June, 1981 LIDS-P-1100

AN OPTIMAL CONTROL APPROACH TO DYNAMIC ROUTING IN NETWORKS

PART II: A MAXIMAL FLOW APPROACH

Mario Jodorkovsky and Adrian Segall
Department of Electrical Engineering
Technion - Israel Institute of Technology
Haifa, Israel

ABSTRACT

This paper presents a new approach for building the feedback solution for the minimum delay dynamic message routing problem for single destination networks. The necessary and sufficient conditions for optimality derived in previous works are interpreted in terms of weighted maximal flow problems. Several properties concerning these problems are obtained and used to develop a simplified algorithm for building the feedback space. The various steps of the algorithm are presented and motivated in detail.

*The work of A. Segall was conducted on a consulting agreement with the Laboratory for Information and Decision Systems at MIT with partial support provided by the Advanced Research Project Agency of the U.S. Department of Defense (monitored by ONR) under contract No. N00014-75-C-1183.

This paper has been submitted to the IEEE Transactions on Automatic Control.

I. INTRODUCTION

In previous papers [1], [2], a model for the analysis of dynamic routing in networks has been proposed and the principles of an algorithm for the solution of the resulting dynamic equations have been introduced. The model gives rise to a linear optimal control problem with linear state and control inequality constraints and linear integral cost functional. The application of the maximum principle to this model dictates that the necessary and sufficient condition for a control law to be optimal is to be the solution of a certain linear program parametrized by time. Furthermore, it is indicated in [2] that the conceptual algorithm for the solution of the dynamic problem can be made implementable for the case when all traffic in the network has a single destination and there are no priorities, and in this paper we restrict our attention to this class of problems.

The main purpose of the present paper is twofold. First we introduce a new interpretation to the (parametrized) linear program that provides the necessary and sufficient condition for optimality in terms of a weighted maximal flow problem. It is well known that algorithms for maximal flow problems are much more efficient than those for general linear programs, so that this interpretation may offer large computation savings. In our case, this point of view provides the additional benefit of allowing us to obtain a series of properties with extensive implications to the development of the actual algorithm. The introduction of this maximal flow approach is the subject of Section II.

The second purpose of the present paper is to present an algorithm for the solution of the optimal control problem. The insight and properties of

the maximal flow approach developed in Section II together with the main properties of the problem in [2, Theorem 4] allow us to considerably reduce the complexity of the algorithm, converting it into an implementable algorithm for moderate size networks. In Section III the algorithm is introduced, motivated and applied to an illustrative example, and several concluding remarks appear in Section IV.

II. THE MAXIMAL FLOW APPROACH

As said before, we restrict our attention to the case of single destination networks with unity weightings in the cost functional. In addition, we assume that all the input rates to the network are zero, so that our problem is to find the optimal control (routing) that empties the network, starting from a given distribution of backlogged traffic. Theorem 4 in [2] gives the main properties of the solution and in the present paper we heavily rely on these results, as well as on the main optimality conditions obtained in [2, Theorems 1,2].

The dynamic equations, constraints and cost functional for the problem are [2, Eq. (7)-(11)]:

$$\dot{\underline{x}}(t) = \underline{B}\,\underline{u}(t) \; ; \quad \underline{x}(t_o) = \underline{x}_o \; ; \quad \underline{x}(t_f) = \underline{0} \; ; \quad \underline{x}(t) \geq 0 \tag{1}$$

$$U = \{\underline{u}(t) \leq \underline{C} \; ; \quad \underline{u}(t) \geq 0\} \tag{2}$$

$$J = \int_{t_o}^{t_f} \Sigma x_i(t)dt \; , \tag{3}$$

where $\underline{B}$ is the incidence matrix of the network and $\underline{C}$ the capacity vector.

Necessary and sufficient conditions for a control law $\underline{u}^*(t)$ to be optimal is to satisfy (1) and also to solve the linear program

$$\min_{\underline{u}(t)\in U} [\underline{\lambda}^T(t)\dot{\underline{x}}(t)], \qquad \text{all } t \in [t_o, t_f], \tag{4}$$

where $\underline{\lambda}(t)$ is a costate vector that satisfies the appropriate backward dynamic equations [2, Eq. (15)-(19)]. In view of [2, Thm. 4d)], there is no loss of generality if we restrict our attention to trajectories with non-positive slopes (forward in time), so that (4) can be rewritten

$$\min[\underline{\lambda}^T\dot{\underline{x}}], \qquad \underline{u} \in U, \quad \dot{\underline{x}} \leq 0 \tag{5}$$

where, for simplicity of notation, we have also suppressed explicit reference to time. A control vector $\underline{u}^*$ satisfying (1) and (5) is said to be globally optimal.

For a given time t and a given trajectory $\underline{x}(\cdot)$, let B and I denote the sets of states x_i for which $x_i(t) = 0$ and $x_i(t) > 0$ respectively (named boundary and interior sets). A control vector satisfying (1) and

$$\begin{aligned} &\min[\underline{\lambda}^T\dot{\underline{x}}], \quad \underline{u} \in U, \quad \dot{x}_i \leq 0 \quad \text{for} \quad x_i \in I \\ &\qquad\qquad\qquad\quad \dot{x}_i = 0 \quad \text{for} \quad x_i \in B \end{aligned} \tag{6}$$

is said to be constrained optimal [2, Note 1]. Clearly, every global optimal control is constrained optimal and [2, Thm. 4c] states that for the case under consideration the opposite is true as well. Consequently, we can regard (6) as replacing (5) in the necessary and sufficient conditions for optimality, and the rest of the present section will be devoted mainly to the analysis of equation (6).

