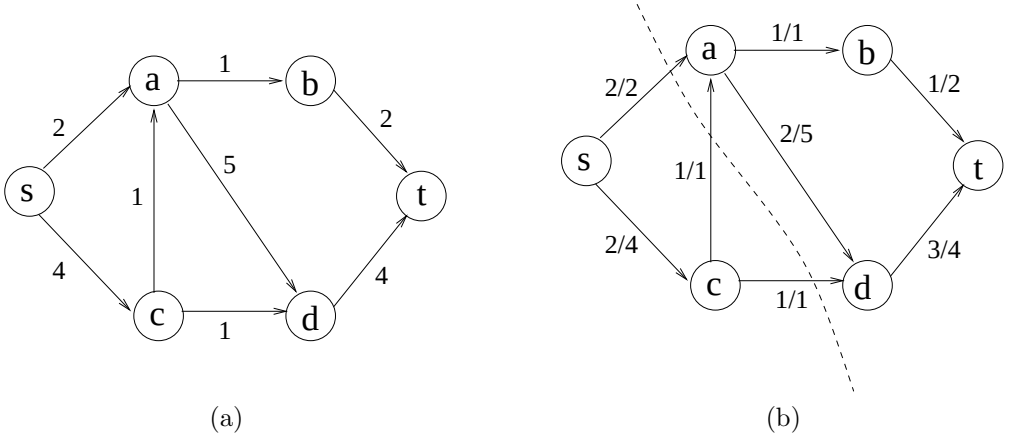
(1) $f(u, v) \leq c(u, v)$ for each $(u, v) \in E$ (capacity constraint)

(2) $f(u, v) = -f(v, u)$ (skew symmetry)

[1] It is also referred to as " The Maximum Concurrent Flow Problem ".

(3) $\sum_{v \in V} f(u,v) = 0 \ \forall \ u \in V \setminus \{s,t\}$ *(conservation of flow).*

The value of flow is defined by $|f| = \sum_{v \in V} f(s,v)$, where s is the source of N. It represents the amount of flow passing out of the source to the sink. Figure 1.1 shows a typical network and a flow in that network. Here f can be viewed as a vector over the directed edges [2] of the network.



(a) (b)

Figure 1.1: (a) A network with edge capacities. (b) The corresponding flow network with $|f| = 4$. The dotted line forms a cut which is also a minimum cut

Definition 1.1.4. *(Flow across a cut) If* (A,B) *is any cut in the network, then the flow across the cut is defined as*

$$f(A,B) = \sum_{u \in A, v \in B} f(u,v)$$

Lemma 1.1.1. *If* f *is a flow in a network* G*, then for any* s,t*-cut* (A,B)*,* $f(A,B) = |f|$.

Proof. We can prove this by considering the flow over the cut and reducing it to the flow from the source vertex which is the actual flow.

[2] The terms directed edge and arc may be used interchangeably.

$$f(A,B) = \sum_{u\in A, v\in B} f(u,v)$$
$$= \sum_{u\in A, v\in B} f(u,v) + \sum_{\{w,w'\}\in A} f(w,w')$$

Since, $\sum_{\{w,w'\}\in A} f(w,w') = 0$. Because for every $\{x,y\} \in A$ both $f(x,y)$ and $f(y,x)$ are added.

$$= \sum_{u\in A, v\in V} f(u,v)$$
$$= \sum_{v\in V} f(s,v) + \sum_{u\in A\setminus\{s\}, v\in V} f(u,v)$$
$$= \sum_{v\in V} f(s,v)$$
$$= |f|$$

□

Definition 1.1.5. *(Residual Graph) If f is a flow in a graph G, then the residual graph is defined as G_f with $c_f(u,v) = c(u,v) - f(u,v)$.*

For a directed edge $(a,b) \in E$, if $c(a,b) = 0$ then in G_f, the residual capacity will be $c_f(a,b) = 0 - f(a,b) = -f(a,b)$. Figure 1.2 gives an example for a residual graph(network).

Lemma 1.1.2. *f' is a flow in G_f iff $f + f'$ is a flow in G.*

Proof. We prove this by proving the following two conditions.

(1) A flow f' is a flow in G_f iff it satisfies $f'(u,v) \le c_f(u,v)$.

$$f'(u,v) \le c_f(u,v)$$
$$\Leftrightarrow f'(u,v) \le c(u,v) - f(u,v)$$
$$\Leftrightarrow f(u,v) + f'(u,v) \le c(u,v)$$
$$\Leftrightarrow f + f'(u,v) \le c(u,v)$$

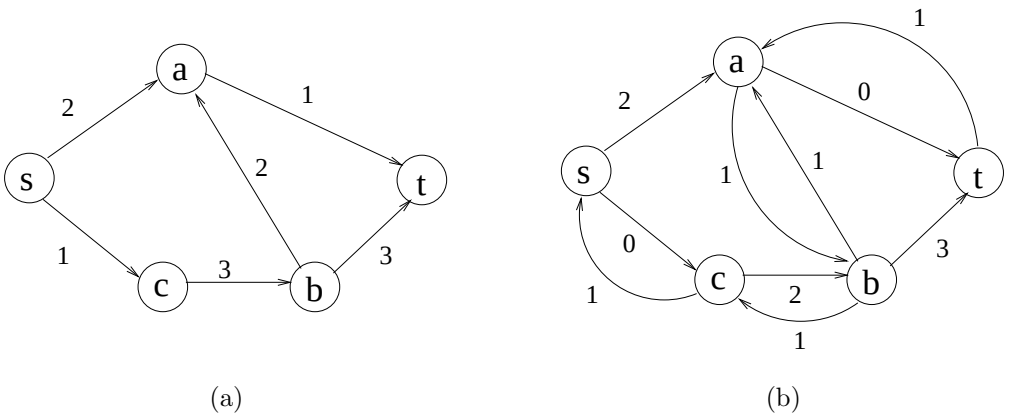


Figure 1.2: (a) A network N of positive integral edge capacities. (b) The residual graph obtained after pushing a flow of 1 through the path $s - c - b - a - t$

(2) $|f \pm f'| = |f| \pm |f'|$

$$\begin{aligned} |f \pm f'| &= \sum_{v \in V} (f \pm f')(s, v) \\ &= \sum_{v \in V} [f(s, v) \pm f'(s, v)] \\ &= \sum_{v \in V} f(s, v) \pm \sum_{v \in V} f'(s, v) \\ &= |f| \pm |f'| \end{aligned}$$

□

Lemma 1.1.3. *If f^* is maximum in G_f, then $f + f^*$ is maximum in G.*

Proof. Suppose if some flow $|g| > |f + f^*|$ in G, then it is easy to see that $g - f$ is a flow in G_f

$$\begin{aligned} \Rightarrow |g - f| = |g| - |f| &> |f + f^*| - |f| \\ &= |f| + |f^*| - |f| = |f^*| \end{aligned}$$

But, this leads to a contradiction to the fact that $|f^*|$ is maximum in G_f.

□

Definition 1.1.6. *(Capacity of a path) The capacity of a path P is given as,* $C(P) = Min_{(u,v)\in P} c(u,v)$

Definition 1.1.7. *(Augmenting Path) An augmenting path P is an s,t-path of positive capacity.*[3]

1.1.2 Bounds

Let (A, B) be any s,t-cut, then the value of the flow is at most the capacity of the cut.

$$|f| \leq C(A,B)$$

Since, from Lemma 1.1.1

$$\begin{aligned} |f| &= f(A,B) \\ &= \sum_{u\in A, v\in B} f(u,v) \\ &\leq \sum_{u\in A, v\in B} c(u,v) = C(A,B) \end{aligned}$$

1.1.3 Proof of the Max-flow Min-cut Theorem

Theorem 1. *(Ford-Fulkerson, 1956) [5] In a Network G, let f be any maximum flow in G, then $\exists$ a cut (A,B) for which $f(A,B) = c(A,B)$*

Proof. The proof given by Ford-Fulkerson is an algorithmic proof. Here, we give the algorithm that iteratively finds an augmenting path and augments the flow and computes the residual graph.

