October 1983 LIDS-P-1336

SECOND DERIVATIVE ALGORITHMS FOR MINIMUM DELAY DISTRIBUTED ROUTING IN NETWORKS†

by

Dimitri P. Bertsekas*
Eli M. Gafni**
Robert G. Gallager***

ABSTRACT

We propose a class of algorithms for finding an optimal quasistatic routing in a communication network. The algorithms are based on Gallager's method [1] and provide methods for iteratively updating the routing table entries of each node in a manner that guarantees convergence to a minimum delay routing. Their main feature is that they utilize second derivatives of the objective function and may be viewed as approximations to a constrained version of Newton's method. The use of second derivatives results in improved speed of convergence and automatic stepsize scaling with respect to level of traffic input. These advantages are of crucial importance for the practical implementation of the algorithm using distributed computation in an environment where input traffic statistics gradually change.

---

†This research was conducted at the M.I.T. Laboratory for Information and Decision Systems with partial support provided by the Defense Advanced Research Projects Agency Grant No. ONR-N00014-75-C-1183 and the National Science Foundation under Grant NSF/ECS 79-19880.

*Room No. 35-210, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, Mass. 02139.

**Computer Science Department, University of California, Los Angeles, California.

***Room No. 35-206, same as*.

1. Introduction

We consider the problem of optimal routing of messages in a communication network so as to minimize average delay per message. Primarily we have in mind a situation where the statistics of external traffic inputs change slowly with time as described in the paper by Gallager [1]. While algorithms of the type to be described can also be used for centralized computation, we place primary emphasis on algorithms that are well suited for real-time distributed implementation. Thus it may be desirable to modify the algorithm of Section 3 to include a line search procedure before using it for centralized computation.

Two critical requirements for the success of a distributed routing algorithm are speed of convergence and relative insensitivity of performance to variations in the statistics of external traffic inputs. Unfortunately the algorithm of [1] is not entirely satisfactory in these respects. In particular it is impossible in this algorithm to select a stepsize that will guarantee convergence and good rate of convergence for a broad range of external traffic inputs. The work described in this paper was motivated primarily by this consideration.

A standard approach for improving the rate of convergence and facilitating stepsize selection in optimization algorithms is to scale the descent direction using second derivatives of the objective function as for example in Newton's method. This is also the approach taken here. On the other hand the straightforward use of Newton's method is inappropriate for our problem both because of large dimensionality and the need for algorithmic simplicity in view of our envisioned decentralized real time loop free implementation. We have thus introduced various approximations to Newton's method which exploit the network structure of the problem, simplify the computations, and facilitate distributed implementation.

In Section 2 we formulate the minimum delay routing problem as a multicommodity flow problem and describe a broad class of algorithms to solve the problem. This class is patterned after a gradient projection method for nonlinear programming [2], [3] as explained in [4], but differs substantively from this method in that at each iteration the routing pattern obtained is loopfree. An interesting mathematical complication arising from this restriction is that similarly as in [1] the value of the objective function need not decrease strictly at each iteration. Gallager's original algorithm is recovered as a special case within our class except for a variation in the definition of a blocked node [compare with equation (15) of [1]]. This variation is essential in order to maintain loopfreedom during operation of our algorithms and, despite its seemingly minor nature, it has necessitated major differences in the proof of convergence from the corresponding proof of [1].

Section 3 describes in more detail a particular algorithm from the class of Section 2. This algorithm employs second derivatives in a manner which approximates a constrained version of Newton's method [3] and is well suited for distributed computation.

The algorithm of Section 3 seems to work well for most quasistatic routing problems likely to appear in practice as extensive computational experience has shown [5]. However there are situations where the unity stepsize employed by this algorithm may be inappropriate. In Section 4 we present another distributed algorithm which automatically corrects this potential difficulty whenever it arises at the expense of additional computation per iteration. This algorithm also employs second derivatives, and is based on minimizing at each iteration a suitable upper bound to a quadratic approximation of the objective function.

Both algorithms of Sections 3 and 4 have been tested extensively and computational results have been documented in [5] and [6]. These results substantiate the assertions made here regarding the practical properties of the algorithms. There are also other related second derivative algorithms [7], [8] that operate in the space of path flows and exhibit similar behavior as the ones of this paper while other more complex algorithms [12], [13] are based on conjugate gradient approximations to Newton's method and exhibit a faster rate of convergence. A survey is given in [15], and a computer code implementation can be found in [16]. These algorithms are well suited for centralized computation and virtual circuit networks but, in contrast with the ones of the present paper, require global information at each node regarding the network topology and the total flow on each link. Depending on the mode of operation of the network this information may not be available. The algorithms of [7], [8] and [12], [13] may also require more computer memory when implemented for centralized computation.