First observe that by defining $y_i = -\dot{x}_i$ for $x_i \in I$ and by explicitly writing the incidence matrix $\underline{B}$ and the vector $\underline{u}$ (see [2, Eq. (1)]), the minimization problem (6) becomes:

$$\max F = \sum_{i:x_i \in I} \lambda_i y_i \tag{7a}$$

with constraints

$$-\sum_j u_{ij} + \sum_k u_{ki} + y_i = 0 \qquad \forall i: x_i \in I \tag{7b}$$

$$-\sum_j u_{ij} + \sum_k u_{ki} = 0 \qquad \forall i: x_i \in B \tag{7c}$$

$$y_i \geq 0, \quad \underline{u} \in U. \tag{7d}$$

Now observe that problem (7) is a (weighted) maximal flow problem for a network that is obtained from the original network by adding an extra node, named source s, and links with infinite capacity connecting s to all nodes i for which $x_i \in I$ (see Fig. 1). The network of Fig. 1b) will be called the network corresponding to problem (7) or the network corresponding to the set I. The main idea of the present work is to exploit properties of this maximal flow problem and algorithms for its solution, in order to obtain properties and algorithms for the solution of our original dynamic control problem. The rest of this section is devoted to the exposition of these properties.

Theorem 1:

(a) The costates $\lambda_i(t)$ corresponding to states $x_i(t) \in I$ are strictly positive.

(b) Any optimal solution of problem (7) with arbitrary positive λ's is also an optimal solution of problem (7) with $\lambda_i = 1 \ \forall x_i \in I$.

Proof: a) Assume that x_i reaches the boundary (forward in time) at time τ with an optimal slope $\dot{x}_i < 0$. Now clearly $\lambda_i(\tau) \geq 0$, since otherwise the necessary and sufficient condition (4) says that it is not optimal for x_i to reach the boundary at τ. Moreover, the dynamic equation for costates corresponding to states travelling on interior arcs [2, Eq. (15)-(17)] is $-\dot{\lambda}_i(t) = 1$ and therefore $\lambda_i(t) > 0$ for all $x_i(t) \in I$.

b) Let F_1 be the optimal value of problem (7) with $\lambda_i = 1$ for all $x_i \in I$ and let y^* be a solution vector of problem (7) with arbitrary $\lambda_i > 0$ for $x_i \in I$. Clearly $\sum_{x_i \in I} y_i^* \leq F_1$ and, on the other hand, the inequality cannot be strict since otherwise, the Maximal Flow Theorem [3] implies that a path can be found between the source and destination nodes of the network corresponding to problem (7) on which the flow can be increased, contradicting the fact that y^* solves (7). Consequently,

$$\sum_{x_i \in I} y_i^* = F_1 \tag{8}$$

implying that y_i^* is a solution of (7) with $\lambda_i = 1 \ \forall x_i \in I$.

From now on, and as a result of Theorem 1 we concentrate on problem (7) with $\lambda_i = 1, \forall x_i \in I$, and use the notations $N(I)$ for the network corresponding to the set I, MFP(I) for the maximal flow problem (7) corresponding to the set I with $\lambda_i = 1, \ \forall x_i \in I$ and max(I) for the maximal flow value corresponding to MFP(I).

Theorem 2:

(a) Suppose there exists two sets of states I and I' such that $I \subset I'$ and max(I) = max(I'). Then all basic optimal solutions of MFP(I) are basic optimal solutions of MFP(I'), and all minimal cuts of $N(I)$ are also minimal cuts of $N(I')$.

b) For a given set I, if there is no set I' satisfying the conditions in (a), then a minimal cut of $N(I)$ is $(X,\bar{X})$, where:

$$X = \{s\} \cup \{x_i \in I\}.$$

Moreover, when $\max(I)$ is achieved, then all links

$$\{(i,k) \in N(I) \mid x_i \in I,\ x_k \notin I\}$$

have flow equal to the corresponding capacity c_{ik} and all links

$$\{(j,i) \in N(I) \mid x_i \in I,\ x_j \notin I\}$$

have zero flow.

c) If there exist two sets I and I' such that $I \subset I'$ and $\max(I) = \max(I')$, then for any set I'' holds $\max(I \cup I'') = \max(I' \cup I'')$.

Proof: a) Take all basic optimal solutions of MFP(I) and add new variables $\{y_i \geq 0,\ \forall x_i \in I'/I\}$ as non-basic variables with value zero. Clearly, all these solutions satisfy the constraints of MFP(I'). Moreover, since $\max I = \max I'$ and the new variables are non-basic, the solutions are also basic optimal solutions for MFP(I'). Take now all solutions of MFP(I') satisfying $\{y_j = 0,\ \forall x_j \in I'/I\}$. Removing from $N(I')$ the links corresponding to $\{y_j, \forall x_j \in I'/I\}$ will give exactly $N(I)$ hence all minimal cuts of $N(I)$ are minimal cuts of $N(I')$.

b) Clearly, every node $\{i \mid x_i \in I\}$ belongs to X since there are no upper bounds on the variables y_i. Now, suppose that there exists a minimal cut $(X',\bar{X}')$ such that $X' = X \cup j$ where $x_j \notin I$. Then we can add to the network $N(I)$ a new variable $y_j \geq 0$ as non-basic with value zero (i.e. a link

connecting the source with node j with zero flow). But note that $I \subset (I \cup x_j)$ and $\max(I) = \max(I \cup x_j)$ contradicting the assumptions. By the Maximal-Flow-Min-Cut Theorem [3] and links connecting X with $\bar{X}$ have flow equal to their capacity and all links connecting $\bar{X}$ with X have no flow.

