FLIGHT TRANSPORTATION LABORATORY
REPORT R68-5

SOME FLEET ROUTING AND SCHEDULING PROBLEMS FOR AIR TRANSPORTATION SYSTEMS

BY: AMOS LEVIN

DEPARTMENT OF
AERONAUTICS AND ASTRONAUTICS
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
CAMBRIDGE, MASSACHUSETTS 02139-4307

ABSTRACT

The purpose of this work is to formulate and develop practicable solution methods to some important fleet routing, scheduling and fleet composition problems. These problems arise in the operation of air transportation systems like the operating domestic and international airlines.

The problem of minimal fleet size to meet a variable schedule, which will be fixed when the system goes into operation, is formulated in several ways as Integer Linear Programs in 0-1 variables. The ILP's obtained are large scale programs and solved here by Land and Doig type Branch and Bound algorithms. The computational experiments with them, which were conducted with MPS/360, have been very sucessful and in the majority of cases, particularly when larger systems are solved, the algorithms terminated at the optimal integer solutions after a single iteration. The problem of scheduling and routing the minimal fleet is then formulated as an ILP which has exhibited equally successful computational results.

The minimal single fleet problem is extended to include some extraneous constraints on service frequencies between and at stations. Computational results with examples are provided. The problem of system design with and without a given fleet size is formulated.

The problem of decomposition of the system into subsystems, each consisting of a single vehicle type is next formulated in several ways for several considerations. These formulations are also given as Integer Linear Programs. The first is proven to have at least one optimal integer solution. Computational experience with the application of the Land and Doig Branch and Bound algorithm to some of the other multi-fleet problems is also given.

A computerized Airline Management Decision System which will use the models and solution methods developed in this work is briefly described in Appendix A. The Crew Scheduling Problem is also briefly discussed in this appendix since its solution procedures must be a part of such a Decision System.

CONTENTS

		Page
CHAPTER I.	INTRODUCTION	
1.1	General Discussion	1
1.2	Overview of the Chapters	3
CHAPTER II.	THE SINGLE FLEET PROBLEM	
2.1	Introduction	5
2.2	Definitions	6
2.3	Chain Decomposition of Acyclic Graphs	7
2.4	Minimizing the Fleet Size to Meet a Fixed Schedule	10
2.5	An 'Improved' Fixed Schedule Solution Generation	15
2.6	Minimizing the Fleet Size for a Variable Schedule	16
2.7	A Branch and Bound Algorithm for ILP(2.1)	20
2.8	A Cut and Branch Algorithm for ILP(2.1)	32
2.9	Alternative Formulation	36
2,10	A Branch and Bound Algorithm for ILP(2.2)	42
2.11	Minimal Flow Formulation; Comparisons	45
2.12	Arc-Chain Minimal Flow Formulation	49
2.13	Scheduling with Departure Preferences	53
2.14	Scheduling with Direct Fleet Costs	60
2.15	Concluding Remarks	63
CHAPTER III.	THE SINGLE FLEET: EXTENSIONS	
3.1	Introduction	64
3.2	Restricted Service Frequencies	64
3.3	System Design	75

3.4 Concluding Remarks 81

CHAPTER IV. THE MULTI-FLEET PROBLEM

4.1 Introduction 82

4.2 Preliminary Discussion and Definitions 83

4.3 Decomposition of the Air Transportation System 87

4.4 System Decomposition with Total Fleet Size Minimization 94

4.5 System Decomposition with Direct Fleet Costs 101

4.6 A Branch and Bound Algorithm for ILP(4.4) 104

4.7 Arc-Chain Formulation 108

4.8 Concluding Remarks 110

CHAPTER V. CONCLUSIONS

5.1 General Remarks 111

5.2 Further Research 112

APPENDIX A. AN OUTLINE OF A COMPUTERIZED AIRLINE MANAGEMENT DECISION SYSTEM (AMDS)

A.1 Introduction 114

A.2 The Crew Scheduling Problem 115

A.3 The Structure of AMDS 120

REFERENCES

CHAPTER I

INTRODUCTION

1.1 General Discussion

The topics which will be treated in this work are concerned with very important operational problems confronting the management of the competitive air transportation system. Some of these problems are significant for transportation systems which have public utility rather than profit orientation. The problems we shall address ourselves to are those connected with fleet utilization, fleet size, routing, fleet composition and scheduling. Work which is related to some of these problems has been carried out by Operations Researchers during the last decade with only partial success. The success has been limited because of two main reasons: 1. When optimal solutions were sought computation times were excessive. 2. When a computationally acceptable method was suggested it usually did not guarantee optimal solutions. In some investigations a combination of both drawbacks is evident. The principal method of attack has been Dynamic Programming, which usually results in a huge state space for any real-life transportation system [3]. A survey of all the significant work done in this area is provided in [18] and will not be given here. Specific references to work which is related to ours will be given in Chapter II.

The purpose of the following chapters is to describe a method of attack which has not, apparently, been extensively

investigated thus far. The basic models are of the discrete-time rather than of the continuous-time variety. This property permits the formulation of the optimization problems as Integer Linear Programs. Our aim is to discuss a realistic model and to obtain the optimal solution to it and consequently, no heuristics or approximations are attempted here. Rather we concentrate on obtaining the optimal integer solutions for the problems and this is what makes this work unique among all the reports which have been published thus far.

As the reader will realize the Integer Linear Programs which will be discussed in the substantive chapters are large for any existing air transportation system. However, the solution methods developed here and the extensive computational evidence gathered by the author in course of this research work lead to the rather gratifying conclusion that the problems with obtaining the optimal integer solutions to these large problems are manageable. The methods which were selected to solve our problems are Branch and Bound methods of the Land and Doig variety [10] . This approach is the only one feasible to us because of the following reasons: 1. Very efficient Linear Programming codes which can handle large enough Programs to accommodate our models have recently become available. (The computational system used throughout this investigation was MPS/360). The Land and Doig approach is based on the repeated solution of Linear Programs. 2. The Branch and Bound methods of the Implicit Enumeration variety (see for example [9]) have performed quite poorly, from computation time point of view, for large problems. They definitely do not have a chance

with problems of the size that we are dealing with.

1.2 Overview of the Chapters

In Chapter II the basic model is discussed. It is concerned with the problem of minimizing the fleet size to meet a schedule subject to some latitude in departure times. Assuming now that management can predict the passenger demand at each departure, the problem of the best routing of the minimal fleet is formulated and solved. Various formulations are presented, the last of which is provided as a suggestion, since no computer code currently exists.

In Chapter III the basic model is extended to include some constraints which may result from legal and other requirements. Also the problem of best utilization of available fleet is discussed. These extensions which are easily made emphasize the advantage in the formulation of the problems as Integer Linear Programs. The results of computational experiments are stressed.

Chapter IV contains extensions into the multi-fleet case. The transportation system will now include different types of aircraft with distinctive properties and the problem of optimal utilization and proper mix of the different types is discussed.

Chapter V is the concluding chapter.

In Appendix A the concept of Integrated Airline Management Decision System is introduced. This system (as yet unimplemented) is computerized and based on MPS/360 [11] . The

methods of Chapter II to Chapter IV are envisaged as the basic computational algorithms in this system and consequently practicability of the methods and precise definitions of the optimization algorithms are stressed throughout this work. The appendix also includes a sketchy discussion of the Crew Scheduling Problem since the author considers solution procedures to it to be an integral part of the Decision System.

CHAPTER II

THE SINGLE FLEET PROBLEM

2.1 Introduction

In this chapter the basic models will be formulated. The primary objective is to minimize the fleet size for fixed and variable schedules of the air transportation system. By variable schedule we mean a schedule that will eventually be fixed in some optimal manner before the system goes into operation. In this chapter it is assumed that only a single type of aircraft is available to the system. The fixed schedule problem is treated as an optimal flow in network problem. The extension for the variable schedule problem leads to Integer Linear Programs for which the solution tools will be Branch and Bound methods and 'Cut and Branch' algorithms which will be explained and formalized in section 2.8.

Sections 2.2 and 2.3 contain some preliminary discussions, definition and a central theorem, which provide the necessary concepts and results for all the following developments. Section 2.4 presents the fixed schedule minimal fleet problem. The first discussion of this problem first appears in Operations Research literature in 1954 [4] where the problem is formulated as a Transportation Problem. The reader may find the formulation of Section 2.4 somewhat simpler. In Section 2.5 a method for the generation of an improved schedule, which is suitable for computerized scheduling and aircraft assignments, is discussed. The variable schedule version is presented in Section 2.6. This problem was treated before in [15] where

a heuristic method which, although efficient, does not guarantee an optimal solution is suggested. More recently some more work has been done and is discussed in [13] . This is a Branch and Bound method which seems to be not particularly efficient and is capable of solving only two-station problems (although it could conceivably be generalized). The method presented here, however, is completely general (any number of stations) and an optimal solution is guaranteed when the algorithms terminate. The rest of this chapter contains discussion of various formulations, each having its own advantages and disadvantages, and some extensions of the basic optimization problem. These extensions have not appeared in literature (as far as the author knows) until now.

2.2 Definitions

A directed graph $G = [N;A]$ is defined to be a collection N, of elements with a subset, A, of NXN, the set of ordered pairs (x_i,x_j) where $x_i, x_j \in N$. The members of N, which is not necessarily finite, are called nodes and the members of A are called arcs. The pictorial representation of G will have a distinct point for each element $x_i \in N$ and an arrow pointing from x_i to x_j if and only if $(x_i,x_j) \in A$.

Let $x_1,x_2,\ldots,x_n$ $(n \geq 2)$ be a sequence of distinct members of N such that $(x_i, x_{i+1}) \in A$ for all $i = 1,2,\ldots,n-1$. Then the sequence of nodes and arcs $x_1, (x_1,x_2), \ldots, (x_{n-1}, x_n), x_n$ is defined to be a chain (or a directed chain). For convenience and consistency in the following discussion a single node $x_i \in N$ will also be defined to be a chain. If it is further stipulated that $x_1 = x_n$ $(n \geq 2)$ then the sequence

is referred to as a cycle (or a directed cycle).

A directed graph which contains no directed cycles is said to be an acyclic directed graph. We shall be concerned with acyclic graphs exclusively.

A graph whose set of nodes, N is divided into two subsets S and T such that $S \bigcup T = N$ and $S \bigcap T = \Phi$, with all arcs of A leading from the nodes of S to the nodes of T is called a bipartite graph. In order to emphasize that a graph G is bipartite we denote it by $G = [S,T;A]$.

When various functions (capacity, flow, etc.) which map the members of A to the real axis are defined, we refer to the graph as a network. It is assumed that the reader is familiar with the various algorithms of network flow theory.

2.3 Chain Decomposition of Acyclic Graphs

A decomposition of an acyclic graph into chains is a partition of N such that the nodes of each part (with their connecting arcs) form a single chain. From the definition of a chain it is clear that the graph can always be decomposed into $|N|$ single node chains. An acyclic graph which is decomposed into five chains is shown in Fig. 2.1, where the chains are indicated by the heavy lines.

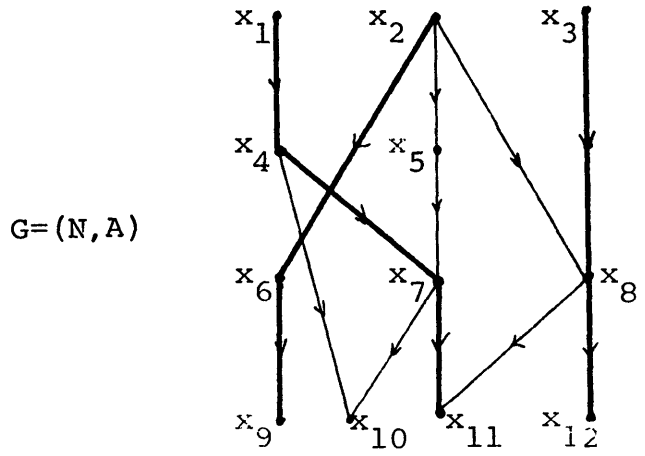


Fig. 2.1

There are two single node chains in this particular decomposition: x_5 and x_{10}.

We are interested in finding the minimal number of chains into which a given acyclic graph can be decomposed, since the path that an aircraft follows on a schedule map (Fig. 2.4) is a chain.

Corresponding to any given acyclic graph $G = [N;A]$ we construct a bipartite graph $G^* = [S,T;A^*]$ where $S = \{s_i\}$, $T = \{t_j\}$, $|S| = |T| = |N|$ and both indices i,j run from 1 to $|N|$. A^* is constructed from G as follows: If $(x_i,x_j) \in A$ there will be an arc (s_i,t_j) in G^*. We get the bipartite network G^* by defining capacity values of 1 on all arcs $(s_i,t_j) \in A^*$.

The following correspondence will be defined between chain decomposition of G and flow from S to T in G^*: If the arc (x_i,x_j) of G is a part of a chain, there will be a flow of one unit from s_i to t_j in G^*. If (x_i,x_j) is not a part of a chain, there will be a flow of value zero in (s_i,t_j). Conversely, (x_i,x_j) will be made a part of a chain if and only if there is a flow on one unit in (s_i,t_j). From the definition of a chain and the selection of capacity values, it is clear that the correspondence is one to one, this because a node in G belongs to one and only one chain. (If x_k is a single node chain there will be positive flow neither out of s_k nor into t_k in G^*.) In Fig. 2.2 the bipartite network and the flows (denoted by the heavy lines) corresponding to the chain decomposed graph of Fig. 2.1 is shown.

The consequences of the following theorem will provide us with a method for finding the minimal number of chains

into which the acyclic graph G can be decomposed.

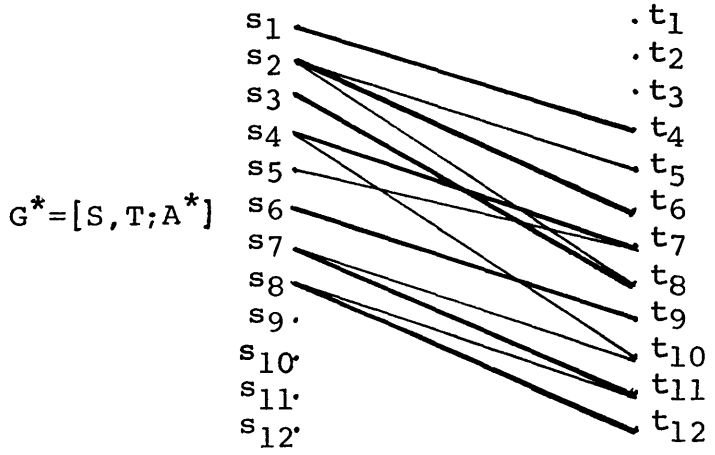


Fig. 2.2

For a chain decomposed acyclic graph, C will be defined to be the set of chains and P the set of arcs which are parts of the chains of C.

Theorem 2.1[6] Let G = [N;A] be a chain decomposed acyclic graph.

Then $|C| + |P| = |N|$.

Proof: The chains will be indexed by k, k=1, ..., $|C|$. The number of nodes which belong to the k-th chain will be denoted by n_k. Since each node belongs to exactly one chain we have:

$$\sum_{k=1}^{|C|} n_k = |N|$$

A chain, by its definition, has one less arcs than nodes. Hence:

$$\sum_{k=1}^{|C|} n_k = \sum_{k=1}^{|C|} (n_k - 1) + |C| = |P| + |C|$$

From which:

$$|N| = |P| + |C| \qquad \text{Q.E.D.}$$

From the indentity stated in Theorem 2.1 the minimization of $|C|$ can be achieved by maximizing $|P|$. But $|P|$ is the value of the flow from S to T in the network G*, a fact which is evident from the definition of G*.

From the discussion above it is clear that the problem of determining the minimal number of chains into which G can be decomposed can be solved by finding the maximal flow in G*.

2.4 Minimizing the Fleet Size to Meet a Fixed Schedule

The results and usefulness of Theorem 2.1 will now be demonstrated in solving the fixed schedule version of our fleet size problem.

An airline is serving the stations A,B and C. There is no direct service between A and C (Fig. 2.3).

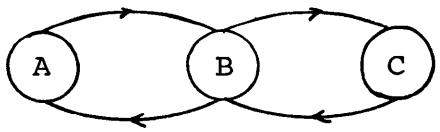
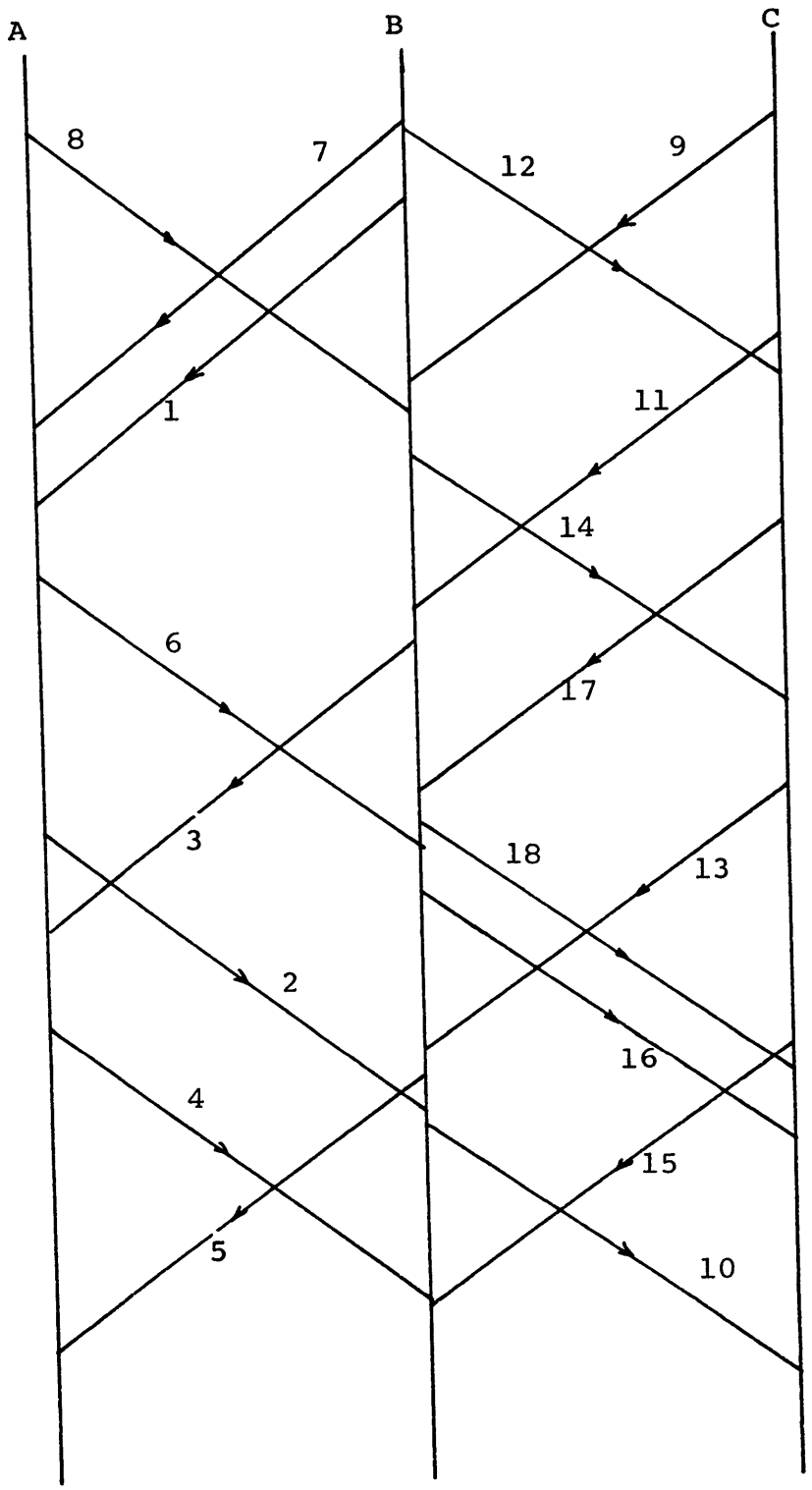


Fig. 2.3

Given the block times between any two stations and fixed required departure times for services, what is the minimal number of airplanes necessary to meet the schedule?

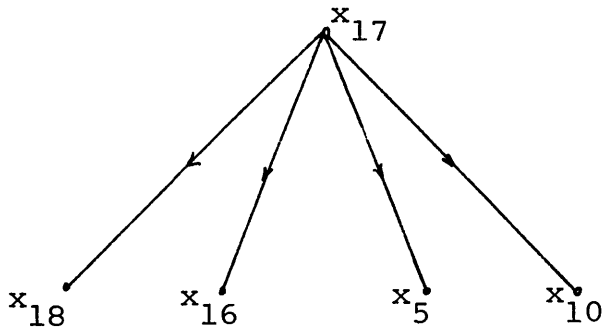
It is convenient to show the schedule on a time expansion of the graph of Fig. 2.3 obtaining the graph shown in Fig. 2.4, which will be referred to as the 'schedule map'.



A Schedule Map

Fig. 2.4

The resulting map is as large as the planning horizon. However, if the schedule is periodic, one period will suffice. Domestic airlines normally have daily periodicity while the international carriers a weekly one. A scheduled service from station X to station Y is shown by a directed arc from X to Y with the origin at the departure time at X and the end at the arrival time in Y. The service number is written alongside the arc. For example, flight number 7 is the earliest service from B to A. An aircraft arriving at any station, X, can connect to any of the departures scheduled out of X after its arrival there. It may be assumed that any necessary delay due to minimal required servicing time of the aircraft before it is ready for the next takeoff is included in the blocktime of its last arrival. For example, an aircraft assigned to flight number 17 (Fig. 2.4) can be subsequently assigned to any one of the flights 18, 16, 5, 10 departing from B later in the day. These possible assignments are shown in the directed graph of Fig. 2.5, where node x_i represents flight number i and the arc (x_i,x_j) a possible sequence of two flights.



Flight Connections

Fig. 2.5

The graph G which shows these relations for the entire schedule map is acyclic (since time progresses in an acyclic fashion). A chain in G describes the services that a particular aircraft will be assigned to. Therefore the problem of minimizing the fleet size to meet the schedule can be stated as a problem of minimal chain decomposition of the graph G.

We are assuming that an aircraft arriving in X will either connect to a departure out of X in the same period or wait in X until the next one; i.e. dead heading is not permitted. (If this assumption is dropped the graph G will have arcs from nodes corresponding to flights arriving at one station into flights departing from another station. And if such an arc is included in a chain, dead heading may take place).

As shown above, the problem can be solved by constructing the bipartite network G*, which can be done directly from the schedule map, and computing the maximal flow in it. For max-flow computations we adjoin to G* a 'master source' s, and a 'master sink' t, together with arcs $(s,s_i)\ \forall i$ and $(t_j,t)\ \forall j$. The capacity values of all these arcs will be set to 1. We shall refer also to the enlarged network as G*. Fig. 2.6 is the network G* constructed from the schedule of Fig. 2.4. The heavy lines indicate unit flows. The flow from s to t is maximal and has a value of 11. Since there are 18 flights in the schedule, by Theorem 2.1 the minimal fleet size required is 7. The assignment of aircraft to flight numbers can be read off directly from G*. For example, the same aircraft will take both flight number 13

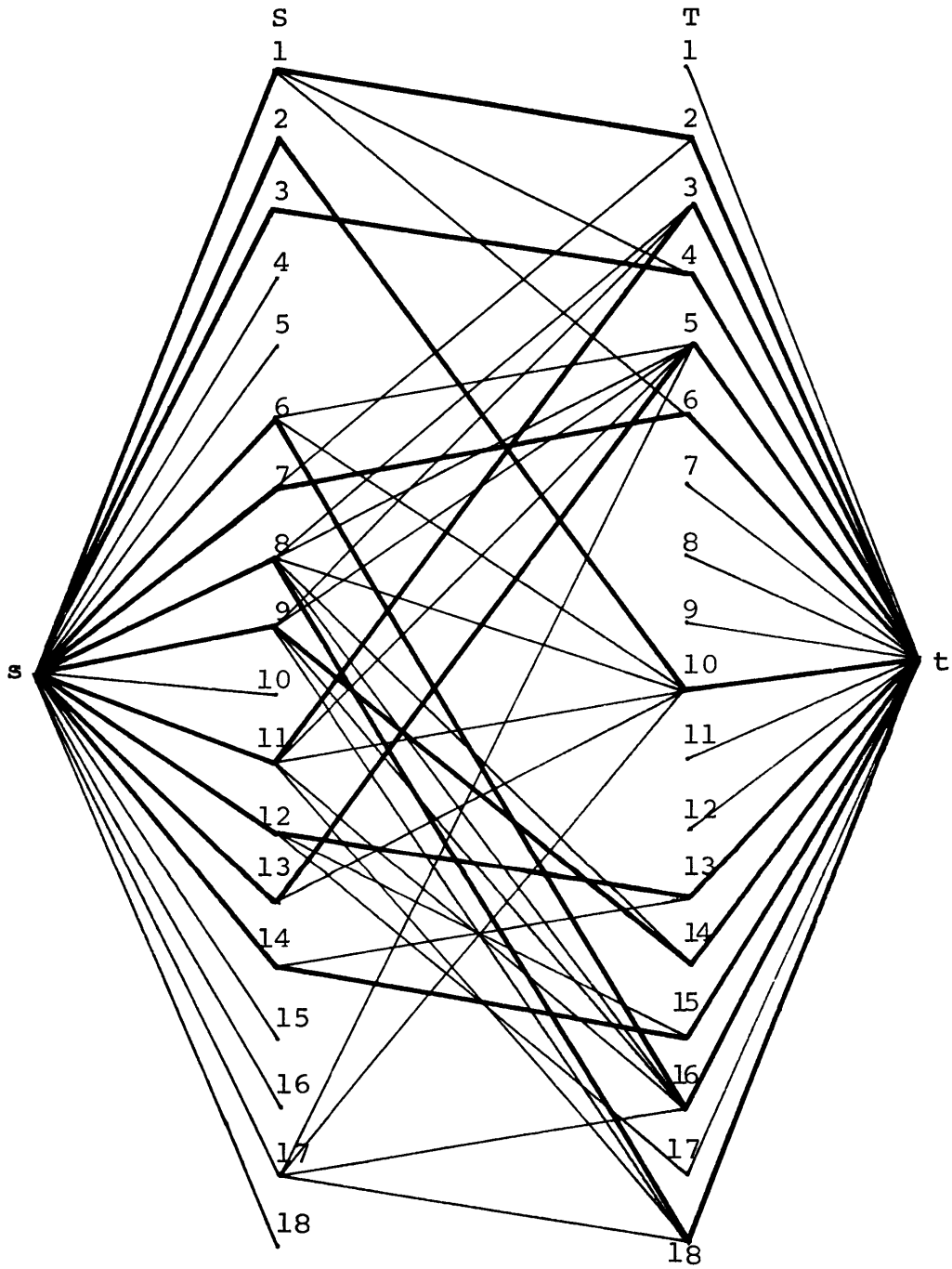


Fig. 2.6

and flight number 5, since there is a unit of flow in (s_{13},t_5). Whereas the maximal flow solution is not in general unique, many combinations of assignments are possible, but so far in our discussion any one is acceptable.