We finally mention that while we have restricted attention to the problem of routing, the algorithms of this paper can be applied to other problems of interest in communication networks. For example, problems of optimal adaptive flow control or combined routing and flow control have been formulated in [9], [10] as nonlinear multicommodity flow problems of the type considered here, and the algorithms of this paper are suitable for their solution

Due to space limitations the proofs of most of the results of the paper have been omitted. The can be found in two reports [11], [14].

2. A Class of Routing Algorithms

Consider a network consisting of N nodes denoted by 1,2,...,N and L directed links. The set of links is denoted by L. We denote by (i,ℓ) the link from node i to node ℓ and assume that the network is connected in the sense that for any two nodes m,n there is a directed path from m to n. The flow on each link (i,ℓ) for any destination j is denoted by $f_{i\ell}(j)$. The total flow on each link (i,ℓ) is denoted by $F_{i\ell}$, i.e.

$$F_{i\ell} = \sum_{j=1}^{N} f_{i\ell}(j).$$

The vector of all flows $f_{i\ell}(j)$, $(i,\ell)\in L$, $j = 1,\ldots,N$ is denoted by f.

We are interested in numerical solution of the following multicommodity network flow problem:

$$\text{minimize} \quad \sum_{(i,\ell)\in L} D_{i\ell}(F_{i\ell}) \qquad \text{(MFP)}$$

$$\text{subject to} \quad \sum_{\ell\in O(i)} f_{i\ell}(j) - \sum_{m\in I(i)} f_{mi}(j) = r_i(j), \quad \forall\, i = 1,\ldots,N,\ i \neq j$$

$$f_{i\ell}(j) \geq 0, \quad \forall\,(i,\ell)\in L,\ i = 1,\ldots,N,\ j = 1,\ldots,N$$

$$f_{j\ell}(j) = 0, \quad \forall\,(j,\ell)\in L,\ j = 1,\ldots,N,$$

where, for $i \neq j$, $r_i(j)$ is a known traffic input at node i destined for j, and $O(i)$ and $I(i)$ are the sets of nodes ℓ for which $(i,\ell)\in L$ and $(\ell,i)\in L$ respectively.

The standing assumptions throughout the paper are:

a) $r_i(j) \geq 0, \quad i,j = 1,\ldots,N, \; i \neq j$

b) Each function $D_{i\ell}$ is defined on an interval $[0,C_{i\ell})$ where $C_{i\ell}$ is either a positive number (the link capacity) or $+\infty$; $D_{i\ell}$ is convex, continuous, and has strictly positive and continuous first and second derivatives on $[0,C_{i\ell})$, where the derivatives at 0 are defined by taking the limit from the right. Furthermore $D_{i\ell}(F_{i\ell}) \to \infty$ as $F_{i\ell} \to C_{i\ell}$.

c) (MFP) has at least one feasible solution, f, satisfying $F_{i\ell} < C_{i\ell}$ for all $(i,\ell) \in L$.

For notational convenience in describing various algorithms in what follows, we will suppress the destination index and concentrate on a single destination chosen for concreteness to be node N. Our definitions, optimality conditions, and algorithms are essentially identical for each destination, so this notational simplification should not become a source of confusion. In the case where there are multiple destinations it is possible to implement our algorithms in at least two different ways. Either iterate simultaneously for all destinations (the "all-at-once" version), or iterate sequentially one destination at a time in a cyclic manner with intermediate readjustment of link flows in the spirit of the Gauss-Seidel method (the "one-at-a-time" version). The remainder of our notation follows in large measure the one employed in [1]. In addition all vectors will be considered to be column vectors, transposition will be denoted by a superscript T, and the standard Euclidean norm of a vector will be denoted by $|\cdot|$, i.e. $x^T x = |x|^2$ for any vector x. Vector inequalities are meant to be componentwise, i.e. for $x = (x_1,\ldots,x_n)$ we write $x \geq 0$ if $x_i \geq 0$ for all $i = 1,\ldots,n$.