The total value of the resulting flow will be $f_1 + f_2 + f_3 \ldots + f_k$, where k is the number of iterations and f_i is the flow obtained in the i^{th} iteration. The above algorithm terminates by producing a maximum flow and also we can show the existence of a cut which will be the minimum cut. The figure 1.3 shows the execution of the *Ford-Fulkerson* algorithm on an example network.

[3] Here path means a directed path from s to t.

Algorithm 1 Maxflow(G)

```
Find an augmenting path P in G
if P does not exist then
  return 0
else
  let f be a flow of value C(P) along P
  return (f + Maxflow(G_f))
end if
```

In order to prove this theorem, we first consider the following three statements. And by showing a circular dependency between them, we prove the theorem.

(1) G has a Maximum flow $|f|$

(2) G_f has no augmenting path

(3) Some cut of G is saturated

As we can see, $1 \Rightarrow 2$ and $3 \Rightarrow 1$. These two are trivial. Because, if there exists an augmenting path, the flow can be further increased by augmenting through that path. But, this contradicts the fact that f is maximum. And, for the second, since the flow can not exceed the capacity of a cut, if some cut is saturated, then the flow has to be maximum.

What remains is to prove is that $2 \Rightarrow 3$. For this, consider two sets of vertices A and B in G_f where A consists of all the vertices reachable from s and B consists of all the vertices from which t is reachable. Note that the graph we consider is the residual graph G_f. Now, we can say that $s \notin B$ and $t \notin A$. If any of the two are not correct, it implies that there exists an augmenting path, which contradicts the hypothesis. Now, consider the cut (A, B), this forms a saturated cut. Hence proved.

□

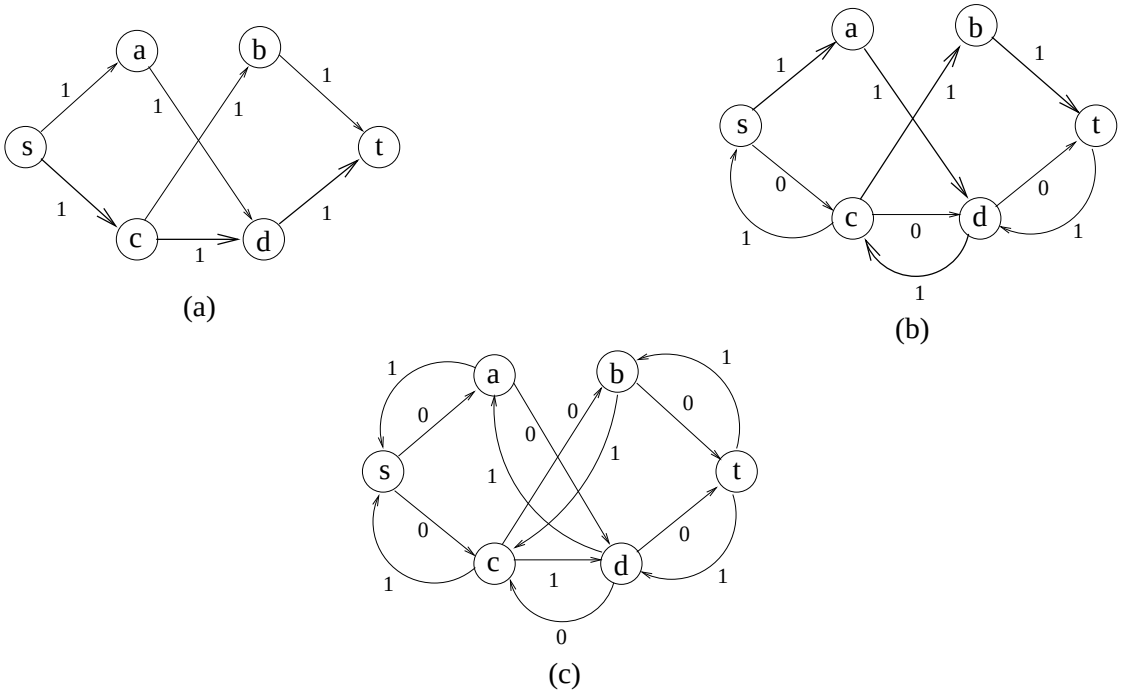


Figure 1.3: (a) The initial graph(network) G. (b) The residual graph G' after sending a flow of 1 unit along the path $s-c-d-t$. (c) The final residual graph G'' after sending a flow of 1 unit along the path $s-a-d-c-b-t$.

Chapter 2

Proving Graph Theorems using Max-flow Min-cut Theorem

In the previous chapter we've seen the proof for *Max-flow Min-cut* theorem. In this chapter we will present other theorems in graph theory that can be viewed as special cases of the *Max-flow Min-cut* theorem. The Menger's theorem [1] characterizes the connectivity of a finite graph. There are two versions of it, the edge version and the vertex version. Below, we present a proof of both the versions using the *Max-flow Min-cut* theorem. The Konig-Egervary theorem [9] describes an equivalence between the maximum matching problem and the minimum vertex cover problem in a bipartite graph. A proof of this theorem is also presented using the above *Max-flow Min-cut* theorem. The results or the proofs presented are not unknown previously. The work presented here is only a consolidation of the known facts scattered in literature to present complete and comprehensive proofs for the graph theoretic min-max results discussed here.

2.1 Preliminaries

Definition 2.1.1. *(Bipartite Graph) A graph $G(V, E)$ in which there exists a partitioning of vertices into two sets X and Y such that $\forall (x, y) \in E \Rightarrow x \in X, y \in Y$*

Definition 2.1.2. *(Vertex cover) A vertex cover in a graph $G(V, E)$ is a set S of vertices such that $\forall (x, y) \in E \Rightarrow$ either $x \in S$ or $y \in S$.*

Definition 2.1.3. *(Minimum Vertex Cover) A vertex cover that contains the minimum number of vertices.*

Definition 2.1.4. *(Matching) A matching in a graph $G(V, E)$ is a set M of edges such that no two edges contain a common vertex.*

Definition 2.1.5. *(Maximum matching) A maximum matching is a matching that contains the maximum number of edges.*

A graph may contain more than one maximum matching and one minimum vertex cover.

Definition 2.1.6. *(Internally-disjoint paths) Two paths are internally-disjoint if neither of them contains a non-endpoint vertex of the other.*

Definition 2.1.7. *(Edge-disjoint paths) Two paths are edge-disjoint if neither of them contains a common edge.*

Definition 2.1.8. *(x,y-separator) Given $x, y \in V(G)$, a set $S \in V(G) - \{x, y\}$ is an x, y-separator if $G - S$ has no x, y-path.*

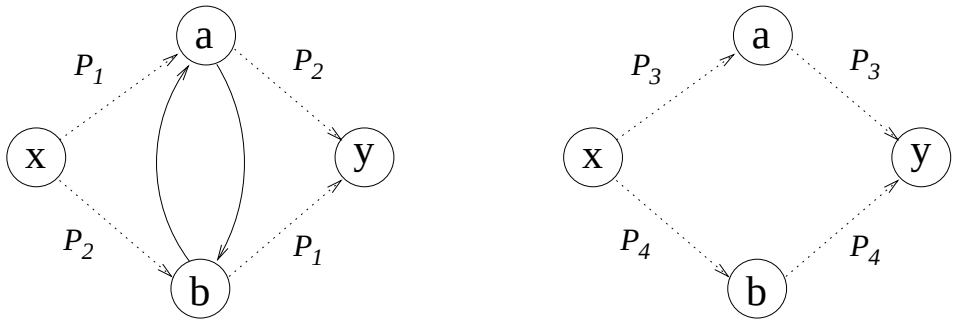
Definition 2.1.9. *(x,y-cut) Given $x, y \in V(G)$, a set $C \in E(G)$ is an x, y-cut if $G - C$ has no x, y-path.*

2.2 Menger's Theorem

In order to apply the *Max-flow Min-cut* theorem, we may need to transform the given undirected graph G into a directed graph G'. This can be done by replacing every undirected edge (a, b) in G with two directed edges (a, b) and (b, a), one forward and backward, in G'.

