D1-82-0265

"Also available from the author"

BOEING SCIENTIFIC RESEARCH LABORATORIES

Maximum Payloads Per Unit Time Delivered Through an Air Network

George B. Dantzig

David L. Johnson

Mathematics Research

June 1963

D1-82-0265

MAXIMUM PAYLOADS PER UNIT TIME DELIVERED THROUGH AN AIR NETWORK

by

Professor George B. Dantzig
University of California, Berkeley

and

David L. Johnson
Mathematics Research Laboratory

Mathematical Note No. 305

Mathematics Research Laboratory

BOEING SCIENTIFIC RESEARCH LABORATORIES

June 1963

Introduction

Optimizing the flow of aircraft through a network of bases requires some modification of the standard network flow algorithms because of the special nature of air transportation. The payload carried by an airplane is a function of the distance between refueling stops, as well as field conditions at the bases; and in general the maximum allowable payload is not the same on all arcs of the network. Thus the shortest route through a network is not necessarily the route of maximum payload flow per hour of flight time, a trivial example being a plane which can fly non-stop from origin to destination but only with zero payload.

In this paper we wish to consider two air network problems analogous to the shortest path and the maximum flow problems. These are the route of maximum payload flow per hour of flight time and the maximum steady-state payload flow through a network with base capacity constraints.

Route of Maximum Payload Flow

Two numbers are required to characterize the flow of payload on an arc of an air network. Referring to Figure 1, the number above each arc is the maximum payload, in thousands of pounds, which can be carried on that arc; the number below the arc is the flight time plus refueling time for that arc. In order to make the algorithm work for the round trip case a directed arc from T to S must be added bearing an artificial payload of infinity and a flight time corresponding to the shortest return path. All other arcs can be either directed or non-directed.

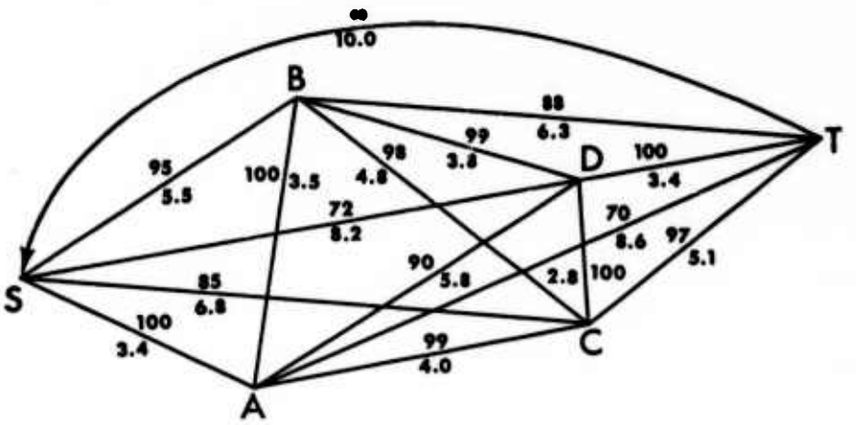


Figure 1

We shall show that the route of maximum payload flow can be found by repeated application of a shortest route algorithm. If we let $R_1, R_2, \ldots, R_n$ be the set of all roundtrip routes from S to T of the network N_1, we can define a subsequence a_i in the following way, where the iteration begins with $i = 1$.

1. Using the shortest route labeling algorithm find a_i, the route of minimum time T_{a_i}.
2. Find the maximum payload, L_{a_i}, which can be carried on route a_i; this is the minimum of the payload limits for the arcs on that route.
3. Delete from the network all arcs with payload limit L_{a_i} or less.
4. Increase i by 1 and repeat steps 1 to 3 until no route exists.

This algorithm yields a sequence of flight times T_{a_i}, payloads L_{a_i} and hence payload flows, $F_{a_i} = L_{a_i}/T_{a_i}$. The maximum payload flow for the network is the maximum flow of this sequence.

For the network in Figure 1 we get the following iterations.

i	a_i	L_{a_i}	T_{a_i}	F_{a_i}	Arcs Deleted
1.	SDTS	72	21.6	3.33	SD,AT
2.	SBTS	88	21.8	4.04	BT,SC
3.	SACTS	97	22.5	4.31	CT,SB,AD
4.	SACDTS	99	23.6	4.19	AC,BC,BD

In this case the algorithm terminates in four iterations from which we pick out the route of maximum payload flow as SACTS.

The number of iterations cannot exceed the number of arcs of the network and is usually much less. It remains to be proved that the maximum flow for the routes examined is also the maximum flow for all the routes.

Proof: We remark first that both the sequence of payloads and flight times are ordered. That is, $L_{a_i} < L_{a_{i+1}}$ and $T_{a_i} \leq T_{a_{i+1}}$ for all i. Now consider any route R_k not in the sequence examined. We can find an i such that $L_{a_{i-1}} < L_{R_k} \leq L_{a_i}$, where we define $L_{a_0} = 0$. At the i^{th} stage of the iteration the shortest time was T_{a_i}; hence $T_{R_k} \geq T_{a_i}$ and $F_{R_k} \leq F_{a_i}$.

