The Application of Network Flow

Khin Khin Chaw[1], Ni Ni Mar[2]

Abstract

In this paper, some basic definitions and the network models of the given problems are firstly introduced. Then the relation between the maximum flow, the size of the maximum matching and the maximum number of edge-disjoint paths in the given network are presented. Finally, the problem of air scheduling is solved by using the maximum flow.

Keywords network models, maximum flow, matching, edge-disjoint path

Introduction

The problem of finding the maximum flow capacity of networks has many applications in Modern Logistics management. Computer programs that can solve maximum flow programs are used by all automatic systems to calculate how shipments are transported by shipping companies (they may use planes, ships, trains, etc.). Another common use of maximum flow problem is in the optimization of design. This includes the design of piping systems for chemical and food-processing plants, water supply of a city, sewage system, etc. In this paper, the maximum flow in networks and its application has been studied. In section 1, some basic definitions and notations followed by [1] and [2] have been described. In section 2, the definitions of matching, maximum flow and edge-disjoint paths based on [1] have been introduced. Then, the theorems which deal with the relation between the maximum flow, the size of maximum matching and the number of edge disjoint paths have been presented. In section 3, we solve the airline scheduling in Rakhine State has been solved by using the edge-disjoint paths based on [3].

1. Basic Definitions and Notations

An **undirected graph** G consists of a set V of **vertices** and a set E of **edges** such that each edge $e \in E$ is associated with an unordered pair of vertices. A **directed graph** (or **digraph**) G consists of a set V of vertices and a set E of edges such that each edge $e \in E$ is associated with an ordered pair of vertices. In a directed graph, the directed edges are indicated by arrows. A graph with numbers on the edges is called a **weighted graph**. If the edge e is labeled k, the weight of the edge e is k.

---

1. Lecturer, Department of Mathematics, Taunggoke Degree College

2. Lecturer, Department of Mathematics, Taunggoke Degree College

The two edges both associated with the same pair of vertices are **parallel edges**. An edge incident on a single vertex is called a **loop**. A graph with neither loops nor parallel edges is a **simple graph**. A graph in which the vertices can be partitioned into disjoint sets V_1 and V_2 with every edge incident on one vertex in V_1 and one vertex in V_2 is called a **bipartite graph**. A **path** in a graph is a finite or infinite sequence of edges which connects a sequence of vertices which are all distinct from one another. A **simple path** is a path with no repeated vertices. A graph G is said to be **connected** if for any two vertices u and v of G, there is a path in G.

A **transport network** is a simple, weighted, directed graph G satisfying:

(a) There is exactly one vertex in G, called the **source**, having no incoming edges.

(b) There is exactly one vertex in G, called the **sink**, having no outgoing edges.

(c) The weight C_{ij} of the directed edge (i, j), called the **capacity** of (i, j), is a nonnegative number.

(d) The undirected graph obtained from G by ignoring the directions of the edges is connected.

The graph of Fig.1 is a transport network. The source is vertex "a" and the sink is vertex "z". The capacity of edge (a, b), C_{ab} is 3 and the capacity of edge (b, c), C_{bc}, is 2.

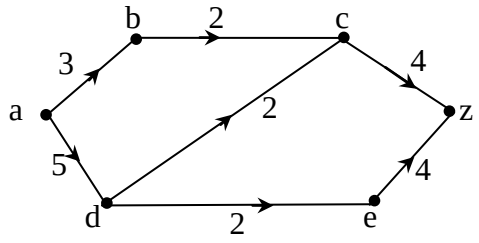

Fig. 1

Let G be a transport network. Let C_{ij} denote the capacity of the directed edge (i, j), a nonnegative number F_{ij} such that $F_{ij} \le C_{ij}$, and for each vertex j, which is neither the source nor the sink, $\sum_i F_{ij} = \sum_i F_{ji}$. We call F_{ij} the flow in edge (i, j). For any vertex j, we call $\sum_i F_{ij}$ the flow into j and $\sum_i F_{ji}$ the flow out of "j". In Fig. 2, we redraw the network of Fig. 1 to show the flow. The flow into vertex "d", $F_{ad} = 3$, and the flow out of vertex "d", $F_{dc} + F_{de} = 1 + 2 = 3$. So it can be seen that the flow in of a vertex equals the flow out of it.

