DYNAMIC NETWORK FLOWS

A DISSERTATION

SUBMITTED TO THE DEPARTMENT OF OPERATIONS RESEARCH

AND TO THE COMMITTEE ON GRADUATE STUDIES

OF STANFORD UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

By

James B. Orlin

May 1981

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

[illegible]
(Principal Adviser)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

George B. Dantzig

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

[illegible]

Approved for the University Committee
on Graduate Studies:

Gerald J. Lieberman
Dean of Graduate Studies &
Research

DEDICATED TO MY FRIENDS AT STANFORD,

TO MY PARENTS, AND TO DONNA

ACKNOWLEDGMENTS

I am truly indebted to my adviser, Professor Arthur F. Veinott, Jr., for guiding the development of the theory in this thesis, suggesting new results, applications, and directions, for simplifying the notation and proofs, and for his painstaking care in improving the exposition. I am also appreciative of the encouragement that he gave me during the course of my dissertation and the independence that he allowed me in pursuing the research.

I am also grateful to my friends and colleagues at M.I.T. for the gentle encouragement that they gave me to complete my dissertation. Moreover, I thank Carole Cusani and Sheila Hill for typing preliminary and final versions of this thesis.

TABLE OF CONTENTS

CHAPTER I

INTRODUCTION AND SUMMARY

This thesis presents and solves three dynamic mathematical programming problems in which the goal is to minimize the long-run average cost per period. In each case we show that the dynamic problem can be reduced to solving a suitable static (one-period) problem. Two of these problems are infinite-horizon integer programming problems involving network flows evolving over time, and they form the core of this thesis. The remaining problem is the dynamic convex programming problem whose solution is an essential sub-result used in solving one of the dynamic network-flow problems and is also of interest in its own right.

Dynamic Convex Programs

The dynamic convex programming problem, discussed in Chapter II, may be described as follows. A decision vector is to be chosen in each of an infinite sequence of periods. There is an associated real or $+\infty$ valued proper stable convex cost function $c(\cdot,\cdot)$ and the cost associated with period i is $c(x, y)$ when x and y are the decision vectors chosen in periods i and $i + 1$. The objective is to choose a sequence of decision vectors that minimizes the long-run average cost per period. If there are no initial conditions and if we require the sequence of decision vectors to be bounded, then we show that we may restrict attention to sequences of decision vectors that are stationary, i.e., the same decision vector is used in each period. The desired common vector x is one that minimizes $c(x, x)$.

Dynamic convex models have applications in various fields including dynamic linear programming, dynamic programming and economic long-range planning.

The problem and its solution play an important role in solving the minimum convex-cost dynamic network-flow problem of Chapter IV. The author (1981) has also used the result to solve the dynamic matching problem, an infinite-horizon dynamic integer programming problem that generalizes the (non-bipartite) weighted matching problem.

The results in Chapter II extend earlier work of Gale (1967) and others who proved a result that implied the optimality of a stationary sequence for a restricted class of underlying cost functions. His proof technique is extended in Chapter II to prove the optimality of stationary sequences for a significantly wider class.

Dynamic Network Flows

Chapters III and IV of this thesis are devoted to two related dynamic network-flow problems. Both may be formulated in terms of the following network structure. Given a finite network, there is associated with each arc a (possibly infinite) integer upper and lower bound on the flow therein and an integral transit time giving the number of periods that flow takes to pass through the arc. A feasible dynamic flow is a sequence of flows that satisfy the upper and lower bounds on the arc flows and also conservation of flow at each node in each period except for the first few periods during which time the flow is initialized. Because such flows are conserved after the first few periods, the net flow in transit in each subsequent period remains constant and is called the throughput.

Maximum-Throughput Dynamic Network Flows

In Chapter III we consider the maximum- (resp., minimum-) throughput dynamic network-flow problem of finding a feasible dynamic flow having maximum (resp., minimum) throughput. To analyze this problem we assume that there is a feasible bounded dynamic flow and define a "dynamic cut" in such a way that the "flow across a dynamic cut" is identically equal to the throughput. We then prove a strong duality result, viz., the maximum throughput is equal to the minimum "upper capacity" of a dynamic cut. Furthermore, we show that there is always an integer stationary dynamic flow that achieves the maximum throughput. This flow may be determined by repeating an integral optimum solution to the corresponding static maximum-linear-profit network-flow problem in which the objective-function coefficient associated with each arc flow is its transit time. Also the minimum-upper capacity dynamic cut may be determined from an optimum solution to the dual of the static linear program.

Minimum Convex-Cost Dynamic Network Flows

In Chapter IV we consider the minimum convex-cost dynamic network-flow problem in which there is also associated with each arc a real-valued convex function of the arc flow between its upper and lower bounds with the function being linear between successive integers. The aim is to find a bounded feasible dynamic flow with fixed throughput that has minimum long-run average cost per period.

This problem is a special case of the dynamic convex programming problem, and the results of Chapter II show that there is a stationary optimal dynamic flow. Furthermore, one such dynamic flow may be determined by repeating a solution of the static minimum convex-cost network-flow problem plus a linear side condition on the arc flows (the fixed-throughput constraint). We show there is a "fractionally extreme" optimal static flow that is the sum of an integer static flow and a flow around a cycle of r/q units for some positive integers $r < q$ and q dividing the absolute transit time of the cycle. This characterization leads to a fast method for rounding the fractional parts of the optimal stationary dynamic flow so as to give an integer feasible dynamic flow with period q and having exactly the same throughput and average-cost per period as the stationary dynamic flow. Hence, this periodic dynamic flow is an integer optimal and an optimal integer dynamic flow.

Applications

Vehicle Routing with Periodic Demands

Consider a transportation firm, such as an airline, that must schedule vehicles each day on a specified set of fixed daily repeating routes. In addition, the firm is allowed to schedule certain optional routes, e.g., deadheading, at a specified cost per route. Dantzig (1962) formulated the problem of determining a feasible stationary schedule that minimizes the number of vehicles needed as a special case of the minimum-throughput dynamic network flow problem, as described in Chapter III. The problem of determining a feasible schedule that minimizes the average daily cost subject to a fixed fleet size is formulated in Chapter IV as a special case of the minimum convex-cost dynamic network-flow problem. Previously, Dantzig (1962) considered the above problem under the added restriction that the schedule is stationary.

Maximum Finite-Horizon Dynamic Network Flows

Ford and Fulkerson (1958) formulated and solved the finite-horizon dynamic network-flow problem which may be described as follows. Given a network with nonnegative-integer transit times, zero lower bounds on arc flows, and integer upper bounds thereon, determine the maximum flow that may be sent from a source to a sink in a fixed number of periods. In Chapter III we show that this problem may be transformed into a special case of the minimum-throughput dynamic network-flow problem, and our solution technique and duality result for the minimum-throughput problem induce the solution technique and duality result given by Ford and Fulkerson (1958) for the finite-horizon problem.

Cyclic Capacity Scheduling and Cyclic Staffing

The cyclic capacity scheduling problem is to find a minimum per-period cost schedule of buying and selling capacity in blocks of consecutive periods so as to satisfy demands for capacity that repeat periodically over time. This problem, which is modeled as a minimum convex-cost dynamic network-flow problem in Chapter IV, generalizes the finite-horizon capacity scheduling problem which was reduced by Veinott and Wagner (1962) to a minimum linear-cost network-flow problem. A special case of the cyclic capacity scheduling problem is the cyclic staffing problem which is to minimize the daily cost of staffing a work-force round-the-clock on shifts of consecutive hours so as to satisfy minimum hourly demands that vary within a day, but repeat daily.

Although the infinite-horizon problem considered above has apparently received no previous attention in the literature, if we require the assignment of workers to be the same every day, then the resulting problem is the 1-day cyclic staffing problem which has received considerable attention. The largest sub-problem of the 1-day cyclic staffing problem to be solved in polynomial time is the case in which the rows of the constraint matrix have "circular ones"; this sub-class was solved by Bartholdi, Orlin, and Ratliff (1980).

The integral solution obtained for the cyclic capacity scheduling problem, and hence for the cyclic staffing problem, has the property that it is obtained by rounding an infinite-horizon continuous stationary solution. For this reason, the number of workers on any specified shift varies by at most one from day to day. In this sense, it is "almost feasible" for the 1-day problem, and always has a daily cost no greater than the minimum daily cost for the 1-day problem.

Periodic Production and Transshipment

Consider a firm that must produce and ship goods so as to satisfy periodically repeating demands, such as for food or petroleum products, and where the cost of producing and shipping goods is convex. The problem of finding a feasible schedule that minimizes the average daily cost given a fixed fleet of delivery vechicles is modeled in Chapter IV as a minimum convex-cost dynamic network-flow problem. Certain assumptions are implicit in the model including the following: no storage is allowed at the demand sites and each vehicle can carry goods for only one demand location at a time. Relaxing either of these assumptions makes the problem NP-hard as proved in the appendix of Chapter IV.

Minimum Cost-to-Time Ratio Circuits

Dantzig, Blattner, and Rao (1965) formulated and solved the "tramp steamer" problem, or equivalently "the minimum cost-to-time ratio circuit" problem. The problem may be viewed as that of choosing an infinite-horizon tour for a tramp steamer that is to travel from port to port so as to minimize its daily costs. The tramp steamer may visit any subset of the ports and in any order, and for any ordered pair of ports there is an associated per trip cost and transit time.

Various applied problems may be modeled as a minimum cost-to-time ratio circuit problem as detailed by Fox (1969). The model has recently been applied by Karp and Orlin (1980) to solve a special case of the cyclic staffing problem, and by Graves and Orlin (1980) to solve a periodic version of the Wagner-Whitin (1958) dynamic economic-lot-size model.

Among optimum tours there is always one in which the steamer travels cyclically in the same order around a subset of ports. We show that the cycle of ports may be obtained by the rounding technique of Chapter IV, thus showing that previous solution techniques may be viewed as a special case of our rounding result.

CHAPTER II

DYNAMIC CONVEX PROGRAMMING

1. INTRODUCTION

We consider herein the dynamic convex programming problem, a deterministic programming problem in which decisions are to be made in each of a countably-infinite number of periods. The cost associated with each period is a stable, proper convex function of not only the decision vector in that period but also of the decision vectors in the subsequent $q - 1$ periods for some fixed $q \geq 2$. This cost function is the same in each period.

The objective is to determine an infinite sequence of decision vectors so as to minimize the average cost per period over the infinite horizon. If we are free to choose the first $q - 1$ decision vectors (i.e., if we ignore the cost in the first $q - 1$ periods) and if the sequence of decision vectors is required to be bounded, then we show that we may restrict attention to an infinite sequence of decision vectors in which the decision vector is the same in all periods. Such a sequence is called stationary. If there is a bounded sequence with a finite average cost per period, then there is a stationary sequence with finite average cost per period. If there is an average cost optimal sequence, then there is a stationary sequence that is average cost optimal.

Dynamic convex models have applications in various fields including large-scale staircase linear programming and control, e.g., Brock and Haurie (1976). Dynamic convex programming models have also been studied by economists interested in long range planning for production, for example by Gale (1967), Brock (1970) and others. Here the decision vector in each period is the ordered pair of available resources in that period and the production to be carried out in the period. The cost depends both on the technologies and the available resources, which in turn depends on the production and available resources in the previous period. The objective is to determine a plan over an infinite horizon that minimizes some specified measure of cost. However, these economic models differ from ours in that (1) they include initial conditions, (2) their optimality criterion is not, in general, average-cost optimality, and (3) they include various additional assumptions and restrictions with economic interpretations.

The theory of dynamic convex programming also has applications to some infinite-horizon dynamic integer programming problems, in particular to the minimum convex-cost dynamic network-flow problem as formulated and solved by Orlin (1981a) and the dynamic matching problem formulated and solved by Orlin (1981b). A prelude to solving both of these problems is to show that a certain stationary continuous solution is optimal for the infinite-horizon integer programming problem with the integrality constraints dropped. Then an optimal integral solution is determined by rounding these continuous solutions so as to maintain optimality for the continuous relaxation of the problem.

The results given herein extend earlier work of Gale (1967) and Brock (1970). Gale proved a lemma which implied the validity of Theorem 1 below in a case in which the convex costs are of a very special type and Brock (1970) extended it. In this paper a more general result is proved which includes Gale's lemma and Brock's extension as a special case. The proof technique used here is a natural extension of that used by Gale.

2. PROBLEM FORMULATION

Let I be either the set of all integers or the set of positive integers, I_t be the set of $i \in I$ with $|i| \leq t$, and $|I_t|$ be the cardinality of I_t. A plan is a sequence $x = (x_i)$ of vectors $x_i \in R^n$ for all $i \in I$. Let q be a positive integer and put $x_{iq} \equiv (x_i, \ldots, x_{i+q-1})$. The cost of x in period i is $c(x_{iq})$, where $c(\cdot)$ is a convex, real or $+\infty$ valued function on R^{qn}, and c has the additional property that its effective domain, denoted dom c, is closed and there is no directional derivative at a vector $x \in \text{dom } c$ that is $-\infty$. This guarantees that $c(\cdot)$ is "stable" in the sense of Rockafellar (1967).

A plan x is feasible if $c(x_{iq})$ is finite for all $i \in I$. The average cost of x is

$$a(x) \equiv \varliminf_{t \to \infty} |I_t|^{-1} \sum_{i \in I_t} c(x_{iq}) .$$

The dynamic convex programming problem is to determine a feasible plan in a given class that is (average-cost) optimal in the class, i.e., for any other feasible plan y in the class we have $a(x) \leq a(y)$.

3. OPTIMALITY OF STATIONARY PLANS

A plan $x = (x_i)$ is called bounded if $\|x\|$ is finite, where the origin and there is a hyperplane strictly separating U therefrom. Thus, there is an n-vector p and a positive real number α (which exists since U is compact) such that $p(v - u) \geq \alpha$ for all $(v - u) \in U$. Substituting x_i for u and x_{i+1} for v, and summing over i we obtain

$$t\alpha \leq \sum_{i=1}^{t} p(x_{i+1} - x_i) = p(x_{t+1} - x_1) \leq 2n\|p\| \cdot \|x\|$$

which is impossible for large t, completing the proof that there is a feasible stationary plan.

Let $(\hat{u},\hat{u})$ be optimal for the convex program

$$\begin{aligned} &\text{minimize} \quad c'(u,v) \\ &\text{subject to} \quad (v - u) = 0 \; . \end{aligned} \tag{1}$$

An optimum exists since there is a feasible stationary plan, dom c' is compact, and c' is continuous on dom c'. We will show that the stationary plan $y = \hat{u}^{\infty}$ satisfies the conditions of the theorem.

The program (1) is stable as there is no directional derivative with value $-\infty$ for any $(u,v) \in$ dom c'. Thus there is a Kuhn-Tucker vector p such that:

$$c'(\hat{u},\hat{u}) \leq c'(u,v) + p(v - u) \quad \text{for all} \quad (u,v) \in R^{2n} \; .$$

On substituting (x_i, x_{i+1}) for (u,v), letting ℓ_t denote the least element of I_t, and summing we get

$$|I_t|^{-1} \sum_{i\in I_t} (c(\hat{u},\hat{u}) - c(x_i, x_{i+1})) = |I_t|^{-1} \sum_{i\in I_t} (c'(\hat{u},\hat{u}) - c'(x_i, x_{i+1}))$$

$$\leq |I_t|^{-1} \sum_{i\in I_t} p(x_{i+1} - x_i) = |I_t|^{-1} p(x_{t+1} - x_{\ell_t}) \leq 2|I_t|^{-1} n\|p\|\|x\| .$$

Taking the limit superior proves the claim.

Finally, we reduce the case $q > 2$ to the case $q = 2$. Suppose $c(\cdot)$ is defined on R^{qn} for $q > 2$, and let $w_i = x_{iq}$ for each $i \in I$. Let V be the set of qn-vectors such that the last $(q - 1)n$ components of the first vector is equal to the first $(q - 1)n$ components of the second vector. Let

$$c'(w_i, w_{i+1}) = \begin{cases} c(w_i) & \text{if } (w_i, w_{i+1}) \in V \\ \infty & \text{otherwise .} \end{cases}$$

Then c' is stable. To see this note that c' is continuous on dom c' which is closed, and any directional derivative of c' at a point $(u,v) \in$ dom c' in the direction (u',v') is greater than or equal to the directional derivative of c at a point u in the direction u' and is thus not $-\infty$. By construction of c', the plan $x = (x_i)$ is feasible (resp., bounded, stationary) if and only if that is so of the corresponding plan $w = (w_i)$. We have proved that the result is true for c' on R^{2qn}, and hence the result is true for c on R^{qn}. ∎

COROLLARY 1. There is a feasible stationary plan for the dynamic convex programming problem if and only if there is a bounded feasible plan.

COROLLARY 2. An optimal stationary plan for the dynamic convex programming problem is also an optimal bounded plan.

COROLLARY 3. A stationary plan u^∞ is an optimal bounded plan for the dynamic convex programming problem if and only if $v = u$ minimizes $c(v^q)$ and $c(u^q)$ is finite.

EXAMPLE 1. (Optimal unbounded plans.) We note that the above results are not true if the restriction of boundedness is dropped. In particular, let $x_i \in R$ for $i = 1, 2, \ldots,$ and let c be defined on the plane as follows:

$$c(u,v) = \begin{cases} u - v & \text{if } -1 \leq u - v \leq 0 \\ \infty & \text{otherwise} . \end{cases}$$

Each stationary plan has zero average cost per period, whereas the plan $x = (x_i)$ with $x_i = i$ for each i has an average cost per period of -1. Furthermore, the existence of feasible unbounded plans does not imply the existence of feasible bounded plans as can be seen if we use the following cost function:

$$c(u,v) = \begin{cases} 0 & \text{if } v - u = 1 \\ \infty & \text{otherwise} . \end{cases}$$

4. EXTENSIONS AND APPLICATIONS

Initial Conditions

In many applications, such as those studied by Gale (1967) and Brock (1970), there are initial conditions on the first decision vector

x_1. Adding such conditions makes the problem significantly more difficult and is beyond the range of this work. To appreciate this, observe that the existence of bounded feasible plans does not guarantee the existence of stationary feasible plans as the following example illustrates.

EXAMPLE 2. (Initial conditions precluding stationary plans.) Let $x_i \in R$ for $i = 1, 2, \ldots$ and let c be defined on the plane as follows:

$$c(u,v) = \begin{cases} 0 & \text{if } u + v = 0 \\ \infty & \text{otherwise} . \end{cases}$$

If we add the restriction that $x_1 = -1$, then the unique feasible plan is given by $x_i = (-1)^i$, which is periodic but not stationary.

Of course, the above example is quite easy to solve; however, the dynamic convex programming problem appears quite difficult in general when initial conditions must be satisfied--even when the problem is specialized to linear programming as below. This author does not know of any efficient solution technique, and it is an open question whether the problem is NP-hard.

Having said this, it is also important to recognize that the imposition of initial conditions will cause no substantial difficulties in many practical problems. The reason is this. Suppose we have found $x = u^\infty$ to be an optimal bounded plan ignoring the initial condition that x_1 is given. Then, in practice, it is often easy to construct vectors $x_2, \ldots, x_p$ such that $x' = (x_1, x_2, \ldots, x_p, u, u, \ldots)$ is a feasible plan for some (small) integer p. For example if c is real valued, then the plan $x' = (x_1, u, u, \ldots)$ is feasible. Also x' has the same average cost as x and so x' is an optimal bounded plan that satisfies the initial condition.

Periodic Cost Functions

The previous analysis considers the case in which the same cost function $c(\cdot)$ is associated with each period. Suppose instead that costs are periodic, i.e., there exist p distinct cost functions $c_1(\cdot), \ldots, c_p(\cdot)$, and c_j is the cost function in period $j + kp$ for $k = 0, 1, \ldots$ and $j = 1, \ldots, p$. As before the objective is to find a sequence with minimum average cost. This is called the periodic convex programming problem with period p.

The periodic case is easily transformed into the dynamic case as follows. First, as before, the case $q > 2$ reduces to the case $q = 2$, which we consider here. For a given plan $x = (x_i)$, form a plan (w_i) with $w_i = (x_{(i-1)p+1}, \ldots, x_{ip})$ for $i = 1, 2, \ldots$. Then define $c'(\cdot)$ on R^{2np} as follows. If $u_i, v_i \in R^n$, so $u = (u_1, \ldots, u_p)$, $v = (v_1, \ldots, v_p) \in R^{np}$, then put

$$c'(u,v) \equiv c_p(u_p, v_1) + \sum_{i=1}^{p-1} c_i(u_i, u_{i+1}) .$$

It is easily verified that $c'(w_1, w_2) + \cdots + c'(w_k, w_{k+1}) = c_1(x_1, x_2) + \cdots + c_p(x_{kp}, x_{kp+1})$. It follows from this formula that if I is the set of positive integers, then the original periodic convex program with period p is equivalent to the stationary convex program with cost function c'. A similar development suffices for the case that I is the set of all integers.

Observe that there is a one-to-one correspondence between the stationary plans w^∞ for the transformed problem and the periodic plans (x_i) with period p (i.e., $x_i = x_{i+p}$ for all $i \in I$) for the original problem. Thus Theorem 1 and its Corollaries have obvious analogs for the periodic problem with period p in which stationary plans are replaced everywhere by periodic plans with period p. In particular, if $u = (u_1, \ldots, u_p)$ is chosen so as to minimize $c'(u,u)$, then the periodic plan $x = (x_i)$ with $x_i = u_{i(\mathrm{mod}\ p)+1}$ for all i is optimal.

Specialization to Linear Programming

If the cost function is piecewise linear in the effective domain, then the directional derivatives in the effective domain are not $-\infty$ and the effective domain is closed. Thus the conditions of Theorem 1 and its corollaries apply. In particular, the results of Theorem 1 apply to the "dynamic linear programming problem" (or infinite staircase linear program), which is the specialization of the dynamic convex programming problem to the case in which $c(\cdot)$ is linear on its effective domain and the latter is in turn polyhedral. It is this specialization which is applied to help solve both the dynamic network flow problem (Orlin (1981a)) and the dynamic matching problem (Orlin (1981b)).

Let $c \in R^n$, $b \in R^m$, and let $A_1, \ldots, A_q$ be $m \times n$ real-valued matrices. A plan $x = (x_i)$ for $i = 1, 2, 3, \ldots$ is <u>feasible</u>

if for each $i = 1, 2, \ldots$

$$A_1 x_i + A_2 x_{i+1} + \cdots + A_q x_{i+q-1} = b \quad \text{and} \quad x_i \geq 0 .$$

The <u>dynamic linear programming problem</u> is to find a feasible plan that is (average-cost) optimal.