Lemma 2.2.1. *In a graph $G(V, E)$, the number of pairwise edge-disjoint x, y-paths in G' are equal to the number of pairwise edge-disjoint paths in G, where G' is the corresponding directed graph.*

Proof. It is easy to see that there will be at least as many pairwise edge-disjoint x, y-paths in G' as there are in G. Because if two paths are edge disjoint in G, they are edge-disjoint in G' also.



(a) (b)

Figure 2.1: (a) The paths P_1 and P_2 where $(a, b) \in P_1, (b, a) \in P_2$. (b) The corresponding paths P_3 and P_4.

To show that there can be no more in G', consider two edge-disjoint paths P_1, P_2 in G'. If there is no edge (a, b) such that $(a, b) \in P_1$ and $(b, a) \in P_2$, then the paths have to be edge-disjoint in G also. But, if such an edge exists, then we can form two new paths that does not have either (a, b) or (b, a). A path P_3 can be formed by joining the x, a-path in P_1 with a, y-path in P_2 and a path P_4 can be formed by joining the x, b-path in P_2 with b, y-path in P_1. Now, the paths P_3 and P_4 does not contain either (a, b) or (b, a). So, all such edges can be eliminated and we can construct two paths corresponding to P_1 and P_2 that are edge-disjoint in G' and these have to be edge-disjoint in G also. In this way all the pairwise edge-disjoint x, y-paths in G' correspond to pairwise edge-disjoint x, y-paths in G. Hence the lemma is proved. □

Lemma 2.2.2. *In a graph $G(V, E)$, the number of pairwise internally-disjoint x, y-paths in G' are equal to the number of pairwise internally-disjoint paths in G, where G' is the corresponding directed graph.*

Proof. Let k be the maximum number of pairwise internally-disjoint x, y-paths in G and U be the set of these paths. Then the corresponding directed paths in G' will also be pairwise internally-disjoint. Let U' be the corresponding set of paths in G'. Then there can not exist any more paths in G' that are pairwise internally-disjoint to all the paths in U'. Suppose there exists such a path P in G' that is internally-disjoint to every path in U'. Then it implies that P does not have a vertex in common with any of the paths in U'. Then the corresponding undirected path in G should also be pairwise internally-disjoint to all the paths in U. But, this leads to a contradiction to the maximality of U in G. Hence $|U'| = k$. □

Theorem 2. *(Menger's,[1927])(Edge version) If x, y are two vertices in a graph G and $(x, y) \notin E(G)$, then the minimum size of an x, y-cut equals the maximum number of pairwise edge-disjoint x,y-paths*

Proof. The graph $G(V, E)$ is an undirected graph. Let k be the maximum number of pairwise edge-disjoint x, y-paths(undirected) in G. The necessary part of the theorem can be easily verified. Since the minimum x, y-cut should contain at least one edge from each x, y-path, it should be at least k. Therefore, the size of the minimum x, y-cut is $\geq k$.

In order to prove sufficiency, we first transform the given undirected graph into a network with x as source and y as sink. Let G' be the new directed graph(Network). Assign unit capacities to all the directed edges in G'. Then by lemma 2.2.1 the number of pairwise edge-disjoint x, y-paths in G' is equal to k. Since the edges are of unit capacity, the maximum flow possible will be equal to the number of pairwise edge-disjoint x, y-paths. So maximum flow in G' is k. Now, by applying the *Max-flow Min-cut* theorem, we can see that the size of the minimum cut(directed) in G' is k. It is easy to see that the corresponding set of undirected edges in G also form a cut in G. And since every cut in G is at least k, this will be minimum.

□

Theorem 3. *(Vertex version) If x, y are two vertices in a graph G and $(x, y) \notin E(G)$, then the minimum size of an x, y-separator equals the maximum number of pairwise internally-disjoint x, y-paths.*

Proof. Let k be the maximum number of pairwise internally-disjoint x, y-paths in G. Then the size of the minimum x, y-separator should be at least k since it should include at least one vertex from each path.

In order to prove sufficiency, we will use the *Max-flow Min-cut* theorem. First, we will transform the undirected graph G to a directed graph G'. From lemma 2.2.2, the number of pairwise internally-disjoint paths in G' are k. Now, we will transform the directed graph G' as follows. Let G'' be the new graph. For every vertex $v \in G'$, include two vertices in G'', v_{in} and v_{out}. v_{in} consists of all the inward edges to v and v_{out} consists of all the outward edges from v. Add a directed edge from v_{in} to v_{out}. Assign unit capacity to all the edges of G''. x_{in} and y_{out} can be left out because there will be no flow into the source vertex and out of sink vertex.

Now, we show that the maximum flow in G'' is k. Clearly, every path in G' correspond to unique path in G''. A path entering a vertex v_{in} has to leave through the directed edge (v_{in}, v_{out}) and this edge is of unit capacity. This implies, no two augmenting paths in G'' can have a common vertex. The k paths in G'' that correspond to the pairwise internally-disjoint x, y-paths in G' will each allow for a flow of one unit. There can not be any more flow. If the flow is more than k, then there exists an augmenting path that is pairwise internally-disjoint to all other augmenting paths in G''. Then the corresponding path in G' will also be pairwise internally-disjoint to all other paths in G''. But, this leads to a contradiction to the fact that k is maximum in G'.

Now, by applying the *Ford Fulkerson* algorithm, we can get a cut of size k.

Select a set S of vertices for each edge in the cut as follows:

(1) If the edge is (v_{in}, v_{out}), select v.

(2) If the edge is (v_{out}, u_{in}) where $\{u, v\} \notin \{s, t\}$, select either u or v.

(3) In case of (x_{out}, v_{in}) or (v_{out}, y_{in}), select the vertex v.

Clearly, the size of S is k. This set also forms a separator in the network G'. Otherwise, there exists at least one more x, y-path in G' from which no edge is included in S. But, the corresponding path in the modified network G'' will be a x, y-augmenting path. This contradicts the fact that f is maximum in G''. Also it is easy to see that this set S will be a separator in original graph G. And since the size of the minimum separator is at least k, this is minimum. Hence proved.

□

2.3 Konig-Egervary Therorem

Theorem 4. *(Konig's theorem,[1931]) In any bipartite graph, the number of edges in a maximum matching equals the number of vertices in a minimum vertex cover.*

Proof. Let $G(V, E)$ be the bipartite graph with partitions X, Y. Let k be the size of a maximum matching, M. Then, the size of the minimum vertex cover is at least k. Since, no two edges in a matching has a common vertex, a vertex cover should consist of at least one vertex for each edge.

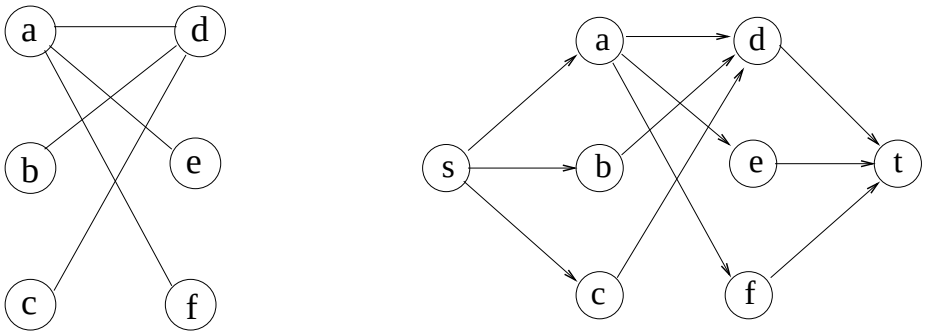
What remains is to show that there exists a vertex cover of k. For this, we first transform the given graph as below. Construct the new graph $G'(V', E')$, where V' consists of all the vertices from G along with two new vertices s, t corresponding to source and sink and add edges to E' as below:

(1) $\forall u \in X, v \in Y$ add a directed edge (u, v)

(2) $\forall u \in X$ add the directed edge (s, u)

(3) $\forall v \in Y$ add the directed edge (v, t)

Assign unit capacities to all the edges in E'.