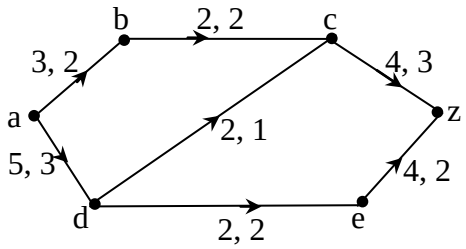

Fig. 2

Now we construct a network model of the airlines for four towns in Rakhine State. We consider the following flights. The edges represent the flights and u_i is the origin and v_i is the destination of the flight "i ", respectively.

u_1: Thandwe (depart 7:45 A.M.) – v_1: Ann (arrive 8:15 A.M.)
u_2: Thandwe (depart 9 A.M.) – v_2: Munaung (arrive 9:45 A.M.)
u_3: Ann (depart 8:30 A.M.) – v_3: Kyuakphyu (arrive 8:45 A.M.)
u_4: Munaung (depart 10:00A.M.) – v_4: Kyuakphyu (arrive 10:15 A.M.)

The graph of the given flights is represented as

G

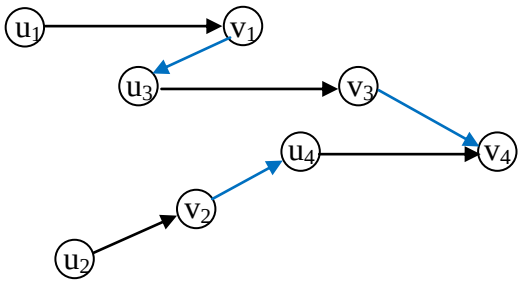

Fig. 3

In Fig. 3, the graph G is not a transport network since there are multiple sources u_i and multiple sinks v_i for i = 1,2,3,4. Therefore we produce an equivalent transport network H by tying the sources into a supersource "s" and tying together the sinks into a supersink "t".

H

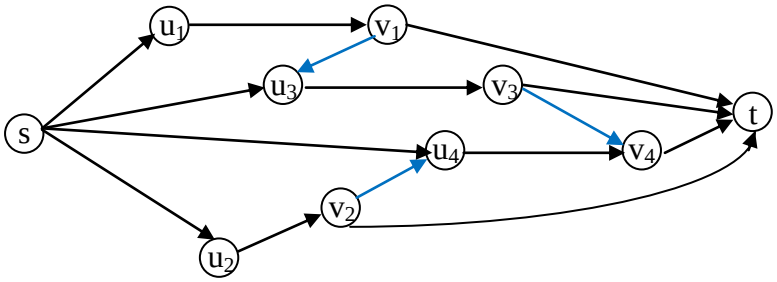

Fig. 4

2. Maximum Flow and Edge-Disjoint Paths

2.1 Definition. Let G be a directed bipartite graph with disjoint vertex sets V and W in which the edges are directed from vertices in V to vertices in W. A **bipartite matching** (simply **matching**) for G is the set of edges E with no vertices in common. **The size of a matching** for G is the maximum number of edges in E containing in that matching.

2.2 Definition. Let F be a flow in a transport network H. Then a **maximal flow** in H is a flow with a maximum value.

In general, there will be several flows having the same maximum value. We can find the maximum flow by using the algorithms (such as Ford-Fulkerson algorithm, Edmonds-Karp algorithm, Dinic's algorithm etc.). The following theorem is the relation between the maximum flow and matching.

2.3 Theorem. [1] Let G be a directed bipartite graph with disjoint vertex sets V and W in which the edges are directed from vertices in V to vertices in W. A flow in the matching network gives a matching in G. Then the vertex $v \in V$ is matched with the vertex $w \in W$ if and only if the flow in edge (v,w) is 1 and the maximum flow equals to the size of the maximum matching.