Observe that the dynamic linear programming problem is indeed a specialization of the dynamic convex programming problem, for given the former problem, we can choose

$$c'(u_1, \ldots, u_q) = \begin{cases} cu_1 & \text{if} \quad A_1 u_1 + \cdots + A_q u_q = b \\ & \text{and} \quad u_1, \ldots, u_q \geq 0 \\ \infty & \text{otherwise} . \end{cases}$$

Theorem 1 and its corollaries show that an optimal stationary plan for the dynamic linear programming problem is also an optimal bounded plan. By Corollary 3, an optimal stationary plan may be found by solving the linear program

$$\begin{aligned} &\text{minimize} \quad cu \\ &\text{subject to} \quad (A_1 + \cdots + A_q)u = b \quad \text{and} \quad u \geq 0 . \end{aligned}$$

ACKNOWLEDGMENT

I wish to thank my thesis supervisor Professor Arthur F. Veinott, Jr. for the help and support that he gave me. In particular, I am grateful for his help in properly formulating the problem and pointing out to me several relevant references. I am especially appreciative of the painstaking care that he took in helping to improve the exposition.

REFERENCES

Brock, W. A. (1970). On Existence of Weakly Maximal Programmes in a Multi-Sector Economy. Rev. Econ. Stud. 37, 275-280.

Brock, W. A. and A. Haurie (1976). On Existence of Overtaking Optimal Trajectories Over an Infinite Time Horizon. Math. Oper. Res. 1, 337-346.

Gale, D. A. (1967). On Optimal Development in a Multi-Sector Economy. Rev. Econ. Stud. 34, 1-18.

Orlin, J. B. (1981a). Minimum Convex-Cost Dynamic Network Flows. Chapter 4 of this thesis.

Orlin, J. B. (1981b). Maximum-Weight Dynamic Matchings. Work in progress.

Rockafellar, R. T. (1967). Duality and Stability in Extremum Problems Involving Convex Functions. Pacif. J. Math. 21, 167-187.

CHAPTER III

MAXIMUM-THROUGHPUT DYNAMIC NETWORK FLOWS

1. INTRODUCTION

The Model and Problem Formulation

Herein, we present and solve the maximum-throughput dynamic network-flow problem, an infinite-horizon integer programming problem that involves flows evolving over time. In a given finite network, referred to henceforth as the "static network", flow is to be sent in all of the arcs in each of an infinite number of periods. Each arc has an upper and a lower bound on the arc flow and an integral transit time, which is the number of periods that it takes for flow to pass through the arc. For a flow to be feasible it must satisfy all upper- and lower-bound constraints, and also satisfy conservation of flow at each node in each period except for the first few periods during which the flow is "initialized". For a given feasible dynamic flow, the throughput in period p is the net amount of flow in transit in that period, i.e., flow that has been initiated in some arc in period p or earlier but has not reached the head of the arc by the beginning of period p. In Section 2, we demonstrate that because there is conservation of flow, the throughput is the same for all periods except possibly the first few.

The maximum- (resp., minimum-) throughput dynamic network-flow problem is to determine a feasible dynamic flow with maximum (resp., minimum) throughput. The maximum- and minimum-throughput problems are equivalent since each may be reduced to the other by multiplying the flows and bounds by -1. While this problem is apparently new, it has antecedents in the literature. For example, it is related to the maximum network-flow problem. Both the dynamic max-flow min-cut formula and the application of dynamic network flows to vehicle scheduling stem naturally from the ideas presented by Ford and Fulkerson (1962).

The minimum-throughput dynamic network-flow problem is closely related to the finite-horizon dynamic maximum-flow problem as presented and solved by Ford and Fulkerson (1958). As is discussed in detail in Section 4, the finite-horizon problem may be transformed into a special case of the minimum-throughput dynamic network-flow problem. Furthermore, both the maximum capacity cut derived in Section 2 and the cut derived by Ford and Fulkerson (1958) are induced by optimum dual solutions to associated static network-flow problems.

Finally, the maximum-throughput dynamic network-flow problem is a specialization of the minimum convex-cost dynamic network-flow problem which is presented and solved in Orlin (1981b). The objective in the latter problem is to find a feasible integral flow with fixed throughput so as to minimize the long-run (Cesàro) average cost per period.

Optimality of Stationary Flows and the Max-Throughput Min-Cut Theorem

We show that if there is a feasible bounded dynamic flow, then the supremum of the throughputs of all feasible dynamic flows equals that of all feasible dynamic flows that are stationary, i.e., the flow in each arc is the same in all periods. If we restrict attention to stationary flows, the maximum-throughput dynamic network-flow problem reduces to that of finding a maximum-profit feasible circulation in the static network with the unit profit of the flow in an arc being its transit time. A stationary flow is obtained by repeating the static circulation in each period over the infinite horizon, and the throughput of the dynamic flow is the profit of the static circulation.

The optimality of stationary flows is proved as part of the main theoretical result in Section 2. Also proved is the Max-Throughput Min-Cut Theorem, which states that the maximum throughput of a feasible flow is the minimum capacity of a cut, where a cut is not defined in terms of the original static network, but rather in terms of an infinite "dynamic network"; each node of the dynamic network is an ordered pair representing a node of the static network and a period of time. As part of the proof we construct a cut and a feasible stationary flow whose throughput is equal to the capacity of the cut.

An Application: Minimizing the Number of Vehicles to Meet a Fixed Periodic Schedule

Consider a transportation firm (e.g., an airline) that must schedule vehicles (e.g., airplanes) each day over an infinite horizon so that certain routes are traveled at the same time each day. In addition, there are certain other routes that the firm may schedule, and deadheading is permitted. The objective is to determine a feasible schedule that minimizes the number of vehicles needed.

Dantzig (1962), in consulting work for United Airlines, considered the above problem under the added restriction that a schedule is stationary, and he modeled the problem as the static version of the minimum-throughput dynamic network-flow problem. By the results of the previous section, the induced stationary schedule he found is optimal over the class of all schedules. As Dantzig and his collaborators observed, the stationary flight schedules induce vehicle schedules that are periodic, but do not necessarily repeat daily.

2. THE MAXIMUM-THROUGHPUT DYNAMIC NETWORK-FLOW PROBLEM

The Static Network and Problem Formulation

A static network is a quintuple $G = (N, A, t, \ell, u)$ in which $N = \{1, \ldots, n\}$ is the node set and A is the arc set of a directed graph, possibly containing loops (i.e., arcs joining a node to itself) and multiple arcs between two nodes. Associated with each arc a is a transit time t_a, which is the (possibly negative) integral number of periods that it takes for flow to travel from the tail of the arc to its head. Also associated with each arc a are (possibly $+\infty$) upper and (possibly $-\infty$) lower bounds u_a and ℓ_a on the flow initiated therein in each period, with $u_a \geq \ell_a$. These networks have also been referred to in the literature as "networks with transit times", for example by Lawler (1976). If a flow begins in one period in the tail of an arc with a negative transit time, then the flow arrives at the arc's head at an earlier period. This somewhat anomalous situation is interpreted in the vehicle-scheduling problem of Section 3 as airplanes that may cross the international date line and arrive the day before they left. Further interpretation of negative transit times is given in an application to cyclic capacity scheduling by Orlin (1981b).

If the tail and head of arc a are i and j

respectively, then we may denote the arc as (i,j) in those cases in which no ambiguity will result. (There may be several arcs with the same tail and head.) For each node $i \in N$, let H_i (resp., T_i) be the set of arcs whose head (resp., tail) is i. Let $t_{max} = \max_{a \in A} |t_a|$.

A <u>dynamic</u> <u>flow</u> $x = (x_a^p)$ is <u>feasible</u> if it satisfies the upper- and lower-bound constraints:

$$\ell_a \leq x_a^p \leq u_a \quad \text{for} \quad a \in A,\ p = 1, 2, 3, \ldots \tag{2.1}$$

and satisfies <u>conservation-of-flow</u> constraints at each node after period t_{max}, i.e.,

$$\sum_{a \in T_i} x_a^p = \sum_{a \in H_i} x_a^{p-t_a} \quad \text{for} \quad i \in N,\ p > t_{max} . \tag{2.2}$$

Conservation of flow is not necessarily satisfied during the first t_{max} periods, which we may view as the initialization periods.

Define the flow in transit in arc a in period p with a feasible dynamic flow x to be

$$\sum_{j=p-t_a+1}^{p} x_a^j . \tag{2.3}$$

The <u>throughput</u> of x is the sum f_x of the flows in transit in all arcs in period t_{max}.

If $t_a \geq 1$, then the flow in transit in arc a in period p is the net amount of flow initiated in arc a in or prior to period p but after period $p - t_a$. If $t_a \leq -1$, the flow in transit in arc a in period p is

$$- \sum_{j=p+1}^{p-t_a} x_a^j \tag{2.3'}$$

which is equal to (2.3) by the conventional method for performing summations in which the lower index is greater than the upper index. If $t_a = 0$, there is no flow in transit in arc a in each period.

To see that the sum (2.3') is really the appropriate interpretation of (2.3), consider one unit of flow sent from the tail of arc a in period p and reaching the head of a in period $p + t_a$. With regard to the conservation-of-flow constraints this unit flow is equivalent to sending negative-one unit of flow from the head of a in period $p + t_a$ and reaching the tail of a in period p. In fact, we can replace arc $a = (i,j)$ with arc $\alpha = (j,i)$ such that $t_\alpha = -t_a$, $\ell_a = -u_a$, and $u_\alpha = -\ell_a$. The definition of flow in transit given in (2.3') is correct because one unit of flow in arc a should give the same contribution to the flow in transit as negative-one unit of flow in arc α.

LEMMA 1. The sum of the flows in transit in each period $p \geq t_{max}$ of a feasible dynamic flow is equal to its throughput.

PROOF. Lemma 1 is shown later in this section to be a special case of Lemma 4. ■

An intuitive proof of the above lemma is as follows.

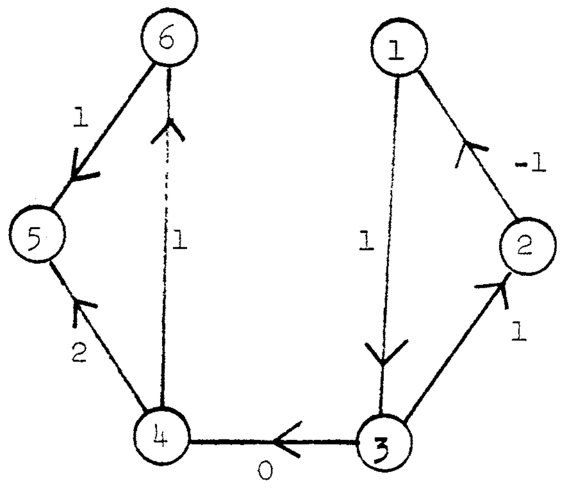
In order to preserve conservation of flow, all flow in transit after period t_{max} must be sent forward upon arriving at the head of an arc. Therefore, the total amount of flow in transit is constant after period t_{max}.

The maximum- (resp., minimum-) throughput dynamic network-flow problem is to determine a feasible dynamic flow with maximum (resp., minimum) throughput. In this section we show that if there is a feasible bounded flow, then the supremum of the throughputs of all feasible flows is the same as that of all stationary feasible flows. Moreover, we define and prove a dynamic analog of the max-flow min-cut theorem of Ford and Fulkerson (1956).

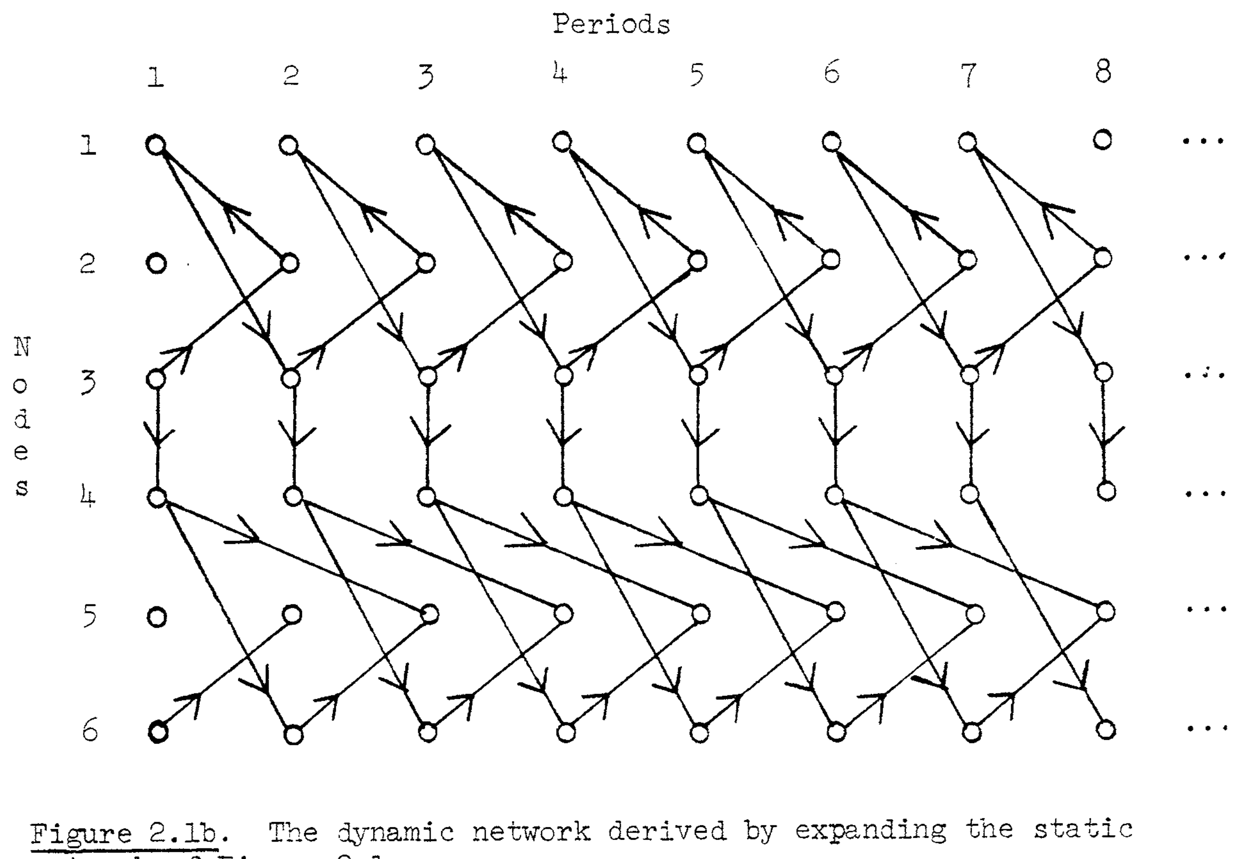
The Dynamic Network

Let $G = (N, A, t, \ell, u)$ be a static network. In order to express flows evolving over time as ordinary network flows expand G into an infinite network, called the dynamic network, and denote it by $G^{\infty} = (N^{\infty}, A^{\infty}, \ell^{\infty}, u^{\infty})$ where $N^{\infty} = \{i^p : i \in N \text{ and } p \in \{1,2,3, \ldots\}\}$. Node $i^p \in N^{\infty}$ represents node i of N in period p. For each arc $a = (i,j) \in A$ and for $p \geq \max(1 - t_a, 1)$ there is an arc $a^p = (i^p, j^{p+t_a}) \in A^{\infty}$, representing the fact that flow may be sent from node i in period p through arc a and arrive at node j in period $p + t_a$. Furthermore, the lower and upper bounds for the flow in arc a^p are the same as those for arc a.

Figures 2.1a and 2.1b show a static network and the corresponding dynamic network. Here and in other diagrams the numbers on arcs refer to transit times.



<u>Figure 2.1a</u>. A static network. The arc numbers are the transit times.



<u>Figure 2.1b</u>. The dynamic network derived by expanding the static network of Figure 2.1a.

The technique of expressing flows over time by expanding the network is now standard. It was used by Ford and Fulkerson (1958) and by many others including Lawler (1976).

Let $x = (x_a^p)$ be a feasible dynamic flow. Let us view x_a^p as the flow in arc a^p of A^∞. Then constraint (2.2) states that conservation of flow is satisfied at node i^p for $p > t_{max}$. Thus a feasible dynamic flow is a circulation in the dynamic network except that conservation of flow is not necessarily satisfied at arcs i^p for $p \leq t_{max}$.

<u>Preliminaries: Paths, Copies, and Cuts</u>

A <u>path</u> in a network (dynamic or not) is an alternating sequence of nodes and arcs $i_o, a_1, \ldots, a_k, i_k$ such that for each $j = 1, \ldots, k$ either a_j has head i_j and tail i_{j-1} or else it has head i_{j-1} and tail i_j. In the former case the arc is called a <u>forward arc</u> of the path; in the latter case it is called a <u>backward arc</u>. A path is called <u>directed</u> if every arc is a forward arc and <u>simple</u> if no node is repeated.

In a static network the <u>transit time</u> of a path is the sum of the transit times of the forward arcs of the path minus the sum of the transit times of the backward arcs.

An example illustrating these concepts is given in Figure 2.2. The path from node 1 to node 4 has forward arcs (2,3) and (3,4). The transit time of this path is the sum of the transit times of (2,3) and (3,4) minus the transit time of (1,2). This value is 1. The path may also be viewed as a path from node 4 to node 1 with transit time -1.

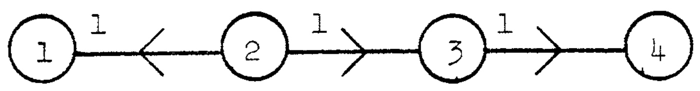


Figure 2.2. A path from node 1 to node 4 with unit transit time.

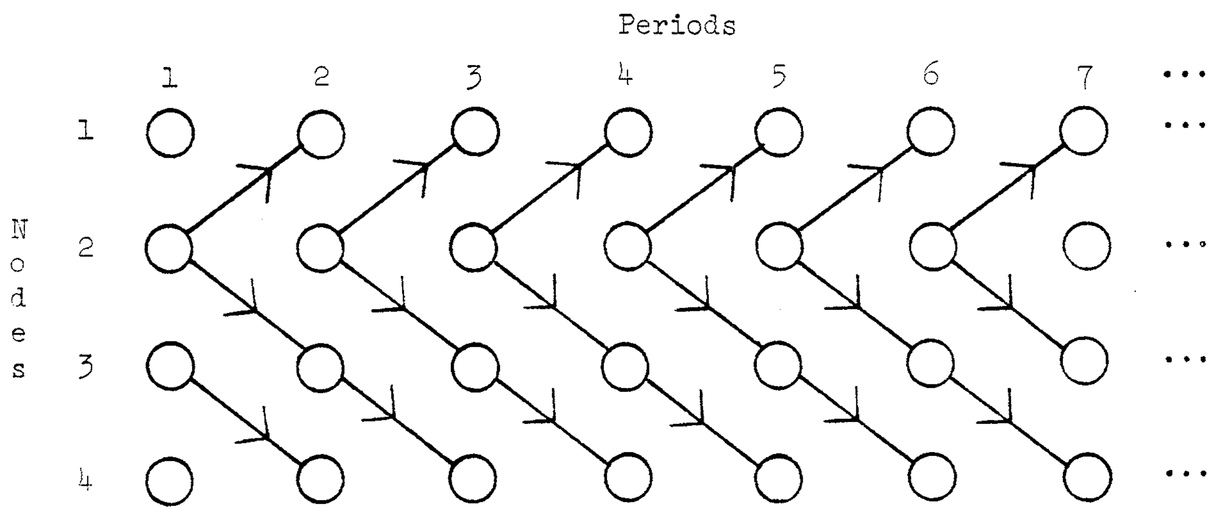


Figure 2.3. The dynamic network derived from the static network of Figure 2.2. The path from 1^p to 4^{p+1} is the p^{th} copy of path in Figure 2.2 for $p \geq 2$.

A cycle is a path in which the initial node is the same as the final node. A cycle is simple if no node is repeated, except that the initial node and the final node are the same.

A static flow $y = (y_a)$ is called feasible if it satisfies the upper and lower bound constraints (2.4) and is a circulation in the static network G, i.e., satisfies (2.5).

$$\ell_a \leq y_a \leq u_a \quad \text{for} \quad a \in A \tag{2.4}$$

and

$$\sum_{a \in H_i} y_a - \sum_{a \in T_i} y_a = 0 \quad \text{for} \quad i \in N . \tag{2.5}$$

Let C be a cycle of G. A flow around C of k units is a static flow $y = (y_a)$ such that

$$y_a = \begin{cases} k & \text{if } a \text{ is a forward arc of } C \\ -k & \text{if } a \text{ is a backward arc of } C \\ 0 & \text{if } a \text{ is not an arc of } C . \end{cases}$$

It is easily verified that a flow around C satisfies the conservation-of-flow constraint (2.5).

If $a \in A$ and $p \geq 1$, then arc $a^p \in A^\infty$ will be called the p^{th} copy of arc a, or simply a copy of a. The p^{th} copy of arc a is not defined for $p \leq -t_a$. Similarly i^p is called the p^{th} copy of node i. Let $P = i_o, a_1, \ldots, a_k, i_k$ be a path in G. Then the p^{th} copy of P is the path P', if one exists, in G^∞ with k arcs such that the initial node is i^p and such that the j^{th} arc of P'

is that copy of arc a_j whose tail is the head of the $(j-1)^{th}$ arc of p'. The definition is illustrated in Figure 2.3, which is the dynamic network corresponding to the path in Figure 2.2. The path from 1^p to 4^{p+1} is the p^{th} copy of the path from node 1 to node 4; this path is not defined for $p = 1$.

LEMMA 2. Let C be a simple cycle of transit time $t \geq 1$ in the static network. Then the infinite number of copies of C in the dynamic network comprise t node-disjoint infinite paths therein.

PROOF. First, ignore the finite number of copies that are not defined. Let the k^{th} copy be the first copy that is defined, and let i^k be the first node of this path. For each $p \geq k$ it is easily verified that the p^{th} copy of C is a path from i^p to i^{p+t}. If we concatenate the p^{th} copies for $p = k, k + t, k + 2t, k + 3t, \ldots$, then we obtain the first infinite path. We obtain $t - 1$ additional paths with initial nodes $i^{k+1}, \ldots, i^{k+t-1}$ in the same manner. These paths are node-disjoint since for $j \neq i$, the node j^p may appear in at most one of the copies of C. ■

The above lemma is illustrated in Figures 2.4a and 2.4b. The copies of the cycle in Figure 2.4a partition into two node-disjoint infinite paths in the dynamic network, as illustrated in Figure 2.4b.