Capacitated Air Network Flow

Unlike most network flow problems, capacity constraints of an air network are on the nodes rather than the arcs. This of course can be handled by treating the nodes as dummy arcs with the prescribed capacity and allowing the real arcs to have infinite capacity. However, a more

difficult aspect of the problem arises from the fact that the base constraints are on the number of planes which can be served per unit of time while the flow being maximized is again payload. In the following paragraphs we describe an algorithm which maximizes payload flow through a network with base capacity constraints on the planes and arc limitations on the payload per plane.

Figure 2 shows a simple network with three intermediate bases, where the arc numbers are the payload limits on the arcs and the node numbers are the base constraints in planes per unit time. There are no flight times shown because we are dealing here with a steady-state problem. For the same reason the return flight is not important. What we are maximizing is the steady-state flow of payload into T when the capacity of the network is saturated.

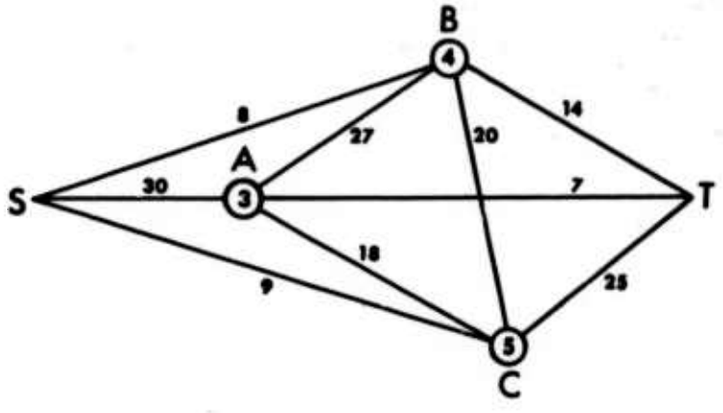


Figure 2

The problem can be formulated as a linear program by enumerating all the routes and determining for each route α a column vector P_α whose i^{th} component is 1 if base i is used and 0 otherwise. The cost component is L_α, the maximum payload which can be carried on route α. This incidence matrix for nodes versus routes for the network in Figure 2 is shown in Table I. It is given here merely to help

	SAT	SBT	SCT	SABT	SBAT	SACT	SCAT	SBCT	SCBT	SABCT	SACBT	SBACT	SBCAT	SCABT	SCBAT	
	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}	b
A	1			1	1	1	1			1	1	1	1	1	1	= 3
B		1		1	1			1	1	1	1	1	1	1	1	= 4
C			1			1	1	1	1	1	1	1	1	1	1	= 5
L	7	8	9	14	7	18	7	8	9	20	14	8	7	9	7	

Table I

identify the routes and is not essential to the calculation. Solving this problem, in general, by any of the usual linear programming procedures is impractical, as the number of routes grows very rapidly with the number of bases so that even the enumeration of the routes becomes a formidable task for any but the most trivial practical problem.

The procedure we shall describe here is very similar to the technique used by Ford and Fulkerson for multi-commodity network flows [5] and consists of the revised simplex procedure modified by using an auxiliary shortest route algorithm to determine the new vector to be brought into the basis at each stage. In this way we need to calculate only a very small part of the relative cost row to determine the pivot column.

At each stage of the iteration we are given a current basis B, whose order is the number of intermediate nodes of the network, and its inverse. The initial basis can be the set of slack variables associated with unused node capacity; or if the one-stop routes are part of the system, the vectors associated with these routes can be used to form the initial basis, as in the example shown in Table I. In either case, then, the initial basis is the identity matrix and the algorithm proceeds as follows.

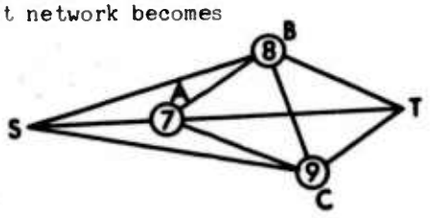
1. Calculate π, the vector of simplex multipliers (prices) for the current basis.
2. Assign the prices to their corresponding nodes (i.e., the dummy arcs) and zero to all other arcs in the original network, and call this price network N_1. Then find a sequence of minimum cost routes according to the following sub-algorithm, initiating the iteration with $i = 1$.
   a. Find the least cost route α_i through the cost network N_i and the payload limit L_{α_i} for the corresponding route through the original network.
   b. Delete from the cost network all arcs with payload limits not exceeding L_{α_i}. Call this network N_{i+1}.
   c. Increasing i by 1, repeat steps a and b until N_{i+1} becomes disconnected and no route exists.
3. If for any route α_i in step 2 we have $S_{\alpha_i} - L_{\alpha_i} < 0$, where S_{α_i} is the sum of the costs along route α_i, then P_{α_i} can be brought into the basis in the usual way to produce a new current basis.