2.5 An 'Improved' Fixed Schedule Solution Generation

It was mentioned in the last section that many minimal chain decompositions of the schedule map are possible. However, it is generally accepted that short assignment times for the aircraft are less desirable than larger ones since there may be an undue pressure on the servicing ground personnel during some periods and low pressure during some others. A more even distribution will be obtained by making the connections as long in time as possible. If it is desired to have the method of the last section generate the chains to be actually used by the transportation system some additional considerations will have to be used.

With each possible connection (x_1,x_j) in G a 'disutility' index, c_{ij}, will be associated. The value of c_{ij} can be set to $T-t_{ij}$ where T is the periodicity of the schedule and t_{ij} is the time between the arrival point of flight number i and the departure time of flight number j. In G* the flow in arc (s_i,t_j) will have a cost of c_{ij}. The max-flow which minimizes the cost over all the max-flows will accomplish the two objectives of minimizing of the size of the fleet and generating 'good' chains in the sense discussed in the last paragraph.

Other considerations may affect the selection of the c_{ij}'s. If the chains obtained are not satisfactory, a larger fleet size may be considered as a possibility. If

the flow in G* is one unit less than the maximal, one more airplane is needed, for the number of chains is one larger than the minimal one. However, they may be sufficiently better, from management's point of view, to justify the use of a larger fleet. Minimal cost of flow of any value can be examined, and final decisions be made using these results.

2.6 Minimizing the Fleet Size for a Variable Schedule

In the last two sections the departure times of all services were given as fixed. However, it is possible that management will be willing to consider other departure times for the same service in the neighborhood of the most desirable one, if the net result will be a reduction of the fleet size. Alternative departure times may be considered simultaneously for a number or all of the services. But sometimes the effect of a perturbation on the departures of just a few services are of interest.

We turn now to the variable schedule version of the fleet size problem. A set of alternative departure times for a particular service from station X to station Y will be shown as a 'bundle' of arcs from X to Y, which have origin nodes at their respective departure times from X. It is specified that one and only one arc out of each bundle will eventually be used is a service from X to Y. The problem is then, to select one flight out of each bundle so that the fleet size required to accomplish the resulting schedule is minimal. A part of such a schedule map is shown in Fig. 2.7a, where service number 1 has three alternative departure times and service number 2 has two. If the earliest alternatives are

are selected

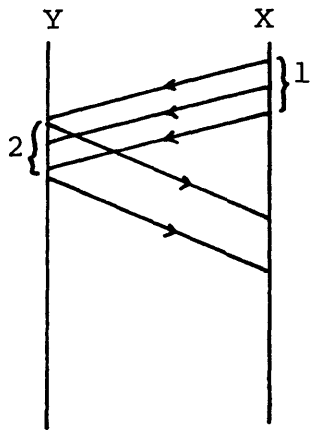
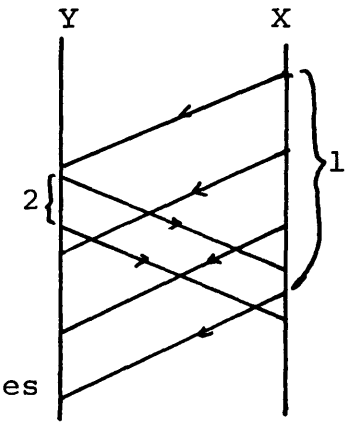


Fig. 2.7a



Service Bundles

Fig. 2.7b

for both services only one aircraft is needed. If the second alternative of service number 1 and the first of number 2 are selected, two airplanes are required. For reasons to be explained later we exclude from consideration cases like the one shown in Fig. 2.7b where the 'bandwidth' of service no. 1 is so large that it is possible to depart on the first arc, return and depart on the last arc of the same service.

In the fixed schedule case there is one and only one arc (s,s_k) and one and only one arc (t_k,t) for service number k in the bipartite graph G*. Now, however, when a bundle of n arcs represent a service there will be a bundle of $n(s,s_i)$ arcs and a bundle of n (t_j,t) arcs in G*, one representative of which will be selected for the optimal schedule. Fig. 2.8 shows the bipartite network corresponding to the schedule map of Fig. 2.7a.

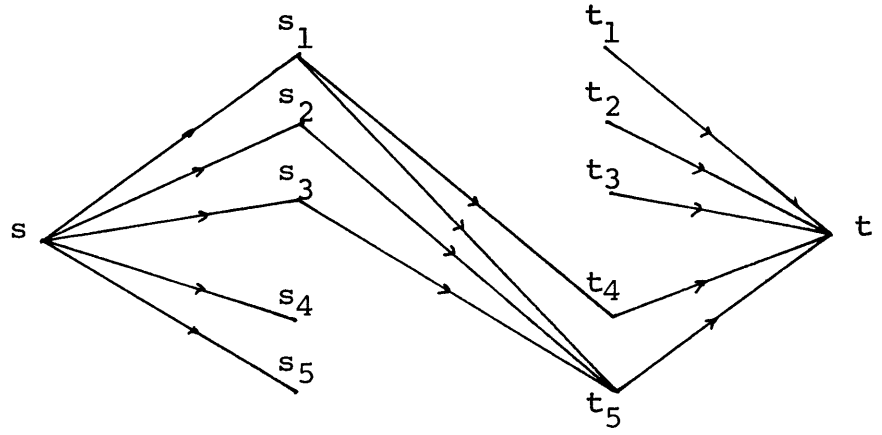


Fig. 2.8

For notational convenience the correspondence between the service numbers and the nodes of S and T in G* will be dropped. The indices will be listed sequentially; e.g. s_4 and t_4 in Fig. 2.8 represent the first arc of bundle number 2 in Fig. 2.7a.

The problem is to select one representative out of each bundle such that the resulting max-flow is maximal over all combinations of selections. Notice that if an arc (s, s_k) is selected its 'counter arc' (t_k, t) must also be selected. This last condition makes it an 'unusual' max-flow problem, and puts it with the class of 'flows with bundle constraints' and 'flow with homologous arcs' network flow problems [1] .

The problem can be formulated as an Integer Linear Programming Problem with 0-1 variables. In order to do so we define the following index sets for G* = [S,T; A*] :

N = the set of services

$K_\ell = \{ k \mid (s, s_k)$ belongs to the ℓ-th bundle$\}$

(There are $|N|$ such sets.)

$A(i) = \{ j \mid (s_i, t_j) \in A^* \}$

$B(j) = \{ i \mid (s_i, t_j) \in A^* \}$

And the following variables:

x_{si} = the flow from s to s_i

x_{ij} = the flow from s_i to t_j

x_{jt} = the flow from t_j to t

u_i = the capacity of arc (s, s_i)

v_j = the capacity of arc (t_j, t)

The ILP (Integer Linear Program) is:

$$\text{(i)} \qquad \max z = \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} x_{si}$$

subject to

$$\text{(ii)} \qquad \sum_{i \in K_\ell} u_i = 1, \qquad \sum_{j \in K_\ell} v_j = 1 \qquad , \forall \ell$$

$$\text{(iii)} \qquad u_i = v_i \qquad , \forall i$$

$$\text{(iv)} \qquad x_{si} \leq u_i \qquad , \forall i$$

(2.1)

$$\text{(v)} \qquad x_{jt} \leq v_j \qquad , \forall j$$

$$\text{(vi)} \qquad x_{si} = \sum_{j \in A(i)} x_{ij} \qquad , \forall i$$

$$\text{(vii)} \qquad x_{jt} = \sum_{i \in B(j)} x_{ij} \qquad , \forall j$$

(All variables are 0-1)

Constraints (ii) specify our requirement that exactly one arc out of each bundle must be selected. Constraints (iii) specify that if an arc (s,s_k) is selected (i.e., if $u_k = 1$) then its counter arc (t_k,t) must be selected too (i.e., $v_k=1$). Constraints (iv) - (vii) are the usual network capacity inequalities and nodal conservation of flow equations. The value of the objective function (i) is the flow from s to t. If the explicit requirement that all variables must be 0-1 is dropped we refer to the problem as LP(2.1) rather than ILP(2.1), and this convention will hold throughout the rest of this work. The optimal solution to LP(2.1) may not, in general, be all-integer, as the example below will demonstrate. However, fortunately, it seems that in many cases it is. In all 'large-scale' problems solved by the author an all-integer solution for LP(2.1) was obtained. A situation which tends to give trouble is the one depicted in Fig. 2.7b and that is why it is desirable to avoid it. Some examples are given below. But in order to deal with the general case we have to resort to the methods of integer programming to obtain the optimal integer solution.

2.7 A Branch and Bound Algorithm for ILP(2.1)

The selection of a successful Branch and Bound algorithm for the solution of a particular problem must strongly depend on the structure of the problem and on computational experience. Since as was mentioned above the author has been unable to produce a 'large' example for which LP(2.1) did not

have an optimal integer solution (about 10 different ones were attempted) we are naturally led to the selection of a Land and Doig type algorithm rather than an Implicit Enumeration type. The selection is strengthened also by the belief that even if the initial optimal solution is non-integer it will take relatively few steps down the Branch and Bound solution tree to obtain the optimal integer solution. The last statement is based on the particular structure of the problem and on computational experience. MPS/360, with which all the computational experiments were conducted is a very efficient system and the solution times which were obtained for the 'large' problems were very satisfactory (see below). Also, experience gained by solving large crew scheduling problems, which is a problem of similar structure (but with restricted cycles rather than chains as elements of interest), by the Land and Doig algorithm leads to the same conclusion*.

The number of variables defined in ILP(2.1) is very large for any real life problem. However, for the present formulation, the partitioning strategy (on which the branching is based) specifies explicitly only a small subset of them, which consists of the u_i's exclusively.

Since there are $|N|$ services in the system at most that number of u_i's will have to be explicitly fixed at 1. In general at each particular node of the tree, some u_i's are fixed at 1 and some others, belonging to other bundles, are explicitly fixed at 0. When u_i is fixed, then by (iii) v_i must be fixed too. The net effect of this is to 'propagate' a change along the chain of which the particular departure is a part, and possibly affect some other chains as well.

---

*See Appendix A

A complete explicit partitioning of the solution space will, in all probability, never be necessary. This observation may point to a good strategy for the selection of variables to be fixed: Of the set of noninteger u_i's start by fixing the ones which currently represent the beginning of a chain.

In order to describe the Branch and Bound algorithm we define the following sets:

$X = \{ x_1, x_2, \ldots \}$, the set of the nodes of the solution tree (the nodes will be indexed sequentially as they are generated)

X_t = the subset of X consisting of all and only the terminal nodes

$I_o(x_k) = \{ i \mid u_i = 0 \text{ at node } x_k \}$

$I_1(x_k) = \{ i \mid u_i = 1 \text{ at node } x_k \}$

and denote by

$\bar{z}(x_k)$ the optimal value of the objective function at node x_k

$S(x_k)$ the optimal solution $\{ u_1, u_2, \ldots \}$ at node x_k

The Branch and Bound Algorithm for ILP(2.1) is:

STEP 1. Create a node x_1 and set $X = X_t = \{ x_1 \}$

STEP 2. Set $I_o(x_1) = \Phi$

STEP 3. Set $I_1(x_1) = \Phi$

STEP 4. Solve LP(2.1) obtaining $S(x_1)$ and $\bar{z}(x_1)$

STEP 5. Set $r = 1$.

STEP 6. If $S(x_r)$ is integer, stop. The optimal objective function has been obtained.

STEP 7. Create two branches and nodes $x_{|X|+1}$ and $x_{|X|+2}$ out of x_r

STEP 8. Set $X = X + \{ x_{|X|+1}, x_{|X|+2} \}$

STEP 9. Set $X_t = X_t + \{ x_{|X|-1}, x_{|X|} \} - \{ x_r \}$

STEP 10. Select an i such that $0 < u_i < 1$

STEP 11. Set $I_o(x_{|X|-1}) = I_o(x_r) + \{ i \}$; $I_1(x_{|X|-1}) = I_1(x_r)$

STEP 12. Set $I_1(x_{|X|}) = I_1(x_r) + \{ i \}$; $I_o(x_{|X|}) = I_o(x_r)$

STEP 13. Solve LP(2.1)' (below) for $k = |X| - 1$ and $k = |X|$ obtaining $S(x_{|X|-1})$, $S(x_{|X|})$, $\bar{z}(x_{|X|-1})$ and $\bar{z}(x_{|X|})$

(i) $$\max \ \bar{z}(x_k) = \sum_{\ell=1}^{N} \sum_{i \in K_\ell} x_{si}$$

S.t.

$$u_i = v_i = 0 \quad , \forall\, i \in I_o(x_k)$$

$$u_i = v_i = 1 \quad , \forall\, i \in I_1(x_k)$$

(ii) $$\sum_{i \in K_\ell} u_i = 1, \qquad \sum_{j \in K_\ell} v_j = 1 \ , \forall \ell$$

$$(2.1)' \quad \text{(iii)} \quad u_i = v_i \qquad \forall i$$

$$\text{(iv)} \quad x_{si} \leqslant u_i \qquad \forall i$$

$$\text{(v)} \quad x_{jt} \leqslant v_j \qquad \forall j$$

$$\text{(vi)} \quad x_{si} = \sum_{j \in A(i)} x_{ij} \qquad \forall i$$

$$\text{(vii)} \quad x_{jt} = \sum_{j \in B(j)} x_{ij} \qquad \forall j$$

STEP 14. Determine r such that $\bar{z}(x_r) = \max_k \{\bar{z}(x_k \in X_t)\}$

STEP 15. Go to STEP 6.

Notice that in the stopping condition it is stated that the optimal objective function corresponding to the optimal integer solution has been obtained. Although $S(x_r)$ is integer it may yet be possible that some of the flow components are not. When $S(x_r)$ is integer we have a definition of a 'simple' network. The incidence matrix of such a network has the unimodularity property, that is, every submatrix has determinant $\pm$ 1 or 0. This property guarantees that every basic feasible solution is integer. In our case, however, this is not true unless we specify that unnecessary constraints and variables are eliminated from the system when branching takes place. This would seem to be an unnecessary complication since when $S(x_r)$ is integer and the flows are not, we can apply a max-flow routine to the defined simple network and obtain the all-integer solution we seek. The value of this max flow is equal to $\bar{z}(x_r)$. (In all cases tested on MPS/360 it has been the author's rather gratifying

experience that if an optimal all-integer solution existed it was invariably the one which was obtained). We prove formally:

Theorem 2.2 When the stopping condition (STEP 6) is met an optimal solution to ILP (2.1) is determined by $S(x_r)$.

Proof: Let $S(x_r)$ be integer with a corresponding optimal objective $\bar{z}(x_r)$. The min cut capacity of the 'simple' network is at least equal to $\bar{z}(x_r)$. It cannot be greater than $\bar{z}(x_r)$ for this would be a contradiction to the definition of $\bar{z}(x_r)$. Hence there is an integer flow of value $\bar{z}(x_r)$ in this network. From the selection of r (STEP 14) it is obvious that no other all-integer solution can have a better objective function.

As an example, the algorithm is applied to the three station problem shown in Fig. 2.10. There are 8 services with two possible departures for each one of them. The bipartite network is shown in Fig. 2.11a. The solution of LP(2.1) for this case gives $u_i = \frac{1}{2}\ \forall i$ and the max-flow in this network is shown by the heavy lines, each of which represents a flow of value ½. The arcs (s,s_i) and (t,t_j) are not shown in this figure. The solution tree for a random selection of a fractional variable is shown in Fig. 2.9a and the one corresponding to the preferred selection of a variable which starts a chain is shown in Fig. 2.9b. Upper and lower bounds on variables are not handled in MPS/360 as

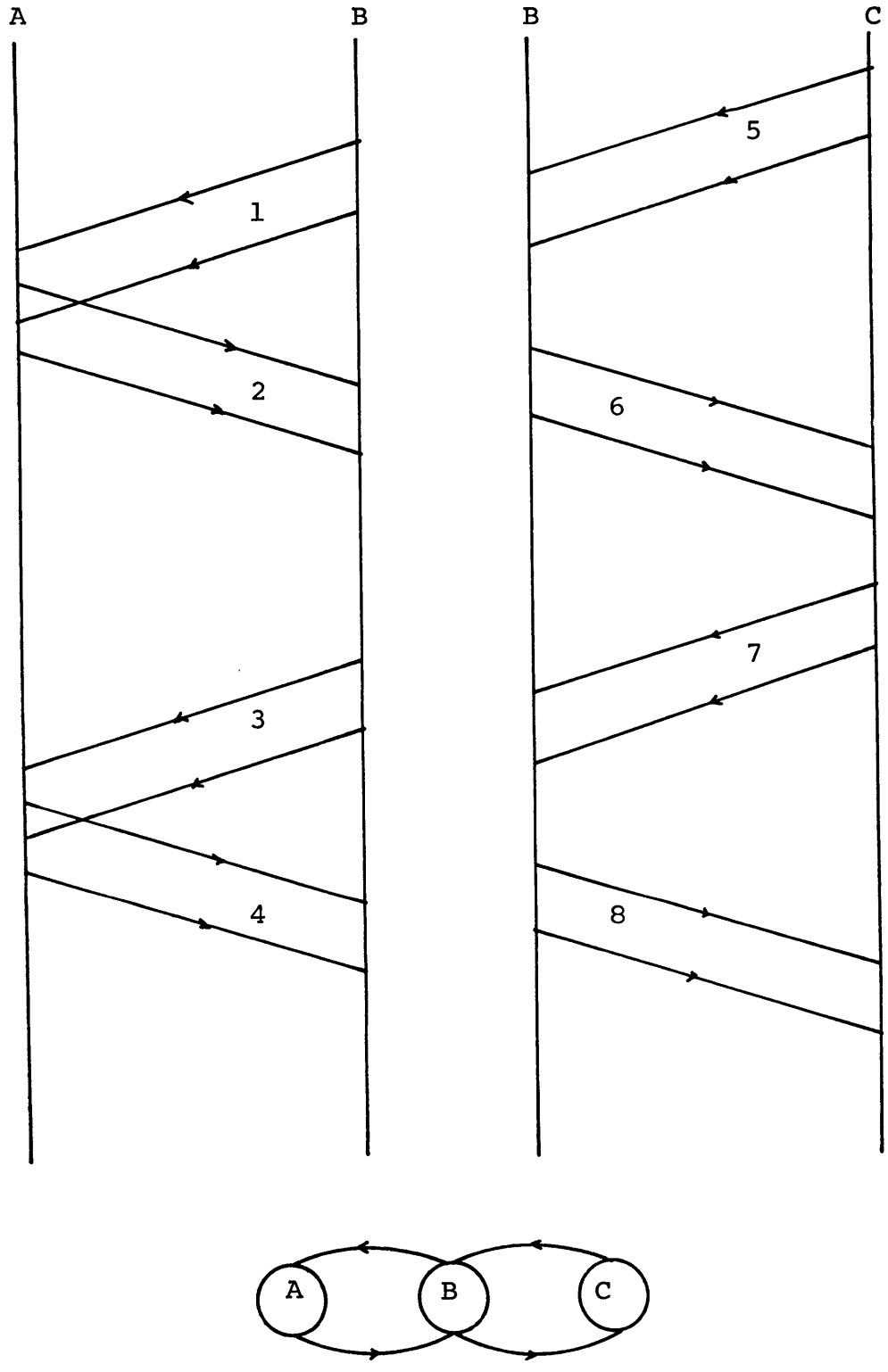


Fig. 2.10

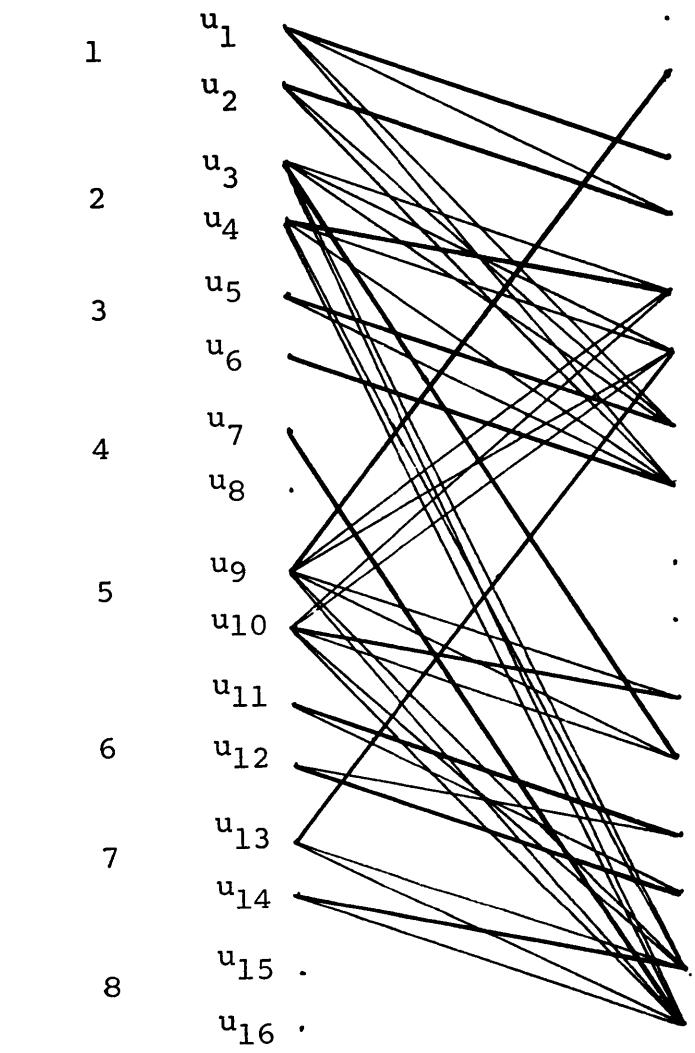


Fig. 2.11a

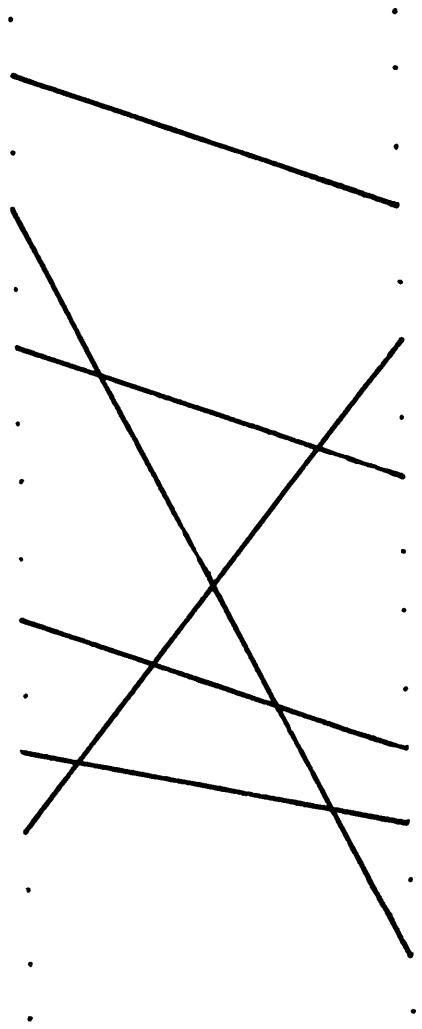


Fig. 2.11b

explicit constraints, which is a great saving in the size of the basis.

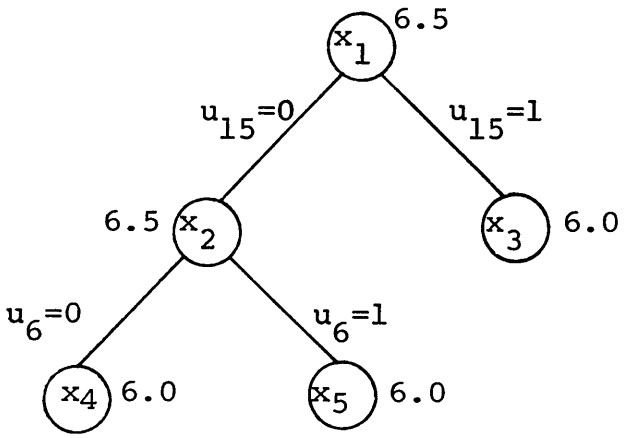


Fig. 2.9a

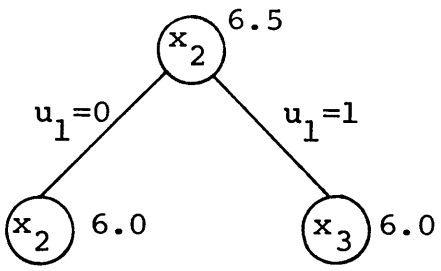


Fig. 2.9b

An optimal integer solution is shown in Fig. 2.11b. The heavy lines represent unit flows (and the flight connections). Since there are 6 such connections at least 2 aircraft are needed to maintain the schedule thus determined.

Notice that compared with the number of variables defined for this problem the number of Branch and Bound iterations is remarkably low. Experiments with other small problems point to the same conclusion.