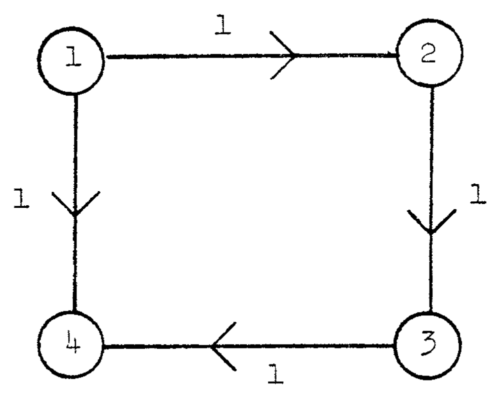


Figure 2.4a. A static network that is a simple cycle of transit time two.

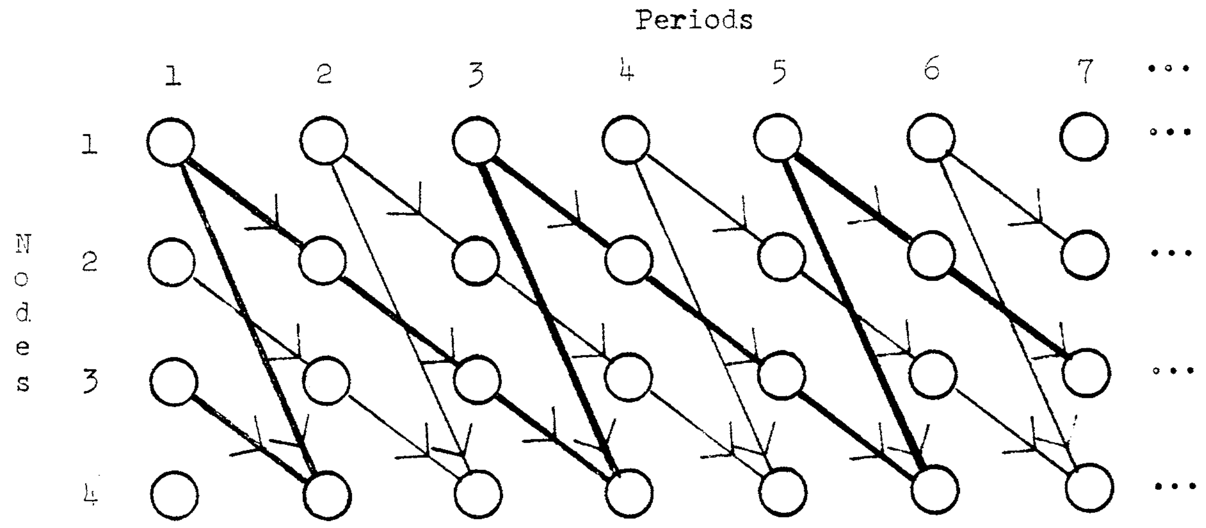


Figure 2.4b. The dynamic network associated with the static network of Figure 2.4a. It has two node disjoint infinite paths, one of which is boldface in the diagram.

A <u>cut</u> in G^∞ is a partition of the nodes of N^∞ into disjoint subsets $S, \overline{S}$ such that S is finite, and $i^p \in S$ for $i \in N$ and $p \leq t_{max}$. This guarantees that conservation-of-flow is satisfied at each node of $\overline{S}$. The nodes of S are called the <u>source nodes</u> while the nodes of $\overline{S}$ are called the <u>sink nodes</u>. A cut $(S,\underline{S})$ is called <u>monotone</u> if $i^p \in \overline{S}$ implies that $i^{p+1} \in \overline{S}$ for all $i \in N$ and $p > t_{max}$.

Given two disjoint sets S, T of nodes in N^∞, let $A(S,T)$ denote the subset of arcs in A^∞ with tail in S and head in T. The <u>upper capacity</u> of a cut $(S,\overline{S})$ is defined to be

$$\sum_{a^p \in A(S,\overline{S})} u_a - \sum_{a^p \in A(\overline{S},S)} \ell_a \qquad (2.6a)$$

and may be interpreted as the maximum net flow from source nodes to sink nodes. The <u>lower capacity</u> of cut $(S,\overline{S})$ is defined to be

$$\sum_{a^p \in A(S,\overline{S})} \ell_a - \sum_{a^p \in A(\overline{S},S)} u_a \qquad (2.6b)$$

and may be viewed as the minimum net flow from source nodes to sink nodes. The upper (resp., lower) capacity is defined to be $+\infty$ (resp., $-\infty$) whenever (2.6a) (resp., (2.6b)) involve any infinite numbers.

A cut is illustrated in Figure 2.5. The dynamic network is the same as in Figure 2.4b. The upper and lower bounds are given in Table 2.1, and the nodes of S are the white nodes of Figure 2.5. The upper capacity of the cut is 2 while the lower capacity is -4.

Arc	Transit Time	Upper Bound	Lower Bound
(1,2)	1	2	0
(1,4)	1	1	1
(2,3)	1	0	-1
(3,4)	1	1	0

Table 2.1. The parameters for the arcs of Figure 2.4a.

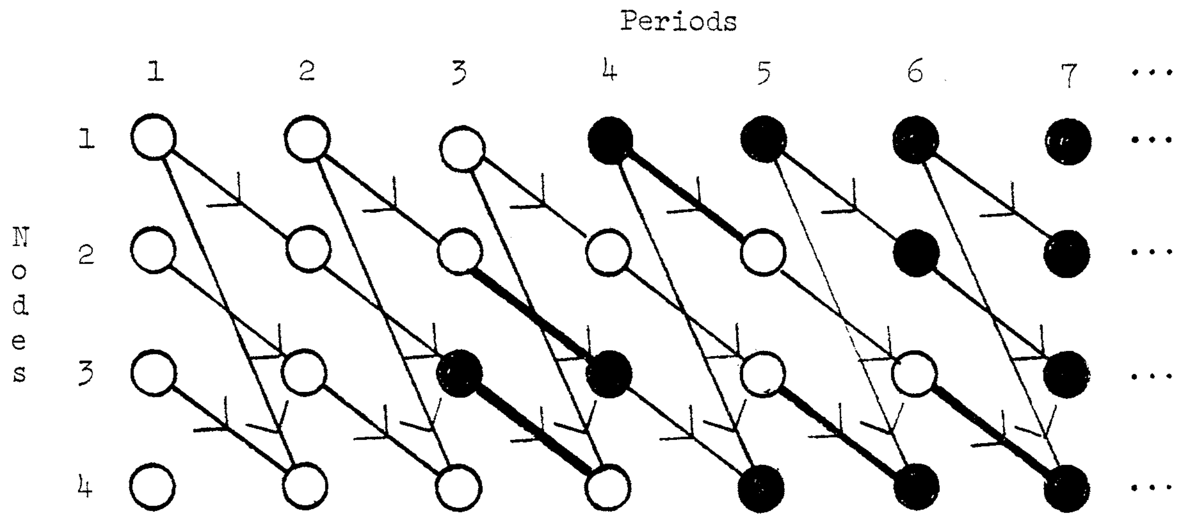


Figure 2.5. A cut in the dynamic network of Figure 2.4. The white nodes are in S. The boldfaced lines are in either $A(S,\overline{S})$ or $A(\overline{S},S)$.

Max-Throughput Min-Cut and the Optimality of Stationary Flows

A dynamic flow $x = (x_a^p)$ is called bounded if its supremum norm is finite and stationary if $x_a^p = x_a^{p+1}$ for each $a \in A$ and $p = 1, 2, 3, \ldots$. A feasible static flow $y = (y_a)$ induces a stationary dynamic flow $y^\infty = (x_a^p)$ with $x_a^p = y_a$ for $p = 1, 2, \ldots$ and $a \in A$. In this subsection we prove our main theoretical results. First, the upper (resp., lower) capacity of any cut provides an upper (resp., lower) bound on the throughput of a feasible flow. Second, if there is a feasible bounded flow, there is a cut whose upper (resp., lower) capacity equals the supremum (resp., infimum) of the throughputs of the stationary feasible dynamic flows.

Let x be a feasible flow and let $(S,\overline{S})$ be a cut. We define the flow across $(S,\overline{S})$ to be

$$\sum_{a^p \in A(S,\overline{S})} x_a^p - \sum_{a^p \in A(\overline{S},S)} x_a^p .$$

LEMMA 4. Let x be a feasible dynamic flow and let $(S,\overline{S})$ be a cut. Then the throughput is equal to the flow across $(S,\overline{S})$ and so is bounded above by the upper capacity of the cut and bounded below by its lower capacity.

PROOF. It is clear that the flow across $(S,\overline{S})$ is bounded above by the upper capacity and bounded below by the lower capacity. It remains to show that the flow across $(S,\overline{S})$ is the throughput.

Consider first the case that $S = \{i^p \in N^\infty : p \leq t_{max}\}$. Then the flow across $(S,\overline{S})$ is exactly the flow in transit in period t_{max}, which is the throughput.

We now prove the result inductively. Let $(S,\overline{S})$ be a cut. Let $S' = S - \{j\}$, where $j = i^r$ is any node of S for which $r > t_{max}$. Then the flow across $(S,\overline{S})$ is equal to the flow across $(S',\overline{S}')$. To see this, note that the difference of the flows across the two cuts is

$$\sum_{a^p \in H_j} x_a^p - \sum_{a^p \in T_j} x_a^p$$

which is zero by conservation of flow.

The lemma now follows inductively, as we may start with any cut $(S,\overline{S})$ and progressively move one node at a time from S to $\overline{S}$ without altering the flow across the cut. Eventually we arrive at the set $S' = \{i^p \in N^\infty : p \leq t_{max}\}$. ■

PROOF OF LEMMA 1. The net flow in transit in period q is the flow across $(S,\overline{S})$ where $S = \{i^p \in N^\infty : p \leq q\}$. Thus, Lemma 1 is a special case of Lemma 4. ■

THEOREM 5. If there is a feasible bounded dynamic flow, then

1° (Sufficiency of Stationary Flows) the supremum (resp., infimum) of the throughputs of all feasible dynamic flows equals that of all feasible stationary flows, with the last supremum (resp., infimum) being attained if it is finite.

2° (Max-Throughput Min-Cut) Moreover, the supremum (resp., infimum) of those throughputs equals the minimum (resp., maximum) upper (resp., lower) capacity of a cut, and this cut may be taken to be monotone.

3° (Integrality) If also the upper and lower bounds are integral, then the stationary flows in 1° may be taken to be integral as well.

PROOF. First suppose that all upper and lower bounds on arc flows are finite. Consider the problem of finding a feasible static flow $y = (y_a)$ that maximizes

$$f_y = \sum_{a\in A} t_a y_a \tag{2.7}$$

Furthermore, as is easily seen from (2.3) and (2.7), $f_y = f_{y^\infty}$, i.e., f_y is the throughput of the stationary flow y^∞ induced by the static flow y. The dual of the above static network-flow problem is the linear program (2.8).

$$\text{Minimize} \quad \sum_{a\in A} \alpha_a u_a - \sum_{a\in A} \beta_a \ell_a \tag{2.8a}$$

subject to

$$\lambda_j - \lambda_i + \alpha_a - \beta_a = t_a \quad \text{for all} \quad a = (i,j) \in A \tag{2.8b}$$

$$\alpha_a, \beta_a \geq 0 \qquad \text{for} \quad a \in A . \tag{2.8c}$$

Let $x^+ = \max(0,x)$, and $x^- = \max(0,-x)$. Since $u_a \geq \ell_a$ for each $a \in A$, we need consider only those solutions to (2.8) for which $\alpha_a = (t_a + \lambda_i - \lambda_j)^+$ and $\beta_a = (t_a + \lambda_i - \lambda_j)^-$. These solutions are feasible for any choice of λ.

Let (λ, α, β) be an integer-valued feasible solution to (2.8) and put $\lambda_{min} = \min_i \lambda_i$. Let p^* be some integer greater than $t_{max} - \lambda_{min}$, and let $S = \{i^p \in N^\infty : p - \lambda_i \leq p^*\}$. Then $(S,\overline{S})$ is a monotone cut whose upper capacity is equal to the objective value of (2.8a) for (λ, α, β). To see this, suppose $a = (i,j)$. If $p - \lambda_i \leq p^*$ and $p - \lambda_j + t_a \geq p^* + 1$, then $a^p = (i^p, j^{p+t_a}) \in A(S,\overline{S})$, and there are $\alpha_a = (t_a + \lambda_i - \lambda_j)^+$ such copies of arc a. Similarly, $a^p \in A(\overline{S},S)$ if $p - \lambda_i \geq p^* + 1$ and $p - \lambda_j + t_a \leq p^*$, and there are $\beta_a = (t_a + \lambda_i - \lambda_j)^-$ such arcs. Thus the upper capacity of the cut is the objective value in (2.8a).

If there is a feasible static flow, then there is an optimal basic static flow y with objective value z and an integral optimal solution (λ, α, β) to (2.8) also with objective value z. The static flow y induces a feasible stationary flow y^∞ with throughput $f_y = z$ and solution (λ, α, β) induces a monotone cut $(S,\overline{S})$ with upper capacity equal to z. By Lemma 4, this stationary flow is optimal.

Consider next the case in which the bounds on some arcs may be infinite. If there is a sequence of feasible stationary flows with throughput unbounded from above, then by Lemma 4 each cut has infinite upper capacity. Suppose instead that there is no sequence of feasible stationary flows with throughput unbounded from above. Then since there is a feasible bounded dynamic flow $x = (x_a^p)$, there is a real number M that is a strict upper bound for both the absolute arc flows $|x_a^p|$ for all a and p and also for the absolute arc flows in each basic static flow (if any exist).

Consider next the static network $G' = (N, A, t, \ell', u')$ where $\ell'_a = \max(\ell_a, -M)$ and $u'_a = \min(u_a, M)$. The resulting dual program (2.8) is feasible and its objective value is bounded below by the throughput f_x by Lemma 4 and what was shown above. Hence, there is a maximum-profit basic static flow y and an optimal dual solution (λ, α, β) for (2.8). By complementary slackness, $u'_a = M$ implies $\alpha_a = 0$, and $\ell'_a = -M$ implies $\beta_a = 0$. Let $(S,\overline{S})$ be the monotone cut induced by (λ, α, β). Then no arc of $A(S,\overline{S})$ (resp., $A(\overline{S},S)$) has an upper (resp., lower) bound equal to M (resp., $-M$). Hence, the capacity of the cut is unaltered if each arc bound of $\pm M$ is replaced by $\pm\infty$, and the throughput of the stationary flow y^∞ induced by y is the upper capacity of $(S,\overline{S})$ in G^∞ from what was shown above, completing the proof.

Finally, the result for the minimum-throughput problem is immediate from that for the maximum-throughput problem because the former reduces to the latter on replacing each (x^p_a, ℓ_a, u_a) by $(-x^p_a, -u_a, -\ell_a)$. ■

EXAMPLE. Consider the static network described in Table 2.2. The corresponding static and dynamic networks are portrayed in Figures 2.6a and 2.6b. A minimum upper-capacity cut is given in Figure 2.6b. The capacity of this cut is 1. If a flow of 1 is sent through the forward arcs of the infinite path of Figure 2.6c and a flow of -1 is sent through the backward arcs, the resulting flow is a feasible stationary flow with a throughput of 1, and is thus optimal.

Arc	(Tail, Head)	Transit Time	Upper Bound	Lower Bound
a_1	(1,2)	0	1	-1
a_2	(1,2)	1	1	0
a_3	(1,4)	1	0	-2
a_4	(2,3)	0	2	1
a_5	(3,4)	0	1	-1
a_6	(3,4)	1	1	0

Table 2.2. The parameters for a static network.

Arc	Upper Bound	Lower Bound	Transit Time
(1,1)	∞	0	1
(2,2)	0	$-\infty$	1
(1,2)	1	1	0

Table 2.3

We note in passing that if x is a maximum-throughput dynamic network flow and $(S,\overline{S})$ is a minimum upper-capacity cut, then $x_a^p = u_a$ for $a^p \in A(S,\overline{S})$ and $x_a^p = \ell_a$ for $a^p \in A(\overline{S},S)$.

Neither 1° nor 3° of Theorem 5 is true if we drop the restriction that there is a feasible bounded dynamic flow because there are no feasible stationary flows in that event. Nevertheless, there may exist unbounded feasible flows and 2° of Theorem 5 may still hold as the next example illustrates.

EXAMPLE. (All feasible dynamic flows are unbounded.) Consider the static network described in Table 2.3 and depicted in Figure 2.7a. There is no feasible stationary flow, although there is a feasible dynamic flow $x = (x_a^p)$ given by

$$x_a^p = \begin{cases} -p \ , & a = (1,1) \\ p + z, & a = (2,2) \\ 1 \ , & a = (1,2) \ , \end{cases}$$

where z is an arbitrary real number. Then $f_x = z$, so the supremum (resp., infimum) of the throughputs of these flows is $+\infty$ (resp., $-\infty$) and this is the upper (resp., lower) capacity of each cut.

<u>An Alternate Proof of 1° of Theorem 5 (Sufficiency of Stationary Flows)</u>

An alternate proof of 1° of Theorem 5 is obtained by applying a result of Orlin (1981a) to the special dynamic linear program given below. However, this proof shows only that stationary flows suffice in the class of <u>bounded</u> feasible dynamic flows whereas the proof of Theorem 5 applies in the class of <u>all</u> feasible dynamic flows.

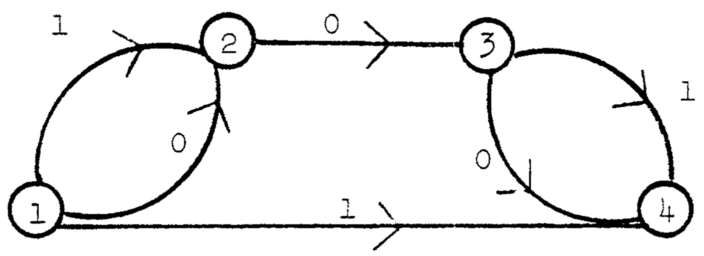


Figure 2.6a. The static network described in Table 2.2.

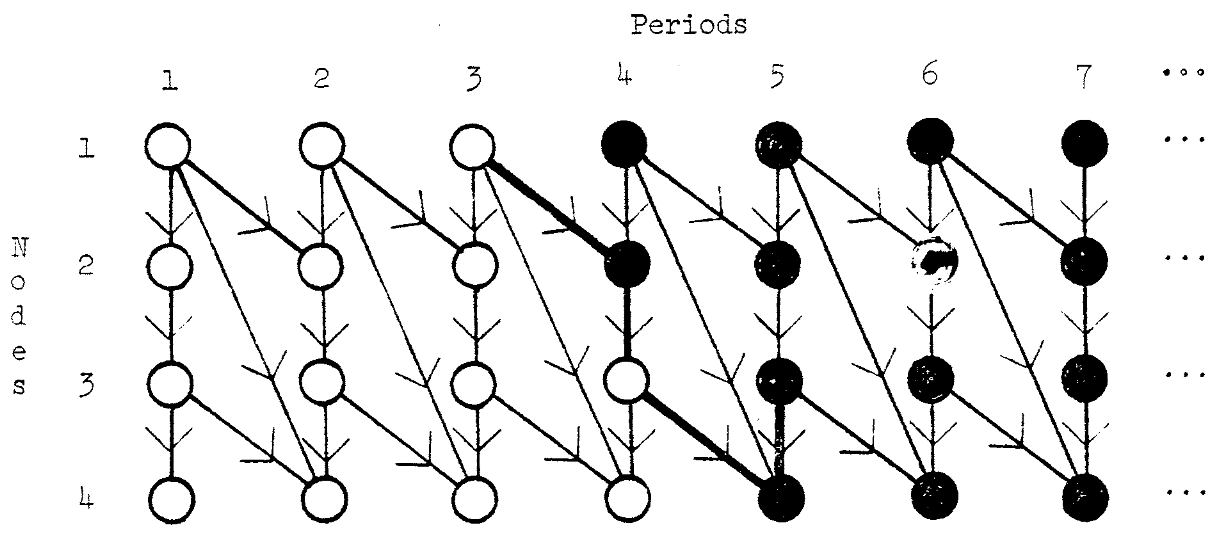


Figure 2.6b. A minimum upper-capacity cut in the dynamic network associated with Figure 2.6a. $A(S,\overline{S}) = \{(1^3,2^4),(3^4,4^5)\}$; $A(\overline{S},S) = \{(2^4,3^4)\}$.

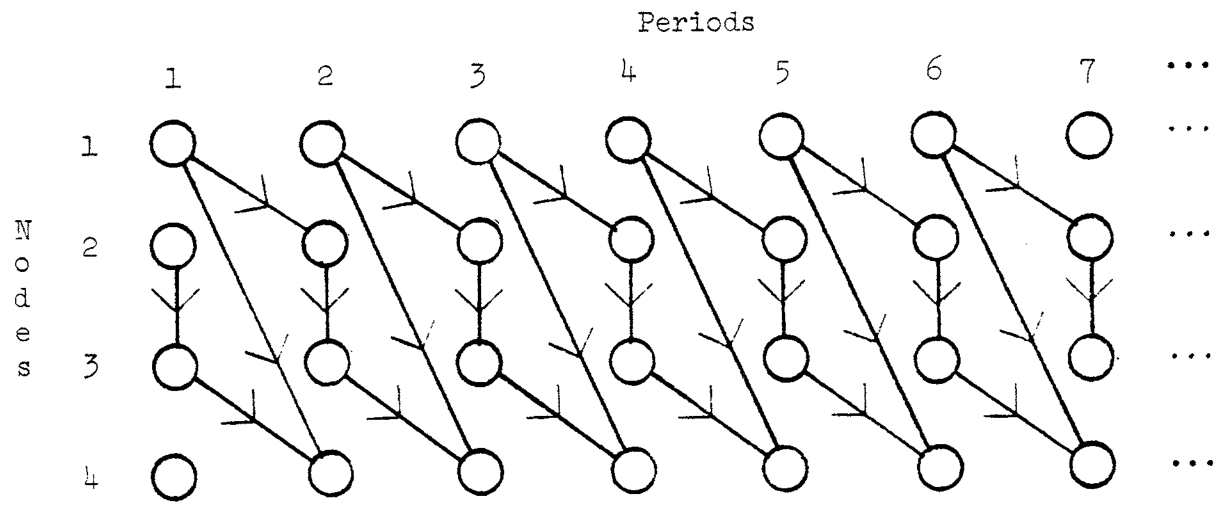


Figure 2.6c. An infinite path in the dynamic network of Figure 2.6b.