c) Consider a solution of MFP(I') satisfying $\{y_j = 0, \forall x_j \in I'/I\}$. Part (a) ensures the existence of such a solution. Now add to the network $N(I')$ links corresponding to $\{y_k \mid x_k \in I'', \; x_k \notin I' \cap I''\}$ and achieve maximal flow in the resulting network $N(I'' \cup I')$ without changing the flow on links $\{y_i \mid x_i \in I'\}$. Now remove the links $\{y_j \mid x_j \in I'/I, \; x_j \notin I''\}$. Since those links have no flow, the maximal flow does not change, therefore $\max(I \cup I'') = \max(I' \cup I'')$.

The proof of the next theorem is somewhat long and therefore not included here:

Theorem 3: [4, Thrms. 3.1, 3.2] Assume that the costates corresponding to states on the boundary are kept at their leave-the-boundary value (cf. [2, Thm. 4a]) at all times. Then all costates satisfy $\dot{\lambda}_i(t) = 0$ or -1 and

$$\lambda_i(t) = \dot{\lambda}_i(t) = 0$$

if and only if the following hold :

(i) $x_i(t) \in B$

(ii) there are no sets I' and I such that $x_i \in I'$, $x_i \notin I$, $I \subset I'$, $\max I = \max I'$ and all states in I have left the boundary before time t backwards in time (i.e. at some time strictly larger than t).

III. THE ALGORITHM FOR BUILDING THE FEEDBACK CONTROL REGIONS

With the results of Section II, we are ready to introduce the algorithm that implements the constructive dynamic programming concept introduced in [2] and builds the feedback control regions that represent the solution to the optimal dynamic routing problem. In order to obtain a complete characterization of the feedback space one has in principle to find all backward optimal trajectories and the corresponding optimal control vectors. However, due to the special nature of the problem, the results of [2] and the properties of Section II of this paper, we can obtain an algorithm that uses only a finite number of trajectories (as opposed to all optimal trajectories). Furthermore, the same results allow us to reduce drastically the number of considered trajectories as well as the number and complexity of the operations needed to generate the trajectories and to translate them into the feedback control space. The result is that we obtain an implementable algorithm, at least for moderate size networks. In what follows we first intuitively explain the possible reductions and then present the algorithm formally.

The first simplification is obtained by observing that from [2, Thm. 4] it is enough to consider piecewise linear trajectories with nonpositive slopes (forward in time) and no breaks between junction points. These can be represented as trajectories with the property that various sets of state variables leave the boundary $x_i = 0$ (backwards in time), never return to it and once a certain set of states leaves the boundary, the optimal control at the leave-the-boundary time remains optimal until $t = -\infty$ unless other states leave the boundary subsequently (backwards in time). Whenever a certain set of state variables is assigned to leave the boundary, the

corresponding set of costates that allows this to happen optimally must be found (and is unique according to [2, Thm. 4a]) and all solutions of the corresponding problem (7) must be obtained. The set of all these solutions will determine the corresponding control feedback region (see [2, Example 1] for illustration). Here, however, we obtain a number of further simplifications: first, since (7) is a linear program, it is enough to find only its basic optimal solutions and obtain their convex hull, since all other optimal solutions are convex combinations of the basic ones; second, Theorem 1 above says that the solutions of problems (7) with $\lambda_i = 1, \forall x_i \in I$, are sufficient and there is no need to solve (7) for arbitrary λ's; third, Theorem 2 a) above further reduces the number of problems of type (7) that need to be considered to those corresponding to sets I for which there is no set I' such that $I \subset I'$ and $\max(I) = \max(I')$; fourth, the fact that (7) with $\lambda_i = \dot{1}$ is a maximal flow problem allows us to use maximal flow algorithms, that are in general much more efficient than general linear programs.

Next, observe that [2, Thm. 4b) and e)] imply that the particular instants when sets of variables leave the boundary (backwards in time) are immaterial and all trajectories with identical sequences of states leaving the boundary construct the same feedback control region (see also [2, Example 1]). Consequently, the leave-the-boundary times can be chosen arbitrarily, providing a further simplification of the algorithm. In addition, we shall show in Operation 3 of the Algorithm that Theorems 2c and 3 above provide a way for identifying certain sequences of states leaving the boundary that are redundant in the sense that they generate feedback control regions that can be built using other sequences. Clearly, the redundant sequences need not be considered,

providing a further saving in the algorithm. Finally, we may mention that Theorem 2b) will allow us to reduce the size of the corresponding networks $N(I)$ on which one needs to work in Operation 2 of the Algorithm.

In what follows we present the algorithm that builds the feedback control regions. Each step of the algorithm is presented, motivated and applied to an example network for illustration. The illustrative network is given in Figure 2.

Operation 1: For each possible subset I of state variables, construct the corresponding network $N(I)$ and solve the corresponding maximal flow problem MFP(I) to obtain a first basic optimal solution and the maximal flow value max(I).

Motivation. As explained before, in order to obtain a complete characterization of the feedback space we have, in principle, to consider all possible trajectories represented by sets of states leaving the boundary (backwards in time). For each set of states leaving the boundary, the corresponding costates should be calculated and the corresponding feedback control region is obtained by finding all solutions to problem (7). In Operation 1, we consider all possible sets of state variables (there are 2^n-1 such sets) and find a first optimal (extremal) solution to problem (7) for each set, as well as the corresponding maximal flow value. Note, in addition, that by Theorem 1 it is enough to find the solutions to problem (7) with $\lambda_i = 1$, $\forall x_i \in I$.