(a) Original Graph (b) Transformed

Figure 2.2: (a) A bipartite graph with partitions $\{a, b, c\}$ and $\{d, e, f\}$. (b) The resulting network after transformation.

We now show that the no of pairwise internally-disjoint s, t-paths in G' is equal to the size of the maximum matching in G. Since no two edges in M have a common vertex, all the s, t-paths corresponding to the edges in M are pairwise internally-disjoint. There can not be any other path that is internally-disjoint to all these paths. Because, if there exists such a path, say $s - x - y - t$, then the size of matching M can be increased by adding the edge (x, y). Note that every path in G' is of length three. This contradicts the fact that M is a maximum matching in G. Therefore, number of pairwise internally-disjoint paths in G' is k.

By applying *Menger's Theorem(Vertex version)* to G', the size of minimum s, t-separator, say W, is k. We show that W forms a vertex cover in G. Since, for every edge $(x, y) \in G$ there exists a s, t-path in G'. For every s, t-path, $s - x - y - t$, the s, t-separator consists of either x or y or both. So, W forms a vertex cover of size k in G.

□

Chapter 3

Max-flow Min-cut Theorem and Konig's Theorem using Total Unimodularity

In this chapter, we will use a different proof technique to prove the Konig's theorem and Max-flow Min-cut theorem. We consider the linear programming formulation of the problem and show that the optimal values of primal and dual are equal. We use the total unimodularity property of coefficient matrix and the fundamental theorem of duality in linear program to drive this equivalence. The total unimodularity of the coefficient matrix helps in determining the integrality of the solution. The proof was given by A Chandra Babu et al.[2]. We have filled a few missing links in this material.

3.1 Preliminaries

Definition 3.1.1. *(Unimodular Matrix) A matrix M over real numbers is said to be unimodular if every square sub matrix of M has determinant equal to 0,1 or -1.*

Examples of totally unimodular matrices,

$$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} -1 & -1 & 0 & 0 & 0 & 1 \\ 1 & 0 & -1 & -1 & 0 & 0 \\ 0 & 1 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 1 & -1 \end{bmatrix}$$

Definition 3.1.2. *(Linear Program) Let P be a maximization problem. Consider this as the primal. Then the linear program formulation can be given as*

$$\begin{aligned} &\textit{Maximize } c^T x \\ &\textit{Subject to } Ax \leq b \\ &x \geq 0 \end{aligned} \tag{3.1.1}$$

Where A is a $m \times n$ matrix and c, x, b are column vectors of order n, n, m respectively. c^T denotes the transpose of c.

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$$

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad c = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix} \quad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

Definition 3.1.3. *(Dual of a Linear Program) The dual of the LPP(P), say $D(P)$, can be given as*

$$\begin{aligned} &\textit{Minimize } b^T y \\ &\textit{Subject to } A^T y \geq c \\ &y \geq 0 \end{aligned}$$

Where y is a column vector of order m.

$$y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

Theorem 5. *(Weak Duality) For any feasible solutions x' and y' of P and $D(P)$, $c^T x'$ is always less than or equal to $b^T y'$.*

Proof. For a feasible x' and y' of P and $D(P)$, the inequality constraints will be satisfied. Consider the inequalities in P and multiply with y'^T on both sides.

$$\begin{aligned} Ax' &\leq b \\ \Rightarrow y'^T Ax' &\leq y'^T b \end{aligned} \tag{3.1.2}$$

Now, consider the objective function $c^T x'$ of P. We have,

$$\begin{aligned} A^T y' &\geq c \\ \Rightarrow (A^T y')^T x' &\geq c^T x' \\ y'^T Ax' &\geq c^T x' \end{aligned} \tag{3.1.3}$$

Now, from equations 3.1.2 and 3.1.3 we have

$$c^T x' \leq y'^T Ax' \leq y'^T b \tag{3.1.4}$$

We can see that $y'^T b = b^T y'$. Since,

$$\begin{aligned} y'^T b &= \begin{bmatrix} y_1 & y_2 & \dots & y_m \end{bmatrix} \times \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \\ &= \begin{bmatrix} b_1 y_1 + b_2 y_2 + \ldots + b_m y_m \end{bmatrix} \\ &= \begin{bmatrix} b_1 & b_2 & \dots & b_n \end{bmatrix} \times \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \\ &= b^T y' \end{aligned} \tag{3.1.5}$$

Therefore,

$$c^T x' \leq y'^T A x' \leq b^T y' \tag{3.1.6}$$

□

Theorem 6. *(Strong Duality) [?] If the primal P has an optimal solution, x^*, then the dual $D(P)$ also has an optimal solution, y^*, such that $c^T x^* = b^T y^*$.*

Theorem 7. *(Unimodular LP) For the LPP(P), if A is a unimodular matrix and b is integral then some optimal solution is integral.*

The proof of the above two theorems is beyond the scope of this report. The proof of the Theorem 6 is found in [?] and the proof of the Theorem 7 is found in [2].

3.2 Konig's Theorem

Theorem 8. *(Konig's theorem,[1931]) In any bipartite graph, the number of edges in a maximum matching equals the number of vertices in a minimum vertex cover.*

Proof. Let $G(V, E)$ be a bipartite graph and X, Y be the two partitions of V. Let $|X| = m, |Y| = n$ and $\{1, 2, \ldots, n\} \in X$,$\{m+1, m+2, \ldots, m+n\} \in Y$.

If E is empty, then the size of both the minimum vertex cover and maximum matching is zero. Without loss of generality we shall assume that E is non empty and $|E| = r$. We shall introduce two variables p and q corresponding to every vertex and edge respectively. The linear program(P) for finding the maximum matching is as follows,

$$\text{Maximize} \sum_{(i,j)\in E} q_{ij} \tag{3.2.1a}$$

$$\text{Subject to} \sum_{j:(i,j)\in E} q_{ij} \leq 1 \qquad \forall i \in X \tag{3.2.1b}$$

$$\sum_{i:(i,j)\in E} q_{ij} \leq 1 \qquad \forall j \in Y \tag{3.2.1c}$$

$$q_{ij} \in \{0, 1\} \qquad \forall (i,j) \in E \tag{3.2.1d}$$

The first two constraints imply that at most one edge can be selected corresponding to every node.

Here, the given problem is an Integer Linear Program. We relax the integrality constraints on q_{ij} so that it can take on decimal values. The resulting LPP, say P', is

$$\begin{aligned} &\text{Maximize} \sum_{(i,j)\in E} q_{ij} \\ &\text{Subject to} \\ &\sum_{j:(i,j)\in E} q_{ij} \leq 1 \qquad \forall i \in X \\ &\sum_{i:(i,j)\in E} q_{ij} \leq 1 \qquad \forall j \in Y \\ &q_{ij} \geq 0 \qquad \forall (i,j) \in E \end{aligned} \tag{3.2.2}$$

Let A be the coefficient matrix of the LPP(P'). By Theorem 3.2.1 below,

A is unimodular. Hence by Lemma 7, there exists an optimal integral solution to P'. Obviously, every element of the solution is either 0 or 1. Otherwise it would violate the constraints. Hence optimal solution of P' is also optimal solution of P. This is the size of the maximum matching in G.