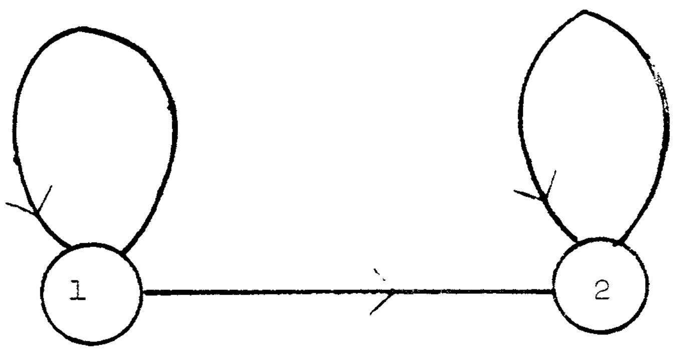


Figure 2.7a. The static network of Table 2.3.

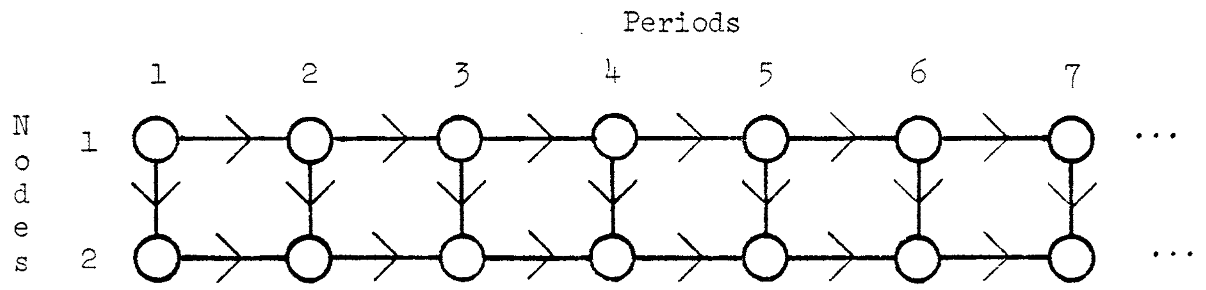


Figure 2.7b. The dynamic network associated with the static network of Figure 2.7a.

Let v^p be the throughput of a feasible dynamic flow in period p. By Lemma 1, v^p is constant for all $p \geq t_{max}$. Thus the maximum-throughput problem can be written as that of

$$\text{maximizing} \quad \lim_{p \to \infty} p^{-1} \sum_{j=1}^{p} v^j$$

subject to

$$\sum_{a \in H_i} x_i^p - \sum_{a \in T_i} x_a^{p-t_a} = 0 \quad \text{for} \quad i \in N \quad \text{and} \quad p > t_{max} ,$$

$$- v^p + \sum_{a \in A} \sum_{j=p-t_a+1}^{p} x_a^j = 0 \quad \text{for} \quad p > t_{max} ,$$

and

$$\ell_a \leq x_a^p \leq u_a \quad \text{for} \quad a \in A, \ p = 1, 2, \ldots .$$

3. MINIMIZING THE NUMBER OF VEHICLES TO MEET A FIXED PERIODIC TRANSPORTATION SCHEDULE

Here we consider a routing problem that arises in the scheduling of vehicles for certain transportation industries, such as airlines and railroads. For convenience, we will borrow terminology from the airline industry.

The problem is to minimize the number of aircraft needed to meet a fixed schedule of daily repeating flights, each of which is either <u>required</u> or <u>optional</u>. The required flights must be flown daily, while the optional flights may be flown on any day at the scheduler's prerogative.

We assume that any plane may fly any route on any day. We do not make any a priori assumptions on the data, and we even allow the contingency that a flight may take several days. (This contingency is of little value in scheduling planes, but might be of value if we were to schedule trains.) Furthermore, we do not require that a feasible schedule be periodic, although our algorithm will always produce a periodic schedule.

Various versions of the above problem have been considered in the literature. Dantzig and Fulkerson (1954) solved the problem of minimizing the number of vehicles to meet a fixed finite-horizon schedule. (It was Fulkerson's first paper on network flows.) The problem of minimizing the number of vehicles to meet a periodic schedule in which all routes are required--so deadheading is not allowed--has been solved by Bartlett (1957) and the problem has been applied to railroad scheduling.

Bartlett was concerned with determining the minimum number of vehicles. For his case, the operating schedule is itself trivial as there are no relevant decisions to make. As pointed out by Bartlett, an obvious necessary condition for optimality is that at no airport is there a plane on the ground for the entire day. Bartlett also showed that this condition is sufficient for optimality, and he gave a simple closed formula for the number of airplanes in such a schedule.

Dantzig (1962) considered various airline scheduling problems including the above problem of minimizing the number of airplanes to meet a fixed periodic schedule under the added restriction that the final flight schedule is stationary. His technique, as described by Simpson (1969), is equivalent to solving the static version of a corresponding minimum-throughput problem. By results of the previous section, the resulting optimal stationary solution that Dantzig obtains is optimal over the class of all feasible (possibly non-stationary) schedules.

As Dantzig and his collaborators noted, in order to satisfy the stationary flight schedule, each airplane flies a periodic schedule, but the total time for a route for one airplane may be a number of days. We repeat Dantzig's technique here both for completeness and so as to illustrate the theory of the previous section.

We formulate the problem as a dynamic flow problem with the static network $G = (N,A,t,\ell,u)$, where each node of N is an ordered pair $\langle i,s \rangle$ with i being a city and s being a time of day. It suffices to consider only those times s for which there is either a departing or arriving flight. We incorporate the required holdover times at airports into the travel times so that we may assume that a plane may take off immediately upon landing. A flight departing at time s from city i and arriving d days later at time r in city j is represented by an arc from (i,s) to (j,r) with transit time d. If it arrives on the same day, then $d = 0$. If it crosses the international date line and arrives on the previous day, then $d = -1$. The upper bound on the flow in each of these arcs is the number of flights that may be flown at this time or $+\infty$ if no such bound exists while the lower bound is 1 or 0 according as the flight is required or optional. We also have arcs which represent the time that the airplanes stay on the ground; the transit time of each of these arcs is 1 or 0 according as the ground time includes or does not include midnight.

An example of this formulation is given in Tables 3.1 and 3.2. The first table describes the flight schedules, while the second table describes the static network. The static network is given in Figure 3.1. Each period of the dynamic network represents a day.

Origin	Destination	Departure Time	Arrival Time	Required Or Optional
City 1	City 2	1 AM	7 AM	Required
City 2	City 3	3 AM	9 AM	Optional
City 3	City 1	5 AM	11 AM	Optional

Table 3.1

Tail	Head	Upper Bound	Lower Bound	Transit Time
<1,1>	<2,7>	1	1	0
<2,3>	<3,9>	1	0	0
<3,5>	<1,11>	1	0	0
<1,11>	<1,1>	∞	0	1
<1,1>	<1,1>	∞	0	1
<2,7>	<2,3>	∞	0	1
<2,3>	<2,3>	∞	0	1
<3,9>	<3,5>	∞	0	1
<3,5>	<3,5>	∞	0	1

Table 3.2

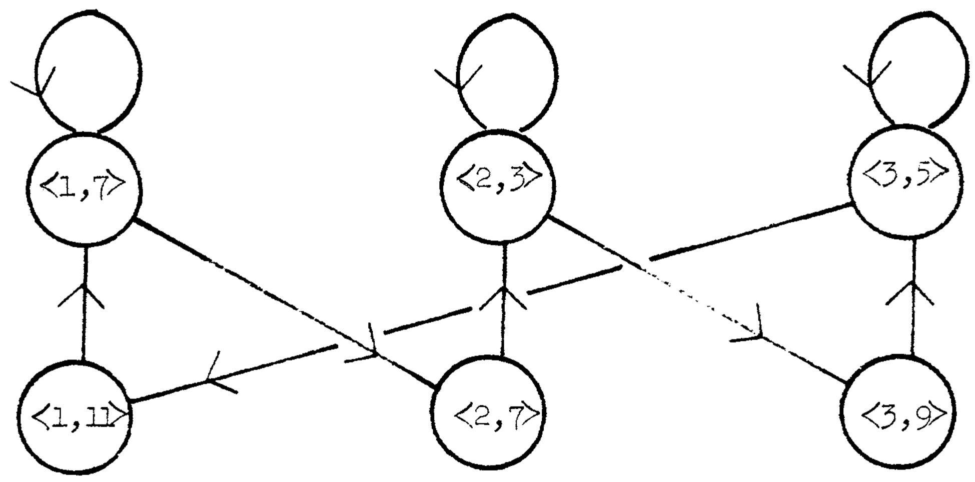


Figure 3.1. A static network for the airplane scheduling problem.

The interpretation of the lower-bound constraint (2.1) is that each required flight be flown. The interpretation of the conservation-of-flow constraint (2.2) is that airplanes are neither created nor destroyed once the schedule is determined. Finally, the net flow in transit in period 1 is equal to the number of airplanes that are either on the ground or in the air at midnight of the first day; thus the throughput is the number of airplanes.

The Solution

The airplane scheduling problem is really an integer programming problem. However, this causes no difficulties as there is always a minimum-throughput static flow that is integral, and this static flow induces a stationary integer dynamic flow that is optimal. Although this stationary flow does provide a time-table for flights and does minimize the number of airplanes, it does not provide the daily schedule for any airplane. Such a daily schedule for airplanes may be determined as follows: let $y = (y_a)$ denote an integral optimal static flow. Decompose the circulation y into a sum of unit flows around directed cycles. The cycles are directed because any backward arc would have a flow equal to -1, which is impossible because all lower bounds are nonnegative. Furthermore, each cycle corresponds to a finite sequence of routes that a single plane may travel so as to start and finish at the same place and at the same time of day, and the transit time of the cycle is the number of days traveled. Each cycle C of transit time t induces t infinite-length paths in the dynamic network by Lemma 2, and each of these paths induces a periodic schedule that repeats every t days for an airplane. In this way, we may determine schedules for all planes.

Let an instance of a flight refer to a flight on a specified day, e.g., the 9 a.m. flight from Boston to Atlanta on April 19, 1979. Then the max-throughput min-cut theorem may be interpreted as follows.

> The minimum number of airplanes needed to meet a fixed periodic schedule is equal to the maximum number of instances of required flights, no two of which may be flown by the same airplane.

In the example given in Table 3.1, the unique optimal tour is given by flying from city 1 to city 2 to city 3 and returning to city 1. In the static network the minimum throughput is obtained by sending one unit of flow around the cycle determined by the node sequence $\langle 1,1\rangle$, $\langle 2,7\rangle$, $\langle 2,3\rangle$, $\langle 3,9\rangle$, $\langle 3,5\rangle$, $\langle 1,11\rangle$, $\langle 1,1\rangle$. Each airplane on this route takes three days. Three airplanes are needed, as the flights from city 1 to city 2 on any three consecutive days must be flown by different airplanes.

4. FINITE-HORIZON DYNAMIC MAXIMUM FLOW

In this section we consider the q-period dynamic maximum-flow problem which Ford and Fulkerson (1958) formulated and solved. The objective in this problem, which we shall henceforth call the <u>q-period problem</u>, is to find a maximum flow from a given source to a given sink in q periods where q is a given positive integer. Ford and Fulkerson showed that there is always an optimal flow (depending on q) that is temporally repeated. Such flows are essentially stationary. This elegant result unfortunately does not seem to generalize to other finite-horizon problems. Specifically, if other optimality criteria are considered, then an optimal flow will not, in general, be stationary. For example, neither minimum-cost flows nor universal flows as defined by Gale (1959) are generally stationary.

In this section we demonstrate that the q-period dynamic maximum flow problem may be transformed into a special case of the minimum-throughput dynamic network-flow problem, which we will refer to in this section as the infinite-horizon problem. In this way, the Ford-Fulkerson results may be viewed as a specialization of the theory developed in Section 2 above.

The q-Period Problem

Let $G = (N, A, t, 0, u)$ be a static network in which each arc has a non-negative transit time, a zero lower bound on its arc flow, and a positive integer-valued upper bound thereon. Let $N = \{1, \ldots, n\}$, and call node 1 the source and node n the sink. Denote by $G^q = (N^q, A^q, t^q, 0, u^q)$ the q-period subnetwork of the dynamic network $G^\infty = (N^\infty, A^\infty, t^\infty, 0, u^\infty)$ induced by the nodes $i^p \in N^\infty$ for $i \in N$ and $1 \leq p \leq q$. Flow may be sent in G^q in periods $1, 2, \ldots, q$ for some fixed q. We call $x = (x_a^p)$ a feasible q-period flow in G^q if it satisfies (4.1), (4.2) and (4.3) below, viz., the upper- and lower-bound constraints

$$0 \leq x_a^p \leq u_a \quad \text{for } a \in A, \quad p = 1, \ldots, q \, , \tag{4.1}$$

the conservation-of-flow equations

$$\sum_{a \in T_i} x_a^p = \sum_{a \in H_i} x_a^{p-t_a} \quad \text{for } i = 2, \ldots, n-1 \text{ and } p = 1, \ldots, q , \tag{4.2}$$

and the flow prior to period 1 is zero, i.e., in the first q periods and at all nodes except for the source and sink,

$$x_a^p = 0 \quad \text{for } a \in A \text{ and } p \leq 0 . \tag{4.3}$$

We assume without loss of generality that $t_a < q$ for each arc a. The <u>q-period problem</u> is to determine a feasible q-period flow that maximizes the amount of flow g_x that arrives at the sink where

$$g_x = \sum_{a \in H_n} \sum_{p=1}^{q-t_a} x_a^p . \tag{4.4}$$

Temporally-Repeated Flows

As we have remarked above, Ford and Fulkerson showed that one maximum q-period flow will be temporally repeated. In order to define this notion, it is convenient to append to the static network G an arc $\alpha = (1,n)$ from source to sink with $t_\alpha = q$, $\ell_\alpha = -\infty$, and $u_\alpha = 0$. Let $G_q = (N, A_q, t, \ell, u)$ be the <u>q-period augmented</u> static network with $A_q = A \cup \{\alpha\}$ and the domain of (t, ℓ, u) extended from A to A_q. (Incidentally, appending α has no effect on the solution to the q-period problem because $t_\alpha = q$.)

If P is a simple directed path in G from source to sink with transit time $r < q$, then $P_q \equiv P \cup \{\alpha\}$ is a simple cycle in

G_q with transit time $r - q$. Thus by Lemma 2, the copies of P_q in G_q^∞ comprise $q - r$ node-disjoint infinite paths and each contains a unique copy of P that is also in G^q. Thus P has $q - r$ copies in G^q.

If P is a simple directed path in G from source to sink with transit time $r < q$, then a positive unit flow in each of the $q - r$ copies of P in G^q is called a temporally-repeated unit flow. A sum of temporally-repeated unit flows is called a temporally-repeated flow.

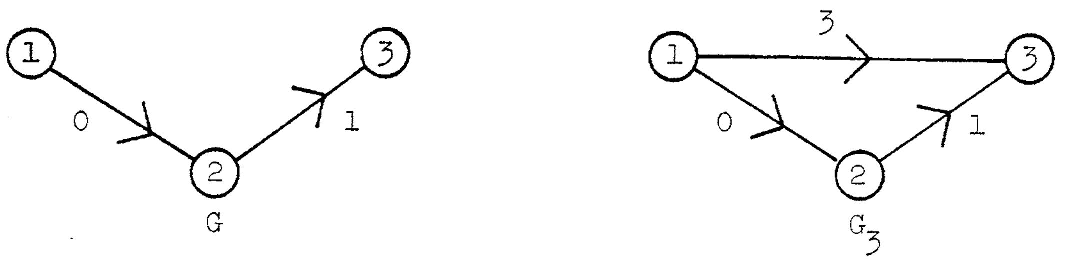
Example

We illustrate these concepts in Figures 4.1-4.3. In Figure 4.1 we give a static network G consisting of a simple directed path P from source to sink with transit time one and the 3-period augmentation G_3 of G comprising a simple cycle with (absolute) transit time two. The two infinite-length paths in G_3^∞ induced by P_3 are portrayed in Figure 4.2, and the boldfaced lines in Figures 4.2 and 4.3 delineate the two copies of P in G^3.

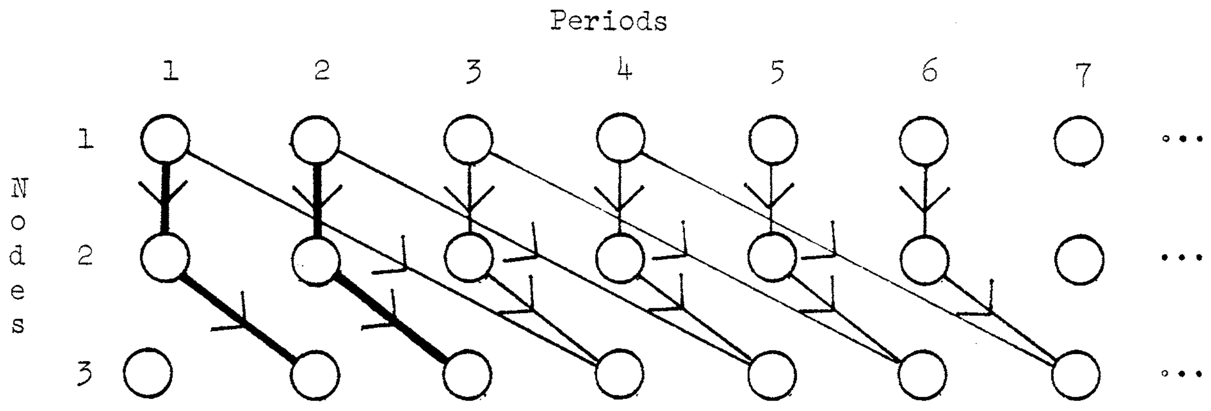
If there are unit upper bounds on flows in all arcs in G, then the maximum 3-period flow from source to sink is evidently two. This flow is achieved by the temporally-repeated (unit) flow that sends unit flows along each of the two copies of P in G^3.

The Infinite-Horizon Problem

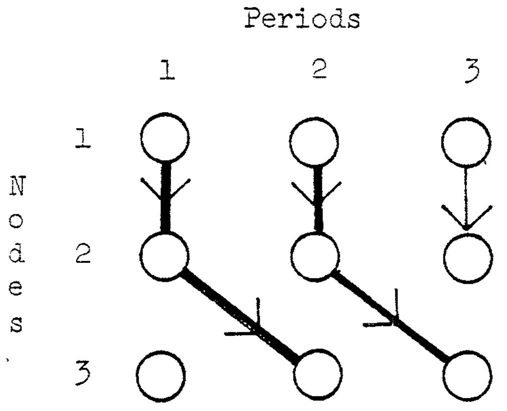
We are now ready to develop the main result of this section, viz., that the problem of finding a maximum q-period flow from source



<u>Figure 4.1</u>. A static network G and its 3-period augmentation G_3.



<u>Figure 4.2</u>. The dynamic network G_3^∞ corresponding to the 3-period augmented static network of Figure 4.1.



<u>Figure 4.3</u>. The 3-period subnetwork G^3 of G^∞ corresponding to the static network of Figure 4.1.

to sink can be reduced to the infinite-horizon problem of finding a feasible dynamic flow in G_q^∞ having minimum throughput. In order to motivate this result, let P be a simple directed path in G and let P' be one of the infinite paths in G_q^∞ induced by P_q. Now a negative unit flow in P' has positive unit flow in each copy of P in P', has negative unit flow in each copy of α in P', and conserves flow at all nodes (including copies of the source and sink) after period q. Thus a negative unit flow in P' sends a positive unit flow from source to sink in G^q (along the unique copy of P in both P' and G^q) and has a negative unit throughput in G_q^∞. Thus, in this simple instance at least, the throughput of a dynamic flow in G_q^∞ is the negative of the induced subflow from source to sink in G^q. This suggests that minimizing throughput in G_q^∞ will indeed maximize flow from source to sink in G^q.

These ideas are illustrated in the example of Figures 4.1-4.3 discussed above. The minimum throughput in G_3^∞ in that example is -2 and is achieved by negative unit flows in each of the two infinite paths in Figure 4.2. That flow induces a subflow from source to sink in G^3, viz., the temporally-repeated (unit) flow discussed above, and attains the maximum 3-period flow of 2.

THEOREM 6. Let y^∞ be a stationary integer minimum-throughput dynamic network flow for the infinite-horizon problem with G_q^∞. Then y induces a temporally-repeated q-period flow x that has maximum q-period flow from source to sink. Furthermore, the negative of the throughput of y^∞ is the q-period flow from source to sink of x.

PROOF. Consider an integer optimal static flow $y = (y_a)$ for G_q that induces a minimum-throughput stationary flow. The circulation y may be written as the sum of unit flows around cycles in G_q. Since we wish to minimize throughput, the transit time of each cycle must be negative because we would delete any cycle with a nonnegative transit time without increasing the throughput of the induced stationary flow. We may also choose the cycles so that each arc in A appears only as a forward arc.

Let C be a cycle in the decomposition, and let $\bar{t}$ be the transit time of C. Since $t < 0$, it follows that C consists of arc α plus a directed path P from node 1 to node n of length $q + t$. This path P induces a temporally-repeated unit flow with $-t$ units of flow arriving at the sink. Thus the temporally repeated flow x induced by these cycles has a flow $g_x = -f_y$ arriving at the sink, because f_y is the sum of the transit times of the cycles.

To complete the proof we will show that x is optimal for the q-period problem. Let $(S,\bar{S})$ be a monotone cut with lower capacity $f_y = -g_x$ as guaranteed by the max-throughput min-cut Theorem 5. Choose p so that $1^{p-q+1}, \ldots, 1^p \in S$ and $1^{p+1}, \ldots, 1^{p+q} \in \bar{S}$, which is possible because the cut $(S,\bar{S})$ is monotone. No copy of α is in the set $A(S,\bar{S})$ because the lower capacity of the cut $(S,\bar{S})$ is $-g_x > -\infty$. Therefore any copy of α with its tail in S must also have its head in S. Hence, $n^{p+1}, \ldots, n^{p+q} \in S$.