Let t_i be the total incoming traffic at node i

$$t_i = r_i + \sum_{m \in I(i)} f_{mi}, \quad i = 1,\ldots,N-1, \tag{1}$$

and for $t_i \neq 0$ let $\phi_{i\ell}$ be the fraction of t_i that travels on link (i,ℓ)

$$\phi_{i\ell} = \frac{f_{i\ell}}{t_i}, \quad i = 1,\ldots,N-1 \ (i,\ell)\varepsilon L.$$

Then it is possible to reformulate the problem in terms of the variables $\phi_{i\ell}$ as follows [1].

For each node $i \neq N$ we fix an order of the outgoing links (i,ℓ), $\ell\varepsilon O(i)$. We identify with each collection $\{\phi_{i\ell} | (i,\ell)\varepsilon L, \ i = 1,\ldots,N-1\}$ a column vector $\phi = (\phi_1^T,\phi_2^T,\ldots,\phi_{N-1}^T)^T$, where ϕ_i is the column vector with coordinates $\phi_{i\ell}, \ell\varepsilon O(i)$. Let

$$\bar{\Phi} = \{\phi | \phi_{i\ell} \geq 0, \ \sum_{\ell\varepsilon O(i)} \phi_{i\ell} = 1, \ (i,\ell)\varepsilon L, \ i = 1,\ldots,N-1\} \tag{2}$$

and let Φ be the subset of $\bar{\Phi}$ consisting of all ϕ for which there exists a directed path $(i,\ell),\ldots,(m,N)$ from every node $i = 1,\ldots,N-1$ to the destination N along which $\phi_{i\ell} > 0,\ldots,\phi_{mN} > 0$. Clearly Φ and $\bar{\Phi}$ are convex sets, and the closure of Φ is $\bar{\Phi}$. It is shown in [1] that for every $\phi\varepsilon\Phi$ and $r = (r_1,r_2,\ldots,r_{N-1})$ with $r_i \geq 0$, $i = 1,\ldots,N-1$ there exist unique vectors $t(\phi,r) = (t_1(\phi,r),\ldots,t_{N-1}(\phi,r))$ and $f(\phi,r)$ with coordinates $f_{i\ell}(\phi,r)$, $(i,\ell)\varepsilon L$, $i \neq N$ satisfying

$$t(\phi,r) \geq 0, \ f(\phi,r) \geq 0$$

$$t_i(\phi,r) = r_i + \sum_{\substack{m\varepsilon I(i) \\ m\neq N}} f_{mi}(\phi,r), \quad i = 1,2,\ldots,N-1$$

$$\sum_{\ell\varepsilon O(i)} f_{i\ell}(\phi,r) - \sum_{\substack{m\varepsilon I(i) \\ m\neq N}} f_{mi}(\phi,r) = r_i, \quad i = 1,\ldots,N-1$$

$$f_{i\ell}(\phi,r) = t_i(\phi,r)\phi_{i\ell}, \quad i = 1,\ldots,N-1, \ (i,\ell)\varepsilon L.$$

Furthermore the functions $t(\phi,r)$, $f(\phi,r)$ are twice continuously differentiable in the relative interior of their domain of definition $\Phi x\{r|r \geq 0\}$. The derivatives at the relative boundary can also be defined by taking the limit through the relative interior. Furthermore for every $r \geq 0$ and every f which is feasible for (MFP) there exists a $\phi\varepsilon\Phi$ such that $f = f(\phi,r)$.

It follows from the above discussion that the problem can be written in terms of the variables $\phi_{i\ell}$ as

$$\text{minimize} \quad D(\phi,r) \triangleq \sum_{(i,\ell)\varepsilon L} D_{i\ell}[f_{i\ell}(\phi,r)] \tag{3}$$

$$\text{subject to } \phi\varepsilon\Phi,$$

where we write $D(\phi,r) = \infty$ if $f_{i\ell}(\phi,r) \geq C_{i\ell}$ for some $(i,\ell)\varepsilon L$.