Consider the LP in equation 3.2.2. We shall derive the dual of this LP formulation. Let p_i and p_j where $i \in X, j \in Y$ be the dual variables corresponding to the first and second set of constraints, respectively. The matrices corresponding to the primal formulation 3.1.1 and dual formulation 3.1.3 problems are as follows,

$$x = \begin{bmatrix} q_{ij} \\ \vdots \\ \vdots \\ \vdots \end{bmatrix} \qquad c = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \qquad b = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \qquad y = \begin{bmatrix} p_1 \\ \vdots \\ p_m \\ p_{m+1} \\ \vdots \\ p_{m+n} \end{bmatrix} \tag{3.2.3}$$

Where the order of the matrices are, x is $r \times 1$, c is $r \times 1$, b is $(m+n) \times 1$ and y is $(m+n) \times 1$. The optimization function with respect to definition 3.1.3 can be given as,

$$b^T y = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ \vdots \\ \vdots \\ 1 \end{bmatrix}^T \times \begin{bmatrix} p_1 \\ \vdots \\ p_m \\ p_{m+1} \\ \vdots \\ p_{m+n} \end{bmatrix} \tag{3.2.4}$$

From Theorem 3.2.1, we know the coefficient matrix is unimodular. In the matrix A, with respect to a variable q_{ij}, every column has exactly two ones corresponding to i^{th} and j^{th} rows. Thus, in A^T every row has two ones, one in the i^{th}

column and other in the j^{th} column. So, the set of constraints, $A^T y \geq c$, for the dual can be given as follows,

$$p_i + p_j \geq 1 \quad \forall (i,j) \in E \tag{3.2.5}$$

Along with the non-negativity constraints, the dual can be given as,

$$\begin{aligned} &\text{Minimize } \sum_{i \in X} p_i + \sum_{j \in Y} p_j \\ &\text{Subject to,} \\ &\qquad p_i + p_j \geq 1 \qquad \forall (i,j) \in E \\ &\qquad p_i \geq 0 \qquad\qquad \forall i \in X \\ &\qquad p_j \geq 0 \qquad\qquad \forall j \in Y \end{aligned} \tag{3.2.6}$$

This actually is the LP formulation for the relaxed minimum vertex cover problem that we see below.

Now, consider the problem for finding the minimum vertex cover in G. Then the LPP formulation, say Q, is given as follows,

$$\begin{aligned} &\text{Minimize } \sum_{i \in X} p_i + \sum_{j \in Y} p_j \\ &\text{Subject to,} \\ &\qquad p_i + p_j \geq 1 \qquad \forall (i,j) \in E \\ &\qquad p_i \in \{0,1\} \qquad \forall i \in X \\ &\qquad p_j \in \{0,1\} \qquad \forall j \in Y \end{aligned} \tag{3.2.7}$$

The above given problem is an ILP(Integer Linear Program). We shall relax the integrality constraints on x_i, x_j in order to obtain a LP. Let Q' be the new LP.

The new LP is as follows,

$$\text{Minimize } \sum_{i \in X} p_i + \sum_{j \in Y} p_j$$

Subject to,

$$\begin{aligned} p_i + p_j &\geq 1 \qquad \forall (i,j) \in E \\ p_i &\geq 0 \qquad \forall i \in X \\ p_j &\geq 0 \qquad \forall j \in Y \end{aligned} \tag{3.2.8}$$

Let B be the coefficient matrix of Q'. It is easy to see that B is the transpose of A. Since A is unimodular, B is also unimodular. Therefore, by Theorem 3.2.1, Q' has an optimal integral solution in which all p_i and q_j are either 0 or 1. Hence, optimal solution to Q' is also optimal solution to Q. That is equal to the size of the minimum vertex cover.

Now, by the Theorem 6, both the problems have same optimal solution. □

Lemma 3.2.1. *The coefficient matrix A of the LPP (P′) is totally unimodular.*

Proof. Clearly, A has $m + n$ rows and r columns in which first m equations contribute the first m rows and second n equations contribute the remaining n rows. Each column of A has exactly two 1's, one in the first m rows and one in the last n rows. All other elements are 0. Let D be a square sub-matrix of order k. We will prove the theorem by using induction on k.

Clearly, for $k = 1$, $|D| = 0$ or 1. Assume that all square sub-matrices of order $k-1$ have determinant equal to $0, 1$ or -1. We shall consider different cases,

(1) If D has at least one column containing only zeros, then $|D| = 0$.

(2) If D has at least one column containing only one one. Then $|D| = \pm|E|$, where E is the sub-matrix obtained from D by deleting the corresponding column and the row containing the one. By induction, $|E| = 0, 1$ or -1. Hence $|D| = 0, 1$ or -1.

(3) If every column of D has exactly two 1. In this case, the first 1 comes from the first m rows of A and the second 1 comes from the later n rows of A. Strictly, every column has exactly a single 1 within the rows that belong to the first m rows of A. So, the by performing row addition on all these rows, we obtain a row with all ones. The same argument applies for the remaining rows of D that came from the last n rows of A. Hence the rows of D are linearly dependent and $|D| = 0$.

Hence A is unimodular. □

3.3 The Max-Flow Min-Cut Theorem

Theorem 9. *(Ford-Fulkerson, 1956) In a Network G, let f be any maximum flow in G, then $\exists$ a cut (A, B) for which $f(A, B) = c(A, B)$*

Proof. Let $N(V, E, c, s, t)$ be the network with $|V| = n$ and $|E| = m$. We shall write the linear programming formulation for the maximum flow.

We want to find the maximal flow that can be sent from the source vertex s to the sink vertex t. Let v be the value of any flow from s to t and x_{ij} be the flow sent along the arc (i, j). Let the vertices be labeled using integers 1 to n such that the source s is labeled as 1 and the sink t is labeled with n. Then the LPP(R)

corresponding to the maximum flow is,

Maximize v

Subject to

$$\begin{aligned}
&\sum_{(i,j)\in E} x_{ij} - \sum_{(k,i)\in E} x_{ki} - v = 0 && \text{if } i = 1\\
&\sum_{(i,j)\in E} x_{ij} - \sum_{(k,i)\in E} x_{ki} + v = 0 && \text{if } i = n\\
&\sum_{(i,j)\in E} x_{ij} - \sum_{(k,i)\in E} x_{ki} = 0 && \text{if } i = 2, 3, \ldots, n-1\\
&x_{ij} \leq c_{ij} && \forall (i,j) \in E\\
&x_{ij} \geq 0 && \forall (i,j) \in E\\
&v \geq 0
\end{aligned} \tag{3.3.1}$$

The first constraint imply that the net flow out of the source vertex 1 is equal to v and the second constraint imply that the net flow into the sink vertex n is v. The third constraint imply that the total flow into any intermediate vertex is equal to the total outflow of that vertex. By relaxing the equality in these three constraints, we will obtain the following inequalities.

$$\begin{aligned}
&\sum_{(i,j)\in E} x_{ij} - \sum_{(k,i)\in E} x_{ki} - v \leq 0 && \text{if } i = 1\\
&\sum_{(i,j)\in E} x_{ij} - \sum_{(k,i)\in E} x_{ki} + v \leq 0 && \text{if } i = n\\
&\sum_{(i,j)\in E} x_{ij} - \sum_{(k,i)\in E} x_{ki} \leq 0 && \text{if } i = 2, 3, \ldots, n-1
\end{aligned} \tag{3.3.2}$$

Let R' be the resulting LPP. In any optimal solution of R', the above three inequalities should satisfy with equality. Otherwise, by adding all the LHS and RHS we get $0 < 0$, a contradiction. Since, on the LHS for every arc (i, j), x_{ij} is added once and subtracted once, so the sum will result in a zero. So, they will satisfy with equality. Therefore, the optimal solution of R' will also be the optimal solution of R.