A three city transportation system is shown in Fig. 2.12, for which the schedule map is drawn in Fig. 2.13. The groups of possible departures for each service are indicated by heavy lines and the service numbers alongside the departure points. There are 26 required services in this system. Three different fleet size problems were solved using the data of Fig. 2.13. The first was a subsystem consisting of services between A and B only (10 services), the second

between A and B and between A and C without the direct services between B and C (18 services), and the third for the entire system (26 services). The solutions of LP(2.1) for all three problems were integer and the one for the complete system is shown in Fig. 2.14a (An alternative solution is shown in Fig. 2.14b). Statistics for the runs are given in table 2.1. These statistics are shown in order to point out the fact that solution times for large LP problems are very reasonable and, therefore, that the Land and Doig Branch and Bound Algorithm has become a feasible method for solving our problems. Also, the good chances of obtaining an integer solution at the first iteration strengthens the author's belief in the efficiency of this approach. Some more computational results will be given in Chapter III.

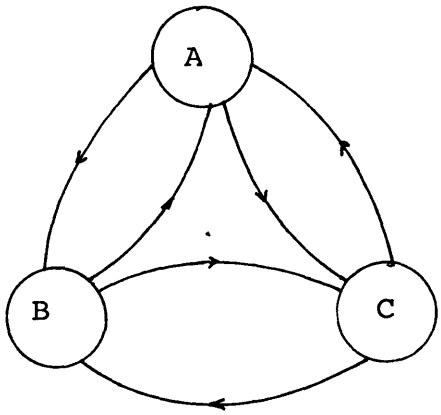


Fig. 2.12

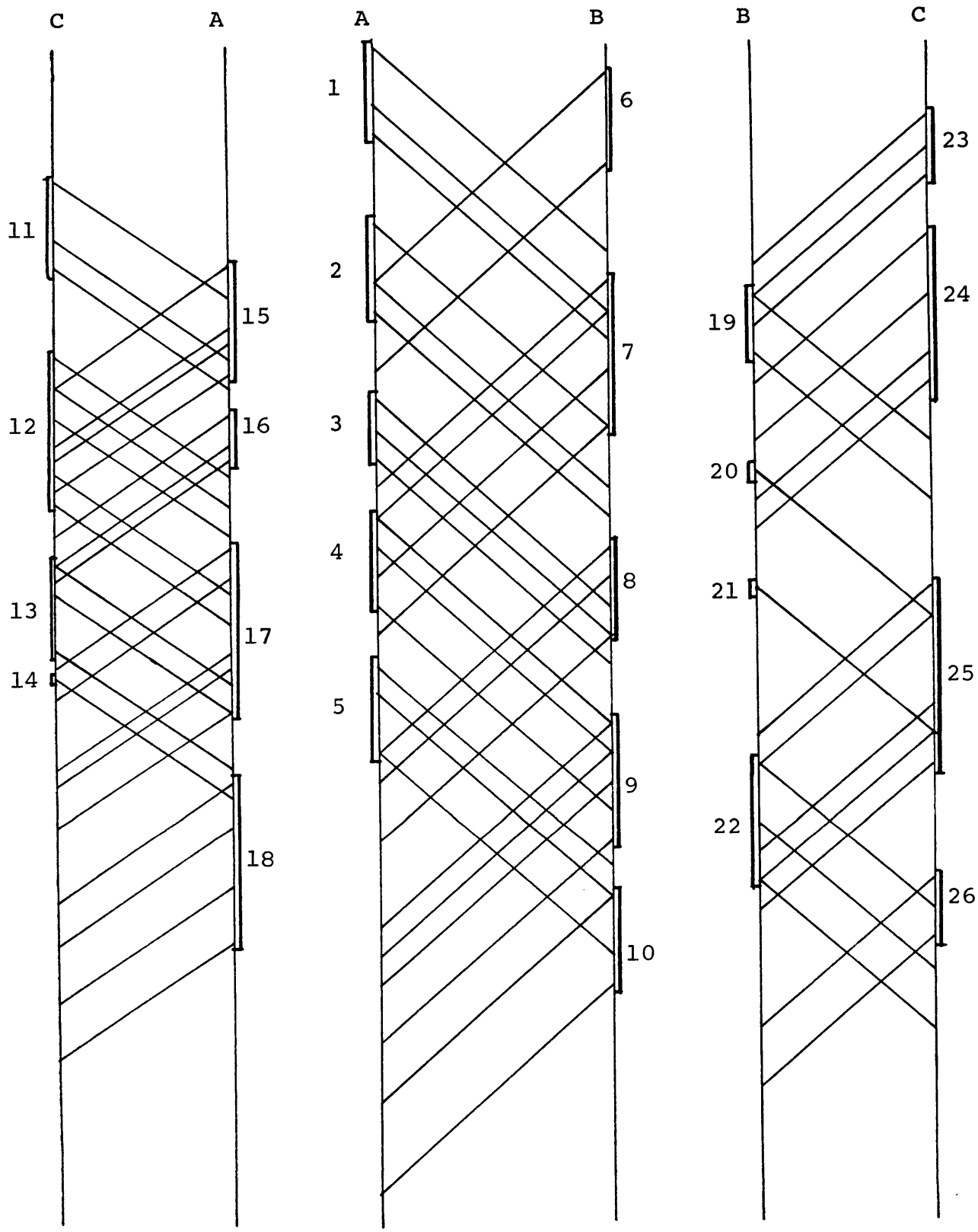


Fig. 2.13

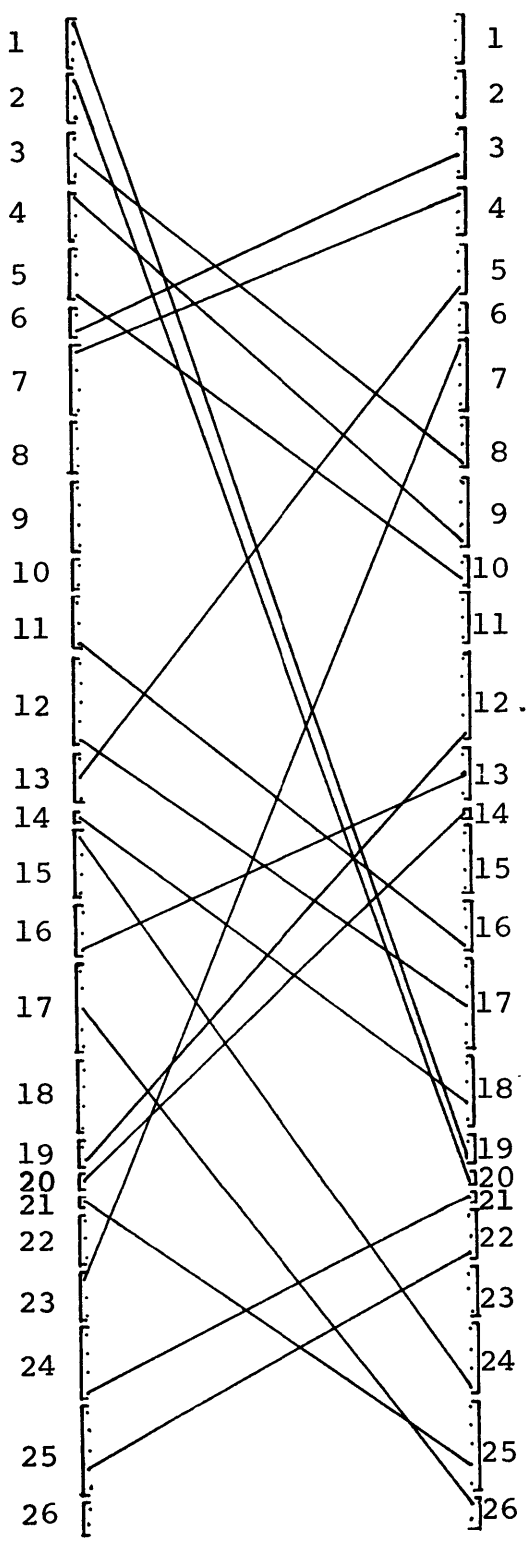


Fig. 2.14a

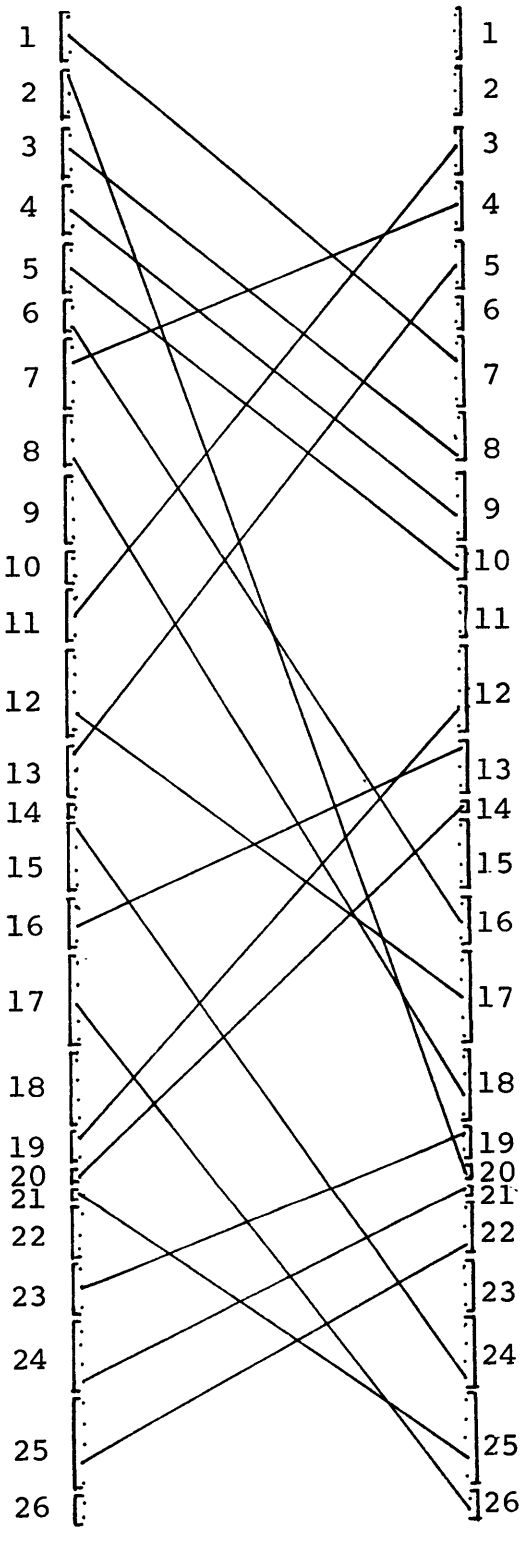


Fig. 2.14b

(Experience with the Implicit Enumeration approach to large crew scheduling problems has been rather discouraging[16]).

Prob.No.	Rows*	Cols.**	Run time***	Setup time	Simplex Algo. time
1	141	344	0.70	0.14	0.10
2	269	789	1.13	0.25	0.39
3	369	1221	2.01	0.39	1.03

*Actually, a considerably smaller number of rows is necessary (see next section)

**Includes one artifical variable for each row.

***Time is given in minutes.

Table 2.1

2.8 A Cut and Branch Algorithm for ILP(2.1)

Suppose that LP(2.1) is solved and $z^o = \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} x^o_{si}$ is the optimal value of the objective function. If the constraint (cut)

$$\sum_{\ell=1}^{N} \sum_{i \in K_\ell} x_{si} = F$$

where $0 \leq F \leq z^o$ is added to the set of constraints of LP(2.1), there exists a feasible solution to the new Linear Program which will be denoted by LP(2.1)''. That this is so is clear from the observation that any amount of flow can be withdrawn from the network defined by the optimal solution $\{u_i\}$ to LP(2.1). If z^o is fractional then an optimal objective function to an integer solution cannot have value greater than $[z^o]$ (the largest integer in z^o). If it can be guaranteed that the solution obtained to the new problem will be all-integer if one exists, then the solution procedure to follow is simple: Solve LP(2.1)'' starting with $F = [z^o]$ and reducing F by one unit until an integer feasible solution is obtained. This cannot be guaranteed, of course, for reasons explained in the last section. Hence some finer cuts are needed. If LP(2.1)'' is solved and a non-integer solution $S(x_1)$ is obtained a Land and Doig type branching procedure can be used to determine whether a feasible integer solution having a flow of value F exists. The terminal nodes of the solution tree may now have infeasible solutions associated with them. If all terminal nodes represent infeasible solutions we have to restart with F set to F-1.

Define the following sets (in addition to those defined earlier)

$S(x_k)$ = the solution at node x_k

X_{tf} = the set of feasible terminal nodes

The Cut and Branch algorithm is:

STEP 1. Solve LP(2.1). If the solution is all-integer stop. The optimal solution to ILP(2.1) has been obtained.

STEP 2. Set $F = [z^o]$ (where z^o is the optimal objective in STEP 1.)

STEP 3. Create node x_2 and set $X = X_{tf} = \{x_1\}$

STEP 4. Set $I_o(x_1) = \Phi$

STEP 5. Set $I_1(x_1) = \Phi$

STEP 6. Solve LP(2.1) with the added constraint $\sum_{\ell=1}^{N} \sum_{i \in K_\ell} x_{si} = F$ obtaining $S(x_1)$

STEP 7. Set $r = 1$

STEP 8. If $S(x_r)$ is integer stop. The optimal solution to ILP(2.1) has been obtained.

STEP 9. Create two branches and nodes $x_{|X|+1}$ and $x_{|X|+2}$ out of x_r.

STEP 10. Set $X = X + \{ x_{|X|+1}, x_{|X|+2} \}$

STEP 11. Select an $i \ni o < u_i < 1$

STEP 12. Set $I_o(x_{|X|-1}) = I_o(x_r) + \{i\}$; $I_1(x_{|X|-1}) = I_1(x_r)$

STEP 13. Set $I_1(x_{|X|}) = I_1(x_r) + \{i\}$; $I_o(x_{|X|}) = I_o(x_r)$

STEP 14. Solve LP(2.1)' with the added constraint (defined in STEP 6) obtaining $S(x_{|X|-1})$, $S(x_{|X|})$

STEP 15. If $S(x_{|X|-1})$ is feasible set $X_{tf} = X_{tf} + \{ x_{|X|-1} \}$

If $S(x_{|X|})$ is feasible set $X_{tf} = X_{tf} + \{x_{|X|}\}$

STEP 16. If $X_{tf} = \Phi$ set $F = F-1$ and go to STEP 6.

STEP 17. Select an $r \ni x_r \in X_{tf}$

STEP 18. Go to STEP 8.

Lemma 2.1 If the conditional of STEP 16 is true there exists no feasible solution to ILP(2.1)''.

Proof: Assume that there exists a feasible solution to ILP(2.1)'' and let $S_o = \{u_1, u_2, \ldots\}$ be one such solution. Let T_r be the tree defined by executing STEP 6-STEP 18 when the conditional of STEP 16 is true. We shall trace a chain in T_r from x_1 to some terminal node as follows: Start at x_1 and follow the branch $u_m = 0$ or $u_m = 1$, where u_m is the variable which defines the partitioning, according to the value of u_m in S_o. Continue by following branches out of subsequent nodes of T_r using the same rule until a terminal node is reached. Let this node be x_k and $x_k \notin X_{tf}$. But here we arrive at a contradiction since in S_o we have $u_i = 0 \ \forall i \in I_o(x_k)$, $u_i = 1 \ \forall i \in I_1(x_k)$ and S_o is assumed to be a feasible solution to ILP(2.1)''.

From Lemma 2.1 and Theorem 2.2 it is clear that when the algorithm stops in STEP 8 an optimal solution to ILP(2.1) is obtained with $z = F$.

The computational success of this algorithm will depend mainly on the size of the tree. If the set of feasible terminal nodes is generally small then there is a good chance that this algorithm will be better than the one described in the last section. However only extensive computational experience with large problems can provide sufficient evidence to make a judgement on this point.

The algorithm was applied to example of Fig. 2.10 using MPS/360. After LP(2.1) was solved giving $z = 6.5$, LP(2.1)'' was set up with $\sum_{\ell=1}^{N} \sum_{i \in K_\ell} x_{si} = 6$ and the first solution was all-integer.

2.9 Alternative Formulation

We now discuss an approach which will result in a substantial reduction in the size of the Linear Programs. In this approach only the variables x_{ij} from LP(2.1) and some other variables to be defined shortly are used. Consider the 4 services example shown in Fig. 2.15a and the bipartite graph in Fig. 2.15b.

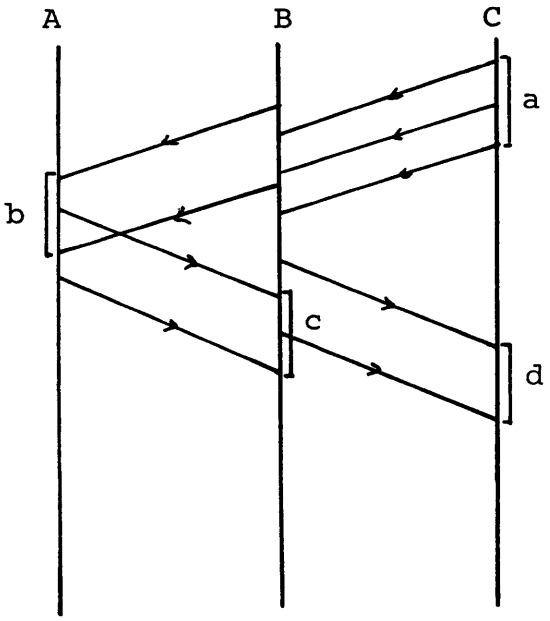


Fig. 2.15a

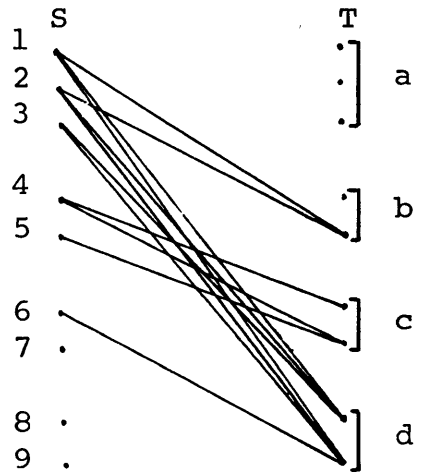


Fig. 2.15b

We shall refer to the condition that exactly one departure must be taken out of each bundle (or, alternatively, that at most one unit of flow can pass through a bundle) as logical condition I. If S is the set of sources and T is

the set of sinks then we refer to the requirement that if a source s_k is selected so must be the sink t_k as logical condition II.

Define the following variables (see ILP(2.1) for the definition of the sets) :

x_{ij} = the flow from s_i to t_j
s_i = the slack at source i
t_j = the slack at sink j

The ILP is:

$$\text{(i)} \quad \max \; z = \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} \sum_{j \in A(i)} x_{ij}$$

S.t.

$$\text{(ii)} \quad \sum_{i \in K_\ell} (s_i + \sum_{j \in A(i)} x_{ij}) = 1 \qquad , \forall \ell$$

$$\text{(2.2) (iii)} \quad \sum_{j \in K_\ell} (t_j + \sum_{i \in B(j)} x_{ij}) = 1 \qquad , \forall \ell \ni |K_\ell| = 1$$

$$\text{(iv)} \quad s_i + \sum_{j \in A(i)} x_{ij} + \sum_{j \in K_\ell - \{i\}} (t_j + \sum_{j \in B(j)} x_{ij}) = 1 \qquad , \forall i \in K_\ell \; , \forall \ell \ni |K_\ell| > 1$$

$$s_i, t_j, x_{ij} = 0,1 \qquad , \forall i,j$$

Constraints (ii) are logical condition I for the set S. Constraints (iii) are logical condition I for those members of T which have only one possible departure (although a summation sign is written only one term results). Condition I for the other members of T will result from constraints

(iv) as will be shown below. The last constraints also induce the satisfying of logical condition II. (The physical interpretation of s_i and t_j is this: If $s_i = 1$ the departure indexed by i is selected and is the last flight on the chain that includes this service. If $t_j = 1$ the departure indexed by j is selected and is the first flight on the chain that includes it). We now prove the statements made in the last paragraph.

Let $X_I = \{ x_{ij} \}$, $S_I = \{ s_i \}$, $T_I = \{ t_j \}$ be a feasible integer solution to ILP(2.2). At least one such feasible solution can be obtained by setting $x_{ij} = 0 \ \forall i,j$ and selecting one arbitrary index k out of every set K_ℓ and setting $s_k = t_k = 1$. These values satisfy (ii)-(iv) as can be checked by substitution. The logical conditions are also trivially satisfied.

Let K_ℓ be an arbitrary bundle. If $|K_\ell| = 1$ then constraints (ii) and (iii) guarantee that logical condition I is satisfied, and so is condition II trivially. Let $|K_\ell| > 1$. Then (ii) guarantees that condition I is satisfied for the set S_{K_ℓ} (the sources of K_ℓ). Suppose it is not satisfied for the set T_{K_ℓ} (the sinks). Then it must be true that either

$$\sum_{j \in K_\ell} (t_j + \sum_{i \in B(j)} x_{ij}) = 0$$

or

$$\sum_{j \in K_\ell} (t_j + \sum_{i \in B(j)} x_{ij}) \geq 2$$

Assume that the first possibility is true, and let $m \in K_\ell$ be an index such that

$$s_m + \sum_{j \in A(m)} x_{mj} = 0$$

(such an index clearly exists since $|K_\ell| \geq 2$ and (ii) holds).
Then

$$s_m + \sum_{j \in A(m)} x_{mj} + \sum_{j \in K_\ell - \{m\}} (t_j + \sum_{i \in B(j)} x_{ij})$$

$$= \sum_{j \in K_\ell - \{m\}} (t_j + \sum_{i \in B(j)} x_{ij}) \leq \sum_{j \in K_\ell} (t_j + \sum_{i \in B(j)} x_{ij})$$

$$= 0$$

which is a violation of constraints (iv).
Assume that the second possibility is true and that

$$\sum_{j \in K_\ell} (t_j + \sum_{i \in B(j)} x_{ij}) = M \qquad \text{where } M \geq 2$$

Let $n \in K_\ell$ be an index such that

$$s_n + \sum_{j \in A(n)} x_{nj} = 1$$

(constraints (ii) guarantee the existence of such an index).
Then from (iv)

$$s_n + \sum_{j \in A(n)} x_{nj} + \sum_{j \in K - \{n\}} (t_j + \sum_{i \in B(j)} x_{ij}) = 1$$

we get

$$\sum_{j \in K_\ell - \{n\}} (t_j + \sum_{i \in B(j)} x_{ij}) = 0$$

hence

$$t_n + \sum_{i \in B(j)} x_{ij} = \sum_{j \in K_\ell} (t_j + \sum_{i \in B(j)} x_{ij}) - \sum_{j \in K_\ell - \{n\}} (t_j + \sum_{i \in B(j)} x_{ij})$$

$$= M - 0 = M$$

Then writing constraints (iv) when i = m (chosen above),

$$s_m + \sum_{j \in A(m)} x_{ij} + \sum_{j \in K_\ell - \{m\}} (t_j + \sum_{i \in B(j)} x_{ij})$$

$$= \sum_{j \in K_\ell - \{m,n\}} (t_j + \sum_{i \in B(j)} x_{ij}) + t_n + \sum_{i \in B(n)} x_{ij}$$

$$= t_n + \sum_{i \in B(n)} x_{ij} = M \geq 2$$

which is a violation of constraints (iv).

We now prove that condition II holds too. Assuming that

$$s_n + \sum_{j \in A(n)} x_{nj} = 1$$

and

$$t_m + \sum_{i \in B(m)} x_{im} = 1$$

where $m,n \in K_\ell$ and $m \neq n$, we have from (iv)

$$s_n + \sum_{j \in A(n)} x_{nj} + \sum_{j \in K_\ell - \{n\}} (t_j + \sum_{i \in B(j)} x_{ij})$$

$$= s_n + \sum_{j \in A(n)} x_{nj} + t_m + \sum_{i \in B(m)} x_{im} + \sum_{j \in K_\ell - \{m,n\}} (t_j + \sum_{i \in B(j)} x_{ij}) = 2$$

which is a violation of constraints (iv).

All the violations of constraints contradict the assumption that we have a feasible integer solution. The proof is therefore complete.

ILP(2.2) for the example of Fig. 2.15 is written below.

$$\max z = x_{15} + x_{18} + x_{19} + x_{25} + x_{28} + x_{29} + x_{38} + x_{39} + x_{46} + x_{47} + x_{57} + x_{69}$$

subject to

$$s_1 + x_{15} + x_{18} + x_{19} + s_2 + x_{25} + x_{28} + x_{29} + s_3 + x_{38} + x_{39} = 1$$

$$s_4 + x_{46} + x_{47} + s_5 + x_{57} = 1$$

$$s_6 + x_{69} + s_7 = 1$$

$$s_8 + s_9 = 1$$

$$s_1 + x_{15} + x_{18} + x_{19} + t_2 + t_3 = 1$$

$$s_2 + x_{25} + x_{28} + x_{29} + t_1 + t_3 = 1$$

$$s_3 + x_{38} + x_{39} + t_1 + t_2 = 1$$

$$s_4 + x_{46} + x_{47} + t_5 + x_{15} + x_{25} = 1$$

$$s_5 + x_{57} + t_4 = 1$$

$$s_6 + x_{69} + t_7 + x_{47} + x_{57} = 1$$

$$s_7 + t_6 + x_{46} = 1$$

$$s_8 + t_9 + x_{19} + x_{29} + x_{39} + x_{69} = 1$$

$$s_9 + t_8 + x_{18} + x_{28} + x_{38} = 1$$

$$x_{ij}, s_i, t_j = 0,1 \ \forall i,j$$

2.10 A Branch and Bound Algorithm for ILP(2.2)

The algorithm for ILP(2.2) will explicitly set only the variables x_{ij}. Define the sets:

$$I_o(x_k) = \{ (i,j) \mid x_{ij} = 0 \text{ at node } x_k \}$$

$$I_1(x_k) = \{ (i,j) \mid x_{ij} = 1 \text{ at node } x_k \}$$

$$S(x_k) = \text{the solution set } \{ x_{ij} \} \text{ at node } x_k$$

The other sets were defined in the last section.

