COLLEGE OF NATURAL AND COMPUTETIONAL SCIENCE DEPARTMENT OF MATHEMATICS

Project Work On Network Flow Problem and Its Application

Submitted in partial fulfillment of requirements for the Bachelor of Science in Mathematics

Prepared by: Samuel Yeda

Advisor: ***Gashaw Merga***

January, 2020

Wolkite, Ethiopia

Contents

WOLKITE UNIVERSITY

Department of Mathematics

The undersigned hereby certify that they have read and recommend to the department of Mathematics for acceptance of a project entitled by in partial fulfillment of the requirements for the Bachelor of Science in Mathematics.

Dated: January, 2021

Advisor: _____

Examiner: _____

Notation

V	the sets of vertices
E	the sets of edges
N	the sources of nodes
t	the sink
c	the capacity of edge

Acknowledgment

First, I would like to express my deepest gratitude to God for giving me patience; I am also grateful to my advisor Gashaw Merga for his helpful discussion, comments and providing the necessary materials in the preparation of this paper. Secondly I would like to extend my thank to my families and all who encouraged me to complete my project and their heart full help while writing the paper. Lastly I would like thanks to Department of mathematics, Wolkite University for giving the necessary materials throughout the preparation of this paper.

Abstract

This project deals about network flow problem. It contains three chapters. The first chapter preliminaries concept of Network Flow problems. The second chapter is deals about the solution method ,and the last chapter deals about network flow problems and its application. Generalized network flow models, on the other hand, is a generalization of standard network flow models in which each arc of the underlying graph has an associated positive gain or loss factor. In this chapter, we wish to introduce the reader to a variety of less apparent network models. Such models are drawn from application areas such as, path, algorithm and decision analysis as well as transportation and communication. Our objective is to illustrate how network modeling can be useful in formulating problems occurring in these diverse areas problems that at first blush seem quite remote from the standard applications of networks.

Introduction

Network flows is an exciting field that brings together what many students, practitioners, and researchers like best about the mathematical and computational sciences. Shortest path problems lie at the heart of network flows. They are alluring to bot researchers and to practitioners for several reasons: they arise frequently in practice since in a wide variety of application settings we wish to send some material (e.g. Maximum flow problem, transportation problem, even though shortest path problems are relatively easy to solve, the design and analysis of most efficient algorithms for solving them requires considerable ingenuity. Consequently, the study of shortest path problems is a natural starting point for introducing many key ideas from network flows, including the use of clever data structures and ideas such as data scaling to improve the worst case algorithmic performance. Therefore, in this and the next chapter, we begin our discussion of network flow algorithms by studying shortest path problems.

Chapter 1

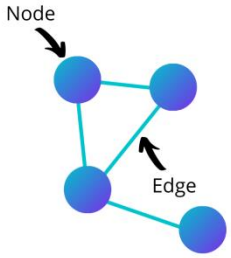
PRELIMINARIES

When one thinks about a network (communication, social, transportation, computer networks etc.), many fundamental questions naturally arise: how well-connected is it, how much data (commodity) can it transport, where are its bottlenecks, etc. In fact, many non-network and non-flow problems areal so frequently solved using network flow framework. Network flows have many practical applications but what makes them a must-know classical topic in algorithms is their elegant theory and beautiful algorithms. The theory itself dates back to 1950s (well before the internet or the web), when Ford and Fulkerson described an augmentation based on the method for finding shortest path problem and maximum flows in a capacitated network. The word flow has natural association with physical commodities such as traffic or data, but mathematically it is an abstract entity which originates at the source nodes .

1.1. Network Flow Problem

Definition 1.1.1. Network is a graph $G = (V, E)$, where V is a set of vertices and E is a set of V's edges a subset of V × V together with a non-negative function c: V × V $\rightarrow \mathbf{R}\infty$, called the capacity function. we may assume that if (u, v) ∈ E then (v, u) is also a member of E, since if $(v, u) \ni E$ then we may add (v, u) to E and then set $c(v, u) = 0$. If two nodes in G are distinguished, a source s and a sink t, then (G, c, s, t) is called a flow network.