Consider the LP for Max-flow in equation 3.3.1. If we try to convert this into matrix form, the corresponding matrices will be,

$$x = \begin{bmatrix} x_{ij} \\ \vdots \\ \vdots \end{bmatrix} \qquad b = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ c_{ij} \\ \vdots \end{bmatrix} \qquad c = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \end{bmatrix} \tag{3.3.3}$$

Now, we shall derive the dual for the above LP in equation 3.3.1. Let u_i, where $1 \leq i \leq n$, be the dual variables corresponding to the first three set of equations (flow constraints) and y_{ij}, where $(i, j) \in E$, be the dual variables corresponding to the fourth set of constraints (capacity constraints). Then the matrix y is,

$$y = \begin{bmatrix} u_1 \\ \vdots \\ u_n \\ y_{ij} \\ \vdots \end{bmatrix} \tag{3.3.4}$$

Therefore, the objective funtion for the dual can be given as,

$$b^T y = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ c_{ij} \\ \vdots \end{bmatrix}^T \times \begin{bmatrix} u_1 \\ \vdots \\ u_n \\ y_{ij} \\ \vdots \end{bmatrix} \tag{3.3.5}$$

Consider the coefficient matrix, say A, of the LP formulation for Max-flow. We shall see the properties of this matrix. There will be $(n+m)$ rows corresponding

to the $(n+m)$ constraints and $m+1$ columns corresponding to m x_{ij} variables. The coefficient matrix A, will look as below,

$$\begin{array}{c} \\ 1 \\ 2 \\ \vdots \\ n \\ x_{ij} \\ \vdots \\ \vdots \\ \vdots \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & \dots & m+1 \end{array} \\ \begin{pmatrix} 1 & a_{2,1} & a_{3,1} & \dots & a_{m+1,1} \\ 0 & a_{2,2} & a_{3,2} & \dots & a_{m+1,2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -1 & a_{2,n} & a_{3,n} & \dots & a_{m+1,n} \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix} \end{array}$$

where $a_{i,j} = -1$ or 0 or 1 $2 \leq i \leq (m+1)$, $\forall$ $1 \leq j \leq n$. This matrix is actually the transpose of the coefficient matrix for the Min-cut LP that we see below in equation 3.3.6. The objective function of the Min-cut is also the same as the function in equation 3.3.5. From this, it is easy to see that the dual of the Max-flow problem is Min-cut.

Now we shall formulate the linear program(T) for finding the minimum cut capacity as follows,

$$\begin{aligned} &\text{Minimize} \sum_{(i,j)\in E} c_{ij}y_{ij} \\ &\text{Subject to} \\ &\qquad -u_1 + u_n \geq 1 \\ &\qquad u_i - u_j + y_{ij} \geq 0 \quad \forall (i,j) \in E \\ &\qquad u_i \in \{0,1\} \qquad \forall i \\ &\qquad y_{ij} \in \{0,1\} \qquad \forall i,j \end{aligned} \tag{3.3.6}$$

The solution to the above LPP will result in a cut such that, corresponding

to a cut $(S, \bar{S})$ of the network N,

$$\begin{aligned} u_i &= 0 \text{ if vertex } i \in S \\ &= 1 \text{ if vertex } i \in \bar{S} \\ y_{ij} &= 1 \text{ if } i \in S, j \in \bar{S} \\ &= 0 \text{ otherwise} \end{aligned} \tag{3.3.7}$$

Now, relax the integrality constraints on u_i and y_{ij}. The resulting LPP(T') will be

$$\begin{aligned} &\text{Minimize } \sum_{(i,j)\in E} c_{ij} y_{ij} \\ &\text{Subject to} \\ &\quad -u_1 + u_n \geq 1 \\ &\quad u_i - u_j + y_{ij} \geq 0 \quad \forall (i,j) \in E \\ &\quad u_i \geq 0 \qquad \forall i \\ &\quad y_{ij} \geq 0 \qquad \forall i, j \end{aligned} \tag{3.3.8}$$

In any optimal solution to T, $(u_i, u_j) = (0.0)$ or $(1, 0)$ or $(1, 1)$ will imply $y_{ij} = 0$ and $(u_i, u_j) = (0, 1)$ imply $y_{ij} = 1$ and hence T will give the capacity of the cut $(S, \bar{S})$.

Now, consider the LPP(T'). Clearly, this is the dual of the LPP(R'). Let B be the coefficient matrix of T'. From Lemma 3.3.1 below, B is unimodular. The column vector b of T' consists either 0's or 1's. Then the optimal solution of T' is also the optimal solution of T.

Now, by Theorem 6, the optimal values of R' and T' are equal. Hence, the optimal values of R and T are also equal. Therefore, the maximum flow in the network is equal to the minimum cut in the network. □

Lemma 3.3.1. *The coefficient matrix B of the LPP(T) is unimodular.*

Proof. Consider the matrix B. Clearly B has $m+1$ rows and $n+m$ columns. Now, let D and E be two partitions of B, such that D consists of the first n columns

and E consists of the second m columns. The matrix D is of order $(m+1) \times n$ and E is of order $(m+1) \times m$. The matrix D will be as below,

$$\begin{array}{c} \\ 1 \\ 2 \\ \vdots \\ m+1 \end{array} \begin{array}{c} \begin{array}{cccc} u_1 & u_2 & \dots & u_n \end{array} \\ \begin{pmatrix} 1 & 0 & \dots & -1 \\ a_{2,1} & a_{2,2} & \dots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m+1,1} & a_{m+1,2} & \dots & a_{m+1,n} \end{pmatrix} \end{array}$$

where $a_{i,j} = -1$ or 0 or 1 $\forall$ $2 \leq i \leq (m+1)$, $1 \leq j \leq n$. And the matrix E will be as below,

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ \vdots \\ m+1 \end{array} \begin{array}{c} \begin{array}{cccc} y_{ij} & \dots & \dots & \end{array} \\ \begin{pmatrix} 0 & 0 & \dots & 0 \\ 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix} \end{array}$$

Every row of C contains exactly one 1 and one -1. Every column of D will contain exactly one 1. Now, we shall prove this by induction on the size of the sub-matrix. let U be the square sub-matrix of order k of B. For, k=1, the element can only be either 1 or 0 or -1. Assume that all square sub-matrices of order $k-1$ have determinant equal to $0, 1$ or -1. Considering for k, the different cases can be as,

(1) U consists of at least one column from D. Every column of D has exactly one 1. Then by deleting that column and the corresponding row, we can get a matrix of order $k-1$. Hence, $|U| = -1$ or 0 or 1.

(2) U consists of columns only from C. Now, if there exists a row with all zeros or one 1 or one -1, then by induction we can see that the determinant of U

will be 1 or 0 or -1. Otherwise, every row of U should contain exactly one 1 and one -1. Then by performing the column addition on all the columns will result in a column with all zeros. Then $|U| = 0$.

□

Chapter 4

The Concurrent Multi-commodity Flow Problem

This chapter consists of introduction to Multi-commodity flow problem. The Multi-commodity flow problem is a more generalized network flow problem. In a multi-commodity flow problem there are $k \geq 1$ commodities each having its own source and sink pair. Because of the multiplicity of the commodities, the problem to be optimized can be defined in several ways. Hence, there exists several variants of the Multi-commodity flow. The problem we are going to discuss is called Concurrent Multi-commodity Flow Problem(CMFP) [15][12]. In this problem, every i^{th} commodity is assigned a demand D_i. Our objective is to maximize a fraction λ, such that there exists a flow of λD_i units for every commodity i. The previously studied maximum flow problem is a special case of Multi-commodity flow problem in which the number of commodities is one($k = 1$). Below, we will give formal definitions and linear program formulation of the problem.