The Branch and Bound algorithm is:

STEP 1. Create a node x_1 and set $X = X_t = \{ x_1 \}$

STEP 2. Set $I_o(x_1) = \Phi$

STEP 3. Set $I_1(x_1) = \Phi$

STEP 4. Solve LP(2.2) obtaining $S(x_1)$ and $\bar{z}(x_1)$

STEP 5. Set r = 1

STEP 6. If $S(x_r)$ is integer stop. Optimal integer solution has been obtained.

STEP 7. Create two branches and nodes $x_{|X|+1}$ and $x_{|X|+2}$ out of x_r.

STEP 8. Set $X = X + \{x_{|X|+1}, x_{|X|+2}\}$

STEP 9. Set $X_t = X_t + \{x_{|X|-1}, x_{|X|}\} - \{x_r\}$

STEP 10. Select $(i,j) \ni 0 < x_{ij} < 1$

STEP 11. Set $I_o(x_{|X|-1}) = I_o(x_r) + \{(i,j)\}$; $I_1(x_{|X|-1}) = I_1(x_r)$

STEP 12. Set $I_1(x_{|X|}) = I_1(x_r) + \{(i,j)\}$; $I_o(x_{|X|}) = I_o(x_r)$

STEP 13. Solve LP(2.2)' for k = |X|-1 and k = |X| obtaining $S(x_{|X|-1})$, $\bar{z}(x_{|X|-1})$ and $S(x_{|X|})$, $\bar{z}(x_{|X|})$

(i) $$\max \bar{z}(x_k) = \sum_{\ell=1}^{N} \sum_{i \in K_\ell} \sum_{j \in A(i)} x_{ij}$$

S.t.

$$x_{ij} = 0 \quad , \forall (i,j) \in I_o(x_k)$$

$$x_{ij} = 1 \quad , \forall (i,j) \in I_1(x_k)$$

(2.2)' (ii) $$\sum_{i \in K_\ell} (s_i + \sum_{j \in A(i)} x_{ij}) = 1 \quad , \forall \ell$$

(iii) $$\sum_{j \in K_\ell} (t_j + \sum_{i \in B(j)} x_{ij}) = 1 \quad , \forall \ell \ni |K_\ell| = 1$$

$$\text{(iv)} \quad s_i + \sum_{j \in A(i)} x_{ij} + \sum_{j \in K_\ell - \{i\}} \left(t_j + \sum_{i \in B(j)} x_{ij} \right) = 1$$

$$, \ \forall \ i \in K_\ell$$

$$, \ \forall \ell \ni |K_\ell| > 1$$

STEP 14. Determine r such that $\bar{z}(x_r) = \max_k \{\bar{z}(x_k \in X_t)\}$

STEP 15. Go to STEP 6

The stopping condition (STEP 6) specifies that only $S(x_r)$ must be integer. However the other variables $\{s_i\}$, $\{t_j\}$ will also be integer or could be trivially set to integer values without a change in $\bar{z}(x_r)$, thus in effect obtaining an all-integer solution. Formally

Theorem 2.3 When the stopping condition (STEP 6) is met an optimal solution to ILP(2.2) is determined by $S(x_r)$.

Proof: Let $S(x_r)$ be integer and select an arbitrary bundle K_ℓ. Then there are four cases:

1. $x_{mj} = 1$, $x_{im} = 1$ for some i,j where $m \in K_\ell$; then
   $s_i = t_i = 0 \ \forall \ i \in K_\ell$
2. $x_{mj} = 1$, $\sum_{j \in K_\ell} \sum_{i \in B(j)} x_{ij} = 0$; then $s_i = 0 \ \forall \ i \in K_\ell$,
   $t_m = 1$
   $t_j = 0 \ \forall \ j \in K_\ell - \{m\}$
3. $x_{im} = 1$, $\sum_{i \in K_\ell} \sum_{j \in A(i)} x_{ij} = 0$; then $s_m = 1$,

$$s_i = 0 \qquad \forall\ i \in K_\ell - \{m\}$$

$$t_j = 0 \qquad \forall\ j \in K_\ell$$

4. $\sum_{i \in K_\ell} \sum_{j \in A(i)} x_{ij} = 0,\ \sum_{j \in K_\ell} \sum_{i \in B(j)} x_{ij} = 0$;

$$\text{then } \sum_{i \in K_\ell} s_i = 1\ ,\quad \sum_{j \in K_\ell} t_j = 1$$

In which case not all s_i, t_j are necessarily integer. Then arbitrarily fix $s_k = t_k = 1$ for some $k \in K_\ell$, thus obtaining an all-integer solution.

from the selection rule in STEP 14 it is clear that the solution thus obtained is the optimal solution to ILP(2.2).

In the formulation of the last section we dealt explicitly only with a small subset of the variables, while in the present formulation the number is considerably higher. Hence it is likely that more Branch and Bound iterations will be required in the latter than in the former. However the size of the basis is considerably smaller here. A decision as to which formulation to use will depend on the size of the transportation system. A more quantitative discussion, with reference to the capacity of MPS/360 will be given in the next section.

2.11 Minimal Flow Formulation; Comparisons

The variable schedule problem can be formulated as a problem of minimal flow with bundle constraints. The schedule map will be transformed into a network by adding some arcs

and defining capacity and flow functions on subsets of arcs of the graph. The vertical lines, representing the stations will be called ''station lines'. The bottom of the station lines will be connected to their top by 'overnight arcs'. The other arcs, connecting one station to another arc the 'flight arcs'. A network thus constructed is shown in Fig. 2.16. The objective is to minimize the sum of the flows on the overnight arcs subject to the

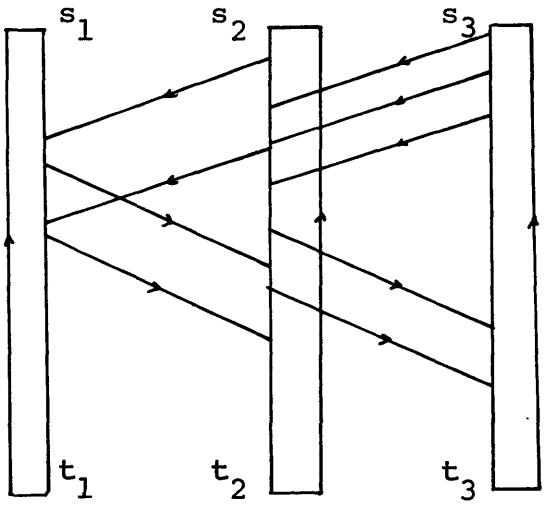


Fig. 2.16

conditions that a unit flow must pass through exactly one arc out of every bundle of flight arcs.

Define the following sets:

K_ℓ = the set of flight arcs belonging to the ℓ-th service

B(i)= the set of nodes such that there is an arc from each one of them to node i

A(i) = the set of nodes such that there is an arc from node i to each one of them

And the variable

f_{ij} = the flow from node i to node j

The ILP for this formulation is:

$$\text{(i)} \qquad \min \quad z = \sum_{(t_i, s_i)} f_{t_i s_i}$$

S.t.

$$\text{(ii)} \qquad \sum_{k \in B(i)} f_{ki} - \sum_{p \in A(i)} f_{ip} = 0 \qquad , \forall\, i$$

(2.3)

$$\text{(iii)} \qquad \sum_{(i,j) \in K_\ell} f_{ij} = 1 \qquad , \forall\, \ell$$

$$\text{(iv)} \qquad f_{ij} = \text{integer} \qquad , \forall\, (i,j)$$

This formulation is perhaps more natural than the preceding two. However it has a significant disadvantage, which will now be discussed.

Linear Programming codes usually restrict the number of rows which can be handled. MPS/360, for example, has a capacity of up to 4095 rows with a (practically) unlimited number of columns. Depending on storage available it can handle very efficiently a smaller number of rows. We shall now compare the number of constraints required for each of the formulations described thus far. But in order to do so we bring LP(2.1) to a minimal form. This can be done by substituting u_j for v_j in (v) and substituting (iv) and (v) in (vi) and (vii) respectively. Thus obtaining the ILP:

$$\text{(i)} \quad \max \; z = \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} \sum_{j \in A(i)} x_{ij}$$

S.t.

$$\text{(ii)} \quad \sum_{i \in K_\ell} u_i = 1 \qquad , \forall \ell$$

(2.1a)

$$\text{(iii)} \quad u_i \geq \sum_{j \in A(i)} x_{ij} \qquad , \forall i$$

$$\text{(iv)} \quad u_j \geq \sum_{i \in B(j)} x_{ij} \qquad , \forall j$$

We shall refer to the formulations of the problem as I, II and III in the order in which they were introduced in this chapter. The number of constraints required are:

for formulation I: $|N| + 2\left(\sum_{\ell=1}^{|N|} |K_\ell|\right)$

for formulation II: $|N| + \sum_{\ell=1}^{|N|} |K_\ell|$

for formulation III: $|N| + 2\left(\sum_{\ell=1}^{|N|} |K_\ell|\right)$ (every arrival and every departure are counted as a node)

The number of variables depends on the problem. The comparison for two specific examples: 1. Fig. 2.10, 2. The entire system of Fig. 2.13, are given below:

	I	II	III	I	II	III
constraints	40	24	40	184	105	184
variables	60	76	48	773	852	243

It is evident from these statistics that formulation II is the best when the system is large and/or the number of

alternative departure times is large. In any case I and II are preferred since they also provide the scheduler with an ability to treat individual connections in any manner desired. He can generate improved (in the sense discussed in section 2.5) schedules or eliminate specific connections from consideration if he feels that it is desirable to do so for any reason.

The large domestic airlines have a total of above 1000 services per day. However the system can frequently be decomposed (geographically) so that it may not be necessary to treat the entire system simultaneously. For example if a subsystem consisting of 500 services and 6 alternative departures for each service is considered, it can be handled by formulation II (using MPS/360) but not by the other two. In the author's judgement this is sufficient for handling present day systems. If the system is small enough it may be desirable to use formulation I (or III) for as was argued earlier it may result in a smaller number of iterations of the Branch and Bound algorithm. In the next section a fourth formulation which reduces the number of constraints even further will be discussed.

2.12 Arc-Chain Minimal Flow Formulation

The minimal flow problem of the last section can be formulated in an Arc Chain form [7]. This formulation reduces drastically the number of constraints. As we shall see the number of constraints in this form is equal to the number of services. The number of columns is extremely large, but it is not necessary to know all of them. Only the columns which

are introduced into the basis must be known, and they are generated as the algorithm progresses. The method of column generation was suggested by Ford and Fulkerson in [8] for computing maximal multicommodity network flows.

The formulation is based on the observation that we could in principle enumerate all the chains that an aircraft can be assigned to and select the minimal number of chains that will accomplish all the services. If $c_1, c_2, .., c_n$ is the list of all the chains, c_j can be represented as a vector with $|N|$ components, one for each service. If $\vec{A}_1, \vec{A}_2, \ldots, \vec{A}_p$ is an enumeration of all flight arcs then we define the following matrix $A = (a_{\ell j})$:

$$a_{\ell j} = \begin{cases} 1 & \text{if } c_j \text{ includes } \vec{A}_i \ni i \in K_\ell \\ 0 & \text{otherwise} \end{cases}$$

(See the definition of K_ℓ in the last section).
And let x_j be the flow along the chain c_j (or, alternatively, a decision variable as to whether c_j will be used in the optimal schedule). Then our optimization problem is:

$$\text{(i)} \qquad \text{minimize } z = \sum_{j=1}^{n} x_j$$

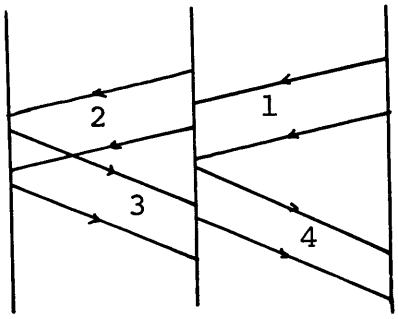
S.t.

(2.4)

$$\text{(ii)} \qquad \sum_{j=1}^{n} a_{\ell j}\, x_j = 1 \qquad \ell = 1, \ldots, |N|$$

$$x_j = 0, 1 \qquad \forall\, j$$

Following the discussion in section 2.5 it is assumed that no chain can include more than one member of a bundle. Fig. 2.17 is an example of the set of constraints for a 4-service system.



x_1	x_5	x_9	x_{13}	x_{17}	x_{19}		
1 1		1 1 1 1 1		1		= 1	(1)
1 1			1 1 1 1	1 1		= 1	(2)
1 1			1 1 1 1 1 1			= 1	(3)
	1 1 1 1 1 1			1 1		= 1	(4)

Fig. 2.17

The number of chains even for this small example is quite large. Chains which cover exactly the same services are represented by the same columns in the matrix, although physically they represent different sets of flight arcs.

In order to proceed with the algorithm all we need is a starting feasible basis. This can be easily obtained by selecting an arbitrary flight arc from each service bundle and defining them as single arc chains (see the first 8 chains of the example given above) thus obtaining a convenient unit matrix. In general, suppose we have a basis B with $\underline{d}_B$ the vector of costs of variables in the basis. Then the simplex multipliers are $\underline{\pi} = \underline{d}_B B^{-1}$ where $\underline{d}_B = \underline{1}_{|N|}$. A chain x_j may be introduced into the basis if

$$\sum_{\ell=1}^{|N|} \pi_\ell a_{\ell j} - 1 > 0 \quad \text{or} \quad 1 + \sum_{\ell=1}^{|N|} (-\pi_\ell) a_{\ell j} < 0.$$

If $-\pi_\ell$ is interpreted as the length of all the flight arcs belonging to the ℓ-th service, a shortest chain algorithm can be used to find the one to be introduced into the basis or to determine that an optimal solution has been obtained if the length of the shortest chain is not less than -1. From this discussion it is clear that explicit knowledge of all the chains is not necessary. A suitable shortest chain algorithm must handle both positive and negative arc lengths. Such an algorithm is described in Chapter III of [7]. A condition that this particular algorithm will be guaranteed to work is that there should be no negative cycles in the graph. But since our graph is acyclic no problems arise.

The case shown in Fig. 2.7b is undesirable particularly in this formulation of the problem, because if we enumerate all chains in a schedule map which includes such cases we are likely to obtain erroneous results. In such cases a chain may include two or more flight arcs which belong to the same service. Notice that in the present formulation there is no way to express this fact to the system of equations in ILP(2.4). Consider for example the schedule map shown in Fig. 2.18. Except for one service which consists of two possible departures (denoted by the two heavy lines) all others are fixed departure services.

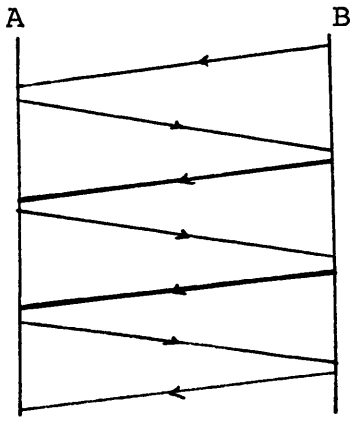


Fig. 2.18

If x_s is the chain consisting of all the flight arcs in the system then $x_s = 1$ is an optimal solution to ILP(2.4). It can be checked however that at least two aircraft are necessary to meet the schedule with a choice of either one of the two alternative departures. In order to guard against such pitfalls our column introduction scheme should introduce only chains which cover a service at most once. Therefore we would like to exclude the situation of Fig. 2.7b.

No computational experience with large systems can be reported since no code currently exists. However it seems that if in the future very large systems, which require more capacity than existing LP codes can provide, are to be solved this method will have to be used.

A Branch and Bound algorithm for this formulation would be based on inclusion and exclusion of particular flight arcs rather than on specific variables (chains). For example if x_s is fractional and covers flight arc i such that $i \in K_\ell$ then one branch out of the node from which branching is currently taking place will have a new schedule map in which all arcs $j \in K_\ell - \{i\}$ are excluded (by putting on them very high costs, for example) and the other node will have the arc i excluded.

2.13 Scheduling with Departure Preferences

Until now it was assumed that management is indifferent to the particular choice of a departure out of a set. However

it may indeed have preferences for some departures over others. In this action the model is extended to the case where the departures are ranked on a preference scale, and we may say that each departure has a 'preference index'. This index may simply be the number of passengers the airline expects to accomodate if the service is provided at the particular departure time. It is possible that the proximity of a competitive service will tap some of the potential passengers, or that the waiting time itself may influence the expected number of passengers. In any case it is assumed that management can assign a reasonable preference index to every departure.

Suppose that ILP(2.1) has been solved and the minimal fleet size required to meet the schedule is known. Out of all schedules that this fleet can accomplish we would like to pick the one which maximizes the sum of the preference indices. This optimization problem will now be formulated as an ILP. First define:

c_i = the preference index of the i-th departure

F = the max-flow obtained by solving ILP(2.1)

Then the Integer Linear Program is:

$$\text{(i)} \quad \max z^* = \sum_{\ell=1}^{N} \sum_{i \in K_\ell} c_i u_i$$

S.t.

$$\text{(ii)} \quad \sum_{i \in K_\ell} u_i = 1, \quad \sum_{i \in K_\ell} v_i = 1 \qquad , \forall \ell$$

$$\text{(iii)} \quad u_i = v_i \qquad , \forall\, i$$

(2.5)

$$\text{(iv)} \quad x_{si} \leq u_i \qquad , \forall\, i$$

$$\text{(v)} \quad x_{jt} \leq v_j \qquad , \forall\, j$$

$$\text{(vi)} \quad x_{si} = \sum_{j \in A(i)} x_{ij} \qquad , \forall\, i$$

$$\text{(vii)} \quad x_{jt} = \sum_{i \in B(j)} x_{ij} \qquad , \forall\, j$$

$$\text{(viii)} \quad \sum_{\ell=1}^{N} \sum_{i \in K_\ell} x_{si} = F$$

F can actually be chosen to be equal to any integer smaller than the maximal, for it may be profitable to use a larger fleet if the increase in revenues induced by a better schedule will justify it. Thus a parametric study may be conducted, with F as a variable, from which the most efficient point can be determined. The smallest F which has to be examined is the maximal flow obtained by selecting the best departure out of a bundle and solving for the maximal flow (which is a fixed schedule problem).

Since F is an integer and we know that there exists at least one all-integer feasible solution to the constraints of LP(2.5), the question which arises now is whether LP(2.5) will always have at least one optimal integer solution. At

present the author is able neither to prove this nor to provide a counter-example, since the problems that were solved had integer optimal solutions. Table 2.2 lists the preference indices for the example of Fig. 2.10. The maximal integer flow in that network was 6 units and LP(2.5) was solved for F = 6. The optimal solution is shown in Fig. 2.19. The reader can check that the more profitable departure out of each bundle is taken. Our large example (Fig. 2.13)

i	c_i	i	c_i	i	c_i	i	c_i
1	75	5	80	9	75	13	78
2	65	6	85	10	65	14	92
3	100	7	85	11	90	15	70
4	75	8	85	12	75	16	100

TABLE 2.2

was also solved with some c_i's which will not be shown here. But they were chosen so that the more 'central' departures out of the bundle are always the better ones. The result is shown in Fig. 2.14b, which can be compared with the rather random arrangement of Fig. 2.14a.

The use of the other two formulations, II and III, will now be briefly discussed. In formulation II the profit of variable x_{ij} is $\frac{1}{2}(c_i + c_j)$, of s_i is $\frac{1}{2}c_i$, and of t_j is $\frac{1}{2}c_j$. It can be easily verified that a solution to the ILP formulated below will properly account for the preference indices of all the

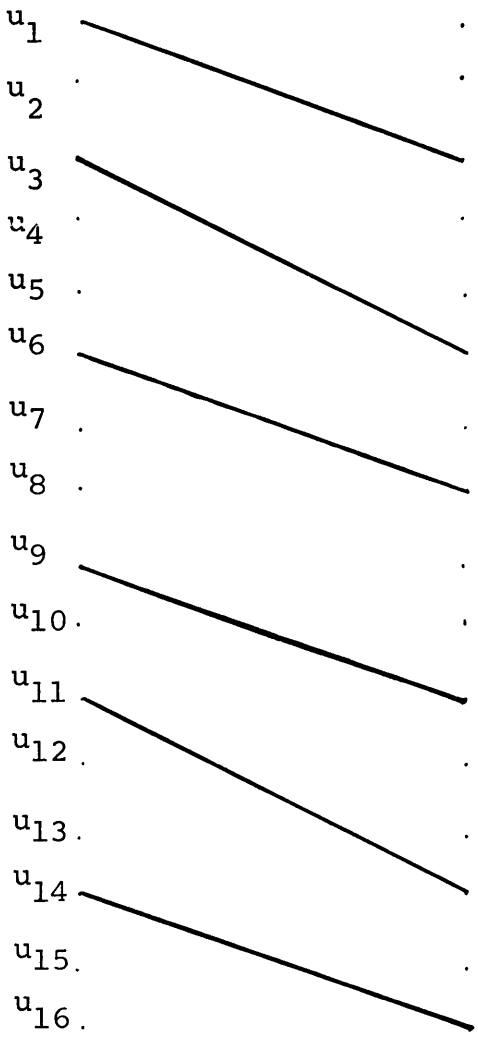


Fig. 2.19

departures chosen. The ILP is

$$\text{(i)} \quad \max \; z^* = \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} \tfrac{1}{2} c_i (s_i + t_i) + \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} \sum_{j \in A(i)} \tfrac{1}{2}(c_i + c_j)\, x_{ij}$$

S.t.

$$\text{(ii)} \quad \sum_{i \in K_\ell} (s_i + \sum_{j \in A(i)} x_{ij}) = 1 \qquad , \forall \ell$$

$$\text{(iii)} \quad \sum_{j \in K_\ell} (t_j + \sum_{i \in B(j)} x_{ij}) = 1 \qquad , \forall \ell \ni |K_\ell| = 1$$

(2.6)

$$\text{(iv)} \quad s_i + \sum_{j \in A(i)} x_{ij} + \sum_{j \in K_\ell - \{i\}} (t_j + \sum_{i \in B(j)} x_{ij}) = 1 \qquad , \forall\, i \in K_\ell$$

$$\text{(v)} \quad \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} \sum_{j \in A(i)} x_{ij} = F \qquad , \forall \ell \ni |K_\ell| > 1$$

$$s_i, t_j, x_{ij} = 0,1 \qquad , \forall\, i,j$$

In the case for which $c_i = 0 \; \forall$ i and we define the costs $-c_{ij}$ for all x_{ij} to be equal to $T - t_{ij}$ for reasons discussed in section 2.5 we solve for an 'improved' schedule in the sense discussed in that section. In general, if c_{ij} can be meaningfully defined to carry a proper weight compared to

the c_i's we may have a cost of $\frac{1}{2}(c_i + c_j) - c_{ij}$ for x_{ij}.

For the fourth formulation d_j will be defined to be the sum of all the preference indices on the chain of departures which x_j covers. Consequently the Integer Linear Program is:

$$\text{(i)} \quad \text{maximize } z^* = \sum_{j=1}^{n} d_j x_j$$

S.t.

$$\text{(2.7)} \quad \text{(ii)} \quad \sum_{j=1}^{n} a_{\ell j} x_j = 1 \qquad \ell = 1, \ldots, |N|$$

$$\text{(iii)} \quad \sum_{j=1}^{n} x_j = F$$

$$x_j = 0, 1 \qquad \forall\, j$$

Here also a shortest route algorithm can be used to determine the chain to be introduced into the basis. The simplex multipliers are $\underline{\pi} = \underline{d}_B B^{-1}$ where $\underline{d}_B$ is the vector of the costs of the basic variables. A chain x_j may be introduced into the basis if

$$\sum_{\ell=1}^{|N|+1} \pi_\ell a_{\ell j} - d_j < 0.$$

Hence define the following lengths on the flight arcs:

$$\pi_\ell - c_i \quad \forall \quad i \in K_\ell \quad , \forall\, \ell$$

Then a shortest route algorithm can be used to determine the next chain to be introduced, or that the present solution is optimal if

$$\sum_{\ell=1}^{|N|} \pi_\ell a_{\ell j} - d_j + \pi_{|N|+1} \geq 0$$

for the shortest route.