*Examples the following figure*1.1



A path

Path is a sequence of edges connecting two specified nodes in a graph: Each edge must have exactly one node in common with its predecessor in the sequence. Edges must be passed in the forward direction.

Residuals

The residual capacity of an arc with respect to a pseudo-flow f, denoted cf, is the difference between the arc's capacity and its flow. That is, $cf\ (e) = c(e) - f(e)$. From this we can construct a residual network, denoted $Gf\ (V, Ef)$, which models the amount of available capacity on the set of arcs in $G = (V, E)$. More formally, given a flow network G,the residual network Gf has the node set V,

arc set $Ef = \{e \in V \times V : cf\ (e) > 0\}$ and capacity function cf.

This concept is used in Ford–Fulkerson algorithm which computes the maximum flow in a flow network. Sometimes, when modeling a network with more than one source, a super source is introduced to the graph. This consists of a vertex connected to each of the sources with edges of infinite capacity, so as to act as a global source. A similar construct for sinks is called a super sink.

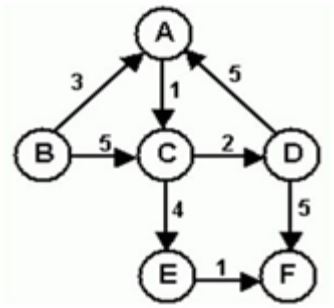
1.2.1 Shortest Path Problem

Definition 1.2.1 shortest path problem is the problem of finding a path between two vertices (or nodes) in a graph such that the sum of the weights of its constituent edges is minimized.

If all the edges in a graph have non -negative then it is possible to find the shortest path from any two vertices .

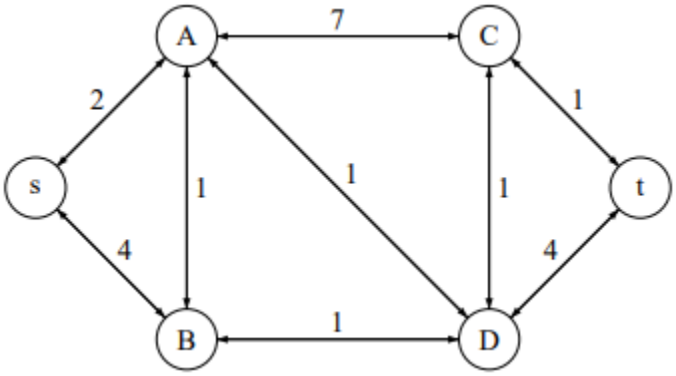
For example in the figure below ,the shortest path from B to F is $\{B, A, C, E, F\}$, with the total cost of nine .

This problem will be defined for a graph that contains non –negative weigths.$Example\ th\ following\ figure$ 1.2



1.3 Maximum Flow Problem

Definition **.1**.3 A flow network is a directed graph $G = (V, E)$ with a source $s \in V$, a sink t $\in V$, and capacities along each edge (described by a function $c: E \rightarrow R$ where c(e) is the capacity of edge e). The following flow network $G. Example\ the\ following\ figure$ 1.3



Chapter 2

SOLUTION METHOD

2.1 Shortest Path Method Dijkstra Algorithm

Dijkstra's algorithm starts by assigning some initial values for the distances from node s and to every other node in the network. It operates in steps, where at each step the algorithm improves the distance values. At each step, the shortest distance from node s to another node is determined

To determine by Dijkstra's algorithm after it has examined node 1; it also shows the corresponding radix heap. To select the node with the smallest distance label, we scan the buckets $0, 1, 2, \ldots, k$ to find the first nonempty bucket. In our example, bucket 0 is nonempty. Since bucket 0 has width 1, every node in this bucket has the same (minimum) distance label. So the algorithm designates node 3 as permanent, deletes node 3 from the radix heap, and scans the arc (3, 5) to change the distance label of node 5 from

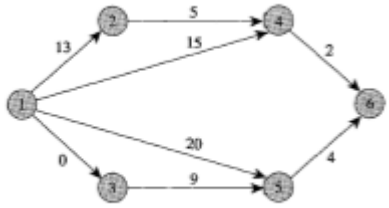


Figure2.1

Node $_i$	1	2	3	4	5	6
Label of $_j$	0	13	0	15	20	∞

Bucket k	0	1	2	3	4	5	6	7
Range k	0	1	[2,3]	[4,7]	[8,15]	[16,31]	[32,63]	[64,127]
Content k	(3)	0	0	0	{2,4}	{5}	0	