Similarly as in [1], our algorithms generate sequences of loopfree routing variables ϕ and this allows efficient computation of various derivatives of D. Thus for a given $\phi\varepsilon\Phi$ we say that node k is <u>downstream</u> from node i if there is a directed path from i to k, and for every link (ℓ,m) on the path we have $\phi_{\ell m} > 0$. We say that node i is <u>upstream</u> from node k if k is downstream from i. We say that ϕ is <u>loopfree</u> if there is no pair of nodes i,k such that i is both upstream and downstream from k. For any $\phi\varepsilon\Phi$ and $r \geq 0$ for which

$D(\phi,r) < \infty$ the partial derivatives $\frac{\partial D(\phi,r)}{\partial \phi_{i\ell}}$ can be computed using the following equations [1]

$$\frac{\partial D}{\partial \phi_{i\ell}} = t_i(D'_{i\ell} + \frac{\partial D}{\partial r_\ell}), \quad (i,\ell)\varepsilon L, \; i = 1,\ldots,N-1 \tag{4}$$

$$\frac{\partial D}{\partial r_i} = \sum_{\ell\varepsilon O(i)} \phi_{i\ell}(D'_{i\ell} + \frac{\partial D}{\partial r_\ell}), \quad i = 1,\ldots,N-1 \tag{5}$$

$$\frac{\partial D}{\partial r_N} = 0$$

where $D'_{i\ell}$ denotes the first derivative of $D_{i\ell}$ with respect to $f_{i\ell}$. The equations above uniquely determine $\frac{\partial D}{\partial \phi_{i\ell}}$ and $\frac{\partial D}{\partial r_i}$ and their computation is particularly simple if ϕ is loopfree. In a distributed setting each node i computes $\frac{\partial D}{\partial \phi_{i\ell}}$ and $\frac{\partial D}{\partial r_i}$ via (4),(5) after receiving the value of $\frac{\partial D}{\partial r_\ell}$ from all its immediate downstream neighbors. Because ϕ is loopfree the computation can be organized in a deadlock-free manner starting from the destination node N and proceeding upstream [1].

A necessary condition for optimality is given by (see [1])

$$\frac{\partial D}{\partial \phi_{i\ell}} = \min_{m\varepsilon O(i)} \frac{\partial D}{\partial \phi_{im}} \qquad \text{if } \phi_{i\ell} > 0$$

$$\frac{\partial D}{\partial \phi_{i\ell}} \geq \min_{m\varepsilon O(i)} \frac{\partial D}{\partial \phi_{im}} \qquad \text{if } \phi_{i\ell} = 0,$$

where all derivatives are evaluated at the optimum. These equations are automatically satisfied for i such that $t_i = 0$, and for $t_i > 0$, the conditions are equivalent, through use of (4) and (5), to

$$\frac{\partial D}{\partial r_i} = \min_{m \in O(i)} \delta_{im} \tag{6}$$

where δ_{im} is defined by

$$\delta_{im} = D'_{im} + \frac{\partial D}{\partial r_m}, \quad \forall\, m \in O(i) \tag{7}$$

In fact if (6) holds for all i (whether $t_i = 0$ or $t_i > 0$) then it is sufficient to guarantee optimality (see [1], Theorem 3).

We consider the class of algorithms

$$\phi_i^{k+1} = \phi_i^k + \Delta\phi_i^k, \quad i = 1,\ldots,N-1 \tag{8}$$

where, for each i, the vector $\Delta\phi_i^k$ with components $\Delta\phi_{i\ell}^k$, $\ell \in O(i)$ is any solution of the problem

$$\text{minimize } \delta_i^T \Delta\phi_i + \frac{t_i}{2\alpha} \Delta\phi_i^T M_i^k \Delta\phi_i \tag{9}$$

$$\text{subject to } \phi_i^k + \Delta\phi_i \geq 0, \quad \sum_{\ell} \Delta\phi_{i\ell} = 0,$$

$$\Delta\phi_{i\ell} = 0, \quad \forall\, \ell \in B(i;\phi^k).$$

The scalar α is a positive parameter and δ_i is the vector with components $\{\delta_{im}\}$ given by (7).

All derivatives in (8) and (9) are evaluated at ϕ^k and $f(\phi^k,r)$. For each i for which $t_i(\phi^k,r) > 0$, the matrix M_i^k is some symmetric matrix which is positive definite on the subspace $\{v_i \mid \sum_{\ell\varepsilon O(i)} v_i = 0\}$, i.e.

$$v_i^T M_i^k v_i > 0, \quad \forall\, v_i \neq 0, \quad \sum_{\ell\varepsilon O(i)} v_{i\ell} = 0.$$

This condition guarantees that the solution to problem (9) exists and is unique. For nodes i for which $t_i(\phi^k,r) = 0$ the definition of M_i^k is immaterial. The set of indices $B(i;\phi^k)$ is specified in the following definition:

Definition: For any $\phi\varepsilon\Phi$ and i=1,...,N-1 the set $B(i;\phi)$, referred to as the set of blocked nodes for ϕ at i, is the set of all $\ell\varepsilon O(i)$ such that $\phi_{i\ell} = 0$, and either $\frac{\partial D(\phi,r)}{\partial r_i} \leq \frac{\partial D(\phi,r)}{\partial r_\ell}$, or there exists a link (m,n) such that $m=\ell$ or m is downstream of ℓ and we have $\phi_{mn} > 0$, $\frac{\partial D(\phi,r)}{\partial r_m} \geq \frac{\partial D(\phi,r)}{\partial r_n}$. (Such a link will be referred to as an improper link).

We refer to [1] for a description of the method for generating the sets $B(i;\phi^k)$ in a manner suitable for distributed computation. Our definition of $B(i;\phi^k)$ differs from the one of [1] primarily in that a special device that facilitated the proof of convergence given in [1] is not employed (compare with equ. (15) of [1]).

The following proposition shows some of the properties of the algorithm. Its proof can be found in [11], [14].

Proposition 1: a) If ϕ^k is loopfree then ϕ^{k+1} is loopfree.

b) If ϕ^k is loopfree and $\Delta\phi^k = 0$ solves problem (9) then ϕ^k is optimal.

c) If ϕ^k is optimal then ϕ^{k+1} is also optimal.

d) If $\Delta\phi_i^k \neq 0$ for some i for which $t_i(\phi^k,r) > 0$ then there exists a positive scalar η_k such that

$$D(\phi^k + \eta\Delta\phi^k,r) < D(\phi^k,r), \quad \forall \eta\in(0,\eta_k]. \tag{10}$$

The following proposition is the main convergence result regarding the class of algorithms (8),(9). Its proof will not be given in view of its complexity and length. It may be found in [11]. The proposition applies to the multiple destination case in the "all-at-once" as well as the "one-at-a-time" version.

Proposition 2: Let the initial routing ϕ^o be loopfree and satisfy $D(\phi^o,r) \leq D_o$ where D_o is some scalar. Assume also that there exist two positive scalars λ,Λ such that the sequences of matrices $\{M_i^k\}$ satisfy the following two conditions:

a) The absolute value of each element of M_i^k is bounded above by Λ.

b) There holds

$$\lambda|v_i|^2 \leq v_i^T M_i^k v_i$$

for all v_i in the subspace $\{v_i \mid \sum_{\ell\notin B(i;\phi^k)} v_{i\ell} = 0\}$.

Then there exists a positive scalar $\bar{\alpha}$ (depending on D_o, λ, and Λ) such that for all

$\alpha\varepsilon(0,\bar{\alpha}]$ and k=0,1,... the sequence $\{\phi^k\}$ generated by algorithm (8),(9) satisfies

$$D(\phi^{k+1},r) \leq D(\phi^k,r) , \quad \lim_{k\to\infty} D(\phi^k,r) = \min_{\phi\varepsilon\Phi} D(\phi,r).$$

Furthermore every limit point of $\{\phi^k\}$ is an optimal solution of problem (3).

Another interesting result which will not be given here but can be found in [11] states that, after a finite number of iterations, improper links do not appear further in the algorithm so that for rate of convergence analysis purposes the potential presence of improper links can be ignored. Based on this fact it can be shown under a mild assumption that for the single destination case the rate of convergence of the algorithm is linear [11].

The class of algorithms (8),(9) is quite broad since different choices of matrices M_i^k yield different algorithms. A specific choice of M_i^k yields Gallager's algorithm [1] [except for the difference in the definition of $B(i;\phi^k)$ mentioned earlier]. This choice is the one for which M_i^k is diagonal with all elements along the diagonal being unity except the $(\bar{\ell},\bar{\ell})$th element which is zero where $\bar{\ell}$ is a node for which

$$\delta_{i\bar{\ell}} = \min_{\ell\varepsilon O(i)} \delta_{i\ell}.$$

We leave the verification of this fact to the reader. In the next section we describe a specific algorithm involving a choice of M_i^k based on second derivatives of $D_{i\ell}$. The convergence result of Proposition 2 is applicable to this algorithm.