Finally, we translate and truncate the cut $(S,\bar{S})$ for G_q^∞ to form a cut $(\bar{S}',S')$ for G^q defined by $\bar{S}' = \{i^r : 1 \leq r \leq q \text{ and } i^{r+p} \in \bar{S}\}$ and $S' = \{i^r : 1 \leq r \leq q \text{ and } i^{r+p} \in S\}$. Observe that the copies of source 1 are in $\bar{S}'$ and the copies of sink n are in S'. Thus the maximum flow in G^q from source to sink is at most the sum of the capacities of all arcs with tail in $\bar{S}'$ and head in S', and this sum is bounded above by the negative of the lower capacity of cut $(S,\bar{S})$ in G_q^∞ which is $-f_y = g_x$. Thus the maximum q-period flow from source to sink is at most g_x, completing the proof. ■

Consider a maximum q-period flow y. If we restrict the flow to the first q-1 periods, is it necessarily a maximum (q-1)-period flow? As shown below, the answer is no. However, Gale (1959) proved that there always is a maximum q-period flow that, when restricted to the first p periods, is also a maximum p-period flow, for all p = 1, 2, ..., q. He called such a flow a "universal maximum flow." In fact, such a flow exists even when the arc capacities vary over time. Gale showed that there does not necessarily exist a stationary universal maximum flow. For example, consider the static network described in Table 4.1 and portrayed in Figure 4.3. If we want a maximum 3-period flow, then the unique universal maximum flow is given in Figure 4.4, while the unique stationary maximum 3-period flow is given in Figure 4.5.

Finally, suppose that we wish to find a minimum-cost feasible q-period flow. Once again we cannot expect an optimal flow to be stationary. To see this, assign a unit cost of -1 to arc (2,3) of the static network of Table 4.1, -2 to arc (1,3) and 0 to all other arcs. Then the minimum-cost stationary flow is that of Figure 4.3, and it has a cost of -4, while the flow in Figure 4.4 has a cost of -5.

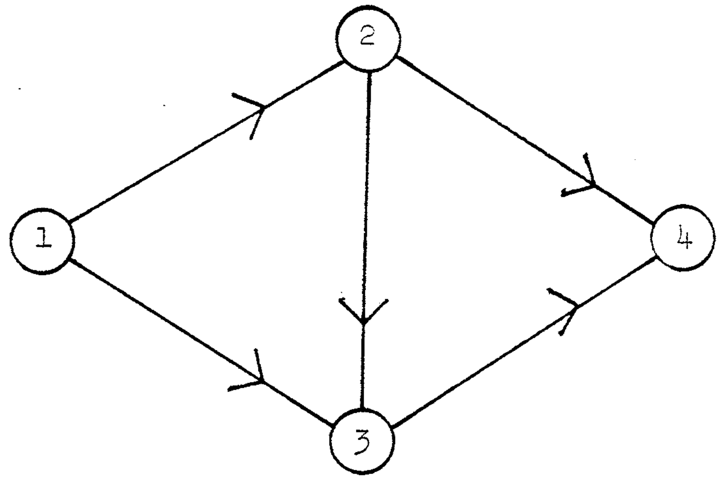


Figure 4.3. The static network described in Table 4.1.

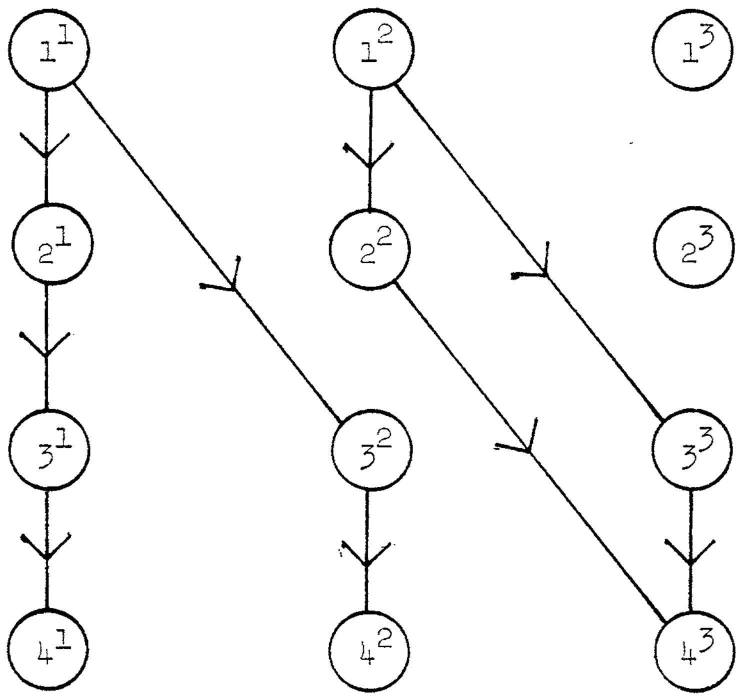


Figure 4.4. The unique optimal universal flow for the network of Table 4.1 is along the four paths given above.

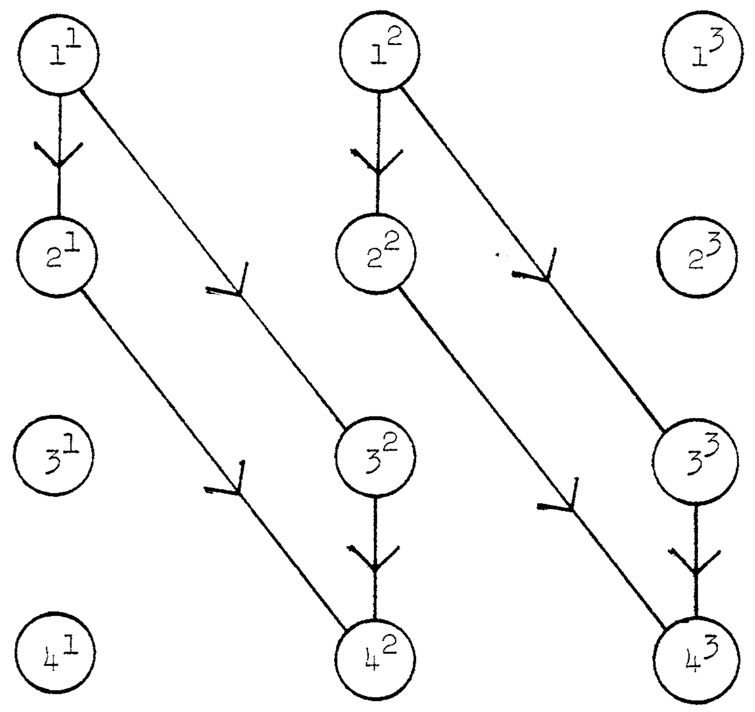


Figure 4.5. The unique optimal temporally-repeated flow for the network of Table 4.1 is along the four paths given above.

Tail	Head	Upper Bound	Lower Bound	Transit Time
1	2	1	0	1
1	3	1	0	1
2	4	1	0	1
3	4	1	0	1
1	4	0	-4	3

Table 4.1. The static network portrayed in Figure 4.3.

The result contrasts with the optimality of stationary flows for the minimum-cost dynamic network-flow problem as proved by Orlin (1981b).

APPENDIX. Everywhere-Conservative Feasible Dynamic Flows

In the maximum-throughput dynamic network-flow problem and again in the minimum convex-cost dynamic network-flow problem of Orlin (1981b), a feasible dynamic flow need not satisfy conservation of flow in the first few periods. This may be a significant relaxation of real-world constraints because conservation of flow usually has a physical interpretation. An interesting open question is how to "phase into" an optimal stationary flow.

In many special instances it is easy to phase into an optimal stationary flow. In the vehicle scheduling problem of Section 3, it suffices to have all vehicle travel to their first departing site. In the cyclic capacity scheduling problem discussed in Orlin (1981b), it is possible to phase into any feasible stationary schedule within one day. However, in general, the "phasing problem" is NP-complete, as is shown in this section.

A feasible dynamic network flow is called <u>everywhere conservative</u> if conservation of flow is satisfied in all periods including periods $1, \ldots, t_{max} - 1$. We refer to such flows as <u>conservative flows</u>, for brevity. In this section we use the terms NP-hard and NP-complete. The reader unfamiliar with these concepts should refer either to Karp (1972) or Garey and Johnson (1979). Informally, a problem X is NP-hard if there is an NP-complete problem which may be polynomially transformed to X, i.e., transformed in polynomial time so that it becomes a special case of X.

THEOREM A.1. Determining whether there is a conservative flow is NP-hard.

PROOF. Consider the following version of the knapsack problem, which was proved NP-complete (and thus NP-hard) by Karp (1972).

KNAPSACK PROBLEM. Given n distinct positive integers $d_1, \ldots, d_n$ and a positive integer b, do there exist non-negative integers $w_1, \ldots, w_n$ such that $d_1 w_1 + \cdots + d_n w_n = b$?

Consider the static network G described in Table A.1. To prove the theorem we show that there is a conservative flow for G if and only if there is a feasible solution for the corresponding knapsack problem.

Arc	a_1	a_2	$\cdots$	a_n	α	β	γ
Tail	1	1	$\cdots$	1	1	2	1
Head	1	1	$\cdots$	1	1	2	1
Lower bound	0	0	$\cdots$	0	-1	-1	1
Upper bound	1	1	$\cdots$	1	-1	-1	1
Transit time	d_1	d_2	$\cdots$	d_n	1	b	$b+1$

Table A.1

First, let G' be G with arcs α, β, and γ deleted. Then there is a feasible solution for the knapsack problem if and only if there is a directed path in G' from node 1 to node 1 with transit time b. We see this as follows: if there is a path with transit time b, let w_i be the number of occurrences of arc a_i in the path. Then the sum of the transit times of the arcs is $w_1 d_1 + \dots + w_n d_n$, which is b. Conversely, suppose w is a solution to the knapsack problem. Consider a path in G' consisting of w_i occurrence of a_i for $i = 1, \dots, n$ (the arcs taken in any order). The resulting path is from 1 to 1 and has transit time b.

To complete the proof of the theorem, we show that there is a conservative flow for G if and only if there is a directed path in G' from node 1 to node 1 with transit time b.

Suppose x is a conservative flow. If we restrict x to copies of α, β, and γ, the resulting feasible dynamic network flow satisfies conservation of flow at all nodes of the dynamic network except 1^1 which has a deficit of one unit and 1^{b+1} which has a surplus of one unit. Since x is a conservative flow, if we restrict x to copies of arcs $a_1, \dots, a_n$ it must consist of a unit flow from 1^1 to 1^{b+1} and conserves flow elsewhere. The unit flow is sent along a path that is a copy of a path in G' from 1 to 1 with transit time b.

Conversely, if there is such a path in G', then there is a copy of the path in the dynamic network from 1^1 to 1^{b+1}. A conservative flow is created by sending 1 unit of flow along this path, 1 unit of flow in each copy of α and β, and -1 unit of flow in each copy of γ. ■

REFERENCES

Bartlett, T. E. (1957). An Algorithm for the Minimum Number of Transport Units to Maintain a Fixed Schedule. Naval Research Log. Quarterly 4, 207-220.

Dantzig, G. B. and D. R. Fulkerson (1954). Minimizing the Number of Tankers to Meet a Fixed Schedule. Naval Research Log. Quarterly 1, 217-222.

Dantzig, G. B. (1962). Consulting work for United Airlines.

Folkman, J. and D. R. Fulkerson (1970). Flows in Infinite Graphs. Journal of Combinatorial Theory 8, 30-44.

Ford, L. R. and D. R. Fulkerson (1956). Maximal Flow Through a Network. Canadian Journal Math. 8, 399-404.

Ford, L. R. and D. R. Fulkerson (1958). Constructing Maximal Dynamic Flows from Static Flows. Operations Research 6, 419-433.

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

Gale, D. (1959). Transient Flows in Networks. Michigan Math Journal 6, 59-63.

Garey, M. R. and D. S. Johnson (1979). Computers and Intractibility: A Guide to the Theory of NP-Completeness. W. H. Freeman and Company, San Francisco.

Glover, F., D. Klingman, C. McMillan (1977). The Netform Concept: A More Effective Model Form and Solution Procedure for Large Scale Non-Linear Problems. Management Science Report No. 77-4, Graduate School of Business Administration, University of Colorado 80309.

Karp, R. M. (1972). Reducibility Among Combinatorial Problems. In R. E. Miller and J. W. Tatcher (eds.) Complexity of Computer Computations. Plenum Press, New York, 85-103.

Lawler, E. L. (1976). Combinatorial Optimization: Networks and Matroids. Holt, Rinehart and Winston, New York.

Maxwell, W. L. and R. C. Wilson (1978). Analysis of Dynamic Material Handling Systems by Network Flow. Working Paper No. 4, Department of Industrial Operations Engineering, The University of Michigan.

Minieka, E. (1973). Maximal Lexicographic and Dynamic Network Flows. Operations Research 21, 517-527.

Orlin, J. B. (1981a). Dynamic Convex Programming. Chapter II of this thesis.

Orlin, J. B. (1981b). Minimum-Convex-Cost Dynamic Network Flows. Chapter IV of this thesis.

Simpson, R. W. (1968). Scheduling and Routing Models for Airline Systems. Report FTL-R68-3, Department of Aeronautics and Astronautics, MIT, 100-107 and 128-134.

CHAPTER IV

MINIMUM CONVEX-COST DYNAMIC NETWORK FLOWS

1. INTRODUCTION

The Model and Problem Formulation

In the sequel we present and solve the minimum convex-cost dynamic network-flow problem, an infinite-horizon integer programming problem that involves network flows evolving over time. This work extends the results of the author (1981b) concerning maximum-throughput dynamic network flows.

We consider a finite network in which there is associated with each arc a real or $+\infty$ valued convex cost of flow therein that is linear between successive integers, and a (possibly negative) integer transit time that is the number of periods for flow to pass through the arc.

A dynamic flow is a sequence of flows initiated in each arc in each period. Such a flow is called feasible if it is bounded, has finite cost in each period, satisfies conservation of flow in all except the first few periods during which the flow is "initialized," and has zero "throughput." The throughput is the net flow in transit in each period and is shown in Orlin (1981b) to be the same in each period after the first few. The throughput may also be viewed as the net amount of flow circulating in the network. This concept was introduced and studied in detail in Orlin (1981b).

A dynamic flow is called optimal in a given class of such flows if it is feasible and has minimum long-run average cost per period among all feasible dynamic flows in the class. Thus, for example, we speak of optimal integer (resp., continuous, stationary, etc.) dynamic flows. The class of integer (resp., continuous, stationary) dynamic flows consists of those for which the flow in each arc in each period is integer (resp., unrestricted, the same in each period).

Finding Optimal Integer Dynamic Flows

Our principal interest centers on finding an optimal integer dynamic flow. Our method for doing this involves two steps, viz., finding a stationary optimal continuous dynamic flow and then rounding that flow to form an integer periodic optimal continuous dynamic flow that is necessarily an optimal integer dynamic flow.

We discuss briefly these two steps. To begin with, it follows from Orlin (1981a) that the sets of optimal stationary and stationary optimal continuous dynamic flows coincide. Also these flows coincide with those induced by repeatedly using an optimal circulation, i.e., a circulation in the static network that has (finite) minimum cost among those with zero throughput. The throughput of a circulation (and of the stationary dynamic flow it induces) is the sum of the products of each arc flow and its transit time, and so is linear in the arc flows. Thus, the optimal-circulation problem is a minimum-cost static network-flow problem with one additional linear (zero-throughput) constraint. Consequently, optimal circulations are not generally integer. Nevertheless, if there is an optimal circulation, there is one that is "fractionally extreme," i.e., if the integer elements of the circulation are fixed, the noninteger elements are uniquely determined. Moreover, the arcs in which flows are noninteger form a simple cycle. Each arc flow in a stationary dynamic flow induced by a fractionally-extreme optimal circulation may be appropriately rounded to an adjacent integer so as to yield an optimal integer (and integer optimal) dynamic flow that repeats every q periods where q is the least common denominator of the fractional parts of the circulation.

Applications

The minimum convex-cost dynamic network-flow problem may be applied to various areas of Operations Research that involve deterministic demands repeating periodically over time. These include the cyclic capacity scheduling problem, the cyclic staffing problem, the periodic production and transshipment problem, the airline scheduling problem, and the minimum cost-to-time ratio circuit problem, as detailed below.

Cyclic Capacity Scheduling and Cyclic Staffing

The cyclic capacity scheduling problem is to find a minimum per-period cost schedule of buying and selling capacity in blocks of consecutive periods so as to satisfy demands for capacity that repeat cyclically over time. This problem generalizes the finite-horizon capacity scheduling problem which was shown by Veinott and Wagner (1962) to reduce to a minimum-cost network-flow problem. A special case of the cyclic capacity scheduling problem is the cyclic staffing problem, which is to minimize the per-day cost of staffing a workforce round-the-clock on shifts of consecutive hours so as to satisfy minimum hourly demands that vary within the day, but repeat from day to day.

The infinite-horizon problem considered above apparently has not been discussed before. However, if the assignment of workers is required to be the same every day, then the resulting problem is the

1-day cyclic staffing problem which has received considerable attention, e.g., Baker (1974) and (1976), Bartholdi et al. (1980), Orlin (1977) and Tibrewala et al. (1972). There is no known polynomial algorithm for the 1-day problem, nor is it known whether the problem is NP-complete.

The integral solution obtained for the cyclic capacity scheduling problem and hence for the cyclic staffing problem has the property that it is obtained by rounding an infinite-horizon continuous-valued solution that repeats each period. For this reason, the number of workers on any specified shift varies by at most one from day to day. In this sense, it is "almost feasible" for the 1-day problem, and always has a daily cost as low as the optimal daily cost for the 1-day problem.

Periodic Production and Transshipment

The periodic production and transshipment problem is to minimize the daily cost of producing and shipping goods (such as food or petroleum products) from city to city so as to satisfy periodically repeating demands, where unit costs are assumed to repeat periodically. We also assume that the number of trucks (or any other cargo carriers) is limited. Finally we need the additional assumption that each truck returns to a production site upon delivery of goods. (If trucks may carry goods to more than one demand location without reloading in between, then the resulting problem is NP-hard.) In Section 6, we model this production and transshipment problem as a special case of the dynamic network-flow problem.

Airplane Scheduling

Consider an airline whose objective is to schedule a fixed number of airplanes to flights so as to maximize its profits (or minimize its costs). A certain set of flights may be flown on any day at the scheduler's prerogative. This problem generalizes that considered by Dantzig (1962) and Orlin (1981b) of determining the minimum number of aircraft for which there is a feasible schedule. Dantzig (1962) also considered the stationary version of the maximum-profit problem. In Section 6, that airplane scheduling problem is modeled as a dynamic network-flow problem.

Minimum Cost-to-Time Ratio Circuits

Dantzig, Blattner, and Rao (1967) formulated and solved the "tramp steamer problem" or equivalently "the minimum cost-to-time ratio circuit problem." This is the scheduling problem of choosing an infinite tour for a tramp steamer which is to travel from port to port so as to minimize its daily costs. The tramp steamer may visit whatever ports it chooses and in any order, and for any pair of ports there is an associated cost per trip and transit time per trip.

There is always an optimum tour in which the steamer travels cyclically in the same order around the ports. The optimal cyclic path is that cycle which maximizes the ratio of the total cost on the cycle to the total transit time. Several authors including Fox (1969), Lawler (1967), and Megiddo (1978) have given efficient algorithms for determining this optimal cycle. Our contribution to this problem in Section 6 is to show that the rounding technique of Section 3 also obtains this cycle, thus showing that the previous methods for solving this problem may be viewed as a special case of rounding.

2. DYNAMIC NETWORK FLOWS

The Static Network and the Problem Formulation

A static network is a quadruple $G = (N, A, t, c)$ in which $N = \{1, \ldots, n\}$ is the node set and A is the arc set of a directed graph, possibly containing loops (i.e., arcs joining a node to itself) and multiple arcs between two nodes. Associated with each arc $a = (i, j)$ is a transit time t_a, which is the (possibly negative) integer number of periods for flow to travel from the tail i of a to its head j. Also associated with arc a is a real- or $+\infty$ valued convex function $c_a(\cdot)$ that is linear between successive integers.

Let H_i and T_i denote the set of arcs of A with head i and tail i, respectively. Let $t_{max} = \max_{a \in A} |t_a|$. In the following we wish to consider flow emanating along arcs in each of an infinite number of periods. With this in mind, we let x_a^p denote the amount of flow originating at the tail of arc a in period P and arriving at the head of arc a in period $p + t_a$. We refer to an infinite vector $x = (x_a^p)$ for $a \in A$ and $p = 1, 2, 3, \ldots$ as a dynamic flow. Call x bounded if its supremum norm is finite, and feasible if it also satisfies the conservation-of-flow constraints

$$\sum_{a \in T_i} x_a^p = \sum_{a \in H_i} x_a^{p-t_a} \quad \text{for } i \in N \text{ and } p > t_{max}, \qquad (2.1)$$

the zero-throughput constraint

$$\sum_{a\in A}\ \sum_{p=t_{max}-t_a+1}^{t_{max}} x_a^p = 0 \ , \tag{2.2}$$

and $c_a(x_a^p)$ is finite for $a \in A$ and $p = 1, 2, \ldots$.

Constraint (2.1) requires that the net amount of flow emanating from each node i in each period $p > t_{max}$ be equal to the net amount of flow entering the node in that period, i.e., there is conservation of flow at each node in each period $p > t_{max}$. In constraint (2.2), the quantity on the left-hand side is the net amount of flow "in transit" in period t_{max}, and we refer to this quantity as the <u>throughput</u> of x. Here we require the throughput to be zero. However, in Section 4 below we show that the above formulation is equivalent to one in which the requirement that the throughput be zero is relaxed and replaced by upper and lower bounds on throughput, or more generally by a convex cost of throughput. The zero-throughput constraint implies that the flow in transit is zero in each period after the first few because of the following lemma proved by the author (1981b).

LEMMA 1. The sum of the flows in transit in each period $p \geq t_{max}$ of a feasible dynamic flow is equal to its throughput.

We use the standard convention that

$$\sum_{p=r-t_a+1}^{r} x_a^p = - \sum_{p=r+1}^{r-t_a} x_a^p \ .$$

Thus the above sum is well defined for $t_a \leq 0$.

A dynamic flow $x = (x_a^p)$ is called <u>integer</u> if x_a^p is integer valued for $a \in A$ and $p = 1, 2, 3, \ldots$. If we wish to emphasize that a dynamic flow is not necessarily integer, we refer to it as being <u>continuous</u>.