4. With this new basis, repeat steps 1 to 3. When $S_{\alpha_i} - L_{\alpha_i} \geq 0$ for all α_i, the flow is maximal.

Example

Let us trace through the steps of the algorithm with the example illustrated in Figure 2. We begin with an initial basis B^0 formed by the vectors associated with the one-stop routes SAT, SBT, and SCT. From the network we find the vector of payload limits $\gamma = (L_{SAT}, L_{SBT}, L_{SCT}) = (7,8,9)$, and the steps of the algorithm are as follows.

1. $\pi = \gamma B^{-1} = (7,8,9)$
2. The cost network becomes



and the sub-iteration is as follows.

i	α_i	S_{α_i}	L_{α_i}	$S_{\alpha_i} - L_{\alpha_i}$	Arcs Deleted
1	SAT	7	7	0	AT
2	SBT	8	8	0	SB
3	SCT	9	9	0	SC
4	SABT	15	14	1	BT
5	SACT	16	18	-2	AC
6	SABCT	24	20	4	BC

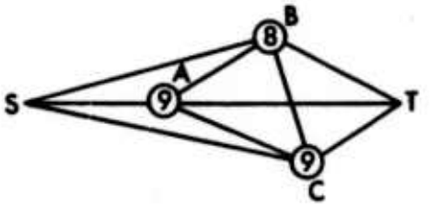
3. Bringing the vector corresponding to SACT into the basis (this corresponds to P_5 in Table I), yields the new basis $B = (P_5, P_2, P_3)$ and the new inverse

$$B^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix}.$$

The new cost vector is $\gamma = (18,8,9)$.

On the second iteration we get the following.

1. $\pi = \gamma B^{-1} = (9,8,9)$
2. The cost network and sub-iteration are



i	a_i	S_{a_i}	L_{a_i}	$S_{a_i} - L_{a_i}$	Arcs Deleted
1	SBT	8	8	0	SB,AT
2	SCT	9	9	0	SC
3	SABT	17	14	3	BT
4	SACT	18	18	0	AC
5	SABCT	26	20	6	BC

3. Since $S_{a_i} - L_{a_i} \geq 0$ for all a_i the flow is maximal.

After calculating the new b vector, $\bar{b} = B^{-1}b$, we find the maximum payload flow is 104 units achieved by 3 planes on route SACT, 4 on SBT, and 2 on SCT.

Since the numbers in the column labeled $S_{a_i} - L_{a_i}$ are actually entries in the relative cost row in the simplex procedure, we know the flow is optimal provided all these elements are non-negative. But we have examined only five of them and it remains to be proved that all the other entries are also non-negative.

Proof: We remark again, as in the preceding section, that the payload limits are ordered. That is, $L_{a_i} < L_{a_{i+1}}$ for all i. Then for any other route R_k not in the sequence of a_i we can find an i such that $L_{a_{i-1}} < L_{R_k} \leq L_{a_i}$, where we define $L_{a_0} = 0$. But at the i^{th} stage of the iteration the route chosen was the least cost route. Hence $S_{a_i} \leq S_{R_k}$, and therefore $S_{a_i} - L_{a_i} \leq S_{R_k} - L_{R_k}$. Thus if all the entries in the column labeled $S_{a_i} - L_{a_i}$ are non-negative, we know that all the entries in the relative cost row are also non-negative and the flow is optimal.

Computational Experience

Neither of the two algorithms described here has been programmed for machine computation because the problems for which they were designed could easily be solved by hand calculation. For the first algorithm the largest problem involved 23 nodes and was solved in less than an hour. The largest problem for the second algorithm involved 11 nodes; but even a network of only 11 nodes has over a million different routes through it. The optimum set of routes was determined in about two hours and required from five to ten iterations depending on the data. The sub-iteration usually required from 8 to 13 steps before the network became disconnected. For hand calculation it was found easier to pivot as soon as the first negative value of $S_{a_i} - L_{a_i}$ was found rather than

calculating all $S_{\alpha_1} - L_{\alpha_1}$ and choosing the most negative. This may not be true for machine calculation and would depend on the relative efficiency of the two iterations.

BIBLIOGRAPHY

1. Dantzig, G. B. "On the Shortest Route Through a Network", Management Science 6(1960), 187-190.

2. _____ and Ferguson, A. R., "The Allocation of Aircraft to Routes - An Example of Linear Programming Under Uncertain Demand", Management Science 3 (1956), 45-73.

3. Ford, L. R. Jr. and Fulkerson, D. R., "Maximal Flow Through a Network", Can. J. Math. 8 (1956), 399-404.

4. _____, "A Simple Algorithm for Finding Maximal Network Flows and an Application to the Hitchcock Problem", Can. J. Math. 9 (1957), 210-218.

5. _____, "A Suggested Computation for Maximal Multi-Commodity Network Flows", Management Science 5 (1958), 97-101.