2.14 Scheduling with Direct Fleet Costs

Until now the operating costs of the fleet were not explicitly introduced into the model. Rather, it was pointed out that management can conduct a parametric study with the variable F taking successively lower integer values. This approach is suitable for the case in which the cost of maintaining the fleet is not a linear function of its size. For example, it is likely that economies of scale are significant and the dependence of the cost of maintaining the fleet on its size looks like the curve of Fig. 2.20a. However, if it is assumed that a linear relation (Fig. 2.20b) is adequate, or that an existing system has already taken care of

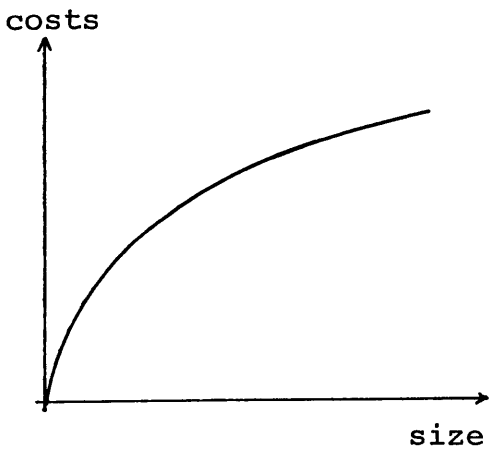


Fig. 2.20a

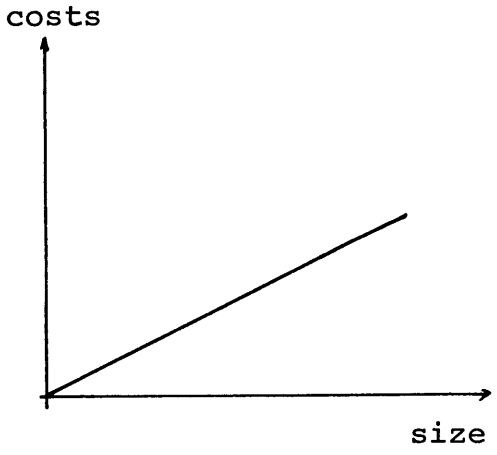


Fig. 2.20b

the initial high expense and now we are concerned with an almost linear region of the relation, the explicit constraint (viii) of ILP(2.5) can be dropped and direct operating costs introduced instead. The ILP which will be defined shortly will determine a fleet size optimally balanced with depar-

ture profits. Let c_o be the cost per period of operating an additional aircraft. Since the number of aircrafts is

$|N| - \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} x_{si}$, the total cost of operating the fleet is

$$c_o(|N| - \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} x_{si})$$

From which the profit of operating the system is

$$\sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} c_i u_i - c_o(|N| - \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} x_{si})$$

$c_o|N|$ being a constant, the ILP is:

$$\text{(i)} \quad \max z^+ = \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} (c_i u_i + c_o x_{si})$$

S.t.

$$\text{(ii)} \quad \sum_{i \in K_\ell} u_i = 1, \quad \sum_{i \in K_\ell} v_i = 1 \qquad , \forall \ell$$

$$\text{(iii)} \quad u_i = v_i \qquad , \forall i$$

(2.8)

$$\text{(iv)} \quad x_{si} \leq u_i \qquad , \forall i$$

$$\text{(v)} \quad x_{jt} \leq v_j \qquad , \forall j$$

$$\text{(vi)} \quad x_{si} = \sum_{j \in A(i)} x_{ij} \qquad , \forall i$$

$$\text{(vii)} \quad x_{jt} = \sum_{i \in B(j)} x_{ij} \qquad , \forall j$$

(all variables 0-1)

When c_o is very high the optimal solution to ILP(2.8) will give the maximal flow in the network. Consequently, by a previous counterexample, LP(2.8) may have non-integer optimal solutions. If c_o is low the fleet is inexpensive to operate, and the best departure will be taken out of each service. Since this fixes the schedule LP(2.8) will have at least one optimal integer solution. The Branch and Bound algorithm which was defined for ILP(2.1) can be used for ILP(2.8) without any changes, since when all u_i's are fixed (at 0,1) the problem becomes one of maximizing the flow in a 'usual' network.

The ILP for formulation II takes the form:

$$\text{(i)}\quad \max z^+ = \sum_{\ell=1}^{|N|} \sum_{i\in K_\ell} \tfrac{1}{2} c_i (s_i + t_i) + \sum_{\ell=1}^{|N|} \sum_{i\in K} \sum_{j\in A(i)} \tfrac{1}{2}(c_i + c_j + c_o)\, x_{ij}$$

S.t.

$$\text{(ii)}\quad \sum_{i\in K_\ell} (s_i + \sum_{j\in A(i)} x_{ij}) = 1 \qquad , \forall\, \ell$$

$$\text{(iii)}\quad \sum_{j\in K_\ell} (t_j + \sum_{i\in B(j)} x_{ij}) = 1 \qquad , \forall\, \ell \ni |K_\ell| = 1$$

(2.9)

$$\text{(iv)}\quad s_i + \sum_{j\in A(i)} x_{ij} + \sum_{j\in K_\ell - \{i\}} (t_j + \sum_{i\in B(j)} x_{ij}) = 1 \qquad , \forall\, i\in K_\ell$$

$$, \forall\, \ell \ni |K_\ell| > 1$$

$$s_i, t_j, x_{ij} = 0,1 \qquad , \forall\, i, j$$

2.15 Concluding Remarks

In this chapter a basic decision model for a single fleet air transportation system was discussed. The fixed schedule model (for which solution methods have already appeared in literature) was introduced as the basis for the following extensions. Next it was assumed that departures are variable within a given 'bandwidth' for each service. The departures are to be fixed so that the size of the fleet which will provide the services is minimal. Four different formulations of this problem were discussed. Solution procedures by the methods suggested in this chapter can be carried out with available LP codes for the first three. The last one, the Arc-Chain formulation, can handle considerably larger systems but no computer code is available. The ILP's which constitute the formulations of the problem exhibit the interesting and welcome property that quite often the corresponding LP's have optimal integer solutions. In case they do not then effective Branch and Bound procedures, by which the optimal integer solutions can be obtained, were formulated. Next the concept of 'preference index' was introduced and the problem of scheduling and routing the minimal fleet was formulated as an ILP and computational results were discussed. Finally, the problem of scheduling with direct fleet costs rather than with the minimal fleet constraint was suggested and formulated as an ILP. Additional computational experience will be presented in the next chapter. Experience with very large systems will have to be obtained in future work.

CHAPTER III

THE SINGLE FLEET: EXTENSIONS

3.1 Introduction

In this short chapter some extensions to the basic Single Fleet Problem will be discussed. The formulations of the problems presented in this work as Integer Linear Programs is of great advantage, since by the introduction of but a few constraints the models can be extended to include many of the real life constraints imposed on the system. We shall be more specific about it in the following section. It is also possible to allow optional services which will be taken only if they contribute to the profitability of the system. Thus management may have higher flexibility in committing itself to specific services. Section 3.3 will include a discussion of total system design, in which it is not committed to the inclusion of any services at all. This chapter is devoted to model formulations and to the presentation of the results of computational experiments.

3.2 Restricted Service Frequencies

In Chapter II it was assumed that the airline intends to include each service on the schedule map. It is possible however that it may want to consider alternative services from station A to station B from which only a subset will be selected. Specifically, the airline plans to have n_{AB} services from A to B, n_{BA} services from B to A etc. The airline may have n_{AB} set for it by regulating agencies. Another problem

is one in which an airline is permitted to fly only n_A flights in and out of A. Management may want to have a higher frequency at A but is restricted (perhaps due to traffic congestion at the airport). In both cases the schedule map will contain more services than will eventually be flown, and we add some constraints to the ILP's of Chapter II. The assumption here is that the system operates in a competitive environment and that consequently, whether a service is or is not included in the resulting system has no effect on the other services in the system, because the competitive carriers absorb the demand projected for the service which is not included.

In order to formulate the Integer Linear Programs we need the following definitions:

$$I_q = \{ \ell \mid K_\ell \text{ is an inbound service to station } q \}$$

$$\emptyset_q = \{ \ell \mid K_\ell \text{ is an outbound service from station } q \}$$

$$L_{pq} = \{ \ell \mid K_\ell \text{ is a service from } p \text{ to } q \}$$

We shall now formulate the ILP for the case in which the transportation system is restricted as to the number of services it can operate at each station. We assume that the first objective of the management is to select the services so that the total fleet size is minimal. Then using the minimal form of formulation I of the last chapter the ILP is:

$$\text{(i)} \quad \max z = \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} \sum_{j \in A(i)} x_{ij}$$

S.t.

$$\text{(ii)} \quad \sum_{i \in K_\ell} u_i \leq 1 \qquad , \forall \ell$$

$$\text{(iii)} \quad u_i \geq \sum_{j \in A(i)} x_{ij} \qquad , \forall i$$

$$\text{(3.1) (iv)} \quad u_j \geq \sum_{i \in B(j)} x_{ij} \qquad , \forall j$$

$$\text{(v)} \quad \sum_{\ell \in I_q} \sum_{i \in K_\ell} u_i = n_q \qquad , \forall q$$

$$\text{(vi)} \quad \sum_{\ell \in \emptyset_q} \sum_{i \in K_\ell} u_i = n_q \qquad , \forall q$$

(all variables 0,1)

Constraints (v) and (vi) are the ones which impose the frequency restrictions at each station. After the minimal fleet size has been found we can formulate the problem which will select the best schedule for this fleet, as was done in Chapter II.

We shall now formulate the ILP for the case in which the management would like to limit the number of services between some or all of the station pairs. Again, we are first interested in the selection which will make the total fleet size minimal. The ILP is:

$$\text{(i)} \qquad \max z = \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} \sum_{j \in A(i)} x_{ij}$$

S.t.

$$\text{(ii)} \qquad \sum_{i \in K_\ell} u_i \leq 1 \qquad , \forall \ell$$

$$\text{(iii)} \qquad u_i \geq \sum_{j \in A(i)} x_{ij} \qquad , \forall i$$

$$\text{(3.2)} \quad \text{(iv)} \qquad u_j \geq \sum_{i \in B(j)} x_{ij} \qquad , \forall j$$

$$\text{(v)} \qquad \sum_{\ell \in L_{pq}} \sum_{i \in K_\ell} u_i = n_{pq} \qquad , \forall (p,q)$$

(All variables 0,1)

The example of Fig. 2.10 is a special case for ILP(2.1) and ILP(3.2) hence the solutions to LP(3.1) and to LP(3.2) are not necessarily integer.

Fig. 3.1 is an example of a shuttle system among 6 cities. There is a total of 54 possible services in the system with three possible departures for each one of them. The system was solved for the minimal fleet size for two cases:

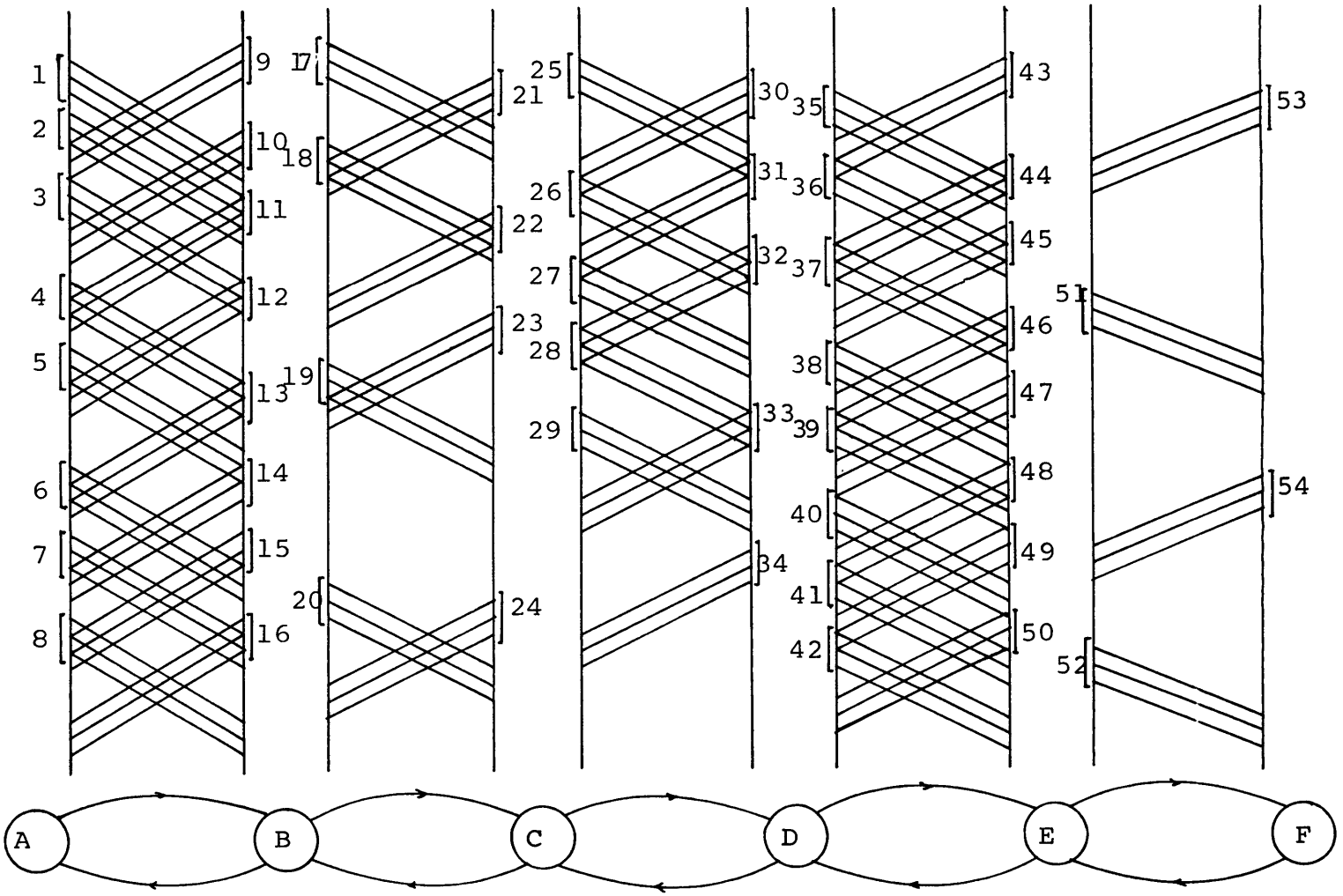

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
A
B
C
D
E
F

Fig. 3.1

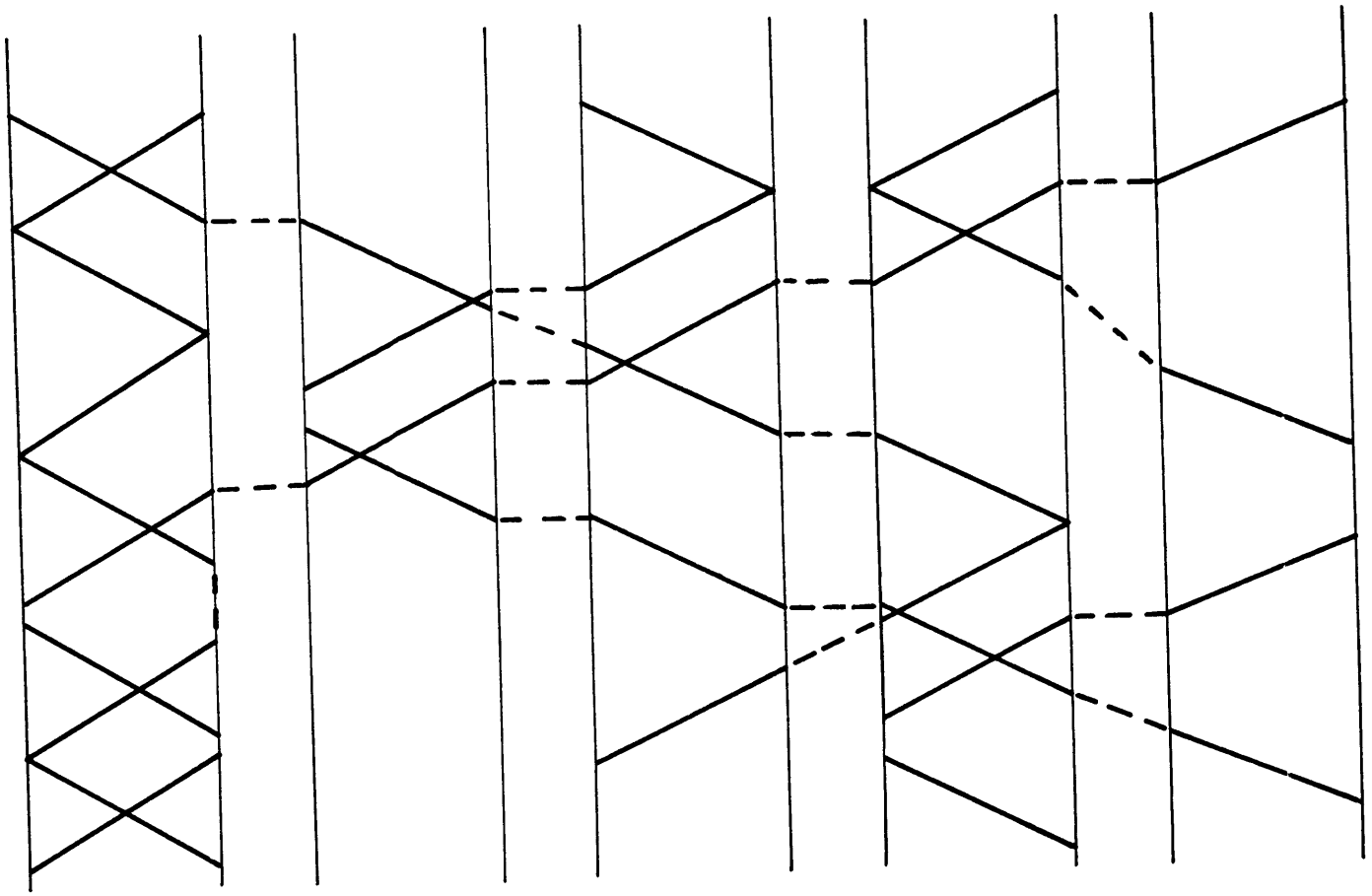


Fig. 3.2

	n_{AB}, n_{BA}	n_{BC}, n_{CB}	n_{CD}, n_{DC}	n_{DE}, n_{ED}	n_{EF}, n_{FE}
Case 1.	8	4	5	8	2
Case 2.	5	2	3	4	2

Table 3.1

The first case is the min-fleet problem (of Chapter II) for the entire system. LP(3.2) gave an integer optimal solution for both cases. Eleven aircraft are needed for the system of case 1 and 5 aircraft for case 2. The resulting schedule for the latter is shown in Fig. 3.2. The connections (x_{ij}'s) are also indicated in this figure by broken lines and the chains which will be followed by each of the five aircraft can be easily traced.

In Chapter II we discussed the procedure for scheduling with preference indices and here we proceed similarly. Subject to the constraint of minimal fleet size management would like to pick the schedule which will bring maximal (projected) revenue. After ILP(3.2) is solved with max-flow of F as the optimal objective, c_i is assigned as a preference index for the i-th departure, and the ILP is:

$$\text{(i)} \qquad \max z = \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} c_i u_i$$

S.t.

$$\text{(ii)} \qquad \sum_{i \in K_\ell} u_i \leq 1 \qquad ,\forall \ell$$

$$\text{(iii)} \quad u_i \geq \sum_{j \in A(i)} x_{ij} \quad , \forall\, i$$

(3.3)

$$\text{(iv)} \quad u_j \geq \sum_{i \in B(j)} x_{ij} \quad , \forall\, j$$

$$\text{(v)} \quad \sum_{\ell \in L_{pq}} \sum_{i \in K_\ell} u_i = n_{pq} \quad , \forall (p,q)$$

$$\text{(vi)} \quad \sum_{\ell = 1}^{|N|} \sum_{i \in K_\ell} \sum_{j \in A(i)} x_{ij} = F$$

(all variables 0,1)

In all cases tested by the author LP(3.3) had an optimal integer solution. For example, for case 2 of Table 3.1, after ILP(3.2) was solved so was ILP(3.3) for a set of preference indices which were randomly selected and ranged from -20 to 80 (Table 3.2). The solution is shown in Fig. 3.3. Notice that although the same number of aircraft is used in both this solution and Fig. 3.2, the schedules are quite different. As a general observation it is true that in any system there are quite a few possible schedules which will use the minimal number of aircraft.

Service No.	c_i	Service No.	c_i	Service No.	c_i
1	30	19	20	37	10
	60		40		10
	0		60		60
2	50	20	50	38	50
	70		30		20
	20		20		70
3	-10	21	10	39	-10
	30		30		50
	60		20		30
4	70	22	40	40	0
	10		40		50
	-10		30		40
5	30	23	70	41	60
	-20		50		-20
	10		80		50
6	40	24	50	42	70
	80		40		20
	40		-10		0
7	20	25	40	43	20
	10		40		50
	50		60		80
8	20	26	70	44	-10
	10		40		60
	-10		-10		20
9	20	27	40	45	70
	60		0		30
	70		20		20
10	0	28	60	46	60
	60		20		50
	40		70		70
11	10	29	30	47	0
	60		20		-20
	-10		0		40
12	40	30	40	48	60
	10		10		50
	70		60		30

Service No.	c_i	Service No.	c_i	Service No.	c_i
	-10		20		50
13	40	31	20	49	20
	60		40		70
	-20		-10		60
14	40	32	20	50	50
	40		70		40
	-10		30		70
15	20	33	40	51	-10
	70		-20		20
	80		70		30
16	50	34	10	52	-10
	0		40		40
	40		80		20
17	30	35	50	53	70
	10		10		40
	20		-20		20
18	70	36	60	54	50
	50		30		60

TABLE 3.2

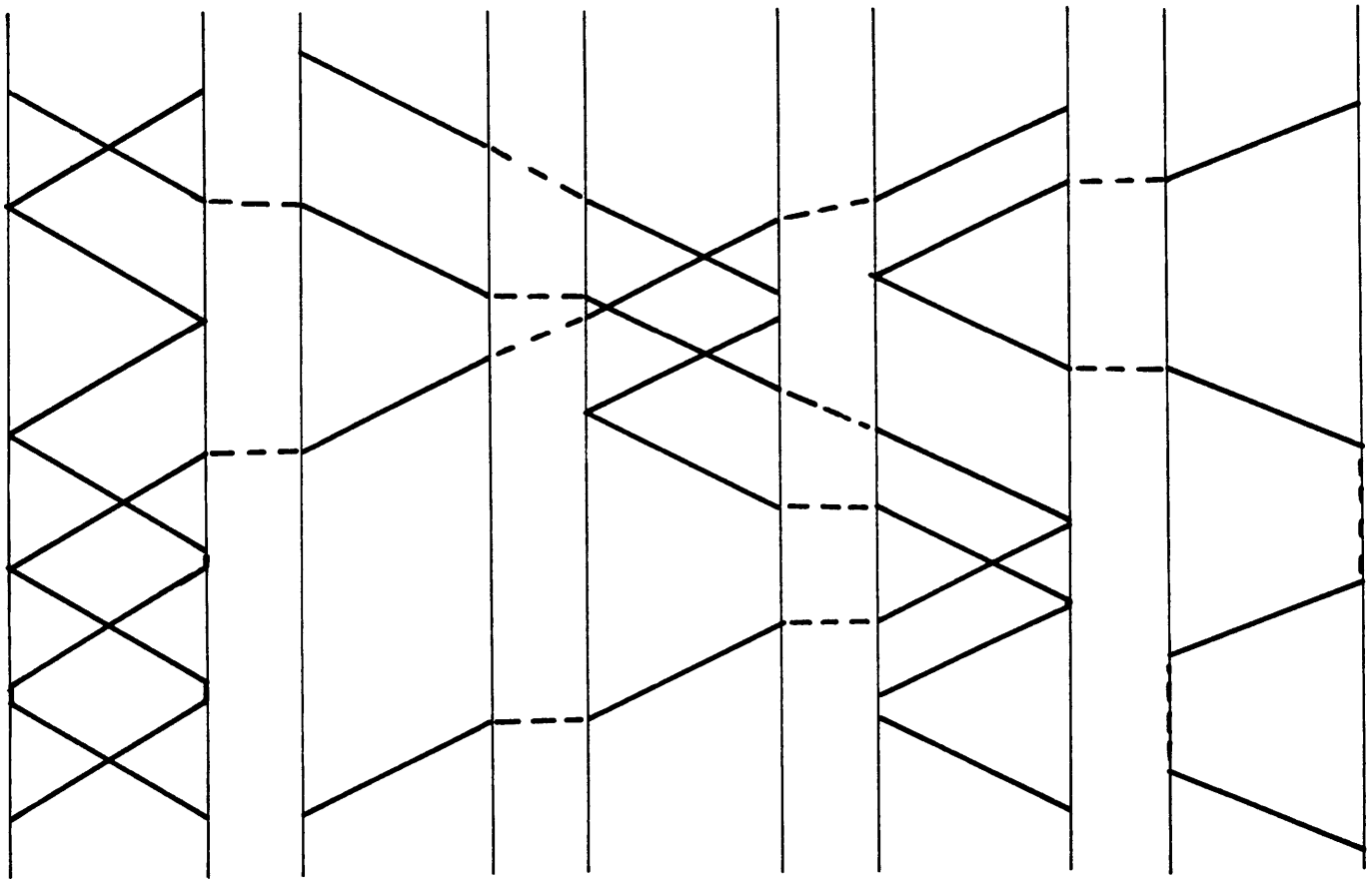


Fig. 3.3

3.3 System Design

In what we term 'System Design' we mean that management is not a priori committed to any number of services at all between any two stations or to frequency of service at any station. We can assume that the transportation system has a number of aircraft available which must be used in the system, or that it has none and would like to determine the size of the fleet which will operate profitably.

Assume now that we are dealing with the second case. The number of services which will constitute the system is

$$\sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} u_i$$

where N is the set of services considered. The size of the fleet will be

$$\sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} u_i - \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} \sum_{j \in A(i)} x_{ij}$$

If c_i is the profit of the i-th departure and assuming linear costs for an aircraft over the period which is equal to c_o (see section 2.15), we get the expense of operating the fleet over the period:

$$c_o \left(\sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} u_i - \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} \sum_{j \in A(i)} x_{ij} \right)$$

The ILP is:

$$\text{(i)}\quad \max\ z = \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} (c_i - c_o)u_i + \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} \sum_{j \in A(i)} c_o\, x_{ij}$$

S.t.

$$\text{(ii)}\quad \sum_{i \in K_\ell} u_i \leq 1 \qquad , \forall\, \ell$$

(3.4)

$$\text{(iii)}\quad u_i \geq \sum_{j \in A(i)} x_{ij} \qquad , \forall\, i$$

$$\text{(iv)}\quad u_j \geq \sum_{i \in B(j)} x_{ij} \qquad , \forall\, j$$

$$\text{(v)}\quad \sum_{\ell \in I_q} \sum_{i \in K_\ell} u_i = \sum_{\ell \in \emptyset_q} \sum_{i \in K_\ell} u_i \qquad , \forall\, q$$

(all variables 0,1)

Constraints (v) specify that in the optimal system there will be an equal number of arrivals and departures at each station. This conditions was always assumed to hold in the models discussed thus far. It guarantees that there will be conservation of aircraft flow in the system; i.e. there will be no net accumulation of aircraft at any station after a period of operation.