In the following for fixed p we let

$$c^p(x) \equiv \sum_{a \in A} c_a(x_a^p) \text{ .}$$

Thus $c^p(x)$ is the sum of the costs of flows beginning in the p^{th} period. Let $c(x) \equiv \underline{\lim}_{r \to \infty} r^{-1} \Sigma_{p=1}^{r} c^p(x)$ be the long-run average-cost of x. A continuous (resp., integer) dynamic flow is called (<u>average-cost</u>) <u>optimal</u> if it is feasible and minimizes $c(\cdot)$ among all such flows. The <u>dynamic network-flow problem</u> is to find an (average-cost) optimal integer dynamic flow.

<u>The Dynamic Network</u>

Let $G = (N, A, t, c)$ be a static network. We expand G into an infinite network $G^\infty = (N^\infty, A^\infty, c^\infty)$, called the <u>dynamic network</u>, where

$$N^\infty = \{i^p : i \in N \text{ and } p \in \{1, 2, 3, \ldots\}\} \text{ ,}$$

for each arc $a = (i, j)$ and $p \geq \max(-t_a + 1, 1)$ there is an arc $a^p = (i^p, j^{p+t_a}) \in A^\infty$, and the cost of a flow x_a^p in arc a in period p is $c_a^p(x_a^p)$ for all $a \in A$ and $p = 1, 2, \ldots$. Node i^p represents node i of N in period p, and it is called the p^{th} <u>copy</u> of node i. Arc a^p represents the ability to send flow from the tail of a in period p reaching its head in period $p + t_a$ and it is called the

p^{th} copy of arc a. Figures 2.1a and 2.1b show a static network and the corresponding dynamic network. The numbers on the arcs of Figure 2.1a are the transit times.

If $x = (x_a^p)$ is a feasible dynamic flow, then x_a^p may be interpreted as the flow in arc a^p of G^∞, and constraint (2.1) represents conservation of flow at each node i^p for $i \in N$ and $p > t_{max}$.

The technique of expressing flows over time by expanding the network is now standard, and was used by Ford and Fulkerson (1962) in their classic text.

Preliminaries: Paths, Copies and Cuts

A path in a network (dynamic or not) is an alternating sequence of nodes and arcs $i_0, a_1, \ldots, a_k, i_k$ such that for each $j = 1, \ldots, k$ either a_j has head i_j and tail i_{j-1} or else it has head i_{j-1} and tail i_j. In the former case the arc is called a forward arc of the path; in the latter case it is called a backward arc. A directed path is a path in which every arc is a forward arc.

For a given static network, the transit time of a path is the sum of the transit times of the forward arcs of the path minus the sum of the transit times of the backward arcs.

An example illustrating these concepts is given in Figure 2.2. The path from node 1 to node 4 has forward arcs (2, 3) and (3, 4). The transit time of this path is the sum of the transit times of (2, 3) and (3, 4) minus the transit time of (1, 2). This value is 1. The path may also be viewed as a path from node 4 to node 1 with transit time -1.

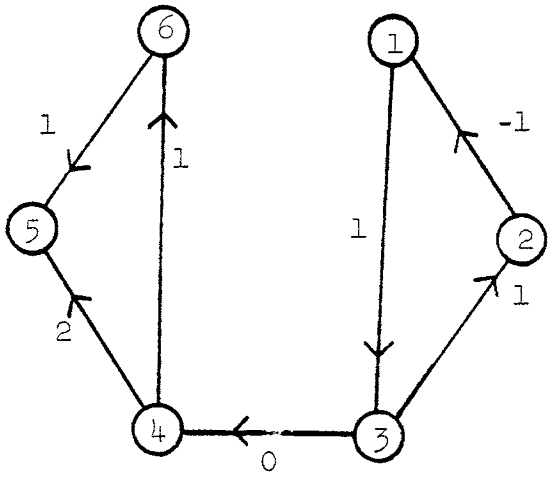


Figure 2.1a. A static network. The arc numbers are the transit times

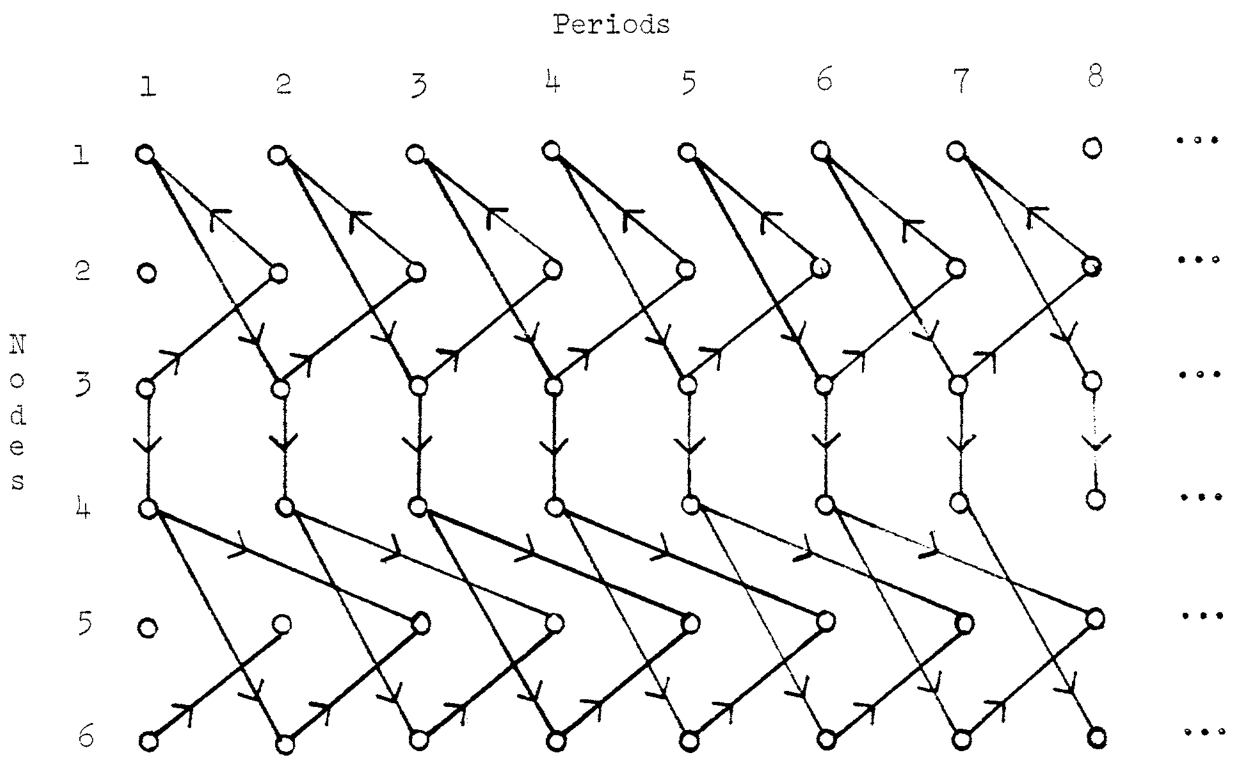


Figure 2.1b. The dynamic network derived by expanding the static network of Figure 2.1a.

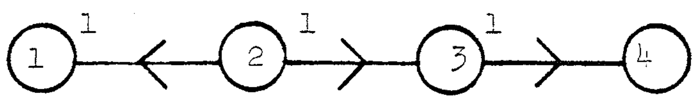


Figure 2.2. A path from node 1 to node 4 with unit transit time.

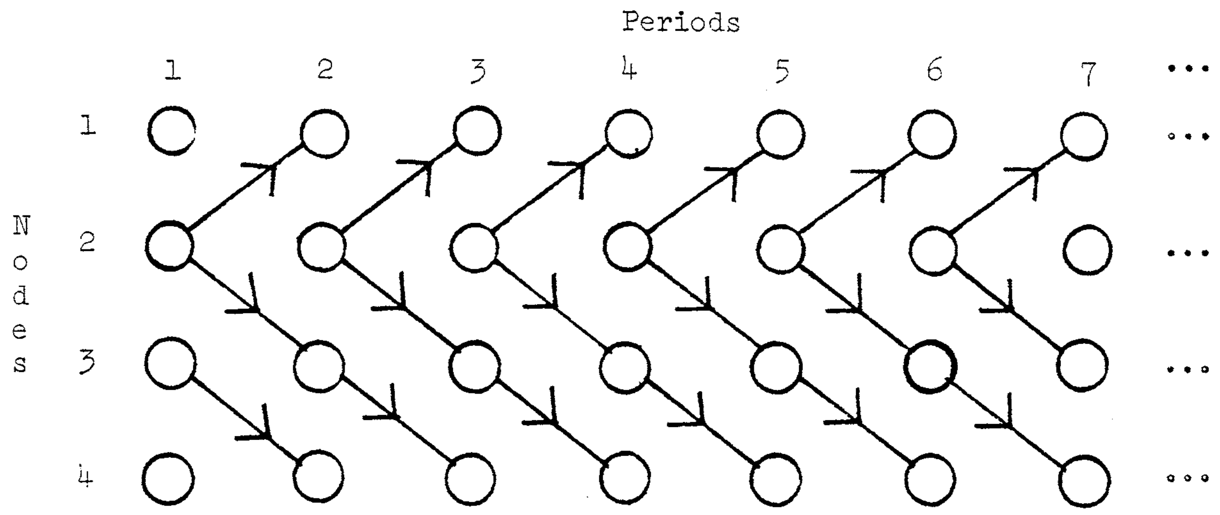


Figure 2.3. The dynamic network derived from the static network of Figure 2.2. The path from 1^p to 4^{p+1} is the p^{th} copy of the path in Figure 2.2.

A <u>cycle</u> is a path in which the initial node is the same as the final node. A cycle is <u>simple</u> if no node is repeated, except that the initial node and the final node are the same.

Let C be a cycle of G. A <u>flow around C</u> of k units is a static flow $y = (y_a)$ such that

$$y_a = \begin{cases} k & \text{if } a \text{ is a forward arc of } C \\ -k & \text{if } a \text{ is a backward arc of } C \\ 0 & \text{if } a \text{ is not an arc of } C . \end{cases}$$

It is easily verified that a flow around C satisfies conservation of flow at each node.

If $a \in A$ and $p \geq 1$, then arc $a^p \in A^\infty$ will be called the p^{th} <u>copy</u> of arc a, or simply a <u>copy</u> of a. The p^{th} copy of arc a is not defined for $p \leq -t_a$. Similarly i^p is called the p^{th} <u>copy</u> of node i. Let $P = i_0, a_1, \ldots, a_k, i_k$ be a path in G. Then the p^{th} <u>copy</u> of P is the path P', if one exists, in G^∞ with k arcs such that the initial node is i_0^p and such that the j^{th} arc of P' is the copy of the arc a_j whose head or tail is the $j - 1^{th}$ node of P'. The definition is illustrated in Figure 2.3, which is the dynamic network corresponding to the path in Figure 2.2. The path from 1^p to 4^{p+1} is the p^{th} copy of the path from node 1 to node 4; this path is not defined for $p = 1$.

The following result was proved by the author (1981b).

LEMMA 2. Let C be a simple cycle with transit time $t \geq 1$ in the static network. Then the infinite number of copies of C in the dynamic network comprise t node-disjoint infinite paths therein. ■

3. THE SOLUTION TECHNIQUE: ROUNDING OPTIMAL CONTINUOUS STATIONARY FLOWS

In this section we give algorithms for finding both optimal continuous and optimal integer dynamic flows. A dynamic flow $x = (x_a^p)$ is called <u>stationary</u> if $x_a^p = x_a^{p+1}$ for all $a \in A$ and $p \geq 1$. The continuous problem is readily solved using the theory developed in Orlin (1981a) which demonstrates that if there is an optimal continuous dynamic flow, then one such flow is stationary. To find an optimal integer dynamic flow, we round the fractional parts of an optimal continuous stationary flow so as to maintain feasibility and the same average cost per period. This results in an optimal integer dynamic flow that is periodic and is also an optimal continuous dynamic flow.

The rounding procedure is subtle, and depends not only on the preliminaries developed in the previous section, but also on the characterization of "fractionally extreme" optimal static flows developed in this section. The naive approach of rounding the fractional parts of flows up in some periods and down in other periods will not, in general, lead to a feasible dynamic flow.

THEOREM 3. (Sufficiency of Continuous Stationary Flows.) If there is a feasible (resp., optimal) continuous dynamic flow, then one such flow is stationary.

PROOF. If we replace the zero-throughput constraint (2.2) by the equivalent constraint that the sum of the arc flows in transit is 0 for each period (the equivalence following from Lemma 1), then the

the continuous dynamic network-flow problem is a special case of the dynamic convex programming problem. The result is then a specialization of Theorem 1 in Orlin (1981a). ■

The requirement that dynamic flows be bounded is crucial in the above result. An example is given in Orlin (1981b) of a problem for which there is no bounded dynamic flow satisfying (2.1) and having finite cost despite the existence of an unbounded dynamic flow having these properties. Moreover, if we append to that example a node 3, and an arc (3, 3) with transit time -1 and zero cost, we may also assume that there is zero throughput.

<u>The Static Network-Flow Problem</u>

Consider the problem of determining an optimal stationary dynamic flow. Since the flow in each period is the same, we may ignore the superscripts on the variables x_a^p and write the problem as the convex programming problem (3.1) of minimizing

$$\sum_{a \in A} c_a(x_a) \tag{3.1a}$$

subject to

$$\sum_{a \in H_i} x_a - \sum_{a \in T_i} x_a = 0 \quad \text{for} \quad i \in N \tag{3.1b}$$

and

$$\sum_{a \in A} t_a x_a = 0 \; . \tag{3.1c}$$

We refer to the vector $x = (x_a)$ as a static flow because any vector satisfying (3.1b) may be viewed as a circulation in the static network. Note that each static flow $x = (x_a)$ induces a stationary dynamic flow $x^\infty = (x_a^p)$ given by $x_a^p = x_a$ for $a \in A$ and $p = 1,2,3, \ldots$ The value of the left-hand side of (3.1c) is called the throughput of the static flow x because it is also the throughput of x^∞. A static flow $x = (x_a)$ is called feasible if (3.1a) is finite and if (3.1b) and (3.1c) hold. A static flow is called fractionally extreme if it is feasible and its fractional elements are uniquely determined given the values of its integer elements. A static flow x is called optimal if it minimizes (3.1a) over all feasible static flows.

As Ranel Erickson has pointed out to me, the characterization of fractionally-extreme flows below is partly implicit in the paper by Chen and Saigal (1977) concerning network-flow problems with linear side constraints. However, our results on the fractional parts of flows are new. Furthermore, the proof here is different.

THEOREM 4. (Characterization of Fractionally-Extreme Static Flows.) A feasible static flow is fractionally extreme if and only if either the flow is integer or else it is the sum of an integer static flow and a flow around a simple cycle of k/t units for some positive integers $k < t$ with t being the transit time of the cycle.

PROOF. Let y denote a feasible static flow. We prove the "if" part first. If y is integer, then it clearly is fractionally extreme. If y is not integer, it is the sum of an integer static flow and a flow of k/t units around a simple cycle C with $0 < k < t$ and t the transit time of C. Let x be another feasible static flow differing from y only on C. Then $z = y - x$ is a circulation, its throughput is zero, and $z_a = 0$ if a is not on the cycle C. Since z is a circulation that takes values only on a simple cycle, in order to satisfy conservation of flow z must be a flow of r units around C. But then the throughput is rt and hence $r = 0$. Therefore $x = y$, and thus y is fractionally extreme.

It remains to establish the "only if" part. To that end assume that y is fractionally extreme and let A' be the subset of arcs of A corresponding to the fractional elements of y. If $A' = \Phi$, the proof is complete. Thus suppose $A' \neq \Phi$. We first note that A' is a union of cycles. To see this, observe that if there were an arc in A' not on a cycle, then there would be a node i with a single arc a' in A' incident to it. Since the net flow through i is zero, it follows that the flow through a' is integral, contrary to the definition of A'.

We next show that there can not be two different cycles in A'. Suppose that C and C' are different cycles with arcs in A' and with respective transit times ℓ and ℓ'. By possibly reorienting and relabeling these cycles, we may assume that $0 \leq \ell \leq \ell'$. Let x be the circulation obtained by adding a flow around C of δ units and a flow around C' of δ' units, where δ and δ' are numbers chosen so that $\delta > 0$ and $\delta\ell + \delta'\ell' = 0$. Then x is a non-zero circulation and its throughput is zero. Furthermore, if δ, δ' are sufficiently close to zero, then $y + x$ is feasible and differs from y only on A', contradicting the fact y is fractionally extreme. Therefore, A' consists of a single simple cycle C.

Since y is a circulation, the net flow through each node is zero, and thus the net flow through each node as restricted to arcs of C is integral. Therefore the fractional parts of flows in the forward arcs of C is the negative of the fractional parts of the flows in the backward arcs of C, and we may write y as an integral flow plus a flow around C of k/ℓ, with $0 < k < \ell$ having no common factors. It remains to show that ℓ divides the transit time t of C. To this end, let y' be the flow derived from y by subtracting a flow of k/ℓ units around C. Then y' is an integral flow with throughput $t(-k/\ell)$. Since an integral flow has integral throughput, it follows that ℓ divides t. ■

Rounding and q-periodic Solutions

A dynamic flow $x = (x_a^p)$ is called q-periodic if $x_a^p = x_a^{p+q}$ for $a \in A$ and $p \geq 1$. The goal of this subsection is to develop a procedure for transforming a fractionally-extreme optimal static flow y into a q-periodic optimal integer flow x, where q is the least common denominator of the fractional parts of the elements of y. The procedure consists of rounding the fractional parts of the elements of y^∞ by increasing and decreasing flows along infinite paths in the dynamic network.

Let P be an infinite path in the dynamic network G^∞. To <u>increase the flow</u> along P by δ is to increase the flow in each forward arc of P by δ and decrease the flow in each backward arc by δ.

In the following we will use the expression $r(\text{mod } p)$ to denote the value q in $[0, p - 1]$ such that $q \equiv r(\text{mod } p)$. For example, $8(\text{mod } 3) = 2$, and $9(\text{mod } 3) = 0$.

If w and x are dynamic flows, we say x <u>rounds</u> w if x is integer and the supremum norm of $x - w$ is less than one.

THEOREM 5. (Optimality of rounding.) If y is a fractionally-extreme static flow and if q is the least common denominator of the fractional parts of the elements of y, then there is a feasible q-periodic dynamic flow x that rounds y^∞ and has the same average cost as y^∞. If also y is an optimal static flow, then x is an optimal continuous dynamic flow.

PROOF. If y is integer, then $q = 1$ and the result is trivially true on choosing $x = y^\infty$. Thus, suppose y is not integer. By Theorem 4, y is the sum of an integer static flow and a flow around a simple cycle C of k/t units for some positive integers $k < t$ with t being the transit time of C. Also q divides t. Let $r = kq/t$, so $r/q = k/t$. Now by Lemma 2, the copies of C consist of t disjoint infinite paths labeled $0, \ldots, t - 1$ so that the i^{th} copy of some given node of C belongs to path $i(\text{mod } t)$ for all $i \geq 1$. Let x be the integer dynamic flow formed from y^∞ by increasing the flow along path i by $(t - k)/t$ if $i \pmod q < r$ and decreasing the flow along path i by k/t otherwise.

Increasing the flow along a path by $(t - k)/t$ maintains conservation of flow at each node of the dynamic network except the first node of the path, rounds the flow in each forward (resp., backward) arc to the next highest (resp., lowest) integer, and increases the throughput by $(t - k)/t$. Decreasing the flow along a path by k/t also rounds all flows along the path to an adjacent integer and decreases the throughput by k/t. Thus x rounds y^∞.

Since q divides t, the flows along k/t paths increase by $(t - k)/t$ and the flows along $(t - k)/t$ paths decrease by k/t. Hence x has the same throughput as y^∞. Since conservation of flow is also maintained, x is feasible.

Moreover, x is q-periodic. To see this consider x_a^p and x_a^{p+q}. If arc a^p of the dynamic network is on path i, then a^{p+q} is on path $j = (i + q) \pmod t$. Therefore, either the flows along paths i and j are both rounded up or both rounded down, and hence $x_a^p = x_a^{p+q}$.

To complete the proof, it remains to show that the average cost of x is the same as that of y^∞, viz., $\Sigma_{a \in A}\, c_a(y_a)$. To see this, observe that for any nonnegative integer p, $\Sigma_{j=p+1}^{p+t}\, x_a^j = ty_a$ for each arc $a \in A$. Since $c_a(\cdot)$ is linear between successive integers, it follows that

$$\lim_{u \to \infty} \frac{1}{u} \sum_{p=1}^{u} c_a(x_a^p) = \frac{1}{q} \sum_{p=1}^{q} c_a(x_a^p) = c_a(y_a) ,$$

completing the proof. ■

EXAMPLE. Consider the static network illustrated in Figure 3.1, and suppose that a fractionally-extreme optimal static flow is: $y_{12} = 1/2$, $y_{23} = 1/2$, $y_{31} = 1/2$, $y_{11} = -2$. The cycle corresponding to the fractional parts has transit-time four and comprises four infinite-length paths in the dynamic network, and the paths are labeled 0, 1, 2, 3 so that node 1^i is on path $i \pmod 4$ for all $i \geq 1$. These four paths are illustrated in Figure 3.2.

The rounding procedure given in the proof of Theorem 5 is to round up the flow in paths 0 and 2 by 1/2 while rounding down the flow in paths 1 and 3 by 1/2. Note that rounding up the flow in paths 0 and 1 and rounding down the flow in paths 2 and 3 would also give an integer optimal flow, but the period length would be four instead of the least common denominator two of the fractional parts of the elements of the static flow.

Representing Optimal Periodic Solutions

Let $x = (x_a^p)$ be an integral optimal dynamic flow that is q-periodic. We call the $q \cdot A$-vector consisting of the values x_a^p for $a \in A$ and $1 \leq p \leq q$ a <u>periodic representation</u> of x. This representation can be easily written on a circular file and is a natural representation of the solution. This representation is usually efficient in practice, but unfortunately is not formally efficient because the value q may be as large as $n \cdot t_{max}$ in the number of bits used to express the input string (because t_{max} can grow exponentially with $|A|$).