Illustration: The networks corresponding to each subset I for the example network of Figure 2 and their corresponding maximal flow appear in Figure 3.

Operation 2: Identify those sets I of state variables for which there is no set $I' \supset I$ such that $\max(I') = \max(I)$. For these sets I use the method indicated in the Appendix to find all optimal basic solutions of the corresponding MFP(I).

Motivation: In principle, in order to obtain an exhaustive set of trajectories as required by the Constructive Dynamic Programming Concept [2], there is need to generate all basic optimal solutions to all MFP(I) determined in Operation 1. Here, however, we were able to obtain large savings in computation due to two results. First, obtaining all solutions to a linear program is an extremely tedious operation even for small dimensions as seen from the literature [5]-[9]. On the other hand, for the particular type of problems under consideration here we were able to develop a Maximal-Flow-specific procedure (see Appendix) that turns out to be especially efficient for degenerate problems, which is the situation in our case. The second saving is obtained from Theorem 2a), indicating that it is sufficient to obtain all basic solutions only to MFP(I) corresponding to those sets I for which there is no proper overset with the same maximal flow value, thereby reducing the number of problems to solve. Although we have no analytic expression for the amount of savings, all examples that we treated provided a great deal of reduction. Furthermore, observe that Theorem 2b) specifies the minimal cut corresponding to each MFP and that only the subnetwork above the cut needs to be considered when finding all solutions, since pivoting under the cut will not lead to new solutions for y_i, $x_i \in I$

Illustration: In the example of Figure 2, we need to consider only the problems corresponding to the networks (a), (b) and (e) of Figure 3. The different

solutions are given in Table 1. Note also that by Theorem 1a) the basic optimal solutions of $MFP\{x_1,x_3\}$ and of $MFP\{x_2,x_3\}$ are also basic optimal solutions of $MFP\{x_1,x_2,x_3\}$ and those of $MFP\{x_1\}$ and $MFP\{x_2\}$ are also basic optimal solutions of $MFP\{x_1,x_2\}$.

Before introducing Operation 3 we may indicate that our goal is to use representative trajectories in the state space in order to construct feedback control regions that cover the entire state space. In principle, in order to ensure complete coverage, one needs to list all possible types of trajectories characterized by sequences of sets of states leaving the boundary. The number $Q(n)$ of possible trajectories for a network with n states ($n+1$ nodes) is given by [10]

$$Q(n) = \sum_{r=1}^{n} \sum_{i=0}^{r} (-1)^i \binom{r}{i} (r-i)^n$$

which turns out to grow as n^n. The computational complexity of considering these sequences (in Operation 3) in order to construct feedback control regions is one of the factors that limits the size of the network that can be handled by the algorithm with reasonable computation effort. However, one of the important results that we have been able to obtain by using the Maximal Flow approach is to develop a criterion in order to discover redundant sequences in the sense that the feedback control regions generated by the corresponding trajectories are in fact contained in such regions generated by another trajectory. The above criterion allows us to check redundancy before proceeding to Operation 4 and since a large number of trajectories usually turn out to be redundant, this considerably reduces the computational requirements of the Algorithm.

Operation 3: List all possible types of trajectories in the state space by writing down all possible sequences of states leaving the boundary and delete redundant sequences by using the following criterion: Consider every pair of sets of states I and I' satisfying $I \subset I'$ and $\max(I) = \max(I')$. every trajectory where at least one state $x_j \in I'/I$ leaves the boundary (backwards in time) after all states of I have left the boundary is redundant and can be deleted.

Illustration: For the network of Figure 2 the list of all possible trajectories appears in Table 2 and all trajectories except 9), 10) and 13) are redundant. For example, sequences 12) and 8) are redundant with $I = \{x_2, x_3\}$ and $I' = \{x_1, x_2, x_3\}$ (the notation is as in Operation 3), sequences 1), 2), 3) are redundant with $I = \{x_1\}$, $I' = \{x_1, x_2\}$ and similarly for the other sequences.

Motivation: To justify the redundancy criterion, consider the trajectory $I_1, I_2, \ldots, I_m, \ldots, I_n, \ldots$, where I_i denotes the set of states that are away from the boundary $x = 0$ during segment i (backwards in time). Since we consider only trajectories with states that do not return to the boundary, we have $I_1 \subset I_2 \subset I_3 \ldots$. Now assume that the above trajectory satisfies the criterion in Operation 3 and with the notations of Operation 3, let m, n be the smallest indices satisfying $I \subseteq I_m$ and $x_j \in I_n$ respectively. By the assumption in Operation 3 we have $m > n$. We show that the above trajectory is redundant by producing another trajectory

$$I_1, I_2, \ldots, I_m, \ldots, \{I_{n-1} \cup x_j\}, I_n, \ldots$$

such that all feedback control regions constructed by the first trajectory are also constructed by the second (note that the only difference between the trajectories is the segment when x_j leaves the boundary).