20 to 9. We check whether the new distance label of node 5 is contained in the range of its present bucket, which is bucket 5. It is not. Since its distance label has decreased, node 5

should move to a lower-indexed bucket. So we sequentially scan the buckets from right to left, starting at bucket 5, to identify the first bucket whose range contains the number 9, which is bucket 4. Node 5 moves from bucket 5 to bucket 4

Node $_i$	2	4	5	6
Label of $_j$	13	15	9	∞

Bucket k	0	1	2	3	4	5	6	7
Range k	[0]	[1]	[2,3]	[4,7]	[8,15]	[16,31]	[32,63]	[64,127]
Content k	0	0	0	0	[2,4,5]	0	0	0

We again look for the node with the smallest distance label. Scanning the buckets sequentially, we find that bucket k = 4 is the first nonempty bucket. Since the range of this bucket contains more than one integer, the first node in the bucket need not have the minimum distance label. Since the algorithm will never use the ranges range(0), ... , range $(k - 1)$ for storing temporary distance labels, we can redistribute the range of bucket k into the buckets $0, 1, \ldots, k - 1$, and reinsert its nodes into the lower-indexed buckets. In our example, the range of bucket 4 is $[8, 15]$, but the smallest distance label in this bucket is 9. We therefore redistribute the range [9, 15] over the lower-indexed buckets in the following manner:

$Range$ (0) = [9],

$Range$ (1)= [10],

$Range$ (2) = [11, 12],

$Range$ (3) = [13, 15],

$Range$ (4)=[0]

Other ranges do not change. The range of bucket 4 is now empty, and we must reassign the contents of bucket 4 to buckets ° through 3. We do so by successively selecting nodes in bucket 4, sequentially scanning the buckets$3, 2, 1$, and inserting the node in the appropriate bucket. The resulting buckets have the following contents:

$Content$ (1) = {5},

$Content$ (1) = 0,

$Content$ (2) = 0,

$Content$ (3) = {2, 4},

$Content$ (4) = 0

This redistribution necessarily empties bucket 4 and moves the node with the smallest distance label to bucket 0.

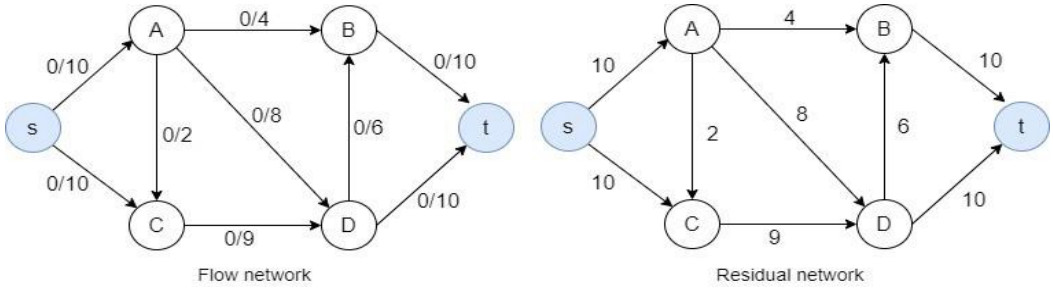
2.2. Solving the Maximum Flow Method, with Ford Fulkerson

However, modeling this problem is really beyond the scope of this article. But this will give you a comprehensive understanding of the Ford Fulkerson method so that you can use it anywhere applicable. But before that let's understand the maximum flow problem.