In general LP(3.4) has non-integer optimal solutions. The Branch and Bound algorithm of section 2.7 can be applied here to obtain the optimal integer solution. Consider, for

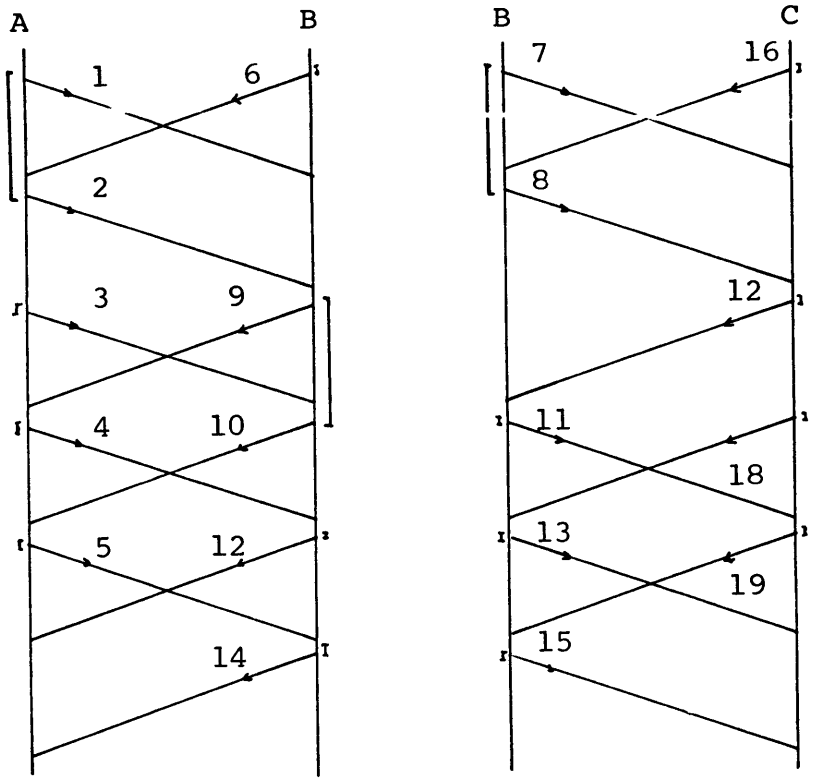


Fig. 3.4a

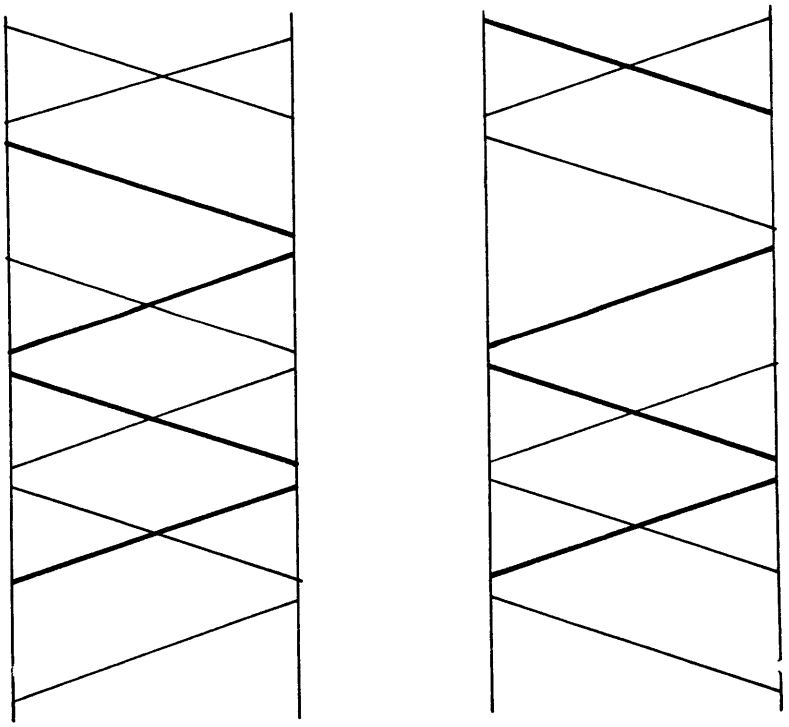


Fig. 3.4b

example, the three-station problem of Fig. 3.4a. The data for this system is given in Table 3.3, where the optimal solution to LP(3.4) is shown too.

Depart. No.	c_i	u_i	Depart. No.	c_i	u_i
1	30	2/3	11	3	0
2	30	1/3	12	20	2/3
3	2	0	13	2	0
4	15	1/3	14	6	0
5	10	2/3	15	2	0
6	14	1/3	16	10	0
7	25	1/3	17	14	1
8	18	2/3	18	12	0
9	15	1/3	19	3	0
10	10	2/3			

$c_o = 40$

TABLE 3.3

The solution tree is shown in Fig. 3.5 where all the terminal nodes represent integer solutions. The optimal integer solution is shown in Fig. 3.4b (by the heavy lined services).

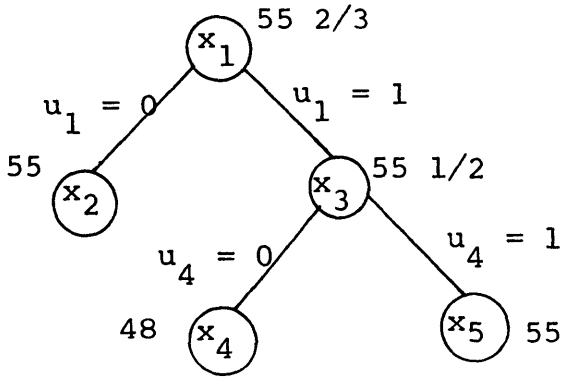


Fig. 3.5

In experiments with 'large'systems the optimal solutions obtained were all-integer. For example, the system of Fig. 3.1 was solved with the data of Table 3.2 and with $c_o = 300$. The optimal solution is shown in Fig. 3.6. The fleet size for the optimal system is 5.

Assume now that p aircraft which management would like to use in the system are available or that it plans on having a fleet of this size. It then wants to fix the schedule which will bring the maximal profit with p aircraft in operation. The ILP for this case is:

$$\text{(i)} \quad \max \quad z = \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} c_i u_i$$

S.t.

$$\text{(ii)} \quad \sum_{i \in K_\ell} u_i \leq 1 \qquad , \forall \ell$$

$$\text{(iii)} \quad u_i \geq \sum_{j \in A(i)} x_{ij} \qquad , \forall i$$

(3.5)

$$\text{(iv)} \quad u_j \geq \sum_{i \in B(j)} x_{ij} \qquad , \forall j$$

$$\text{(v)} \quad \sum_{\ell \in I_q} \sum_{i \in K_\ell} u_i = \sum_{\ell \in \emptyset_q} \sum_{i \in K_\ell} u_i \qquad , \forall q$$

$$\text{(vi)} \quad \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} u_i - \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} \sum_{j \in A(i)} x_{ij} = p$$

(all variables 0,1)

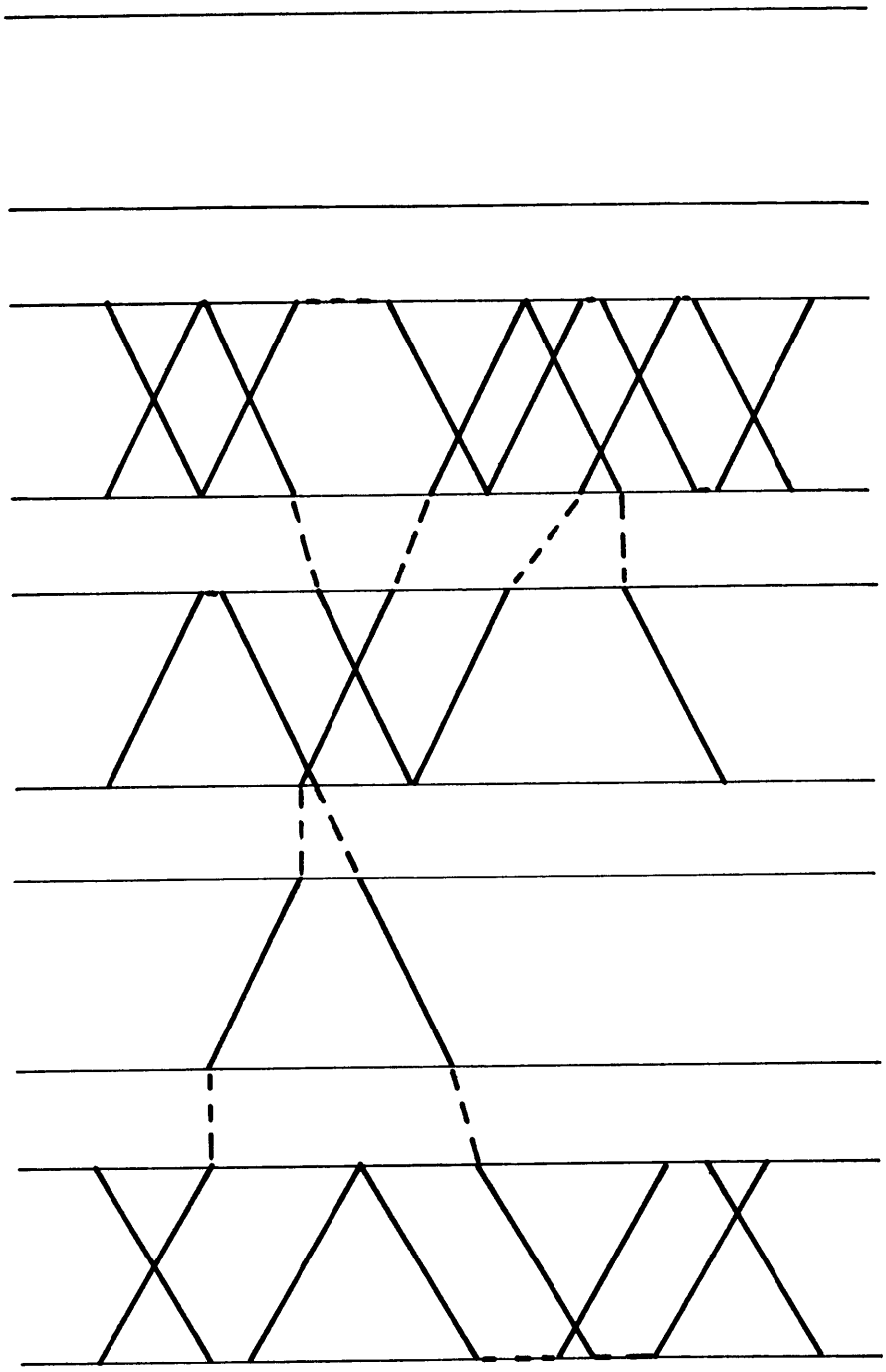


Fig. 3.6

As the reader can verify, it is the last constraint which imposes the fleet-size requirement. No computational experiments were conducted with ILP(3.5).

3.4 Concluding Remarks

In this chapter we discussed some model extensions, together with computational experience. Simplex algorithm times (using MPS/360) for all the problems discussed were below 1 minute. Any combination of the constraints which were discussed can, of course, be included in any particular model. The other formulations of Chapter II were not included because they would simply be repetitious. Formulation II could be profitably used here also to reduce the number of constraints. In the next chapter we extend the discussion to a transportation system which operates more than one type of aircraft.

CHAPTER IV
THE MULTI-FLEET PROBLEM

4.1 Introduction

In Chapter II it was assumed that the transportation system uses only one type of aircraft or, alternatively, that the subsystem consisting of a single type was considered. All large air transportation systems do indeed use several types of aircraft, and in the present chapter we shall address ourselves to some optimization problems connected with the operation of a multi-fleet system. The reason for operating a multi-fleet is, of course, the difference in operating costs between different types of aircraft which have different capacities. Thus it may be profitable to use a low capacity aircraft for a low demand service while it may not be so if a high capacity aircraft is used.

Section 4.2 includes some preliminaries and definitions which will be used in the following sections. In section 4.3 a 'system decomposition problem' is formulated. This section provides a tool for decomposing the transportation system (in some optimal manner) into subsystems each consisting of a unique aircraft type.

Sections 4.4-4.6 also present various formulations of the system decomposition problem but with some additional constraints and operating costs, which at the same time introduce computational difficulties like matrix size problems and integrality of the optimal solutions. An Arc-Chain formulation of some of the models of this chapter is discussed in section 4.7.

4.2 Preliminary Discussion and Definitions

It was stated in the introduction to this chapter that the aircraft types differ mainly in capacity and in operating costs. Lower capacity aircraft have lower operating costs and therefore it may be costly to assign a high capacity airplane to an (expectedly) low demand service. On the other hand a potentially large number of passengers will have to be rejected if a low capacity aircraft is assigned to a high demand service. It may still be a sound decision, however, to assign a small aircraft if demand exceeds capacity only slightly, when the difference is not sufficiently high to balance the higher operating costs of the larger aircraft. We shall restrict the following discussions to a two-fleet system, but generalizations to an n-fleet system can be easily made.

In order to define suitable Integer Linear Programs for the models, we have to transform the discussion of the last paragraph into a more quantitative format. The operating costs of an aircraft type will be expressed in terms of a 'breakeven point' which is the number of passengers required to just make the operation of the aircraft for a particular service profitable. There will be two preference indices for each departure, one for each of the vehicle types. The preference index of type-1 aircraft for a given departure will be obtained by subtracting the breakeven point from the number of passengers it will carry if the departure is taken multiplied by the fare. This procedure looks reasonable enough but, of course, any number of factors can be included in the computation of the preference index.

When the entire system is being divided into subsystems according to the aircraft types we have to impose the constraint that there must be conservation of 'vehicle flow' in each subsystem. Namely, that there are as many arrivals as there are departures at each station of each subsystem. Otherwise there will be an accumulation of aircraft in some stations. This constraint is superfluous in the single fleet case since it is automatically satisfied.

The schedule map will now have two copies, one for each aircraft type. Superscripts will distinguish variables and sets. Fig. 4.1 is the two-copy version of the schedule map of Fig. 2.7a. Out of the total of six departures which correspond to service number 1 (1^1 and 1^2) exactly one will be selected. If the one selected belongs to 1^1 the service will be taken by a type-1 aircraft. If it belongs to 1^2 a type-2 aircraft will be assigned to service no. 1.

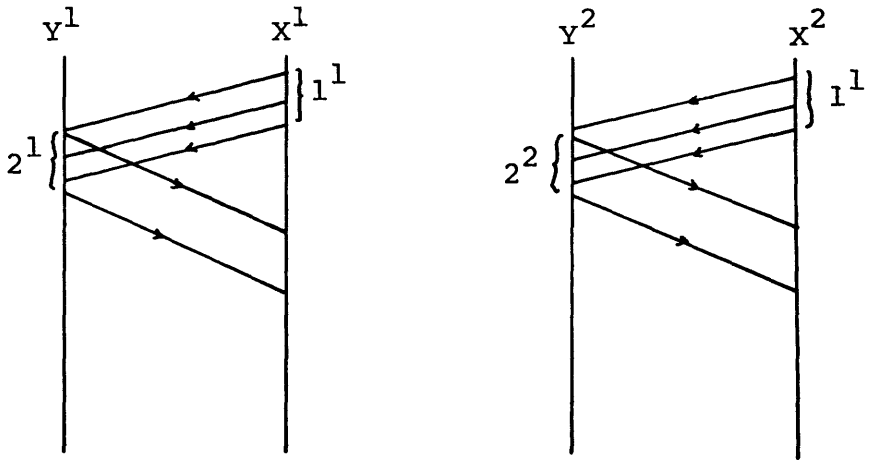


Fig. 4.1

The bipartite network corresponding to a two-fleet problem will have two copies, joined together at the nodes s and t (Fig. 4.2). This network will be denoted by $G^* = [S^1, S^2, T^1, T^2; A^1, A^2]$. Before proceeding we define some sets and variables for G* which will be used in the formulations of the optimization problems in the rest of this chapter. We shall number the stations sequentially and use q as an index for this sequence.

Define the following sets:

$$A_m(i) = \{ j \mid (s_i^m, t_j^m) \in A^m \}$$

$$B_m(j) = \{ i \mid (s_i^m, t_j^m) \in A^m \} \qquad (m = 1,2)$$

$$I_q^m = \{ \ell \mid K_\ell^m \text{ is an inbound service to station } q \}$$

$$\emptyset_q^m = \{ \ell \mid K_\ell^m \text{ is an outbound service from station } q \}$$

(Subsequently we shall drop the superscript from K_ℓ^m and assume that we have two <u>exact</u> copies, namely, that each copy has as many alternative departures for each service as the other).

And the following variables:

x_{si}^m = the flow in (s, s_i^m)

x_{ij}^m = the flow in (s_i^m, t_j^m)

x_{jt}^m = the flow in (t_j^m, t) $\qquad (m = 1,2)$

u_i^m = the capacity of (s, s_i^m)

v_j^m = the capacity of (t_j^m, t)

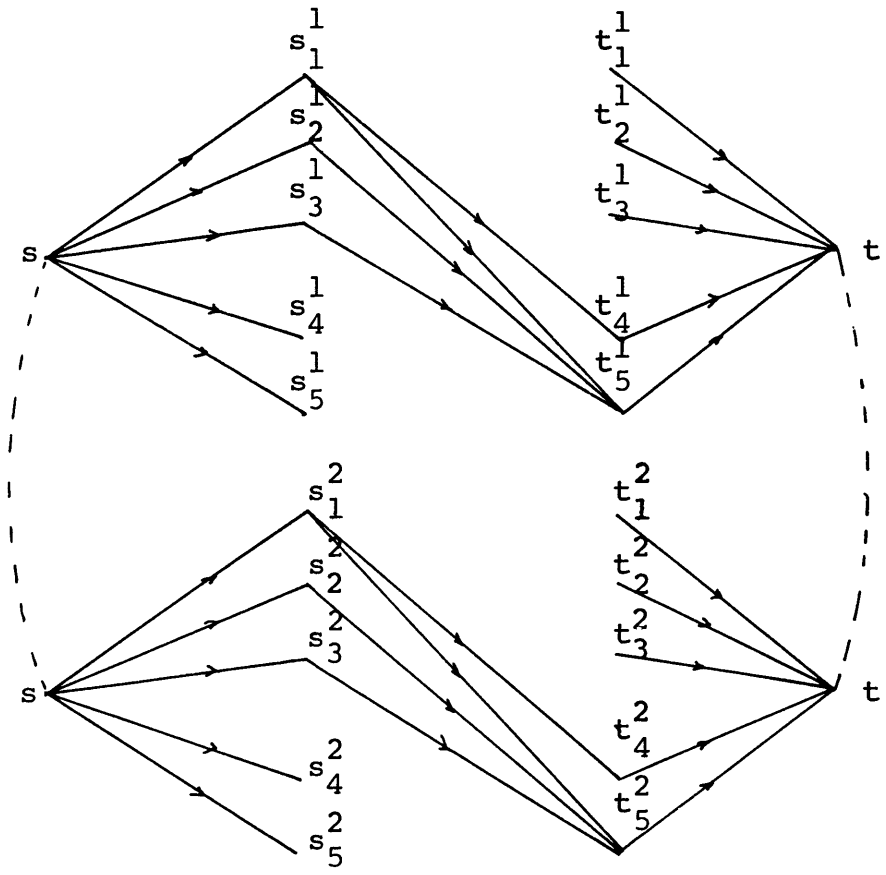


Fig. 4.2

Note that our assumption of two copies implicitly states that the two types fly at the same speed. This assumption is not crucial, however, and the discussion can be generalized to the case where only one-to-one correspondences is defined between services of the two copies.

The sizes of the Integer Linear Programs defined in this chapter will be larger than the ones defined in Chapter II. They may indeed be too large to handle by present day systems. There may be an advantage (or a necessity) to decompose the system into its subsystem in some optimal manner and then

treat each one of them separately by the methods of Chapter II. The system decomposition problem which will now be formulated is only partially satisfactory. Although the optimally decomposed system will bring maximal revenue, no fleet size considerations will be incorporated in this formulation. Namely, the total fleet size required (including both types) may be larger than the one that would have been obtained had the entire system been optimized with the minimal total fleet size requirement as a constraint. This is not to say that the 'total' optimization is always better than the decomposition-first method. Each subsystem will be optimized with respect to its own fleet size and the total fleet size thus obtained may well be the minimal one anyhow, or that the total revenue is sufficient to justify a larger fleet.

4.3 Decomposition of the Air Transportation System

Let c_i^m be the preference index of the i-th departure if it is taken by a type m (m = 1,2) aircraft. As argued in the last section in general $c_i^1 \neq c_i^2$. Then the following ILP is a formulation of the system decomposition problem:

$$\text{(i)} \quad \max z = \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} (c_i^1 u_i^1 + c_i^2 u_i^2)$$

S.t.

$$\text{(4.1) (ii)} \quad \sum_{i \in K_\ell} (u_i^1 + u_i^2) = 1 \qquad , \forall \ell$$

$$\text{(iii)} \quad \sum_{\ell \in I_q^1} \sum_{i \in K_\ell} u_i^1 = \sum_{\ell \in \emptyset_q^2} \sum_{i \in K_\ell} u_i^1 \qquad , \forall q$$

$$u_i^1, u_i^2 = 0,1 \qquad , \forall i$$

Constraints (ii) specify that the ℓ-th service will be taken by exactly one of the two types. Constraints (iii) are the conservation of aircraft flow equations for type-1 aircraft. Similar constraints for type-2 will be automatically satisfied since the model specifies that there are as many arrivals as there are departures at each station on the schedule map. The objective function is self-explanatory.

LP(4.1) has a 'small' number of constraints as the reader can check. However, what is equally important is that it has at least one optimal integer solution. This can be argued in a few ways, but in order to provide a formal proof a preliminary discussion is necessary.

In section 2.2 the notion of cycle was defined. A circuit is defined to be a cycle $x_o,(x_o,x_1),x_1,\ldots,x_n,(x_n,x_o),x_o$ for which $x_i = x_j$ for any i and j is allowed, i.e. it is permitted to visit a node more than once when a circuit is traversed. An Euler circuit is a circuit whose arcs are traversed only once.

The outward degree of a node x_i is the number of arcs which originate from x_i. The inward degree of x_i is the number of arcs terminating there. An Euler Graph is a graph for which the inward and outward degrees are equal for all of the nodes. In [2] it is proved that there exists an Euler circuit (or a <u>set</u> of Euler circuits if the graph is disconnected) which includes all the arcs of an Euler graph.

We now define a 'station map' to be a graph in which the stations of the transportation system are represented as nodes and each service, regardless of how many alternative departures are considered, as a single arc. Fig. 4.3 is the station map of for the system of Fig. 2.10.

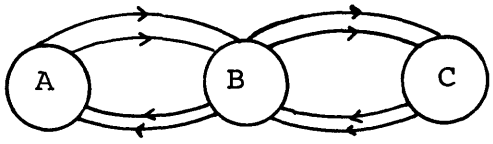


Fig. 4.3

For the two-fleet problem there will be two copies to the station map: one for each subsystem. The station map is an Euler Graph since we have assumed that there is an equal number of arrivals and departures at each station over the period.

Let $V = \{ V_1, V_2, \ldots \}$ be the set of feasible integer solutions to LP(4.1) where $V_i = \{ u_1^1, u_1^2; u_2^1, u_2^2; \ldots \}$ and $u_i^1, u_i^2 = 0,1$. (This set is not empty since, for example if $u_i^2 = 0 \; \forall$ i (iii) of LP(4.1) is automatically satisfied and (ii) has at least one integer solution: one arbitrary departure out of each service.) Let V_o be an arbitrary, not necessarily integer, feasible solution to LP(4.1). If u_i^m is interpreted as a <u>flow</u> in the m-th copy in the i-th departure arc, then V_o describes a circulatory flow in the two copies having a value of $\sum_{i \in K_\ell} u_i^m$ in the arc which represents the ℓ-th service in the m-th station map. It is not important, for the current discussion, to distinguish among the alternative departures of service ℓ .

Each member of V specifies integer valued circulations in the two copies. In other words, to each $V_i \in V$ corresponds

a selection of Euler circuits such that each service is covered exactly once: either in the first or in the second copy.

Let $G^+ = [N; A]$ be a sourceless-sinkless network and define:

u_{ij} = the upper capacity of arc $(i,j) \in A$

ℓ_{ij} = the lower capacity of arc $(i,j) \in A$

f_{ij} = the flow in arc $(i,j) \in A$

$(I,\bar{I})$ = any cut in G^+ $(I \bigcup \bar{I} = N,\ I \bigcap \bar{I} = \Phi)$ consisting of the arcs $(i,j) \ni i \in I,\ j \in \bar{I}$

Theorem 4.1 Let u_{ij}, ℓ_{ij} be integer $\forall (i,j)$, then a necessary and sufficient condition that an integer feasible flow exists in G^+ is that

$$\sum_{(I,\bar{I})} u_{ij} \geqslant \sum_{(\bar{I},I)} \ell_{ij}$$

holds for all $I \subseteq N$.

Proof: See Theorem 11.1 and the infeasibility conditions of the 'out of kilter' algorithm of section III.11 in [7].

Lemma 4.1 Let $u_{ij} = 1$, $\ell_{ij} = 0$ $\forall (i,j) \in A$ of G^+. And let $f_{ij} > 0$, not necessarily integer, $\forall (i,j) \in A$ be a feasible flow. Then there exists an Euler circuit (or a set of Euler circuits) in G^+ which includes all the saturated arcs of A.

Proof: Let $A_s \subseteq A$ be the set of saturated arcs and redefine $\ell_{ij} = 1 \ \forall \ (i,j) \in A_s$. Let $(I, \bar{I})$ be an arbitrary cut in G^+. In the redefined network the flow is still feasible, and since the net flow across any cut is 0 it must be true that

$$\sum_{(I,\bar{I})} u_{ij} \geq \sum_{(\bar{I},I)} \ell_{ij}$$

simply by accounting for all flow across the cut.
By theorem 4.1 an integer feasible circulation exists in the redefined network G^+. Then construct such a feasible flow (by the Out of Kilter procedure, for example). From conservation of flow, the subgraph of G^+ consisting of all the saturated arcs is now an Euler Graph. From our previous discussion, there exists an Euler circuit which includes all the arcs of this subgraph. This proves the lemma.