Below we give another representation whose length always is bounded by a polynomial in the length of the input string, and is easily

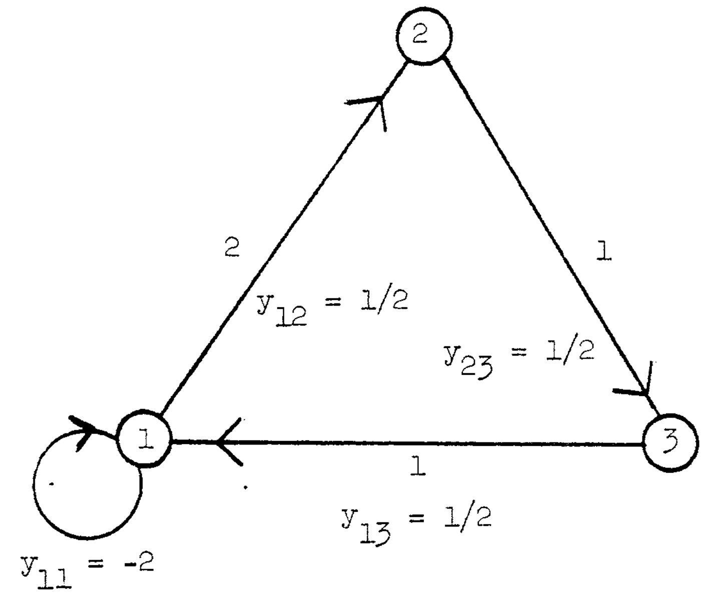


Figure 3.1. A static network with a given optimal static flow. The fractional flows lie on a cycle with transit time 4.

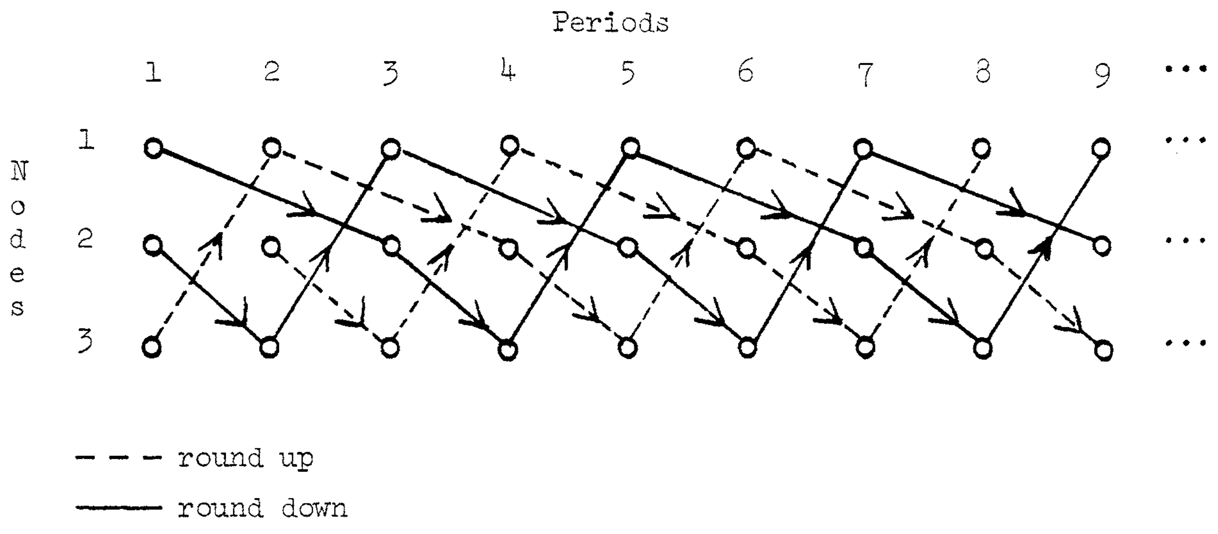


Figure 3.2. The four paths of the dynamic network of Figure 3.1 corresponding to the arcs with non-integer flow. If the fractional parts are rounded in the indicated way, the resulting flow is an integer optimal flow.

calculated starting from a fractionally-extreme optimum static flow y with a fractional flow r/q around a cycle C of transit time t.

To this end, label the t paths that comprise the infinite number of copies of C by $0, 1, \ldots, t - 1$ as in the proof of Theorem 5. For each arc $a \in C$, let d_a be chosen so that the d_ath copy of a is on path 0, and hence the pth copy of a is on path i for $i = (p - d_a)(\text{mod } t)$. Let F (resp., B) denote the subset of arcs of A that are forward (resp., backward) arcs of C. Then we can construct an integer optimal dynamic flow as follows:

$$x_a^p = \begin{cases} y_a & \text{if } a \in A - F - B \\ \lceil y_a \rceil & \text{if } a \in F \text{ and } (p - d_a)(\text{mod } q) \in [0, r - 1] \\ & \text{or } a \in B \text{ and } (p - d_a)(\text{mod } q) \in [r, q - 1] \\ \lfloor y_a \rfloor & \text{if } a \in F \text{ and } (p - d_a)(\text{mod } q) \in [r, q - 1] \\ & \text{or } a \in B \text{ and } (p - d_a)(\text{mod } q) \in [0, r - 1] \, . \end{cases}$$

Since a^p is on path j for $j = (p - d_a)(\text{mod } q)$, the above rounding of y^∞ corresponds to rounding up the flow in path i for $i(\text{mod } q) \in [0, r - 1]$ and rounding down the flows in the other paths. We will call the sextuple (y, F, B, d, r, q) a modular representation of x because any value x_a^p may be calculated from the sextuple using only modular arithmetic. The modular representation is a formally efficient representation even if the period length q is large, but it has the obvious drawback that more computation must be carried out before calculating the flows on the arcs.

4. VARIANTS OF THE MINIMUM CONVEX-COST DYNAMIC NETWORK-FLOW PROBLEM

Just as there are several equivalent variants of the minimum-cost network-flow problem, there are several equivalent variants of the dynamic network-flow problem. In fact, some of the applications given in Sections 5 and 6 are naturally expressed in terms of these variants. All of the variants below are expressed in terms of convex costs. However, we note that upper and lower bounds on a variable u can be written in terms of a convex cost $c(\cdot)$ on u such that $c(u) = \infty$ whenever u is less than the given lower bound or greater than the given upper bound.

Convex Costs on Flow into, out of, and through Nodes

For any node i and period p, the inflow (resp., outflow) is the sum of all flows entering (resp., leaving) node i in that period. The throughflow is the outflow minus the inflow. Consider a network (N, A, t, c) in which there are associated convex costs f_j^i, f_j^O, f_j^t on the inflow, outflow, and throughflow respectively of node $j \in N$ in each period $p \geq t_{max}$ where the convex costs are real or $+\infty$ valued and are linear between successive integers. Furthermore, relax the node throughflow constraints (2.1) and assume that the sum of the throughflows of all nodes in any period is 0. (If the sum of the throughflows is non-zero, then the throughput will change from period to period, as is evident from Lemma 1 and its proof in Orlin (1981b).)

We can reduce this problem to standard form by replacing the network G with a new network $G' = (N', A', t', c')$ constructed as follows. Replace each node $j \in N$ by three nodes of G' labeled

j^i, j^0, j^t, and add an additional node $n + 1$. For each arc $a = (j, k) \in A$ create an arc (j^0, k^i) in A', and refer to this arc as the copy of (j, k) in G'. The arc (j^0, k^i) has the same transit time and cost as (j, k). Finally, for each $i = 1, \ldots, n$, add the following three arcs each with transit time equal to zero: arc (j^i, j^t) with cost f^i_j, arc (j^t, j^0) with cost f^0_j, and arc $(n + 1, j^t)$ with cost f^t_j. This transformation is depicted in Figures 4.1a and 4.1b.

We form a 1:1 correspondence between flows in G and G' as follows. Let x be a dynamic flow in G (not necessarily satisfying conservation of flow through nodes). For each arc $a \in A$, let the flow in the copy $a' \in A'$ be the same in each period as the flow in a. Let the flow in all other arcs in periods $p \geq t_{max}$ be the unique amount of flow required so that conservation of flow is satisfied. Thus the flow in arc (j^i, j^t) (resp., (j^t, j^0)) is the inflow (resp., outflow) of node j in G, and the flow in arc $(n + 1, j^t)$ is the throughflow of j in G. (Note that the throughflow of $n + 1$ of G' is the sum of the throughflows of the nodes $1, \ldots, n$ of G, and this sum is 0). The above correspondence between dynamic flows of G and feasible dynamic flows of G' is easily invertible and is 1:1.

From the above, it is easy to see that the average cost of a feasible dynamic flow in G' is the average cost of the corresponding dynamic flow in G.

Convex Cost of Throughput

In the dynamic network-flow problem, the throughput is required to be fixed at zero. Consider the problem derived by relaxing the

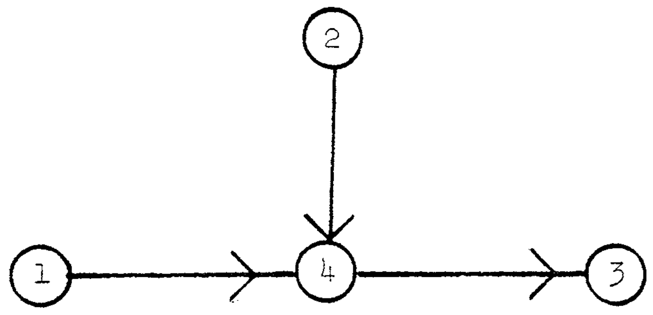


Figure 4.1a. A static network. We assume that there are convex costs on the inflow, outflow, and throughflow.

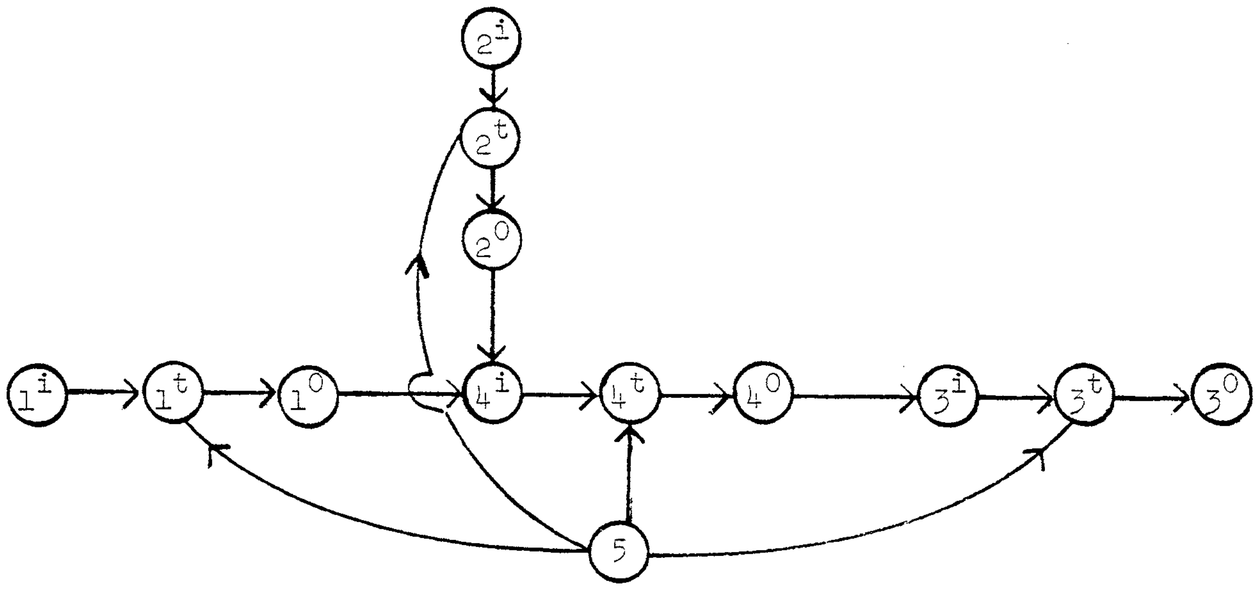


Figure 4.1b. The static network in Figure 4.1b after the transformation representing convex costs on inflow, outflow and throughflow.

throughput constraint and replacing it by a proper convex cost $\bar{c}(\cdot)$ on throughput such that the cost is linear between successive integers. (This cost is treated as a long-run average cost). If $G = (N, A, t, c)$ is the original static network with cost $\bar{c}(\cdot)$ on throughput, then we can transform it into standard form by creating a new network $G' = (N', A', t', c)$ with $N' = N \cup \{n + 1\}$ and $A' = A \cup \{\alpha\}$ where $\alpha = (n + 1, n + 1)$ is an arc with $t'_\alpha = -1$ and $c_\alpha(\cdot) = \bar{c}(\cdot)$. All other costs and transit times in G' are the same as the corresponding values in G.

Let x be a dynamic flow for G satisfying the conservation-of-flow constraints (2.1) but not necessarily the throughput constraint, and let f be the throughput of x. We associate with x a feasible dynamic flow y of G' as follows:

$$y_a^p = \begin{cases} x_a^p & \text{for } a \in A \\ f & \text{for } a = \alpha . \end{cases}$$

Then the average cost per period for y is the same as that for x, and the throughput of y is 0.

The above correspondence is also 1:1. To see this, suppose that y is a feasible dynamic flow for G'. Since conservation of flow holds at node $n + 1$, we must have $y_\alpha^p = y_\alpha^{p+1}$ for $p > t_{max}$. Deleting all instances of arc α leaves the corresponding dynamic flow x of G with throughput y_α.

Periodically Repeating Parameters

In the dynamic network-flow problem, the transit times and costs repeat from period to period. Consider the generalization of the dynamic network-flow problem in which the parameters repeat every k periods, i.e., for some $k > 1$ the transit times and costs of the flow initiated in arc a in period p are the same as in period $p + k$ for all $a \in A$ and $p = 1, 2, 3, \ldots$.

Let N and A be the nodes and arcs of the original periodic problem. We reduce the periodic problem to the standard dynamic problem by expanding the original network into a static network $G' = (N', A', t', c')$ with

$$N' = \{i^p : i \in N \text{ and } p = 1, \ldots, k\} .$$

The static network G' represents flow in a block of k consecutive periods which we call an <u>epoch</u>, and i^p denotes node i in period p of an epoch. Furthermore, there is an arc α in A' from i^r to j^p with transit time t'_α and cost $c'_\alpha(\circ)$ if there is an arc (i, j) in the original problem such that the transit time in period r is $p - r + kt'_\alpha$, and its cost is c'_α.

The transit time in the k-period expanded network differs from that in the original static network because transit time in the expanded network is measured in epochs. A flow in the expanded static network from i^r to j^p with transit time t'_α epochs corresponds to a flow with transit time $p - r + kt'_\alpha$ periods in the original static network.

EXAMPLE. Let G be the network depicted in Figure 4.2 and described in Table 4.2, where transit times for arcs (1, 2) and (2,3) oscillate between the two given values. To represent G as a static network with repeating parameters we expand G into the 2-period representation G' pictured in Figure 4.3 and described in Table 4.3.

Head	Tail	Transit Time
1	2	1 or 2
2	3	3 or 4
3	1	5

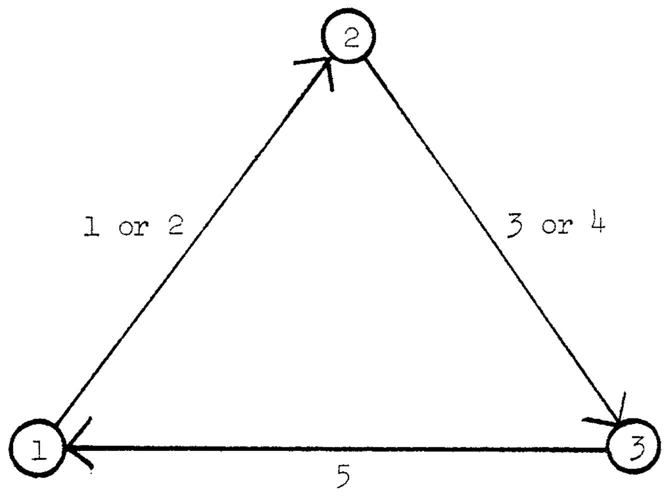
Table 4.2. The static network of Figure 4.1.

Head	Tail	Transit Time
1^1	2^2	0
1^2	2^2	1
2^1	3^2	1
2^2	3^2	2
3^1	1^2	2
3^2	1^1	3

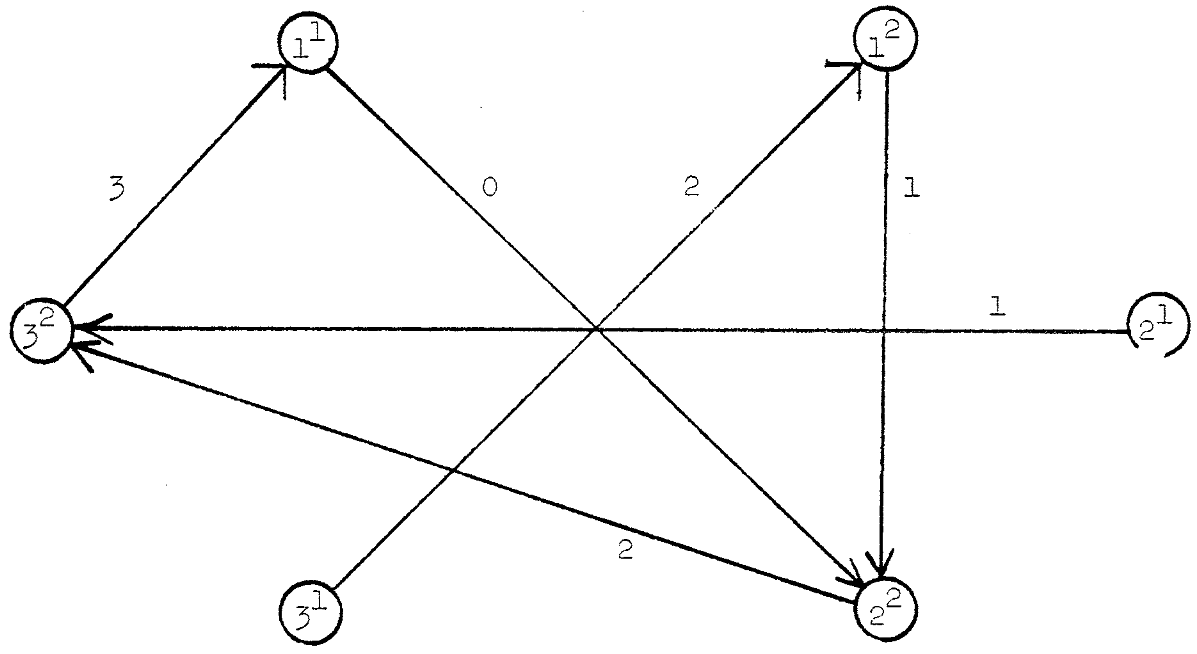
Table 4.3. The 2-period static network of Table 4.2.

5. CYCLIC CAPACITY SCHEDULING

The cyclic capacity scheduling problem is to minimize the per period cost of buying and selling integral amounts of capacity for intervals of time so as to satisfy periodically repeating demands for capacity. Veinott and Wagner (1962) showed that the finite-horizon version of the problem is a static finite-horizon network-flow problem.



<u>Figure 4.2</u>. A static network with its parameters repeating every two periods.



<u>Figure 4.3</u>. The 2-period expansion of the static network in Figure 4.2.

The Problem and Parameters

To model the cyclic capacity scheduling problem we first partition the infinite horizon into an infinite number of epochs, and each epoch is divided into n periods. Period i of epoch p is represented as i^p. The interval stretching from the beginning of period i^p to the beginning of period j^r is represented as $[i^p, j^r)$.

In the problem formulation, we let b_i denote the demand for capacity in period i^p, independent of p. This demand must be satisfied exactly. However, the case of excess or deficient capacity is included as a special case of this problem because the ability to have excess or deficient capacity in a period is the same as the ability to freely sell or buy capacity for that period. The opportunities for buying and selling capacity are periodic in that if capacity may be bought or sold in interval $[i^p, j^r)$, then it may also be bought or sold in $[i^{p+1}, j^{r+1})$.

The Associated Dynamic Network-Flow Problem

In the following model, the conservation-of-flow constraint is replaced with the constraint that the throughflow of node i is exactly d_i (defined below) in period p for $p > t_{max}$. This variant was shown to be equivalent to the dynamic network-flow problem in Section 4.

Let $G = (N, A, t, c)$ be a static network in which each node of N represents a period. If capacity may be bought or sold in the interval $[i^p, j^r)$, then there is an associated arc (i, j) of A with transit time $r - p$ epochs. The buying of capacity is represented by a positive flow in the arc, and the selling of capacity is represented by a negative flow in the arc. There are convex costs of buying and selling capacity.

The demand on flow through node i is

$$d_i = \begin{cases} b_1 - b_n & \text{for } i = 1 \\ b_i - b_{i-1} & \text{for } i = 2, 3, \ldots, n \end{cases}$$

and the throughput is fixed at $b = b_n$, which may be modeled as in Section 4.

To interpret the constraints, first observe that the throughflow of node i^p is equal to the net amount of capacity in period i^p minus the amount of capacity in the previous period. Furthermore, the throughput in epoch p is exactly the amount of capacity in period n^p because any flow in transit in epoch p represents capacity bought at a period j^r for $r \leq p$ and sold in or after period 1^{p+1}. Thus any feasible flow is such that:

1) the associated capacity in n^p is b_n for $p \geq t_{max}$,
2) the associated capacity in i^p is b_i for $p > t_{max}$, and
3) the costs are finite.

To see that (2) is satisfied, note that the capacity in 1^{p+1} is the capacity in n^p plus d_1, which is $b_n + d_1 = b_1$. Inductively, the capacity in period i^{p+1} for $i > 1$ is the capacity in period $(i - 1)^{p+1}$ plus d_i, which is b_i.

Cyclic Staffing Models

The cyclic capacity scheduling problem may be applied to two different cyclic staffing models according as the periods are either days or hours. In both cases, capacity is viewed as the number of persons working.

Consider first a workforce model in which each person works for five consecutive days at a time. Demand varies from day to day but repeats from week to week. The objective is to minimize the average number of excess workers per day. This is modeled in Table 5.1 and is portrayed in Figure 5.1 as a static network in which each node of the static network represents a day of the week. For instance, the arc (1, 6) represents the five day workstretch from the beginning of Sunday to the end of Friday (or equivalently the beginning of Saturday), while a negative flow in arc (1, 2) represents an excess number of workers on Sunday.

Next consider a staffing problem in which demands vary from hour to hour but repeat from day to day. Each person works a shift of consecutive hours, and shifts start at various times within the day. This model has potential applications in scheduling telephone operators round the clock. Furthermore, other organizations such as restaurants, bus companies, taxi-cab companies, and police forces staff workforces round the clock to meet demands that vary (to a major extent) periodically.