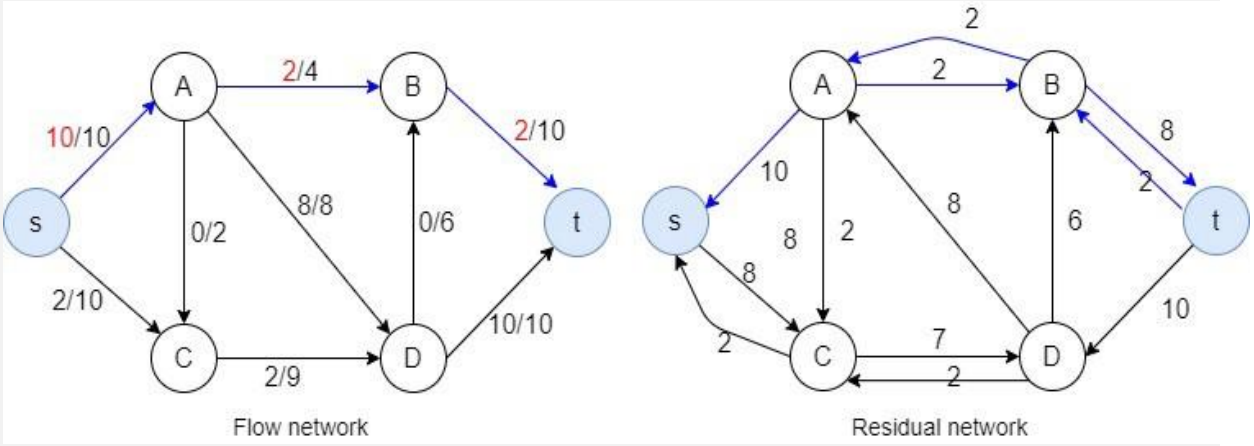
2.2.1. What is the maximum flow problem?

What is the greatest rate at which material can be sent (shipped) from source to the sink without violating any capacity constraints.

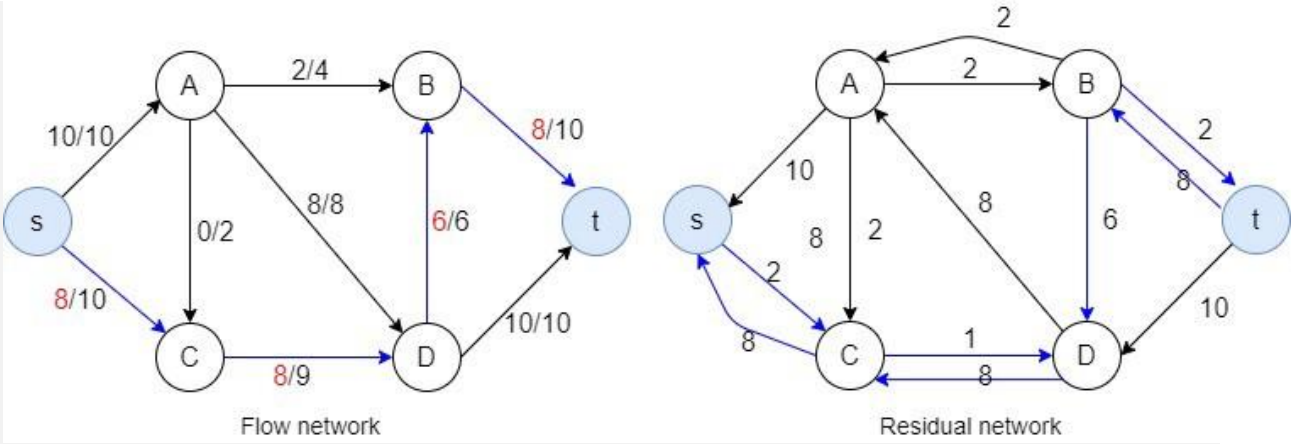
Given a flow network G with source s and sink t, the maximum flow problem is an optimization problem to node a flow of maximum value from s to t. Flow network G=(V, E), is essentially just a directed graphs. The Ford Fulkerson method, also known as 'augmenting path algorithm' is an effective approach to solve the maximum flow problem. The Ford Fulkerson method depends on two main concepts and they are, e same vertices as the original network with one or two edges for each edge in the original network. And it indicates the additional possible flow through the network. For each edge we'll calculate the additional flow as follows. So, to create a residual network we will take our original network and will update the capacity of each edge with the additional flow corresponding to that edge. Then we will also add reverse edges to indicate the amount of flow currently going across the edges of the original network. Given a flow network $G = (V, E)$, the augmenting path is a simple path from s to t in the corresponding residual graph of the flow network. Now we have the basic knowledge to understand the Ford Fulkerson method. Let's step through this method using the following:$Example2$.