4.1 Preliminaries

Definition 4.1.1. *(Multi-commodity Network) A Multi-commodity Network is a directed graph $G(V, E, c)$ with vertex set V and an arc set E in which every directed edge $(i, j) \in E$, has a non negative capacity $c(i, j) \geq 0$, $c : V \times V \longrightarrow \mathbf{R}^+$. There are $k \geq 1$ commodities $K_1, K_2, \ldots, K_k$. For each commodity i, there is an ordered pair (s_i, t_i) representing the source and sink of that commodity where $(s_i, t_i) \in V \times V$ and $s_i \neq t_i$.*

The flow of a commodity is similar to that of a flow in a single-commodity flow network. Let f^i represent the flow of the i^{th} commodity.

Definition 4.1.2. *(Flow of a Commodity) A flow of a commodity is a mapping* $f^i : E \longrightarrow R$ *denoted by* f^i_{uv} *or* $f^i(u,v)$*, subject to the following constraints:*

(1) $f^i(u,v) \leq c(u,v)$ *for each* $(u,v) \in E$ *(capacity constraint)*

(2) $f^i(u,v) = -f^i(v,u)$ *(skew symmetry)*

(3) $\sum_{v \in V} f^i(u,v) = 0 \ \forall \ u \in V \setminus \{s_i, t_i\}$ *(conservation of flow).*

The value of the flow of a commodity i is given by $|f_i| = \sum_{w \in V} f^i(s_i, w)$.

Definition 4.1.3. *(Concurrent Multi-commodity Flow Problem) Given a Multi-commodity Network along with demands* $D_1, D_2, \ldots, D_k$ *corresponding to the* k *commodities, the objective is to assign flow to commodities so as to maximize a fraction* λ *such that for every commodity* i*, the value of the flow of the commodity* $|f_i|$ *is at least* λD_i*. The assignment should satisfy the following constraints along with the flow constraints,*

$$\sum_{i=1}^{k} f^i(u,v) \leq c(u,v) \ \forall (u,v) \in E \tag{4.1.1}$$

4.2 Linear Programming Formulation

Below, we give the LP formulation for the Concurrent Multi-commodity Flow problem(CMFP).

Maximize λ

Subject to

$$\begin{aligned} &\sum_{i=1}^{k} f^i(u,v) \leq c(u,v) && \forall (u,v) \in E \\ &\sum_{w \in V} f^i(u,w) = 0 && \forall 1 \leq i \leq k,\ \forall u \in V - \{s_i, t_i\} \\ &\sum_{w \in V} f^i(s_i,w) \geq \lambda D_i && \forall 1 \leq i \leq k \end{aligned} \tag{4.2.1}$$

The above formulation is very intuitive and straight forward. We will now see another formulation of the same problem which uses paths. Let P represent the set of all non-trivial paths in the network and P_j be the set of paths corresponding to the commodity j (paths from s_i to t_i). Let $x(\alpha)$ be a variable corresponding to every path $\alpha \in P$. Then, the LPP formulation (primal) can be given as,

Maximize λ

Subject to

$$\begin{aligned} &\sum_{\alpha : e \in \alpha} x(\alpha) \leq c(e) && \forall e \in E \\ &\lambda D_j - \sum_{\alpha \in P_j} x(\alpha) \leq 0 && \forall 1 \leq j \leq k \\ &x(\alpha) \geq 0 && \forall \alpha \in P \end{aligned} \tag{4.2.2}$$

The dual problem for the above linear program can be interpreted as assigning weights(z_j) to the commodities and lengths($y(e)$) to edges such that for any commodity i, the length of every path from s_i to t_i should be at least z_i. The length of a path is given as the sum of the lengths of all the edges in that path. The LPP

formulation for the dual is as follows,

$$\text{Minimize } \sum_{e \in E} c(e)y(e)$$

Subject to

$$\begin{aligned} &\sum_{e \in \alpha} y(e) \geq z_j \qquad \forall \alpha \in P_j,\ \forall j \\ &\sum_{1 \leq j \leq k} D_j z_j \geq 1 \\ &l(e) \geq 0 \qquad\qquad \forall e \in E \\ &z_j \geq 0 \qquad\qquad \forall j \end{aligned} \tag{4.2.3}$$

We shall consider the example in the figure 4.1 with two commodities, say K_1 and K_2. Let the demands be D_1 and D_2 respectively. First, the primal formulation of the problem using paths is given. We shall then derive the dual formulation of the problem.

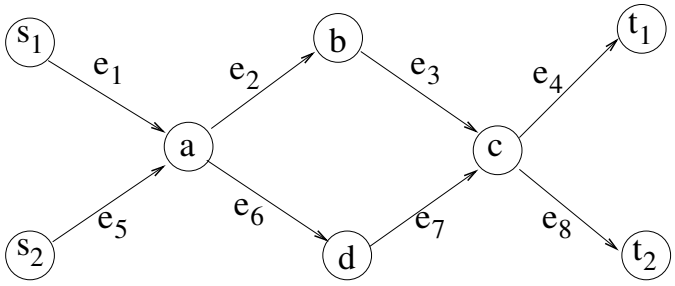


Figure 4.1: An example of a two commodity flow network with unit demands on the commodities.

In the example figure, there are two paths from source to sink for each of the commodities K_1 and K_2. The edges are labeled $e_1, e_2, \ldots, e_8$. Let us name the paths with respect to edges as follows,

$$\begin{aligned}
\alpha_1 &\rightarrow e_1 - e_2 - e_3 - e_4 \\
\alpha_2 &\rightarrow e_1 - e_6 - e_7 - e_4 \\
\alpha_3 &\rightarrow e_5 - e_2 - e_3 - e_8 \\
\alpha_4 &\rightarrow e_5 - e_6 - e_7 - e_8
\end{aligned} \tag{4.2.4}$$

The paths α_1, α_2 belong to commodity K_1 and the paths α_3, α_4 belong to commodity K_2. The LP formulation for the example is as follows,

Maximize λ

Subject to

$$\begin{aligned}
x(\alpha_1) + x(\alpha_2) &\leq c(e_1) \\
x(\alpha_1) + x(\alpha_4) &\leq c(e_2) \\
x(\alpha_1) + x(\alpha_4) &\leq c(e_3) \\
x(\alpha_1) + x(\alpha_2) &\leq c(e_4) \\
x(\alpha_3) + x(\alpha_4) &\leq c(e_5) \\
x(\alpha_3) + x(\alpha_2) &\leq c(e_6) \\
x(\alpha_3) + x(\alpha_2) &\leq c(e_7) \\
x(\alpha_3) + x(\alpha_4) &\leq c(e_8) \\
\lambda D_1 - x(\alpha_1) - x(\alpha_2) &\leq 0 \\
\lambda D_2 - x(\alpha_3) - x(\alpha_4) &\leq 0 \\
x(\alpha_i) &\geq 0 \qquad\qquad \forall 1 \leq i \leq 4
\end{aligned} \tag{4.2.5}$$

Now, multiply the constraints with the dual variables($y(e)$ and z_j) on both sides.

$$
\begin{aligned}
&[\ x(\alpha_1) + x(\alpha_2) \leq c(e_1)\ ]\ y(e_1)\\
&[\ x(\alpha_1) + x(\alpha_4) \leq c(e_2)\ ]\ y(e_2)\\
&[\ x(\alpha_1) + x(\alpha_4) \leq c(e_3)\ ]\ y(e_3)\\
&[\ x(\alpha_1) + x(\alpha_2) \leq c(e_4)\ ]\ y(e_4)\\
&[\ x(\alpha_3) + x(\alpha_4) \leq c(e_5)\ ]\ y(e_5)\\
&[\ x(\alpha_3) + x(\alpha_2) \leq c(e_6)\ ]\ y(e_6)\\
&[\ x(\alpha_3) + x(\alpha_2) \leq c(e_7)\ ]\ y(e_7)\\
&[\ x(\alpha_3) + x(\alpha_4) \leq c(e_8)\ ]\ y(e_8)\\
&[\ \lambda D_1 - x(\alpha_1) - x(\alpha_2) \leq 0\ ]\ z_1\\
&[\ \lambda D_2 - x(\alpha_3) - x(\alpha_4) \leq 0\ ]\ z_2
\end{aligned}
\tag{4.2.6}
$$