1-Day Cyclic Staffing

The cyclic staffing problem treated above appears to be a new model; however, if we add the restriction that a schedule repeats every epoch, then the resulting problem is the 1-day cyclic staffing problem. Many special cases of this problem have been considered in the literature,

Workstretch	Tail	Head	Lower Bound	Upper Bound	Transit Time
Sunday-Thursday	1	6	0	∞	0
Monday-Friday	2	7	0	∞	0
Tuesday-Saturday	3	1	0	∞	1
Wednesday-Sunday	4	2	0	∞	1
Thursday-Monday	5	3	0	∞	1
Friday-Tuesday	6	4	0	∞	1
Saturday-Wednesday	7	5	0	∞	1

Table 5.1a

Capacity	Tail	Head	Lower Bound	Upper Bound	Transit Time
Sunday	1	2	$-\infty$	0	0
Monday	2	3	$-\infty$	0	0
Tuesday	3	4	$-\infty$	0	0
Wednesday	4	5	$-\infty$	0	0
Thursday	5	6	$-\infty$	0	0
Friday	6	7	$-\infty$	0	0
Saturday	7	1	$-\infty$	0	1

Table 5.1b

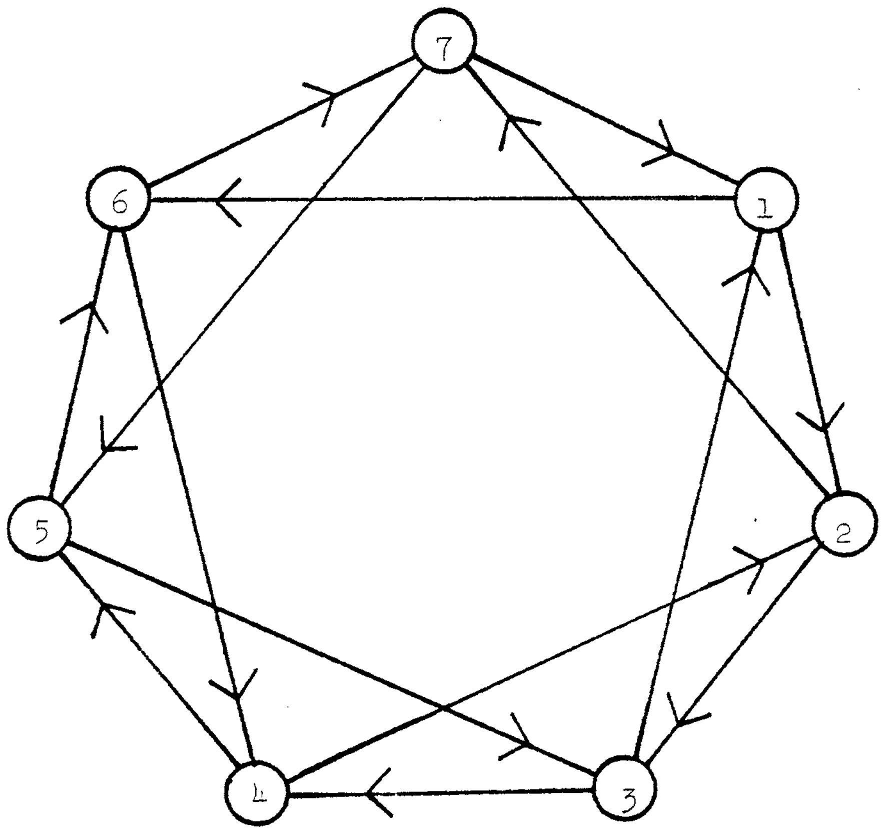


Figure 5.1. The static network for the 5-day workforce scheduling problem of Table 5.1.

for example, Baker (1974), Bartholdi and Ratliff (1978), Bartholdi et al. (1980), and Tibrewala et al. (1971). In the problem illustrated in Figure 5.1, the added restriction would imply, for instance, that the number of persons on the workstretch from Sunday to Friday is the same each week.

The largest subclass of the 1-day cyclic staffing problem to be solved with a polynomial algorithm is that by Bartholdi, Orlin and Ratliff (1980), which includes the subclass of cyclic staffing problems in which all shifts have the same length. There is no known polynomial algorithm that solves the entire class of 1-day cyclic staffing problems; however, the problem is not yet classified as NP-complete.

Although the cyclic capacity scheduling problem differs from the 1-day cyclic staffing problem in that solutions are not required to repeat each epoch, the solution obtained for the cyclic capacity scheduling problem "almost repeats" each epoch. The solution is obtained with the rounding algorithm in Section 3, and thus the integer solution varies by at most one from epoch to epoch. In terms of staffing, the number of persons on any shift varies by at most one from day to day.

6. OTHER APPLICATIONS

In recent years networks have been applied to a variety of situations including production, transshipment, inventory, and workforce models (see, for example, Glover and Klingman (1977)). When the above models involve parameters that repeat periodically over time, they may often be modeled as dynamic network flows. Below, we give a list of potential applications that is not intended to be complete, but is offered as a representative sampling.

The Minimum Cost-to-Time Ratio Circuit Problem

This first application differs from the others in that it is a previously solved problem. Dantzig, Blattner, and Rao (1967) introduced the "tramp steamer problem" which involves a steamer visiting n distinct ports. Traveling from port i to port j takes t_{ij} days and earns a profit of p_{ij} dollars, and both the transit time and profit are independent of the starting time for the trip. The objective is to determine an infinite-horizon tour that maximizes the average daily profit. Although the problem is phrased as a transportation problem, it has applications in several areas as detailed in Fox (1969). The problem is also equivalent to deterministic semi-markov decision chains, as described by Fox.

The static network has n nodes, one for each port, and for each pair i, j of distinct nodes there is an arc (i, j) with transit time t_{ij} and unit cost $-p_{ij}$. The upper and lower bounds on arc flows are 1 and 0 respectively, and the throughput is fixed at 1, representing the tramp steamer.

Dantzig, Blattner and Rao formulated the problem with the above static network. They observed that each basic solution of the static network-flow problem is a flow around a <u>circuit</u>, which is a simple directed cycle. Each circuit induces an infinite horizon tour. Ports are traveled in the order that they appear on the circuit, and the average daily cost is the ratio of the cost of traveling the circuit to its transit time. Thus an optimal circuit has the minimum cost-to-time ratio and induces an optimum tour. This tour is exactly the same tour determined by the rounding procedure in Section 3. To see this, note that if t is the transit time of the circuit C, then the optimum continuous-valued static solution has a flow $1/t$ on each of the arcs on the circuit and 0 elsewhere, thus achieving a throughput equal to 1. In the dynamic network, the flows on $t - 1$ of the resulting t infinite-length paths are rounded down while the flow on the remaining path is rounded up. It is this last path that gives the optimum tour. Thus the rounding procedure of Section 3 is a generalization of the method for solving the tramp steamer problem.

Incidentally, the minimum cost-to-time ratio circuit problem has excited a fair amount of interest in recent years. Many authors including Lawler (1967), Rinaldi (1975), Megiddo (1978), and Fox (1969) have attempted to find efficient methods for computing the optimum circuit. Lawler gave an $O(n^3 \log(n + t_{max}))$ algorithm, and more recently Megiddo found an $O(n^4 \log n)$ algorithm. In the special case in which each transit time is either 0 or 1, Karp and Orlin (1980) discovered an $O(n^3)$ algorithm. This special case has applications in the area of workforce scheduling (Bartholdi et al. (1980)), and in cyclic lot sizing (Graves and Orlin (1980)).

Airplane Routing

Consider the problem of scheduling a fixed number of aircraft for a partially fixed periodic schedule of daily repeating flights between n cities. Each flight is expressed in terms of (1) its departure site and time, e.g., Boston at 3 PM, (2) its arrival site and time, (3) its cost (or profit), and (4) whether it is required or optional. A required flight must be flown each day, whereas an optional flight may be flown at the scheduler's prerogative.

The aircraft are considered to be identical in that any aircraft may fly any route. Furthermore, we do not require that the flight schedule repeat daily. Indeed, an optimal schedule may repeat only after a number of days.

The problem of minimizing the number of aircraft needed to fly the schedule was proposed and solved by Dantzig (1962) and Orlin (1981b). Rather than repeat the way to model this problem, we refer the reader to (1981b). The difference here is that we associate a cost with each flight and we require a fixed number of aircraft. Other than that, the detailed explanation in (1981b) is appropriate for the minimum-cost model.

Dantzig (1962) formulated the variant of the above problem in which the final schedules are constrained to be stationary. His formulation is the static network-flow variant (3.1) of the dynamic network-flow problem subject to the additional constraint that flows are integer valued. In general, this integrality requirement will result in a static solution whose cost is greater than that of an optimal dynamic network flow. Moreover, there is no known polynomial algorithm for solving this integer programming problem.

Dantzig's heuristic solution technique is to assign a unit price p (Lagrange multiplier) to each airplane and to replace the constraint on the fixed number f of airplanes by an associated cost in the objective function. Assuming that there is a feasible schedule with at most f airplanes, then as the price p is varied parametrically from 0 to ∞, the number of airplanes in the resulting schedule decreases monotonically until there is a value p^* for which there are alternative optima with

f' and f'' airplanes respectively with $f' \leq f \leq f''$. In this way, one can obtain optimal solutions for the problems in which the number of airplanes is fixed at either f' or f'', and Dantzig's heuristic is to select the "preferred" of the two schedules.

Cyclic Production, Storage and Transshipment

Consider a number of cities with demand for a certain good that varies periodically over time, e.g., demand for bread or for petroleum products. We assume that demand is satisfied by shipping goods in a fixed number of trucks from a number of supply/production sites, where the cost of production is assumed to be convex. We restrict our attention to the case in which each truck must unload all of its goods at the demand site upon arriving. (The case in which a single truck can service many demand sites without reloading is NP-complete, as is proved in the appendix.) The objective is to determine a production schedule and shipping schedule over time so as to minimize the daily cost.

Below we consider two cases of the problem for which demands and costs repeat daily. The periodic problem may be formulated using the technique in Section 4 for expanding static networks.

The problem is formulated as follows:

(1) production site i has a convex cost $c_i(\cdot)$ of production for $i = 1, \ldots, r$;

(2) demand site i has a demand b_i for goods repeating daily for $i = r + 1, \ldots, n$;

(3a) the number of trucks is bounded above and below, <u>or</u>

(3b) storage is allowed at the production and demand sites at a convex cost.

Constraints (3a) and (3b) are related in that flow is measured in goods traveling over time; if storage is allowed, then storage will be interpreted as throughput as will a truck traveling. Simultaneously allowing both (3a) and (3b) results in a problem that is NP-complete, as proved in the appendix, even in the case in which there is but one truck.

The static network has node set $\{1, 2, \ldots, n\}$. For each production site i and storage site j, there are arcs (i,j) and (j,i) with transit times that are the number of days of travel time between the two sites. Thus the static network is a complete directed bipartite graph. For $i = r + 1, \ldots, n$ there is an additional constraint that the flow into i is b_i in each period, and for $i = 1, \ldots, n$ the flow out is bounded above by u_i and below by ℓ_i. These constraints are modeled via the transformations given in Section 4. Combined with the conservation-of-flow constraint (2.2), these constraints guarantee that (1) and (2) are satisfied.

If no out-of-truck storage is allowed (in both models goods may be stored in the trucks), then the flow in transit is the number of trucks. Thus upper and lower bound constraints on the number of trucks may be modeled via upper and lower bound constraints on the throughput and can be modeled as in Section 4. If the number of trucks is not restricted, then we may model storage at site i by a loop (i,i) with transit time 1 and an appropriate convex storage cost.

We note that if the throughput is not bounded, then the static flow problem is a circulation problem with no additional side constraints, and thus each basic solution is integral. In this case, the rounding of Section 3 is not necessary, as the stationary optimal flow is integral.

Appendix. The NP-Hardness of Some Storage and Transshipment Problems.

In this section two of the problems described in Section 6 are shown to be NP-hard. We refer the reader who is unfamiliar with the concepts of NP-completeness to either Karp (1972) or Garey and Johnson (1979).

The first problem X_1 that we wish to consider is the problem of finding a feasible infinite-horizon tour for the production/transshipment problem of Section 6 under the additional proviso that a truck may service several demand sites before returning to a supply site. The second problem X_2 that we consider is the production/transshipment problem in which there is exactly one truck and storage is allowed at each site. To show that these two problems are NP-hard, we show that each includes as a special case the hamiltonian circuit problem, which is to find a simple directed cycle of n arcs in a directed graph. This problem was proved to be NP-complete by Karp (1972).

THEOREM A.1. Problem X_1 is NP-hard.

PROOF. Let $G = (N, A)$ be a directed graph. We transform the hamiltonian circuit problem on G into a production transshipment problem as follows:

i) node 1 is the unique supply node with a maximum supply of $n-1$ units per period;

ii) nodes $2, \ldots, n$ are demand nodes with a demand of 1 unit each per period;

iii) the transit time for each arc with head 1 is one; all other arcs have a transit time of 0;

iv) each arc has an associated unit cost of 1, which is the cost of a truck traveling the route; and

v) there is exactly one truck and it may carry $n-1$ units of goods.

We now claim that there is a hamiltonian circuit in G if and only if there is a feasible transshipment schedule with an average cost per period of n.

If there is a hamiltonian circuit, then an optimal schedule consists of traveling along the route induced by the circuit in each period, incurring a cost of n. Conversely, any feasible routing with one truck must consist of the truck picking up and delivering $n-1$ units of goods in each period. The total cost in some period is n only if the circuit in G induced by the truck's route consits of n arcs, and is thus a hamiltonian circuit. ∎

THEOREM A.2. Problem X_2 is NP-hard.

PROOF. Let $G = (N, A)$ be a directed graph. We transform the hamiltonian circuit problem into X_2 as follows. Construct $G' = (N', A')$ as below.

i) $N' = \{1, 2, \ldots, n, 1', 2', \ldots, n'\}$. Node i is a supply node with a supply of one unit per period. Node i' is a demand node with a demand of one unit in periods $2n$, $4n$, $6n$, ... and a demand of zero in all other periods.

ii) The arcs of A' are (i, i') for $i = 1, \ldots, n$ and (i', j) for each arc (i, j) of A, and all lower and upper bounds are 0 and 1, respectively.

iii) Storage is allowed at all sites except in periods $2n$, $4n$, $6n$, ...

This transformation is portrayed in Figures A.1 and A.2. We now claim that there is a feasible routing for the above problem if and only if there exists a hamiltonian circuit in G.

If there is a hamiltonian circuit $i_1, i_2, \ldots, i_n$ in G, then a feasible periodic tour is induced by the circuit in G' $i_1, i_1', i_2, i_2', \ldots, i_n, i_n', i_1$. Conversely, if there is a feasible tour, then in periods $2n, \ldots, 4n-1$, the truck must visit each of the sites $1', \ldots, n'$. However, this is only possible if there is a hamiltonian tour in G' and hence a hamiltonian tour in G.

The above transformation was into a periodic version of X_2. However, using the transformation of Section 4 it is easy to transform the above in polynomial time to a stationary version of X_2. ■

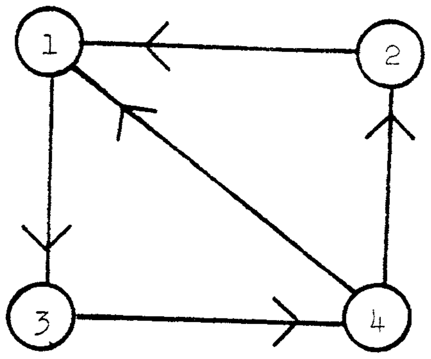


Figure A.1. A static network for a production and transshipment problem.

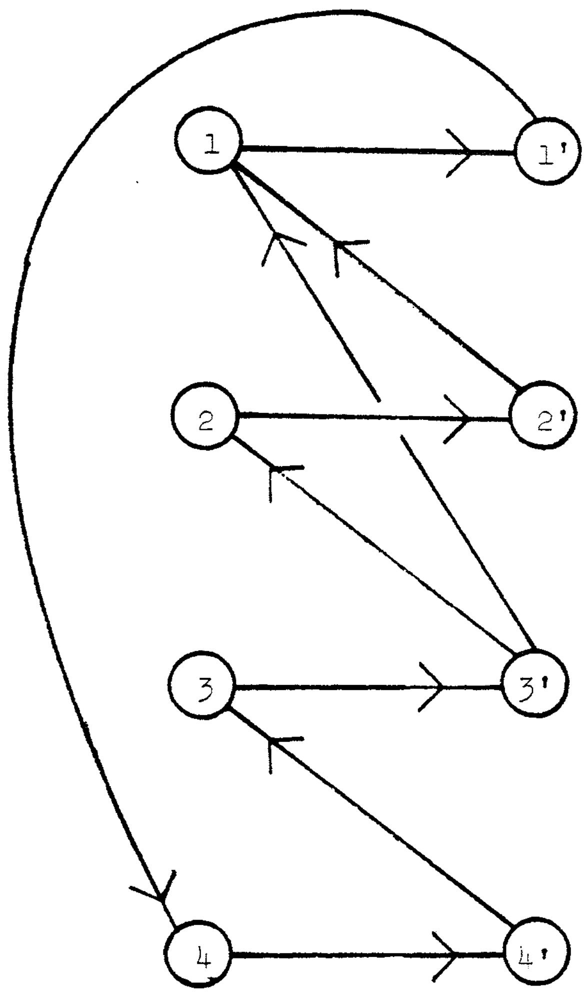


Figure A.2. The static network for the diagram in Figure A.1 after the transformation given in the proof of Theorem A.2.

ACKNOWLEDGMENT

I am indebted to my thesis supervisor, Professor Arthur F. Veinott, Jr., for helping to pose the periodic capacity scheduling problem; urging that I reformulate the results I found for that problem in terms of dynamic network flows and that I then generalize the theory to arbitrary dynamic networks; guiding the development of the theory; suggesting new results, applications, and directions; simplifying the notation and proofs; and for his painstaking care in improving the exposition.

REFERENCES

Baker, K. R. (1974). Scheduling a Full-Time Workforce to Meet Cyclic Staffing Requirements. Management Science 20, 1561-8.

Baker, K. R. (1976). Workforce Allocation in Cyclical Scheduling Problems. Operational Research Quarterly 27, 155-167.

Bartholdi, J. J. and H. D. Ratliff (1978). Unnetworks, with Applications to Idle Time Scheduling. Management Science 24, 850-858.

Bartholdi, J. J., III, J. B. Orlin, and H. D. Ratliff (1980). Cyclic Staffing via Integer Programs with Circular Ones. Operations Research 24, 1074-1085.

Chen, S. and R. Saigal (1977). A Primal Algorithm for Solving a Capacitated Network_Flow Problem with Additional Linear Constraints. Networks 7, 59-80.

Dantzig, G. B. (1962). Consulting work for United Airlines.

Dantzig, G. B., W. Blattner, and M. R. Rao (1967). Finding a Cycle in a Graph with Minimum Cost to Times Ratio with Application to a Ship Routing Problem. In P. Rosentiehl (ed.). Theory of Graphs. Dunod, Paris, Gordon and Breach, New York, 77-84.

Dantzig, G. B. and D. R. Fulkerson (1954). Minimizing the Number of Tankers to Meet a Fixed Schedule. Naval Research Logistics Quarterly 1, 217-222.

Edmonds, J. and R. M. Karp (1972). Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems. Journal of the Association of Computing Machinery 19, 248-264.

Folkman, J. and D. R. Fulkerson (1970). Flows in Infinite Graphs. Journal of Combinatorial Theory 8, 30-44.

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

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

Fox, B. (1969). Finding Minimal Cost-Time Ratio Circuits. Operations Research 17, 546-550.

Garey, M. R. and D. S. Johnson (1979). Computers and Intractibility: A Guide to the Theory of NP-Completeness. W. H. Freeman and Co., San Francisco.

Glover, F., D. Klingman, and C. McMillan (1977). The Netform Concept: A More Effective Model Form and Solution Procedure for Large Scale Nonlinear Problems. Management Science Report No. 77-4, Graduate School of Business Administration, University of Colorado.

Graves, S. C. and J. B. Orlin (1980). The Infinite-Horizon Dynamic Lot-Size Problem with Cyclic Demand and Costs. Working Paper, Operations Research Center, M.I.T.

Karp, R. (1975). On the Computational Complexity of Combinatorial Problems. Networks 7, 45-68.

Karp, R. M. and J. B. Orlin (1980). Parametric Shortest Path Algorithms with an Application to Cyclic Staffing. Discrete Applied Mathematics, to appear.

Lawler, E. L. (1967). Optimal Cycles in Doubly Weighted Linear Graphs. In P. Rosentiehl (ed.) Theory of Graphs. Dunod, Paris, Gordon and Breach, New York, 209-214.

Lawler, E. L. (1967). Combinatorial Optimization: Networks and Matroids. Holt, Rinehart and Winston, New York.

Maxwell, W. L. and R. C. Wilson (1978). Analysis of Dynamic Material Handling Systems by Network Flow. Working Paper No. 4, Department of Industrial and Operations Engineering, The University of Michigan.

Megiddo, N. (1978). Combinatorial Optimization with Rational Objective Functions. In Proceedings of the 10th ACM Symposium on the Theory of Computation, San Diego, California, 1-12.

Orlin, J. B. (1981a). Dynamic Convex Programming. Chapter 2 of this thesis.

Orlin, J. B. (1981b). Maximum-throughput Dynamic Network Flows. Chapter 3 of this thesis.

Orlin, J. B. (1977). Quick Optimal Weekly Scheduling with Two Consecutive Days Off. Technical Report 77-1, Department of Operations Research, Stanford University, Stanford, California.

Rinaldi, S. (1975). Optimal Constrained Cycles in Graphs. In S. Rinaldi (ed.). Topics in Combinatorial Optimization. CISM Courses and Lectures No. 175, Springer-Verlag, New York, 45-68.

Segal, M. (1974). The Operator Scheduling Problem: A Network Flow Approach. Operations Research 22, 803-24.

Simpson, R. W. (1968). Scheduling and Routing Models for Airline Systems. Report FTL-R68-3, Department of Aeronautics and Astronautics, MIT, 100-107 and 128-134.

Tibrewala, R., D. Philippe, and J. Browne (1972). Optimal Scheduling of Two Idle Periods. Management Science 19, 71-5.

Veinott, A. F., Jr. and H. M. Wagner (1962). Optimal Capacity Scheduling - I and II. Operations Research 10, 518-46.