Observe first that, again with the notations of Operation 3, we have

$$\max(I \cup x_j) = \max(I' \cup x_j) = \max(I') = \max(I)$$

where the equalities follow from Theorem 2c), the fact $x_j \in I'$ and $\max(I') = \max(I)$. This implies, again by Theorem 2c), that

$$\max(I_{n-1} \cup x_j) = \max(I \cup x_j \cup (I_{n-1}/I)) = \max(I \cup (I_{n-1}/I)) = \max(I_{n-1}).$$

Consequently, I_{n-1} and $I_{n-1} \cup x_j$ satisfy the conditions of Theorem 2a), i.e. all extremal solutions of MFP(I_{n-1}) are also extremal solutions of MFP($I_{n-1} \cup x_j$). Moreover, by Theorem 3, the costate functions are identical for both trajectories, implying that the feedback control region resulting from the segment corresponding to I_{n-1}, in the first trajectory results also from the segment corresponding to $I_{n-1} \cup x_j$ in the second.
Since the remaining segments of both trajectories are the same, the resulting feedback control regions are identical and hence the first trajectory is redundant.

Operation 4: For every remaining trajectory from the list of Operation 3, and starting with value zero for all costates carry out the following steps for every segment of the trajectory, starting at the first segment:

(a) Set $\lambda_i' = \lambda_i + 1 \quad \forall x_i \in I$.

(b) From among all solutions $\{\underline{y}^*\}$ of MFP(I) obtained in Operation 2 choose those that maximize the expression

$$\sum_{x_i \in I} \lambda_i y_i^*$$

(c) Every solution obtained in (b) determines a ray in the state space and a corresponding set of controls. Let V be the set of rays determined

by all these solutions and construct the region

$$R_{p-1} = C_o(R_p \cup V)/R_p$$

where R_{p-1} is the feedback control region obtained from the current step, R_p is the feedback control region obtained in the preceding step (i.e. from the previous segment of the trajectory) and C_o denotes convex hull (see [13, p. 175]). In the constructed feedback control region any control from the abovementioned set that does not take the state outside of the region is optimal.

Illustration: In our three dimensional example of Figure 2, in order to illustrate the constructed feedback control regions (that are convex polyhedral cones with vertex at the origin [2, Thm. 3] it is easiest to draw a plane defined by the points with coordinates (0,0,1), (0,1,0), (1,0 0) (see Fig. 4). The remaining trajectories of Operation 3 are 9), 10), 13) of Table 2, and the costates to be considered according to Operation 4a) appear in Table 3. The control that maximize the expression in Operation 4b) are given in Table 4. Each segment of each trajectory determines a feedback control region which is drawn in Figure 4 and the pair appearing in the angular parentheses $< >$ in each region indicates the corresponding trajectory and segment respectively. For example in trajectory 9, the first segment generates the x_3 axis and the second one generates the cone determined by the rays (0,0,1), (0,2/3,2/3), (2/3,0,2/3), excluding the x_3 axis. The corresponding optimal controls appear in Table 4.

Motivation: Since we consider only non-redundant sequences Theorem 3 dictates that every costate is identically 0 when its corresponding state is on the boundary and grows with slope 1 (backwards in time) when its state is away from the boundary. Consequently the costates behave as in Operation 4a). From Theorem 1b) follows that all controls optimizing the expression of Operation 4b) appear in the lists generated by Operation 2 and consequently it is enough to check the value of this expression and select the controls providing the maximal value.

IV. CONCLUSIONS

In this paper we presented an approach leading to the implementation of the algorithm suggested in [1], [2] for finding a feedback solution to the problem of dynamic routing in networks. The linear programs arising from the necessary and sufficient conditions for optimality were transformed into maximal-flow-in-network problems and several properties concerning their solutions were obtained. The properties lead to the development of an implementable algorithm for building the feedback space for the problem of dynamic routing in single destination networks with zero inputs and when the cost functional has no priorities. Two inherent features of the problem limit the maximal size of the networks for which the algorithm is applicable under moderate computational resources. The first is the need of considering

a great amount of piecewise linear trajectories in state space to assure complete covering of the feedback space. The second is the need for finding all solutions to the linear programs defined by the above trajectories to obtain the optimal controls. Taking advantage of the degenerancy of the linear programs, the algorithm severely reduces the number of trajectories that have to be considered. In the example of Section III only 3 out of 13 trajectories had to be considered. In [4] an example of a network with four states is presented in which only 18 out of 75 trajectories have to be considered. Moreover, a simple method for finding all solutions of the linear programs was developed. The method is based on pivoting operations on networks and in [4] several examples are presented when the largest one consists of a five states network that lead to 184 different solutions and representations with a reasonable CPU time (7.41 sec. on an IBM 370/168 computer).

APPENDIX

The Method for Finding all Solutions to the Linear Programs

The method suggested here is based on techniques of pivoting-on-networks. The concepts used here can be found in [12]. Recall that Operation 2 of the algorithm dictates that all optimal solutions have to be found only for those MFP($\mathcal{I}$)'s for which there is no set $\mathcal{I}' \supset \mathcal{I}$ satisfying $\max(\mathcal{I}') = \max(\mathcal{I})$. At maximal flow, the corresponding networks $N(\mathcal{I})$ can be reduced to networks containing only the nodes in X, when $(X,\bar{X})$ is the minimal cut close to the source (see Theorem 2b)). For a reduced network $N_r(\mathcal{I})$ the first (extended) basic optimal solution is given by:

$$y_i = b_i \qquad \forall x_i \in N_r(\mathcal{I})$$

$$u_{ik} = 0 \qquad \forall x_i, x_k \in N_r(\mathcal{I}) \cap N(\mathcal{I}) \ .$$

The links y_i are taken to be basic and the remaining links non-basic. Starting at these first optimal extended basic solution the method picks a non-basic link and looks for a cycle formed by basic links and the considered non-basic link. The existence of such a (unique) cycle is ensured since every extended basic solution corresponds to a spanning tree in the network. A pivot is performed on the cycle reaching a new extremal solution or, if the cycle is degenerate, another representation of the original extremal solution (i.e., the non-basic link is declared basic, and a basic link becomes non-basic while the flow on the links of the cycle does not change). Every new solution or representation is numbered and kept in memory (if it does not exist there already). The above process is performed for all non-basic links corresponding to every solution and/or representation

in memory until no new solution or representation can be reached. This ensures that at the end of the process all optimal solutions had been found. Finally, only one representation per extremal solution is kept.