Proof. Let a(z) represent the source (sink) in the matching network and suppose that a flow is given. Suppose that the edge $(v,w), v \in V$, $w \in W$, has flow 1. The only edge into the vertex v is (a,v). This edge must have flow 1, thus the flow into the vertex v is 1. Since the flow out of v is also 1, the only edge of the form (v,x) having flow 1 is (v,w). Similarly, the only edge of the form (x,w) having flow 1 is (v,w). Therefore, if E is the set of edges of the form (v,w) having flow 1, the members of E have no vertices in common, thus E is a matching for G. So that the number of vertices in V matched is equal to the value of the corresponding flow.

2.4 Definition. Let H be a transport network. Two paths are said to be **edge-disjoint** in H if they have no edge in common.

2.5 Definition. A flow F in a transport network H is called **0/1-flow** if every edge has either no flow on it, or one unit of flow.

2.6 Lemma.[3] Let F be a 0/1-flow in a transport network H with flow value k. Then there are k edge-disjoint paths between "s" and "t" in H.

Proof. By induction on the number of edged in H that has one unit of flow assigned to them by F. If k = 0 then there is nothing to prove.

Otherwise, start traversing the graph H from "s" travelling only along edges with flow 1 assigned to them by F. We mark such an edge as used, and do not allow one to travel on such an edge again. There are two possibilities:

(i) We reached the target vertex "t". In this case, we take this path, add it to the set of output paths, and reduce the flow along the edge of the generated path π to 0. Let H' be the resulting flow network and F' be the resulting flow. We have $|F'| = k-1$, H' has less edges, and by induction, it has $k-1$ edge-disjoint paths in H' between "s" and "t". Together with π this forms k such paths.

(ii) We visit a vertex v for the second time. In this case, our traversal contains a cycle C, of edges in H that have the flow 1 on them. We set the flow along the edges of C to 0 and use induction on the remaining graph (since it has less edge with flow 1 on them). The value of the flow F did not change by removing C, and as such it follows by induction that there are k edge-disjoint paths between "s" and "t" in H.

2.7 Theorem.[3] Let G be a directed, bipartite graph with each edge has a capacity 1. Then the maximum number of edge-disjoint paths is equal to the maximum flow in G.

Proof. Suppose there are k edge-disjoint paths P_1, P_2, …, P_k. Set $F_e = 1$ if e participates in some path P_i; otherwise, set $F_e = 0$. Since paths are edge-disjoint, F is a flow of value k. Conversely, suppose that the maximum flow value is k. By integrality theorem, there exists $\{0,1\}$ flow F of value k. Consider the edge (s, v) with $F_{sv} = 1$. By conservation, there exists an edge (v, w) with $F_{vw} = 1$. We continue until reach t, always choosing a new edge. Then we produce k edge-disjoint paths.

The above theorem shows the relation between the maximum flow and edge-disjoint paths.

3. Application of Maximum Flow

Now we solve the problem of airline scheduling if we are given the flights that the airline needs to serve. We can find the minimum number of airplanes needed to carry out this schedule. For example, see Fig. 3. We can use the same airplane for two segments "i" and "j "if

the destination of "i "is the origin of the segment "j" and there is enough time in between the two flights for the required maintenance. The edge- disjoint paths of the network H in Fig. 4 can be seen as below.

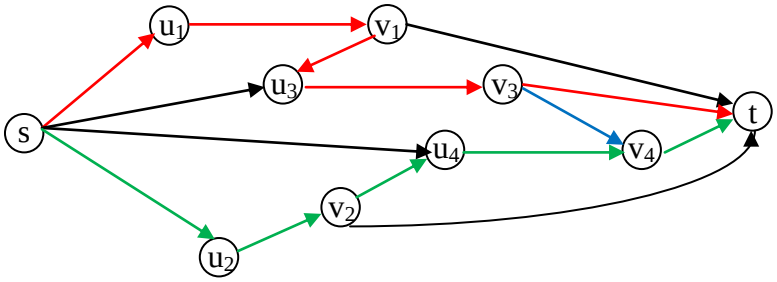

Fig. 5

If we denote the supersource "s" as Yangon and the supersink "t" as Sittwe, then the new schedule can be extended as follow:

1. Yangon (depart 7:00 A.M.) - Thandwe (arrive 7:30 A.M.)
2. Thandwe (depart 7:45 A.M.) - Ann (arrive 8:15 A.M.)
3. Ann (depart 8:30 A.M.) – Kyuakphyu (arrive 8:45 A.M.)
4. Kyuakphyu (depart 9:00 A.M.) - Sittwe (arrive 9:15 A.M.)
5. Yangon (depart 9:00 A.M.) - Thandwe (arrive 9:30 A.M.)
6. Thandwe(depart 9:45 A.M.) - Munaung (arrive 10:15 A.M.)
7. Munaung (depart 10:30A.M.) - Kyuakphyu (arrive 10:45 A.M.)
8. Kyuakphyu (depart 11:0 0A.M.) - Sittwe (arrive 11:15 A.M.)

Now Fig. 5 represents the transport network of the new schedule. Since we have two edge-disjoint paths from "s" to "t", the number of airplanes needed to serve for the new schedule is 2. So we use only two airplanes for the flights from Yangon airport to the airports in Rakhine State. In the following map, the flights of the required schedules can be seen.

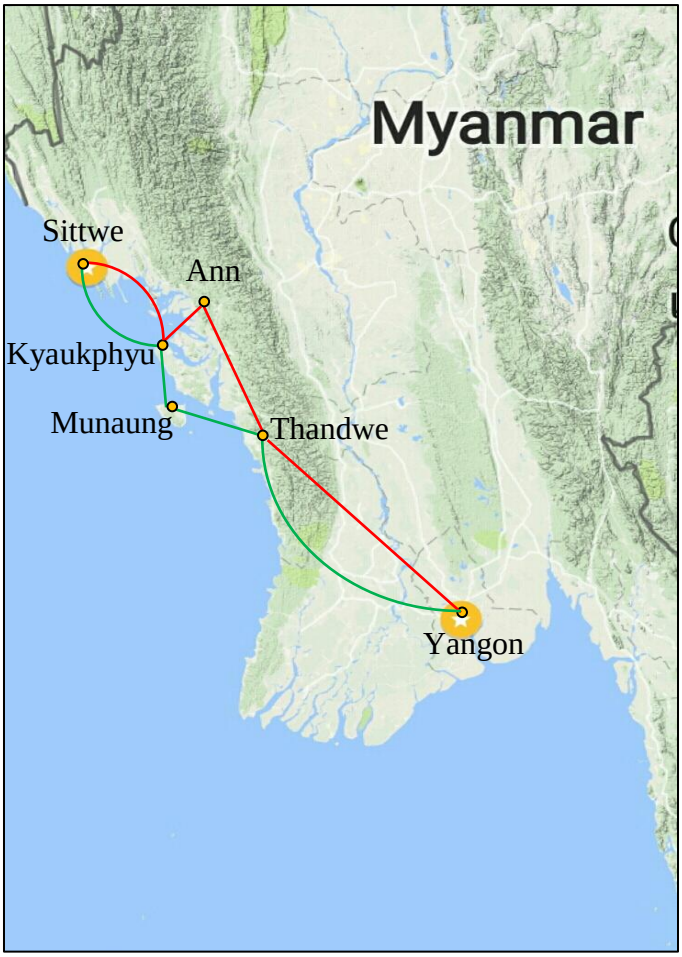

Fig. 6

Conclusion

In this research paper, the problem can be solved if certain information about flights are given. It is sure that an airline needs to provide and generate a profitable schedule.

Acknowledgement

We would like to express our sincere thanks to Dr. Myint Swe, Principal of Taunggoke Degree College for his permission to write this research paper. We also wish to record our deep gratitude to Dr. Khin Sandar Win, Professor and Head of Department of Mathematics, Taunggoke Degree College for her encouragement and invaluable suggestions.

References

[1] Richard Johnsonbaugh, "*Discrete Mathematics*", Macmillan Publishing Company, 1990.

[2] Jerrold W.Grossman, "*An Introduction to Concepts, Methods and Applications*", Macmillan Publishing Company, 1990.

[3] CS 473: Fundamental Algorithms, Spring 2011, "*More Network Flow Applications*", https://courses.engr.illinois.edu.