Lemma 4.2 The convex set defined by the constraints of LP(4.1) is the convex hull of the members of V.

Proof: Denote the convex set defined by LP(4.1) by R and

let $V_o \in R$ be arbitrary. It was argued above that V_o defines circulation flows in the two copies of the station map. Also a set of Euler circuits which include each service exactly once (in either but not in both copies) defines some $V_i \in V$. Select the first copy and trace an Euler circuit which includes all its saturated arcs. This is possible by lemma 4.1. Let A_c^1 be the set of arcs included in this circuit and let A_c^2 be the set of arcs in the second copy which includes the arcs representing the same services as the members of A_c^1. If A^2 is the set of arcs of the second copy then $A^2 - A_c^2$ is an Euler subgraph and there exists an Euler circuit which covers all the members of this subgraph. Trace such a circuit. The circuit traced in the two copies define some $V_p \in V$. V_p will be completely specified, however, only if we specify the <u>departure</u> taken out of each service. Since we trace arcs for which $\sum_{i \in K_\ell} u_i^m > 0$, there is at least one departure $i \in K_\ell$ and an arbitrary one can be selected. Let

$$\lambda_p = \min \left\{ \min_i (u_i^1), \min_i (u_i^2) \right\}$$

for those u_i^m's included in the Euler circuits traced, and define $V_o' \equiv V_o - \lambda_p V_p$. The components of V_o' satisfy:

$$\sum_{i \in K_\ell} (u_i^1 + u_i^2) = 1 - \lambda_p \qquad , \forall \ell$$

$$\sum_{\ell \in I_q^1} \sum_{i \in K_\ell} u_i^1 = \sum_{\ell \in \emptyset_q^1} \sum_{i \in K_\ell} u_i^1 \qquad , \forall q$$

If $\lambda_p = 1$ then $V_o = V_p$, else $\frac{1}{1 - \lambda_p} V_o' \in R$, in which case the

procedure described above can be applied to $\frac{1}{1-\lambda_p} V_o'$ and repeated until

$$0 = V_o - \sum_i \lambda_i V_i \quad \text{and} \quad \sum_i \lambda_i = 1 \quad .$$

which proves the lemma.

Theorem 4.2 LP(4.1) has at least one optimal integer solution.

Proof: Define $\underline{c} = \{ c_1^1, c_1^2; c_2^1, c_2^2; \ldots \}$. Let V_o be a feasible solution to LP(4.1). Then

$$z^* = \underline{c}\, V_o = \underline{c} \sum_i \lambda_i V_i$$

$$= \sum_i \lambda_i \underline{c}\, V_i$$

Let $\underline{c}V_k = \max_i \{ \underline{c}\, V_i \}$, then

$$z^* \leq \sum_i \lambda_i \underline{c}\, V_k$$

$$= \underline{c}\, V_k$$

But V_k is an integer solution to LP(4.1) by definition.

Q.E.D.

Observe that it is sufficient to select the best departure out of each bundle in each copy when the system decomposition problem is being solved. This is true because a service is assigned to one of the two types, and it may as well be the best departure.

As an example we shall optimally decompose the three-station system of Fig. 2.4. The data for the system are as follows: There are two types of aircraft available:

	Breakeven Point	Capacity
1.	60	100
2.	80	150

The demand data and the preference indices computed from them are given in Table 4.1. (It is assumed that A-B and B-C services have the same fare).

Flight No.	Demand	c_i^1	c_i^2	Flight No.	Demand	c_i^1	c_i^2
1	160	40	70	10	150	40	70
2	70	10	-10	11	40	-20	-40
3	180	40	70	12	90	30	10
4	60	0	-20	13	200	40	20
5	20	-40	-60	14	110	40	30
6	140	40	60	15	40	-20	-40
7	80	20	0	16	30	-30	-50
8	50	-10	-30	17	140	40	60
9	100	40	20	18	170	40	70

Table 4.1

The optimally decomposed system is shown in Fig. 4.4. The heavy-lined flight arcs constitute the type-1 subsystem. The broken-lined arcs constitute the second subsystem.

4.4 System Decomposition with Total Fleet Size Minimization

In the last section we discussed a method for an optimal decomposition of the air transportation system. It was assumed

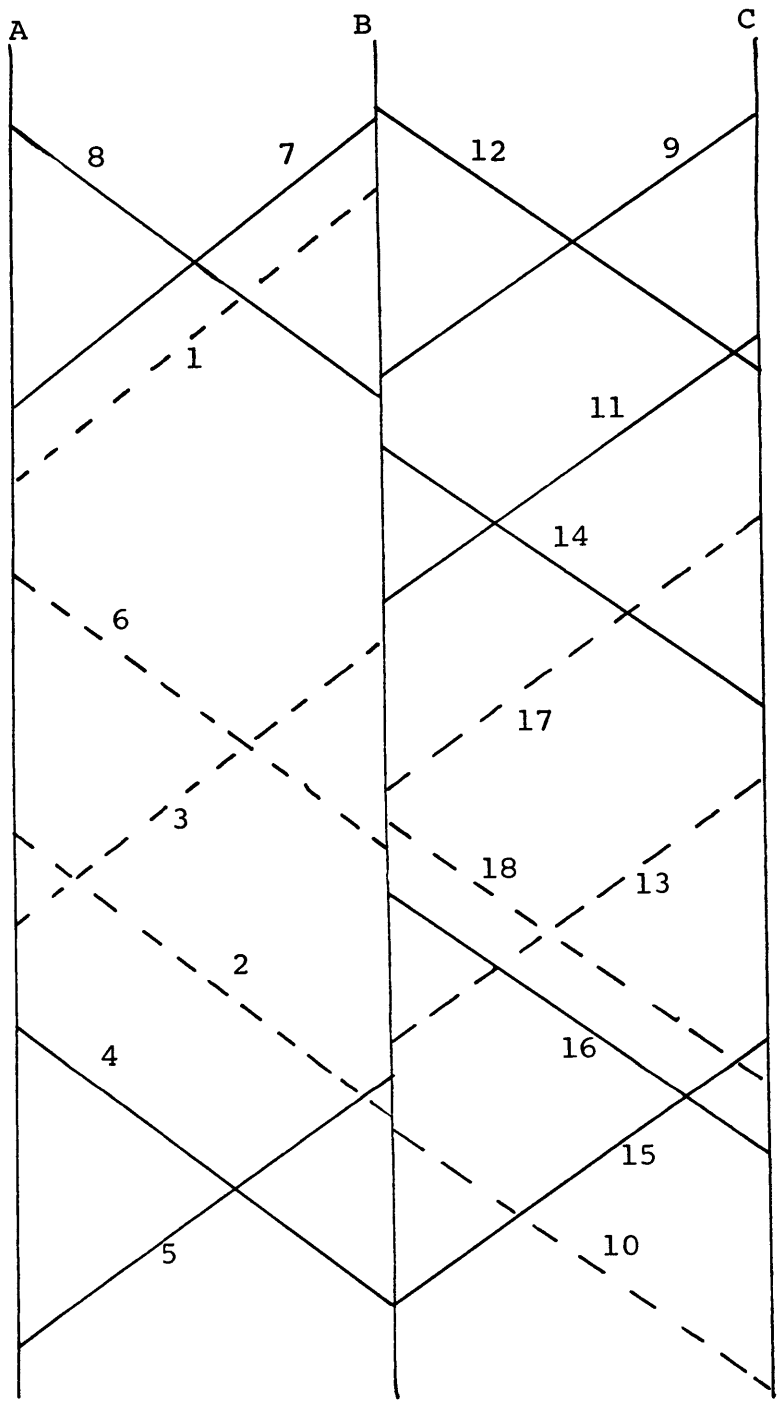


Fig. 4.4

that any fleet size optimization procedures will be carried out in each subsystem independently. If the system is large, so that there is high utilization of aircraft anyway, this may be a satisfactory procedure. When the system is smaller the main objective may be the minimization of the size of the entire fleet. Also, for large systems, if the computer codes can accomodate larger size Linear Programs it may be desirable to perform 'total' optimization directly.

The optimization procedure will be carried out as follows: First solve for the minimal fleet size using ILP(2.1), and let F be the maximal flow obtained. Subject to the minimal fleet size constraint solve the two-fleet problem using the following ILP:

$$\text{(i)} \quad \max z^* = \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} (c_i^1 u_i^1 + c_i^2 u_i^2)$$

S.t.

$$\text{(ii)} \quad \sum_{i \in K_\ell} (u_i^1 + u_i^2) = 1 \qquad , \forall \ell$$

$$\text{(4.2) (iii)} \quad \sum_{\ell \in I_q^1} \sum_{i \in K_\ell} u_i^1 = \sum_{\ell \in \emptyset_q^1} \sum_{i \in K_\ell} u_i^1 \qquad , \forall q$$

$$\text{(iv)} \quad \sum_{j \in A_1(i)} x_{ij}^1 \leq u_i^1, \quad \sum_{j \in A_2(i)} x_{ij}^2 \leq u_i^2 \qquad , \forall i$$

$$\text{(v)} \quad \sum_{i \in B_1(j)} x_{ij}^1 \leq u_j^1, \quad \sum_{i \in B_2(j)} x_{ij}^2 \leq u_j^2 \qquad , \forall j$$

$$\text{(vi)} \quad \sum_{\ell=1}^{\lceil N \rceil} \sum_{i \in K_\ell} \left(\sum_{j \in A_1(i)} x_{ij}^1 + \sum_{j \in A_2(i)} x_{ij}^2 \right) = F$$

(all variables 0-1)

ILP(4.2) is written in its minimal form (see section 2.11). Constraint (vi) is the requirement that the total fleet size is minimal. The other constraints should be familiar.

Just like the case with LP(2.5) the author is unable to produce an example which has a non-integer optimal solution. Neither is he able to prove that in general LP(4.2) has at least one integer optimal solution.

As an example consider the system of Fig. 4.5a. There are three services with alternative departures. All the other services are fixed. The minimal number of aircraft necessary to meet the schedule is 5. Subject to this constraint the system is decomposed into two subsystems with aircraft types for which the data was given in the last section. The data for Fig. 4.5a is given in Table 4.2, and the optimally decomposed system is shown in Fig. 4.5b. Again, it is assumed that all fares are equal. Three type-1 aircraft and two type-2 aircraft are necessary to accomplish the schedule of the subsystems.

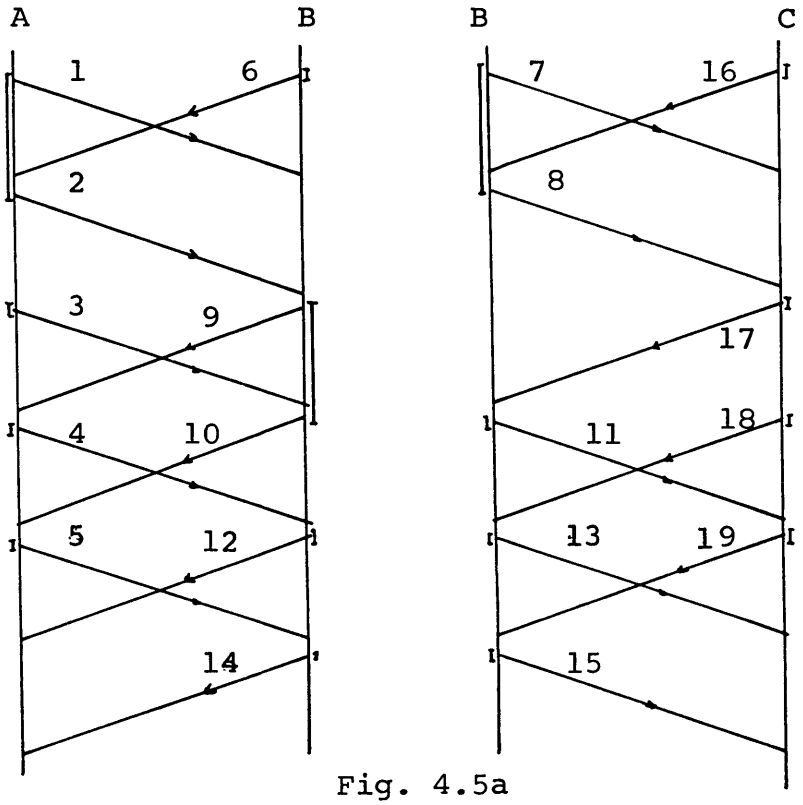


Fig. 4.5a

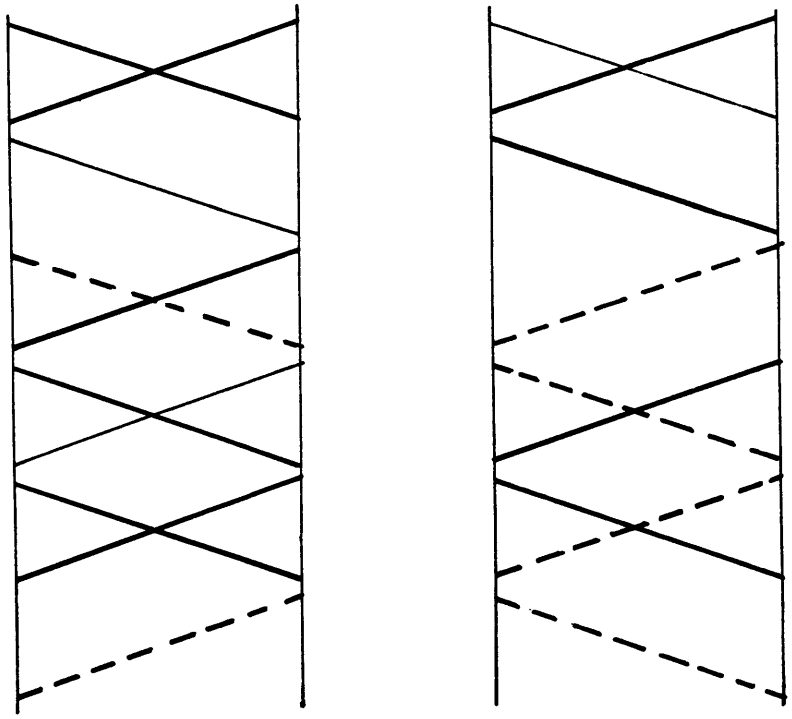


Fig. 4.5b

Depart.No.	Demand	c_i^1	c_i^2	Depart.No.	Demand	c_i^1	c_i^2
1	140	40	60	11	170	40	70
2	30	-30	-50	12	80	20	0
3	150	40	70	13	100	40	20
4	80	20	0	14	60	0	-20
5	50	-10	-50	15	160	40	70
6	100	40	20	16	40	-20	-40
7	60	0	-20	17	130	40	50
8	70	10	-10	18	90	30	10
9	110	40	30	19	140	40	60
10	130	40	50				

TABLE 4.2

Other computational experiments were conducted with LP(4.2). Invariably they resulted in all-integer optimal solutions.

Formulation II will again introduce a considerable saving in the number of constraints. Instead of $s_i; t_j$ we now have the variables s_i^1, s_i^2; t_j^1, t_j^2 with their obvious interpretations. The ILP is:

$$\text{(i)} \quad \max z^* = \sum_{\ell=1}^{N} \sum_{i \in K_\ell} \tfrac{1}{2} \left[c_i^2 (s_i^1 + t_i^1) + c_i^2 (s_i^2 + t_i^2) \right]$$

$$+ \sum_{\ell=1}^{N} \sum_{i \in K_\ell} \left[\sum_{j \in A_1(i)} \tfrac{1}{2} (c_i^1 + c_j^1)\, x_{ij}^1 \right.$$

$$\left. + \sum_{j \in A_2(i)} \tfrac{1}{2} (c_i^2 + c_j^2)\, x_{ij}^2 \right]$$

S.t.

$$\text{(ii)} \quad \sum_{i \in K_\ell} \left(s_i^1 + \sum_{j \in A_1(i)} x_{ij}^1 + s_i^2 + \sum_{j \in A_2(i)} x_{ij}^2\right) = 1 \qquad ,\forall\, \ell$$

$$\text{(iii)} \quad s_i^1 + \sum_{j \in A_1(i)} x_{ij}^1 + \sum_{j \in K_\ell - \{i\}} \left(t_j^1 + \sum_{i \in B_1(j)} x_{ij}^1\right)$$

$$+ \sum_{j \in K_\ell} \left(t_j^2 + \sum_{i \in B_2(j)} x_{ij}^2\right) = 1 \qquad ,\forall\, i \in K_\ell,\ \forall\, \ell$$

$$\text{(4.3) (iv)} \quad s_i^2 + \sum_{j \in A_2(i)} x_{ij}^2 + \sum_{j \in K_\ell - \{i\}} \left(t_j^2 + \sum_{i \in B_2(j)} x_{ij}^2\right)$$

$$+ \sum_{j \in K_\ell} \left(t_j^1 + \sum_{i \in B_1(j)} x_{ij}^1\right) = 1 \qquad ,\forall\, i \in K_\ell,\ \forall\, \ell$$

$$\text{(v)} \quad \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} \left(\sum_{j \in A_1(i)} x_{ij}^1 + \sum_{j \in A_2(i)} x_{ij}^2\right) = F$$

$$\text{(vi)} \quad \sum_{\ell \in I_q^1} \sum_{i \in K_\ell} \left(s_i^1 + \sum_{j \in A_1(i)} x_{ij}^1\right) = \sum_{\ell \in \emptyset_q^1} \sum_{i \in K_\ell} \left(s_i + \sum_{j \in A_1(i)} x_{ij}^1\right)$$

(all variables 0,1) $\qquad ,\forall\, q$

The proof that this is indeed the correct ILP follows exactly the same lines as the proof of section 2.9.

4.5 System Decomposition with Direct Fleet Costs

In the last section c_i^m was computed by subtracting some quantity (breakeven point) from some expected income from operating a service. We now propose a different formulation which will use the direct operating costs of an aircraft over the entire period, rather than the breakeven point. This approach has already been discussed in section 2.14. For the two-fleet case we have c_o^1 as operating costs for type-1 aircraft over the period and c_o^2 for type-2 aircraft. Let N_1 be the set of services which will be taken by type-1 and N_2 the set taken by type-2 aircraft($|N_1| + |N_2| = |N|$). The costs of operating the fleets over the period are

$$z^1 = c_o^1 \left(|N_1| - \sum_{\ell \in N_1} \sum_{i \in K_\ell} \sum_{j \in A_1(i)} x_{ij}^1 \right) \quad \text{for type-1}$$

and

$$z^2 = c_o^2 \left(|N_2| - \sum_{\ell \in N_2} \sum_{i \in K_\ell} \sum_{j \in A_2(i)} x_{ij}^2 \right) \quad \text{for type-2}$$

The net operating income of the period is

$$\sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} (c_i^1 u_i^1 + c_i^2 u_i^2) - z^1 - z^2$$

But

$$|N_1| = \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} u_i^1, \quad |N_2| = \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} u_i^2$$

Hence the ILP is:

$$\text{(i)} \quad \max z^+ = \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} (c_i^1 - c_o^1) u_i^1 + (c_i^2 - c_o^2) u_i^2$$

$$+ \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} \left(c_o^1 \sum_{j \in A_1(i)} x_{ij}^1 + c_o^2 \sum_{j \in A_2(i)} x_{ij}^2 \right)$$

S.t.

$$\text{(ii)} \quad \sum_{i \in K_\ell} (u_i^1 + u_i^2) = 1 \qquad , \forall \ell$$

$$\text{(iii)} \quad \sum_{\ell \in I_q^1} \sum_{i \in K_\ell} u_i^1 = \sum_{\ell \in \emptyset_q^1} \sum_{i \in K_\ell} u_i^1 \qquad , \forall q$$

$$\text{(4.4) (iv)} \quad \sum_{j \in A_1(i)} x_{ij}^1 \leqslant u_i^1 , \quad \sum_{j \in A_2(i)} x_{ij}^2 \leqslant u_i^2 \qquad , \forall i$$

$$\text{(v)} \quad \sum_{i \in B_1(j)} x_{ij}^1 \leqslant u_j^1, \quad \sum_{i \in B_2(j)} x_{ij}^2 \leqslant u_j^2 \qquad , \forall j$$

$$\text{(vi)} \quad \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} \left(\sum_{j \in A_1(i)} x_{ij} + \sum_{j \in A_2(i)} x_{ij}^2 \right) = F$$

(All variables 0-1)

ILP(4.4) is different from ILP(4.2) only in the values of the c_i^m's and in the form of the objective function. LP(4.4) does not always have optimal integer solutions. If constraint (vi) of ILP(4.4) is eliminated a new optimization problem is obtained in which the optimal size of the fleet will be determined by the ILP. This is what management may desire if there is a net increase in revenues as a result of higher combinatorial flexibility possible with a larger fleet. If, for example, the value of c_o^1 is very high compared with the other cost coefficient of z^+ we have a problem of max-flow in the first copy. From this it is obvious that the optimization problem does not always have an optimal integer solution. The Branch and Bound algorithm of the next section is applicable for the present problem without any changes.

4.6 A Branch and Bound Algorithm for ILP(4.4)

As an example consider the data in table 4.3 for the system of Fig. 4.5a. This data obviously does not represent any 'real' situation but was selected because it gives a 'bad' initial optimal solution which was selected in order to demonstrate the power of the Land and Doig type algorithm to be defined below.

Depart.No.	c_i^1	c_i^2	u_i^1	u_i^2	Depart.No.	c_i^1	c_i^2	u_i^1	u_i^2
1	30	0	2/3	0	11	3	0	0	1
2	30	0	1/3	0	12	20	0	0	1
3	2	0	0	1	13	2	0	0	1
4	15	0	1/3	2/3	14	6	0	0	1
5	10	0	2/3	1/3	15	2	0	1/3	2/3
6	14	0	1/3	2/3	16	10	0	1/3	0
7	35	0	0	0	17	14	0	1	0
8	18	0	1	0	18	12	0	0	1
9	15	0	1/3	0	19	3	0	0	1
10	10	0	2/3	0					

$c_o^1 = 40, \quad c_o^2 = 0$

Table 4.3

Table 4.3 also shows the values of the variables, u_i^m's, which constitute an optimal solution to LP(4.4).

The Branch and Bound algorithm is similar to the one defined in section 2.7. Here, however, there is a possibility

of generating terminal nodes which represent infeasible solutions. The infeasibilities may arise because of the inclusion of constraints (iii) and (vi), although by the definition of F at least one optimal integer solution exists. Define the following sets (others were defined in Chapter II):

$$I_o^m(x_k) = \{ i \mid u_i^m = 0 \text{ at node } x_k \} \qquad m = 1,2$$

$$I_1^m(x_k) = \{ i \mid u_i^m = 1 \text{ at node } x_k \} \qquad m = 1,2$$

$$S(x_k) = \{ u_1^1, u_1^2, u_2^1, u_2^2, \ldots \}$$

The algorithm is:

STEP 1. Create a node x_1 and set $X = X_{tf} = \{x_1\}$

STEP 2. Set $I_o^m(x_1) = \Phi \qquad m = 1,2$

STEP 3. Set $I_1^m(x_1) = \Phi \qquad m = 1,2$

STEP 4. Solve LP(4.4) obtaining $S(x_1)$ and $\bar{z}^+(x_1)$

STEP 5. Set $r = 1$

STEP 6. If $S(x_r)$ is integer stop. The optimal solution has been obtained.

STEP 7. Create two branches and nodes $x_{|X|+1}$ and $x_{|X|+2}$ out of x_r

STEP 8. Set $X = X + \{ x_{|X|+1}, x_{|X|+2} \}$

STEP 9. Set $X_{tf} = X_{tf} - \{ x_r \}$

STEP 10. Select $(i,m) \ni 0 < u_i^m < 1$

STEP 11. Set $I_o^m(x_{|X|-1}) = I_o^m(x_r) + \{i\}$; Copy the other three sets of $x_{|X|-1}$ from x_r.

STEP 12. Set $I_1^m(x_{|X|}) = I_1^m(x_r) + \{i\}$; Copy the other three sets of $x_{|X|-1}$ from x_r.

STEP 13. Solve LP(4.4) for $k = |X|-1$, $k = |X|$ with the following additional constraints, obtaining $S(x_{|X|-1})$, $S(x_{|X|})$, $\overline{z}^+(x_{|X|-1})$, $\overline{z}^+(x_{|X|})$

(vii) $u_i^1 = 0, \forall\, i \in I_o^1(x_k)$; $u_i^2 = 1, \forall\, i \in I_1^1(x_k)$

(viii) $u_i^2 = 0, \forall\, i \in I_o^2(x_k)$; $u_i^2 = 1, \forall\, i \in I_1^2(x_k)$

STEP 14. If $S(x_{|X|-1})$ is feasible, $X_{tf} = X_{tf} + \{x_{|X|-1}\}$

If $S(x_{|X|})$ is feasible, $X_{tf} = X_{tf} + \{x_{|X|}\}$

STEP 15. Determine $r \ni \overline{z}(x_r) = \max_k \{\overline{z}(x_k \in X_{tf})\}$

STEP 16. Go to STEP 6.

The algorithm was applied to the problem of Table 4.3 and the tree is shown in Fig. 4.6. All the terminal nodes represent integer solutions.