Note that from two different representations of an extremal solution, different new extremal solutions can be reached, therefore, it is essential to consider all the representations of the extremal solutions in the process. Unfortunately, the number of solutions and representations is not known a priori; only an upper bound is known since every basic solution corresponds to a spanning tree of the network (the number of spanning trees of a network can be calculated), however, not every spanning tree leads to a feasible basic solution. Therefore, in practice this number is meaningless.

REFERENCES

[1] A. Segall, "The Modelling of Adaptive Routing in Data Communication Networks", IEEE Trans. on Comm., COM-25, No.1, pp. 85-95, January 1977.

[2] F.H. Moss, and A. Segall, "An Optimal Control Approach to Dynamic Routing in Networks", IEEE Trans. on Autom. Control. To appear.

[3] L.R. Ford, Jr. and D.R. Fulkerson, "Maximal Flow Through a Network", Canadian J. Math., 8(3), pp. 339-404, 1950.

[4] M. Jodorkovsky, and A. Segall, "A Maximal Flow Approach to Dynamic Routing in Communication Networks", E.E. Pub. No. 358, Technion - Israel Institute of Technology, Haifa, August 1979.

[5] M.L. Balinski, "An Algorithm for Finding All the Vertices of Convex Polyhedral Sets", J. Soc. Indust. Appl. Math., Vol. 9, No.1, March 1961.

[6] N.V. Chernikova, "Algorithm for Finding a General Formula for the Nonnegative Solutions of a System of Linear Inequalities", U.S.S.R. Computational Mathematics and Mathematical Physics, 5, pp. 228-233, 1965.

[7] J. Bloom, "A Program for Chernikova's Algorithm", Electronic Systems Lab., M.I.T., March 1976.

[8] T.H. Mattheiss, "An Algorithm for Determining Irrelevant Constraints and all the Vertices in Systems of Linear Inequalities", Operation Research, Vol. 21, No.1, 247, Jan.-Feb. 1973.

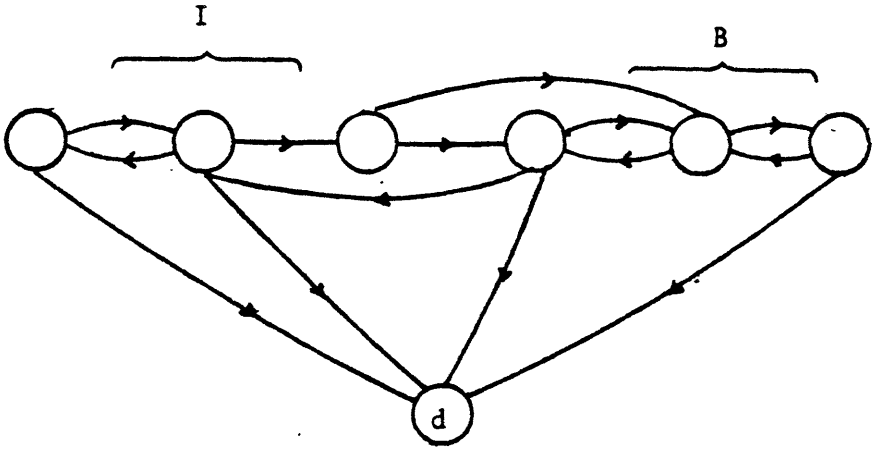
[9] D.C. Rubin, "Vertex Generation and Cardinality Linear Programs", Operations Research, 23, pp. 555-564, 1975.

[10] S. Even, Algorithmic Combinatorics, the MacMillan Company, N.Y., 1973.

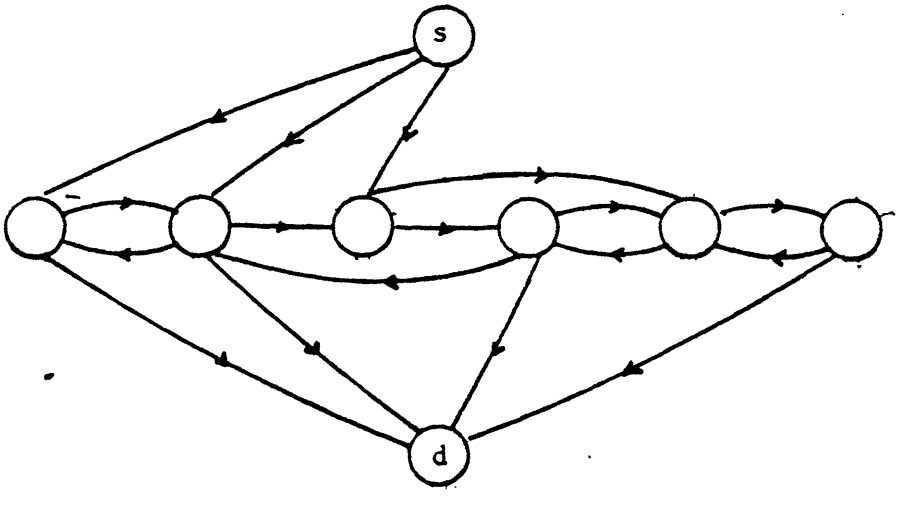
[11] P. Sarachik, "Clearing of Congested Multi-Destination Networks", Proc. Conf. on Research Directions in Computer Control of Urban Traffic Systems, pp. 231-250, 1979.