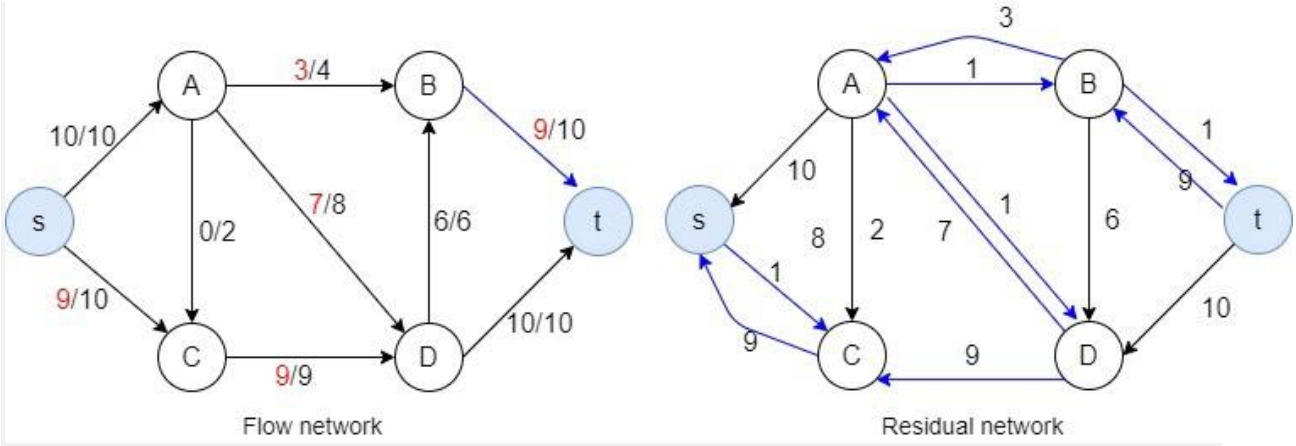
Step 1: Now, find an augmenting path in the residual network. Here, I select the path $\boldsymbol{s} -> \boldsymbol{A} -> \boldsymbol{D} -> \boldsymbol{t}$. Then we have to identify the bottleneck capacity (i.e. maximum flow for that path) for the selected path. As you can see, along this path, the bottleneck capacity is 8. Now without violating the capacity constraint, update the flow values of the edges in the augmenting path. Then you will get the following path $\boldsymbol{s} -> \boldsymbol{A} -> \boldsymbol{B} -> \boldsymbol{t}$, the bottleneck capacity is 2.



Step 4: augmenting path $\boldsymbol{s} -> \boldsymbol{C} -> \boldsymbol{D} -> \boldsymbol{B} -> \boldsymbol{t}$, the bottleneck capacity is 6.



Step 5: augmenting path $\boldsymbol{s} -> \boldsymbol{C} -> \boldsymbol{D} -> \boldsymbol{A} -> \boldsymbol{B} -> \boldsymbol{t}$, bottleneck capacity is 1.



Look! Now there are no paths left from the *s* to *t* in the residual graph. So, there is no possibility to add flow. This means the Ford Fulkerson method is complete and we are ready to find the maximum flow .The maximum flow method is $9 + 10 = 19$

Chapter 3

Application

In this chapter, we discuss briefly the most important applications of network flow problems.

3.1.1 Shortest path problem

Roughly speaking, the shortest-path problem is to find, well, the shortest path from one specific node to another in a network (N,A). In contrast to earlier usage, the arcs connecting successive nodes on a path must point in the direction of travel. Such paths are sometimes referred to as directed paths. To determine a shortest path, we assume that we are given the length of each arc. To be consistent with earlier notations, let us assume that the length of arc (i, j) is denoted by C_{IJ} . Naturally, we assume that these lengths are non-negative. To find the shortest path from one node (say, s) to another (say, r), we will see that it is necessary to compute the shortest path from many, perhaps all, other nodes to r. Hence, we define the shortest-path problem as the problem of finding the shortest path from every node in N to a specific node $r \in N$.