By combining the equations on the left hand and the right hand sides we get the following inequality,

$$
\begin{aligned}
&[\ x(\alpha_1) + x(\alpha_2)]\ y(e_1) + [\ x(\alpha_1) + x(\alpha_4)]\ y(e_2) +\\
&[\ x(\alpha_1) + x(\alpha_4)]\ y(e_3) + [\ x(\alpha_1) + x(\alpha_2)]\ y(e_4) +\\
&[\ x(\alpha_3) + x(\alpha_4)]\ y(e_5) + [\ x(\alpha_3) + x(\alpha_2)]\ y(e_6) +\\
&[\ x(\alpha_3) + x(\alpha_2)]\ y(e_7) + [\ x(\alpha_3) + x(\alpha_4)]\ y(e_8) +\\
&\qquad [\ \lambda D_1 - x(\alpha_1) - x(\alpha_2)\ ]\ z_1 +\\
&\qquad\quad [\ \lambda D_2 - x(\alpha_3) - x(\alpha_4)\ ]\ z_2 \leq \sum_{i=1}^{8} c(e_i) y(e_i)
\end{aligned}
\tag{4.2.7}
$$

Now, represent the inequality in terms of $x(\alpha)$,

$$\begin{aligned}
&[\ y(e_1)+y(e_2)+y(e_3)+y(e_4)-z_1\ ]\ x(\alpha_1)+\\
&[\ y(e_1)+y(e_6)+y(e_7)+y(e_4)-z_1\ ]\ x(\alpha_2)+\\
&[\ y(e_5)+y(e_6)+y(e_7)+y(e_8)-z_2\ ]\ x(\alpha_3)+\\
&[\ y(e_5)+y(e_2)+y(e_3)+y(e_8)-z_2\ ]\ x(\alpha_4)+\\
&\qquad\qquad [\ D_1z_1+D_2z_2\ ]\ \lambda \leq \sum_{i=1}^{8} c(e_i)y(e_i)
\end{aligned} \tag{4.2.8}$$

Now, with respect to the objective function of the primal, the dual can be formulated as below,

$$\text{Minimize } \sum_{i=1}^{8} c(e_i)y(e_i)$$

Subject to

$$\begin{aligned}
&y(e_1)+y(e_2)+y(e_3)+y(e_4)-z_1 \geq 0\\
&y(e_1)+y(e_6)+y(e_7)+y(e_4)-z_1 \geq 0\\
&y(e_5)+y(e_6)+y(e_7)+y(e_8)-z_2 \geq 0\\
&y(e_5)+y(e_2)+y(e_3)+y(e_8)-z_2 \geq 0\\
&D_1z_1+D_2z_2 \geq 1\\
&y(e_i) \geq 0\ \forall 1 \leq i \leq 8\\
&z_j \geq 0\ \forall 1 \leq j \leq k
\end{aligned} \tag{4.2.9}$$

This is the resulting LPP formulation for the dual of the example we considered in figure 4.1. The dual we have given in equation 4.2.3 is a generalization of this resulting formulation.

The Multi commodity flow problem is very well studied in combinatorics. Unlike single commodity flow, the structural properties of this problem are not well known when the number of commodities is greater than two($k > 2$). This

problem can be solved in polynomial time using linear programming. However, the problem of finding an integer flow is NP-Complete when $k \geq 2$.

Chapter 5

Conclusion

In this thesis, we reviewed the classical *Max-flow Min-cut* theorem and its proof using *Ford-Fulkerson* algorithm. We have also presented the *Konig's* theorem and *Menger's* theorem as consequences of the *Max-flow Min-cut* theorem. While the above proofs are very well established, they are proved in an algorithmic perspective. In the third chapter, we have presented the proofs for *Konig's* theorem and *Ma-flow Min-cut* theorem using a complete different technique based on the total unimodularity property of the coefficient matrix in their linear program formulation. Finally, we have briefly discussed about Multi-commodity flow and the Concurrent Multi-commodity Flow Problem(CMFP).

Although these results and the formulations are not new, an attempt was made to present complete proofs for the results from first principles, and the material does not seem to be consolidated and presented elsewhere in an easily accessible form.

Many more primal-dual relations exist in graph theory and the approach generalize to investigation into these relations and discovering LP based proofs for those Min-Max relations in graph theory. Hence, this approach is a general tool and the results presented here are just sample cases.

This study gives an insight into different techniques and tools that can be used to prove primal-dual relations in Graph theory. Though we have not established any new results, the treatment given in this thesis gives us a good scope of

applying these techniques in order to establish new results in both graph theory and combinatorics. We have also attempted to apply a technique called *Lagrangian relaxation* [7] from linear programming to some of these relations in order to gain some insights into its effectiveness. But, the work related that is not included in this thesis as it did not yield insightful results. A possibility is to apply different techniques in combination and try to investigate the outcome which could lead to interesting observations.

Bibliography

[1] Ron Aharoni and Eli Berger. Menger's theorem for infinite graphs. Inventiones Mathematicae, 176(1):1–62, 2009.

[2] A Chandra Babu, P.V. Ramakrishnan, and C.R. Seshan. New proofs of konig-egervary theorem. and maximal flow-minimal cut capacity. theorem using o.r. techniques., August 1990.

[3] Thomas Cormen. Introduction to algorithms. The MIT Press, Cambridge Mass., 2. ed. edition, 2001.

[4] Thomas Cormen. Introduction to algorithms. The MIT Press, Cambridge Mass., 2. ed. edition, 2001.

[5] L. R. Ford and D. R. Fulkerson. Maximal flow through a network. Canadian Journal of Mathematics, 8:399–404, January 1956.

[6] Anders Forsgren. Optimization online - an elementary proof of optimality conditions for linear programming. Technical Report TRITA-MAT-2008-OS6, Department of Mathematics, Royal Institute of Technology (KTH), SE-100 44 Stockholm, Sweden, June 2008.

[7] Arthur M. Geoffrion. Lagrangian relaxation for integer programming. In Michael Jünger, Thomas M. Liebling, Denis Naddef, George L. Nemhauser, William R. Pulleyblank, Gerhard Reinelt, Giovanni Rinaldi, and Laurence A. Wolsey, editors, 50 Years of Integer Programming 1958-2008, pages 243–281. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[8] Alon Itai. Two-Commodity flow. Journal of the ACM, 25(4):596–611, October 1978.

[9] Dénes Kőnig. Gráfok és alkalmazásuk a determinánsok és a halmazok elméletére. Matematikai és Természettudományi Értesítő, 34:104–119, 1916.

[10] Eugene Lawler and Eugene Lawler. Combinatorial optimization: Networks and matroids. In 4.5. Combinatorial Implications of Max-Flow Min-Cut Theorem, 4.6. Linear Programming Interpretation of Max-Flow Min-Cut Theorem, pages 117–120. Dover, 2001.

[11] Eugène L. Lawler. Combinatorial Optimization: Networks and Matroids. Courier Dover Publications, March 2001.

[12] Tom Leighton and Satish Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. J. ACM, 46(6):787–832, November 1999.

[13] László Lovász and M. D Plummer. Matching theory. North-Holland : Elsevier Science Publishers B.V. ; Sole distributors for the U.S.A. and Canada, Elsevier Science Pub. Co., Amsterdam; New York; New York, N.Y., 1986.

[14] Christos Papadimitriou. Combinatorial optimization : algorithms and complexity. Dover Publications, Mineola N.Y., 1998.

[15] Farhad Shahrokhi and D. W. Matula. The maximum concurrent flow problem. J. ACM, 37(2):318–334, April 1990.

[16] Douglas B. West. Introduction to Graph Theory. Prentice Hall, 2 edition, September 2000.