[12] M.S. Bazaraa, and J. Jarvis, Linear Programming and Network Flows, John Wiley and Sons, Inc., 1977

[13] F.H. Moss, The Application of Optimal Control Theory to Dynamic Routing in Data-Communication Networks, Report ESL-R-721, M.I.T., February 1977.



(a)



(b)

Figure 1 - (a) Original network
(b) Network corresponding to set I

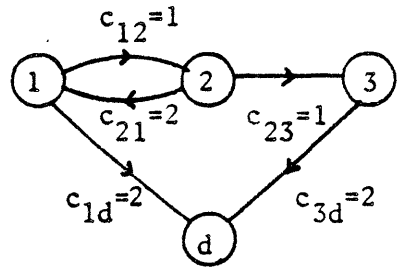
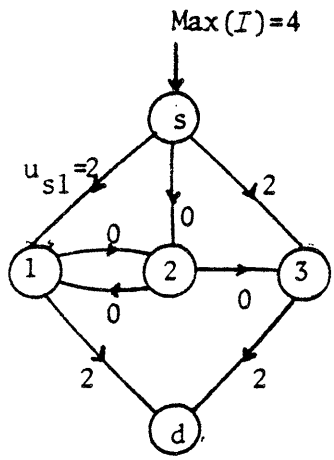
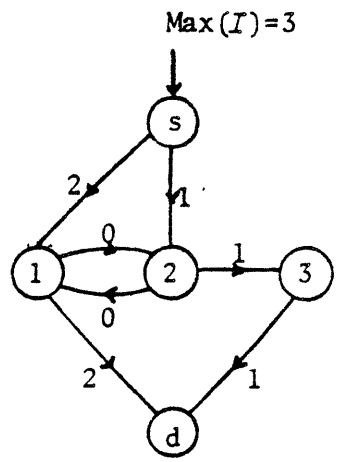