The destination node r is called the root node. shortest path problem is the problem of finding a path between two vertices (or nodes) in a graph such that the sum of the weights of its constituent edges is minimized.

3.1.2 The Maximum-Flow Problem

The subject of this section is the class of problems called maximum-flow problems. These problems form an important topic in the theory of network flows. There are very efficient algorithms for solving them, and they appear as sub problems in many algorithms for the general network flow problem. However, our aim is rather modest. We wish only to expose the reader to one important theorem in this subject, which is called the Max-Flow Min-Cut Theorem. Before we can state this theorem we need to set up the situation. Suppose that we are given a network (N,A), a distinguished node s ∈N called the source node, a distinguished node $t \in N$ called the sink node, and upper bounds on the arcs of the network uij, $(i,j) \in A$. For simplicity, we shall assume that the upper bounds are all finite (although this is not really necessary). The objective is to "push" as much flow from $s\ to\ t$ as possible. To solve this problem, we can convert it to an upper-bounded network flow problem as follows. First, let cij = 0 for all arcs $(i,j) \in A$, and let $bi = 0$ for every node $i \in N$. Then add one extra arc (t,s) connecting the sink node t back to the source node s, put a negative cost on this arc $(say, cts = -1)$, and let it have infinite capacity $uts = \infty$. Since the only non-zero cost is actually negative, it follows that we shall actually make a profit by letting more and more flow circulate through the network. But the upper bound on the arc capacities limits the amount of flow that it is possible to push through. $A\ cut, C,$ is a set of nodes that contains the source node but does not contain the sink node (see Figure 3.1) The capacity of is defined as

$$K(c) = \sum_{\substack{i \in c \\ j \ni c}} uij$$

Note that here and elsewhere in this section, the summations are over "original" arcs that satisfy the indicated set membership conditions. That is, they don't include the arc that we added connecting from t back to s. (If it did, the capacity of every cut would be infinite which is clearly not our intention). Flow balance tells us that the total flow along original arcs connecting the cut set C to its complement minus the total flow along original arcs that span these two sets in the opposite direction must equal the amount of flow on the artificial arc (t,s). that is

$$X_{ts} = \sum_{\substack{i \in c \\ j \ni c}} xij - \sum_{\substack{i \ni c \\ j \in c}} xij$$

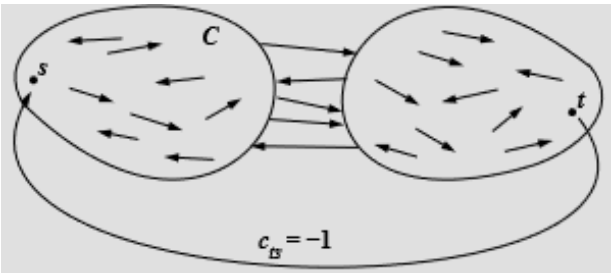


Figure 3.1
A cut set C for a maximum flow problem.

Summary

In this chapter we introduced the network flow problems that we study in this book and described few scenarios in which these problems arise. We began by giving a linear programming formulation of the minimum cost flow problem and identifying several special cases: the shortest path problem, the maximum flow problem, the assignment problem, the transportation problem, and the circulation problem. We next described several The Ford Fulkerson method (max-min flow)problem, the generalized network flow problem, and the multi commodity flow problem. Finally, we described the application of network flow . Although these two problems are not network flow problems per se, we have included them in this book because they are closely related to several network flow problems and because they arise often in the context of network optimization. Networks are pervasive and arise in numerous application settings. . Network flow problems also arise in surprising ways in optimization problems that on the surface might not appear to involve networks at all. The applications we have considered offer only a brief glimpse of the wide ranging practical importance of network flows; although our discussion of applications in this chapter is limited, it does provide at least one example of each of the network models that we have introduced in this chapter.

REFERENCECS

- Ahuja, R., Magnanti, T. & Orlin, J. Network Flows: Theory, Algorithms, and Applications, Prentice Hall, Englewood Cliffs, NJ. 240, 257 in (1993).
- Christofides, N. Graph Theory: An Algorithmic Approach, Academic Press, New York. 240 in (1975).