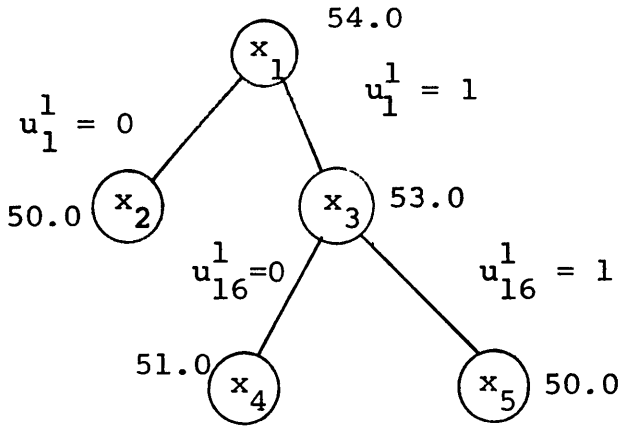


Fig. 4.6

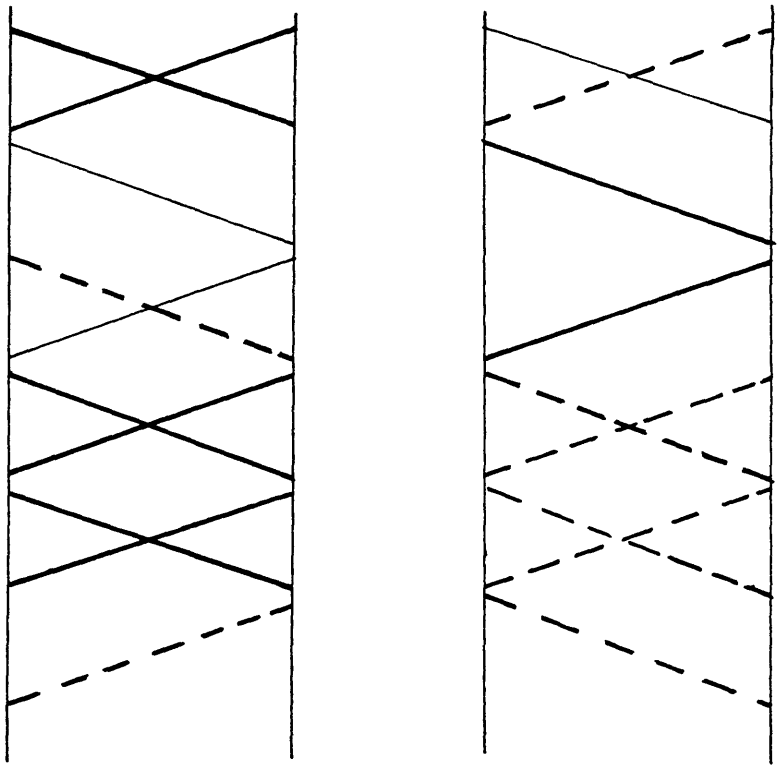


Fig. 4.7

Formulation II can be derived directly from LP(4.3) and LP(4.4) and will not be given here.

4.7 Arc-Chain Formulation

The Arc-Chain formulations for single fleet problems were discussed in sections 2.12 and 2.13. The principle of this formulation is the implicit enumeration of all possible chains that an aircraft may follow. In this chapter there are two copies of the schedule map and any chain can be traced in either one of them. Hence we have twice as many chains to consider. Here, moreover, we add the explicit requirement of conservation of aircraft flow for each type over the period. The Arc-Chain equivalent of LP(4.2) will now be formulated.

Define:

x_j^m = the j-th chain on the m-th copy (m = 1,2)

d_j^m = the total profit of the j-th chain on copy m.

F = the minimal number of aircraft necessary to accomplish the schedule on any copy.

$\emptyset_q$ = the set of chains which start at the q-th station.

I_q = the set of chains which end at the q-th station

n = the number of chains on any copy

The ILP is:

$$\text{(i)} \quad \text{maximize } z^* = \sum_{j=1}^{n} (d_j^1 x_j^1 + d_j^2 x_j^2)$$

S.t.

$$\text{(ii)} \quad \sum_{j=1}^{n} a_{\ell j} \, (x_j^1 + x_j^2) = 1 \qquad \ell = 1, \ldots, |N|$$

$$\text{(4.5) (iii)} \quad \sum_{j=1}^{n} (x_j^1 + x_j^2) = F$$

$$\text{(iv)} \quad \sum_{j \in I_q} x_j^1 = \sum_{j \in \emptyset_q} x_j^1 \qquad , \forall \, q$$

$$x_j^1, \; x_j^2 = 0, \; 1 \qquad , \forall \, j$$

LP(4.5) can be solved by a column generation technique similar to the one described in section 2.13. Only in this case we have to examine the shortest route on both copies in order to determine the next column to be introduced into the basis. Here, in addition we have the constraints (iv) which require some changes in the column generation procedure. As a result of the constraint for station q all chains originating there will have an additional cost term, and the same for all chains terminating there. Hence each station line of the first copy will start and end with arcs having some costs which are determined from those components of $\underline{\pi}$,the shadow prices, which correspond to constraints (iv). Every chain originating at q must include the starting arc of q and every chain terminating there must include the end arc of q.

Since the number of stations is generally very small compared to $|N|$, we still have a system with a very small number of constraints compared to the other formulations.

4.8 Concluding Remarks

In this chapter some multi-fleet models and optimization problems were formulated. Some variations are possible to suit particular conditions and requirements. For example, suppose the system already has p_1 type-1 and p_2 type-2 aircraft available and management would like to use all of them in the newly optimized system. From section 4.5, the fleet sizes for type-1 and type-2 aircraft are, respectively,

$$\sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} (u_i^1 - \sum_{j \in A_1(i)} x_{ij}^1) \text{ and}$$

$$\sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} (u_i^2 - \sum_{j \in A_2(i)} x_{ij}^2)$$

Hence we must add the constraints

$$\text{(i)}' \quad \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} (u_i^1 - \sum_{j \in A_1(i)} x_{ij}^1) \geq p_1$$

$$\text{(ii)}'' \quad \sum_{\ell=1}^{|N|} \sum_{i \in K_\ell} (u_i^2 - \sum_{j \in A_2(i)} x_{ij}^2) \geq p_2$$

to the constraints of ILP(4.4) in order to obtain the appropriate optimization problem.

Some other constraints could similarly be imposed to meet specific requirements. But the author feels that the basic formulations were completely discussed in this chapter.

CHAPTER V
CONCLUSIONS

5.1 General Remarks

The emphasis in this work has been on the development of models for practical applications and practicable solution methods. As the reader probably realizes no attempt has been made to include economic justification for the basic model except for the fact that it looks reasonable and it offers great possibilities for applications to real-life transportation systems.

The largest system solved by the author included 54 services (Chapter III) which is almost a fair-size system. The application of the Land and Doig 'Branch and Bound' techniques to our formulations has been very successful and, in the author's opinion, the only reasonable algorithm to obtain the optimal integer solutions. But what is more important, as a result of the computational experiments with MPS/360 it seems that for some of our problems optimal integer solutions are obtained at the first iteration, and this has been particularly true when the larger transportation systems were solved. Computing time statistics have also have been very good. As far as problem size is concerned it is the author's opinion that all present day systems can be handled by LP procedures like those of MPS/360 (provided sufficient core storage is available). Although some geographical decomposition of the larger system may be necessary for some of the optimization

problems (see next section).

5.2 Further Research

This work reemphasizes the importance of continuing research in the area of 0-1 Integer Programming problems. Formulation II of section 2.9 is in the special form of the 0-1 problem which has occupied researchers for quite a few years now [14] . Results so far have been of limited success.

Two questions were left open: 1. Does LP(2.5) have at least one optimal integer solution? 2. Does LP(4.2) have at least one optimal integer solution? It will be very valuable to be able to answer these questions by providing either proofs or counterexamples.

How good is the system decomposition procedure which was formulated in section 4.3? An answer to this question can be given only after extensive experimentation with large-scale systems and in comparison with solutions to ILP(4.2).

Since some of the existing air transportation systems are quite large it is desirable to obtain a procedure by which a system can be decomposed into geographical subsystems (in contradistinction to decomposition by vehicle type) which will not harm the over-all optimal solutions. Some experimentation with large systems will be necessary here too.

The Arc-Chain formulation uses a considerably smaller number of constraints. It will be useful or necessary (although probably expensive) to write a code for this formulation and obtain computational results.

Formulation II of section 2.9 introduces a sizeable reduction in the number of constraints. At the same time the number of variables increases. What is needed is a procedure to reduce the number of variables without a loss in optimality.

Finally, a problem for the econometrician is to devise a procedure for the determination of the c_i's which we have been using in our models.

The author plans to continue working on these problems.

APPENDIX A
AN OUTLINE OF A COMPUTERIZED AIRLINE MANAGEMENT DECISION SYSTEM (AMDS)

A.1 Introduction

The purpose of this appendix is to describe a computerized airline management decision system. AMDS is intended to be built around MPS/360 whose Linear Programming procedures are the basic optimization tools in this decision system. Airline management may want to solve many of its short range and long range planning problems by using the tools of LP and optimal flows in networks algorithms. The AMDS which is being sketched here will be so easy for management to use, that even the manager who is totally unfamiliar with the computational algorithms of mathematical programming will be able to use it with little effort. He will have to punch just a few control cards or, possibly, type only a few commands into a time-sharing console and obtain the results he will request.

The Crew Scheduling problem has occupied airline O.R. groups for some time now [17] . This optimization problem will be a part of the AMDS in addition, of course, to the problems which are the subject of this thesis. The next section will briefly describe the formulation of the problem and will include some of the author's ideas concerning solution procedures.

The description of the AMDS is meant to be very sketchy and consequently many of the details are missing. No

programming has been done. The construction of the entire system is certainly a major programming effort although the structure of MPS/360 will greatly facilitate the task.

A.2 The Crew Scheduling Problem

In an airline transportation system with many cities only a small number of them serve as 'crew bases', where airline crews and their families live. Out of these bases crews are assigned to all the scheduled flights. There are union limitations on the duty time of crews [16], which impose severe restrictions on the tours (or flight sequences) to which crews can be assigned. Also, overnighting crews away from their bases, meals, etc. are expenses that the airlines must take care of. The problem of crew assignment to flight tours can be formulated as an ILP in the following way:

$$\text{(i)} \quad \min \; z = \sum_{j=1}^{n} c_j x_j$$

S.t.

$$\text{(A.1)} \qquad \text{(ii)} \quad \sum_{j=1}^{n} a_{ij} x_j = 1 \qquad \forall\, i \in N$$

$$\text{(iii)} \quad x_j = 0, 1$$

where x_j represents a feasible tour of a crew (namely, a tour which meets all union requirements) and i is an index over the set of services, N. This formulation requires the enumeration of all the feasible tours. If tour j covers flight number i then $a_{ij} = 1$, otherwise $a_{ij} = 0$. The number

c_j represents the cost to the company associated with the j-th tour. If $c_j = 1 \ \forall \ j$ we are looking for the minimal number of crews necessary to meet the schedule. In general, LP(A.1) has non-integer optimal solutions. Various Implicit Enumeration type algorithms have been suggested to solve ILP(A.1). The computational record is quite poor [16] . Based on his experience with the Land and Doig type algorithm for fleet size problems the author has suggested the use of this technique for ILP(A.1). Computational experiments (conducted by a colleague) have been very successful and indicate that this method is considerably better than all of the enumerative algorithms. The number of constraints is equal to the number of flights in the system and the number of columns is usually very large. But this is just the right situation for MPS/360. The solution procedure is, then, to solve LP(A.1) and then apply a Land and Doig Branch and Bound algorithm (of the type formulated in Chapter II) until the optimal integer solution is obtained.

Another formulation of the Crew Scheduling problem has been pointed out to the author by Fred Steiger of Swissair. Let $G = [S,T;A]$ be a bipartite graph in which S represents the feasible tours and T the services, i.e. $|S|$ will be equal to the number of feasible tours (n of ILP(A.1)) and $|T|$ is equal to the number of flights in the transportation system ($|T| = |N|$). There will be an arc $(s_k, t_\ell) \in A$ if the tour which s_k represents includes flight ℓ . As done before we adjoin to G the nodes s and t and the arcs $(s,s_k) \ \forall \ k$, $(t_\ell, t) \ \forall \ \ell$ obtaining the graph shown in

Fig. A.1.

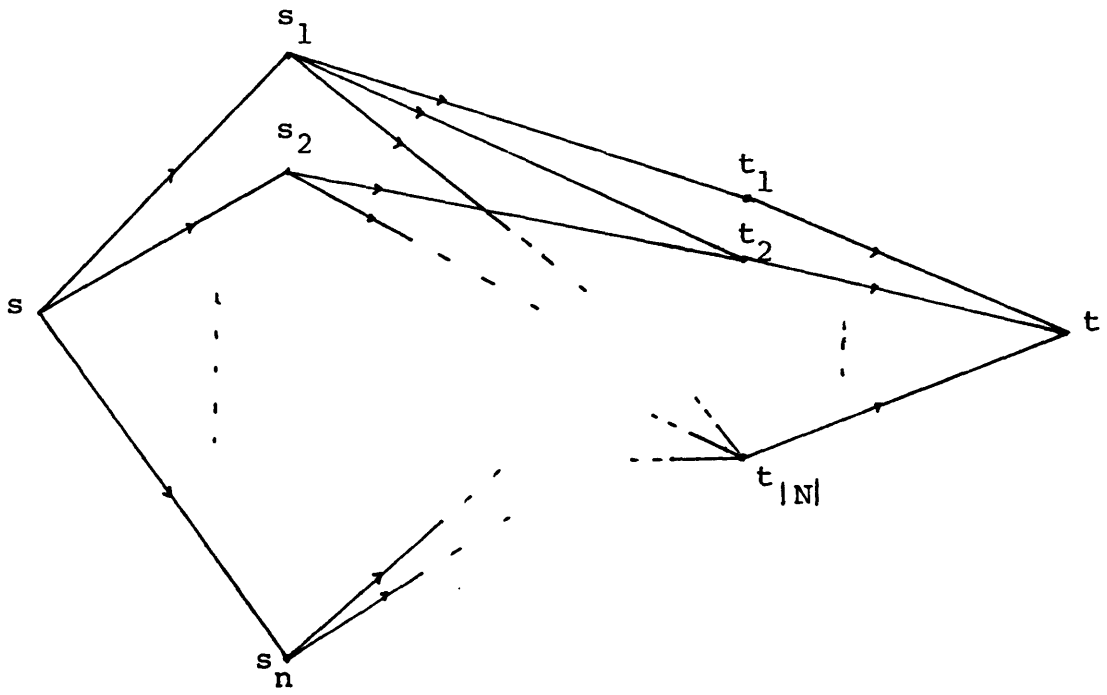


Fig. A.1

Define the following capacities for the arcs of G, where u_{ij} is upper and ℓ_{ij} is lower:

$u_{sk} \equiv u_{s,s_k}$ = the number of flights which are covered by tour k (or the number of (s_k, t_ℓ) arcs which originate at s_k)

$\ell_{sk} \equiv \ell_{s,s_k} = 0 \ \forall$ k

$$u_{k\ell} \equiv u_{s_k, t_\ell} = 1 \ \forall \ (k,\ell)$$

$$\ell_{k\ell} \equiv \ell_{s_k, t_\ell} = 0 \ \forall \ (k,\ell)$$

$$u_{\ell t} \equiv u_{t_\ell, t} = 1 \ \forall \ \ell$$

$$\ell_{\ell t} \equiv \ell_{t_\ell, t} = 1 \ \forall \ \ell$$

And the costs:

$c_{sk} \equiv c_{s, s_k}$ = the cost of tour k (= c_k of ILP(A.1)).

all the other costs are equal to zero.

The optimization problem has been formulated now as one of obtaining the feasible flow which minimizes costs, but with the following condition: each arc is either empty or saturated. The last condition renders the problem nonlinear, and no straightforward algorithms for solving it are currently available. The formulation of this optimal flow problem is:

$$\text{(i)} \quad \min \ z = \sum_{k=1}^{n} \frac{c_{sk}}{u_{sk}} \ x_{sk}$$

$$\text{(ii)} \quad x_{sk} = \sum_{\ell \in A(k)} x_{k\ell} \qquad \forall \ k$$

$$\text{(iii)} \quad x_{\ell t} = \sum_{k \in B(\ell)} x_{k\ell} \qquad \forall \ \ell$$

(A.2)

$$\text{(iv)} \quad 0 \leq x_{k\ell} \leq 1 \qquad \forall \ (k, \ell)$$

(v) $x_{\ell t} = 1$ $\forall\ \ell$

(vi) $x_{sk} = 0,\ u_{sk}$ $\forall\ k$

The author is suggesting the following Branch and Bound Procedure for this problem. This procedure was motivated by the ideas brought forth in [5] . The discussion will be sketchy.

Suppose we relax constraint (vi) and substitute the following constraint for it:

(vi)' $0 \leq x_{sk} \leq u_{sk}$

The new problem thus obtained is a usual optimal flow in a network problem which can be solved by the 'Out of Kilter' algorithm, say. If for some k in the optimal solution $0 < x_{sk} < u_{sk}$ create two branches: $x_{sk} = 0$ and $x_{sk} = u_{sk}$ and solve the two optimal flow in network problems. Continue per Chapter II. Eventually the optimal solution to MP(A.2) is obtained. The advantage of this method is, of course, that it uses a network flow algorithm instead of a LP at each node, but the author expects the tree to be considerably larger than the one representing the Land and Doig procedure for ILP(A.1). No computational experience can, as yet, be reported.

The Crew Scheduling problem has been discussed because the author considers it to be, together with fleet size problems, the most important component of AMDS. We shall

assume that ILP(A.1) will be used in AMDS for solving it.

A.3 The Structure of AMDS

The discussion of AMDS will be divided into the following topics:

1. Data Files and file-maintenance procedures
2. Preoptimization Procedures
3. Optimization Procedures
4. Management Report Generation

It is the intention of the author to expose the reader only the very basic ideas:

1. The data files will consist of all the basic data which are needed in the preoptimization and optimization procedures. These data will include all the services of the system: their departure and arrival times, alternative departures and preference indices for each one of them. This file will be the basic one for fleet-size problems. The Crew Scheduling procedure requires a fixed schedule system and a different file may be necessary for that purpose. These files will be uniquely identified by names and a file dictionary. File maintenance procedures will provide management with file-deletion and file-addition capabilities as well as the ability to modify individual data items via a time-sharing console or by batch processing, using a minimal number of control cards.

2. The preoptimization procedures will include all the programs which operate on the basic data files and gen-

erate temporary and/or intermediate files. They will be stored, in the format required by the optimization procedures. The preoptimization procedures will be matrix generators for the various ILP's or data for network flows algorithms (for the fixed schedule min-fleet problem for example). The intermediate files will be permanently stored in order to save as much future preoptimization processing as possible.

3. MPS/360 [11] will be the central component of the optimization procedures. This system can be augmented by user-written programs to which computational results of the LP algorithms are accessible. Thus a FORTRAN coded routine which will manage the Branch and Bound procedure can be added to the MPS program library. This program will be brought to core by the MPS executor [12]whenever an optimal solution to the LP's are obtained, will check the solution and further branching or termination decisions will be made by it.

4. After the optimal integer solution to the problem at hand has been obtained, management reports will be generated by a user written program, which will transform the filed solution into a report in a format easily interpreted and used by management, scheduling personnel and others. The report-generating programs can also be made part of the MPS/360 program library. MARVEL processor of MPS/360 (unimplemented as yet) might also be used to generate reports.

The execution of every MPS/360 job is controlled by MPS control statements. These consist of procedure names such as PRIMAL (which causes the execution of a LP optimization

algorithm) or the name of a user written Branch and Bound routine, for example. All these statements must be logically related, of course, and can be grouped and defined as a MACRO statement, which may be added to MPS MACRO library. Hence, a person who would like to perform the crew scheduling procedure for his system (or a part of it) will have to use only a single control statement which might look like this:

```
CREWSCHD('FILNAME','OUTPUTA')
```

where the first parameter identifies for the processing programs which data file to use for this particular scheduling. The second parameter designates the output device. The necessity for other parameters will of course arise when the development of AMDS progresses.

Fig. A.2 is a schematic drawing of AMDS.

AMDS

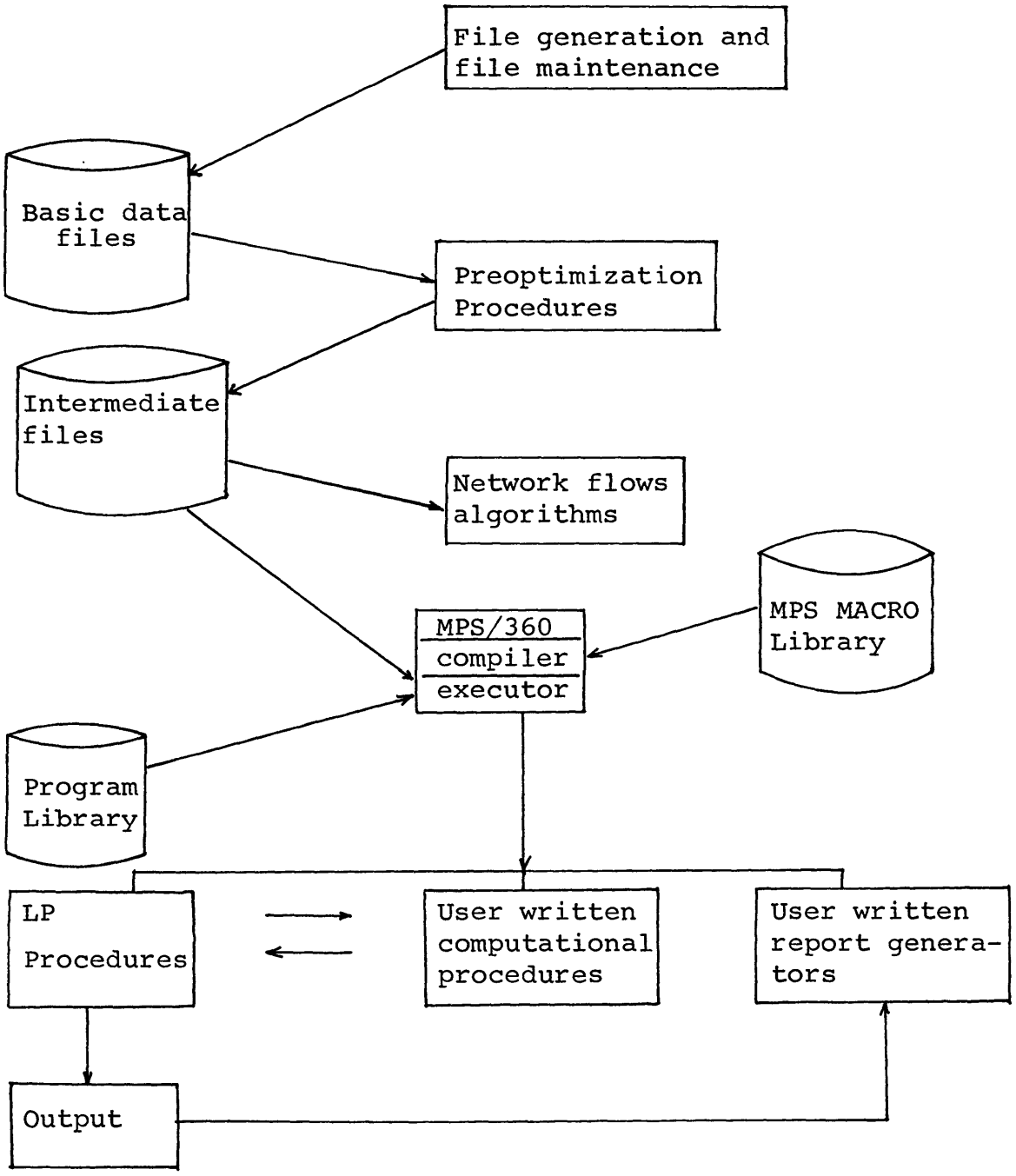

File generation and file maintenance
Basic data files
Preoptimization Procedures
Intermediate files
Network flows algorithms
MPS MACRO Library
MPS/360
compiler
executor
Program Library
LP
Procedures
User written computational procedures
User written report generators
Output

Fig. A.2

REFERENCES

1. Berge, C., and Ghouila-Houri, A., Programming, Games and Transportation Networks, Methuen (London) and Wiley (New York), 1965.

2. Berge, C., The Theory of Graphs, Methuen (London) and Wiley (New York), 1962.

3. Devanney, J. W., Transportation Scheduling Under Multi-Dimensional Demands, Ph.D. Thesis, M.I.T., September 1967.

4. Dantzig, G. B., and Fulkerson, D.R., Minimizing the Number of Tankers to Meet a Fixed Schedule, Naval Research Logistics Quarterly, Vol. I, 1954, pp. 217-222.

5. Efroymson, M.A., and Ray, T. L., A Branch-Bound Algorithm for Plant Location, Operations Research, May-June, 1966.

6. Ford, L. R., and Fulkerson, D. R., Flows in Networks, Princeton University Press, 1962, pp. 62-63.

7. Ford, L.R., and Fulkerson, D. R., Flows in Networks, Princeton University Press, 1962.

8. Ford, L.R., and Fulkerson, D.R., A Suggested Computation for Multi-Commodity Network Flows, Management Science, Vol. 5, No. 1, October 1958.

9. Gue, R. L., Liggett, J.C., and Cain, K.C., Analysis of Algorithms for the Zero-One Programming Problem, Communications of the ACM, Vol. 11, No. 12, December 1968.

10. Land, A.H., and Doig, A., An Automatic Method of Solving Discrete Programming Problems, Econometrica, Vol. 28, 1960, pp. 497-520.

11. MPS/360, Application Description, IBM Manual H20-0136-1, 1966.

12. MPS/360, Read Communications Format, IBM Manual H20-0372-0, 1967.

13. Martin-Löf, A., Operations Research Center Report, M.I.T., (as yet unpublished).

14. Pierce, J.F., Application of Combinatorial Programming to a Class of All-Zero-One Integer Programming Problems, IBM Cambridge Scientific Center, July 1966.

15. Simpson, R.W., Computerized Schedule Construction for an Airline Transportation System, Flight Transportation Laboratory Report FT-66-3, M.I.T., December 1966.

16. Steiger, F., and Liederer, M., An Accelerated Enumeration Method for Combinatorial Programming Problems, A paper submitted at the IFIP congress in Edinburgh, 1968.

17. Steiger, F., Activity Report of the AGIFORS Study Group 'Crew Scheduling', A paper presented at the 7th AGIFORS Symposium, October 2-5, 1967.

18. Young, D., Scheduling Revisited, NBS Report, October 1967.