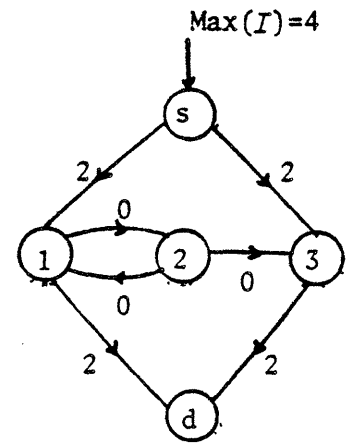
Figure 2 - Example network



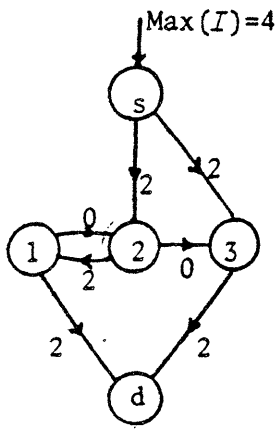
(a) $I = \{x_1, x_2, x_3\}$



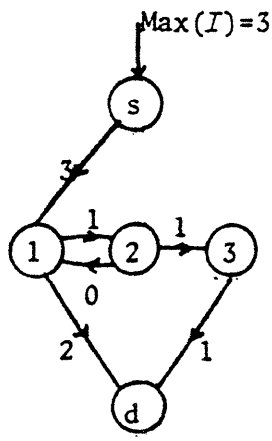
(b) $I = \{x_1, x_2\}$



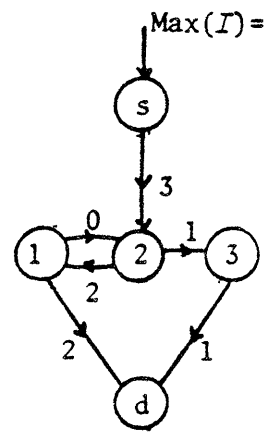
(c) $I = \{x_1, x_3\}$



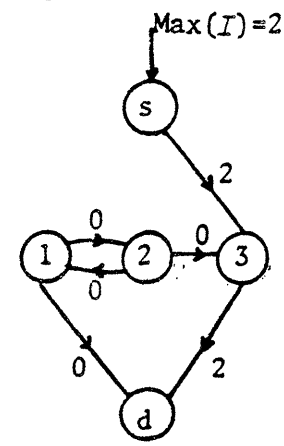
(a) $I = \{x_2, x_3\}$



(e) $I = \{x_1\}$



(f) $I = \{x_2\}$



(g) $I = \{x_3\}$

Figure 3 - Maximal Flow of $N(I)$

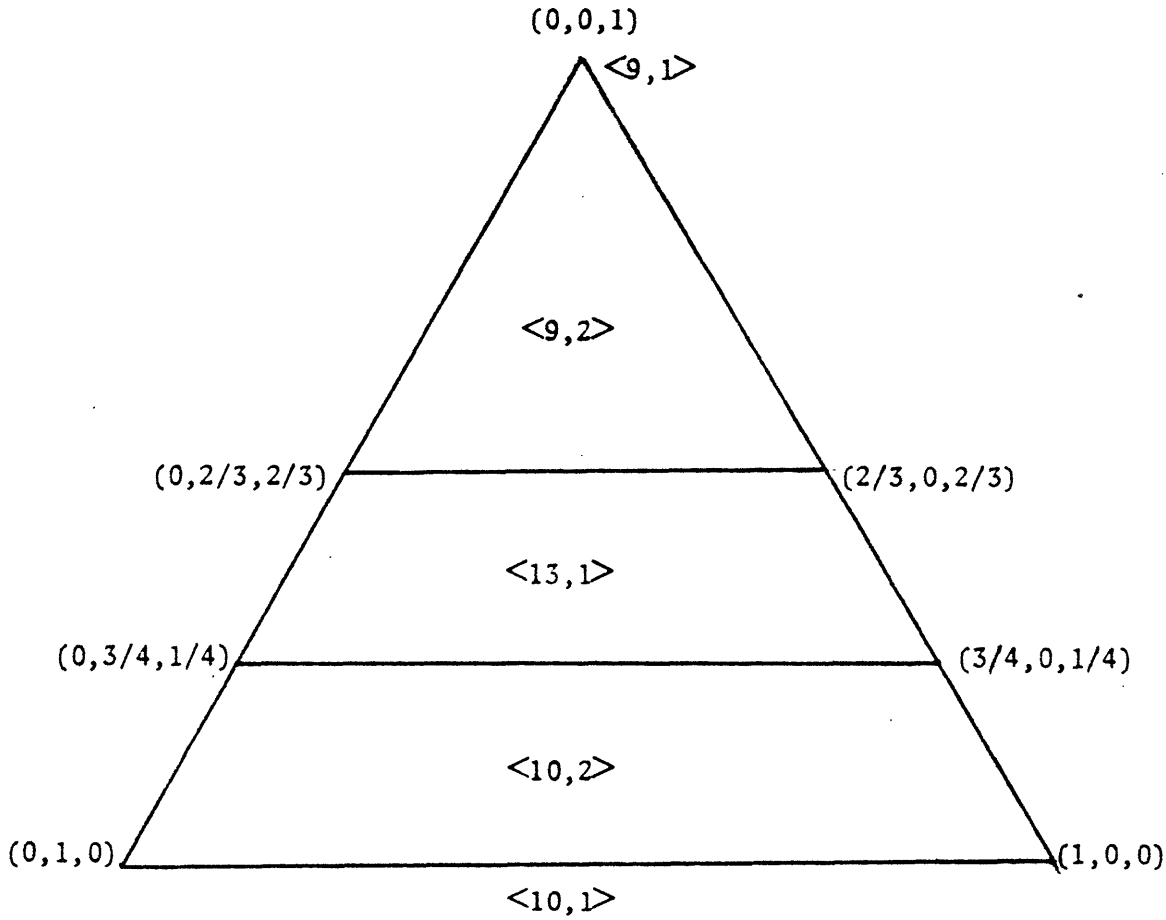
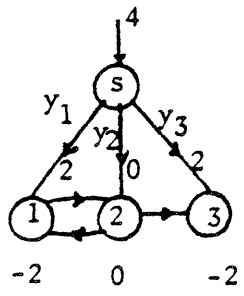


Figure 4 - Projection of Feedback Control Regions

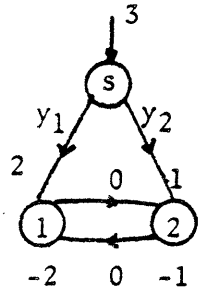
(a)



Different solutions and Representations: 18

	y_1	y_2	y_3	u_{12}	u_{21}	u_{23}	u_{1d}	u_{3d}
i)	2	0	2	0	0	0	2	2
ii)	0	2	2	0	2	0	2	2
iii)	3	0	1	1	0	1	2	2
iv)	0	3	1	0	2	1	2	2
v)	2	0	2	1	1	0	2	2
vi)	2	1	1	0	0	1	2	2
vii)	1	1	2	1	2	0	2	2
viii)	1	2	1	1	2	1	2	2

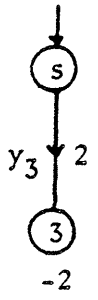
(b)



Different solutions and Representations: 8

	y_1	y_2	u_{12}	u_{21}	u_{23}	u_{1d}	u_{3d}
i)	2	1	0	0	1	2	1
ii)	3	0	1	0	1	2	1
iii)	0	3	0	2	1	2	1
iv)	1	2	1	2	1	2	1

(c)



	y_3	u_{12}	u_{21}	u_{23}	u_{1d}	u_{3d}
i)	2	0	0	0	0	2

Table 1 - Subnetworks and their different solutions

1)	x_1	x_2	x_3
2)	x_1	x_3	x_2
3)	x_1	x_2,x_3	
4)	x_2	x_1	x_3
5)	x_2	x_3	x_1
6)	x_2	x_1,x_3	
7)	x_3	x_1	x_2
8)	x_3	x_2	x_1
9)	x_3	x_1,x_2	
10)	x_1,x_2	x_3	
11)	x_1,x_3	x_2	
12)	x_2,x_3	x_1	
13)	x_1,x_2,x_3		

Table 2 - List of trajectories of states leaving the boundary

Trajectory (from Table 2)	First segment	Second segment
9	(0,0,1)	(1,1,2)
10	(1,1,0)	(2,2,1)
13	(1,1,1)	-

Table 3 - Costate values $\lambda = (\lambda_1,\lambda_2,\lambda_3)$

Trajectory	First segment	Second segment
9	c) i)	a) i), ii), v), vii)
10	b) i), ii), iii), iv)	a) iii), iv), vi), viii)
13	a) i) - viii)	-

Table 4 - Controls maximizing expression appearing in Operation 4b) (entries correspond to controls determined in Table 1)