3. An Algorithm Based on Second Derivatives

A drawback of the algorithm of [1] is that a proper range of the stepsize parameter α is hard to determine. In order for the algorithm to have guaranteed convergence for a broad range of inputs r, one must take α quite small but this will lead to a poor speed of convergence for most of these inputs. It appears that in this respect a better choice of the matrices M_i^k can be based on second derivatives. This tends to make the algorithm to a large extent scale free, and for most problems likely to appear in practice, a choice of the stepsize α near unity results in both convergence and reasonably good speed of convergence for a broad range of inputs r. This is supported by extensive computational experience some of which is reported in [5] and [6].

We use the notation

$$D''_{i\ell} = \frac{\partial^2 D_{i\ell}}{[\partial f_{i\ell}]^2}$$

We have already assumed that $D''_{i\ell}$ is positive in the set $[0,C_{i\ell})$. We would like to choose the matrices M_i^k to be diagonal with $t_i^{-2} \frac{\partial^2 D(\phi^k,r)}{[\partial \phi_{i\ell}]^2}$ along the diagonal. This corresponds to an approximation of a constrained version of Newton's method (see [3]), where the off-diagonal terms of the Hessian matrix of D are set to zero. This type of approximated version of Newton's method is often employed in solving large scale unconstrained optimization problems. Unfortunately the second derivatives $\frac{\partial^2 D}{[\partial \phi_{i\ell}]^2}$ are difficult to compute. However, it is possible to compute easily upper and lower bounds to them which, as shown by computational ex-

periments, are sufficiently accurate for practical purposes.

Calculation of Upper and Lower Bounds to Second Derivatives

We compute $\frac{\partial^2 D}{[\partial \phi_{i\ell}]^2}$ evaluated at a loopfree $\phi \in \Phi$, for all links $(i,\ell) \in L$ for which $\ell \notin B(i;\phi)$. We have using (4)

$$\frac{\partial^2 D}{[\partial \phi_{i\ell}]^2} = \frac{\partial}{\partial \phi_{i\ell}} \{t_i (D'_{i\ell} + \frac{\partial D}{\partial r_\ell})\}.$$

Since $\ell \notin B(i;\phi)$ and ϕ is loopfree, the node ℓ is not upstream of i. It follows that $\frac{\partial t_i}{\partial \phi_{i\ell}} = 0$ and $\frac{\partial D'_{i\ell}}{\partial \phi_{i\ell}} = D''_{i\ell} t_i$. Using again the fact that ℓ is not upstream of i we have $\frac{\partial t_i}{\partial r_\ell} = 0$, $\frac{\partial D'_{i\ell}}{\partial r_\ell} = 0$ and it follows that

$$\frac{\partial^2 D}{\partial \phi_{i\ell} \partial r_\ell} = \frac{\partial}{\partial r_\ell} \frac{\partial D}{\partial \phi_{i\ell}} = \frac{\partial}{\partial r_\ell} \left\{ t_i (D'_{i\ell} + \frac{\partial D}{\partial r_\ell}) \right\} = t_i \frac{\partial^2 D}{[\partial r_\ell]^2}.$$

Thus we finally obtain

$$\frac{\partial^2 D}{[\partial \phi_{i\ell}]^2} = t_i^2 (D''_{i\ell} + \frac{\partial^2 D}{[\partial r_\ell]^2}). \tag{11}$$

A little thought shows that the second derivative $\frac{\partial^2 D}{[\partial r_\ell]^2}$ is given by the more general formula

$$\frac{\partial^2 D}{\partial r_\ell \partial r_m} = \sum_{(j,k) \in L} q_{jk}(\ell) q_{jk}(m) D''_{jk}, \quad \forall \ell, m=1,\ldots,N-1 \tag{12}$$

where $q_{jk}(\ell)$ is the portion of a unit of flow originating at ℓ which goes through link (j,k). However calculation of $\frac{\partial^2 D}{[\partial r_\ell]^2}$ using this

formula is complicated, and in fact there seems to be no easy way to compute this second derivative. However upper and lower bounds to it can be easily computed as we now show. By using (5) we obtain

$$\frac{\partial^2 D}{[\partial r_\ell]^2} = \frac{\partial}{\partial r_\ell} \{\sum_m \phi_{\ell m}(D'_{\ell m} + \frac{\partial D}{\partial r_m})\}.$$

Since ϕ is loopfree we have that if $\phi_{\ell m} > 0$ then m is not upstream of ℓ and therefore $\frac{\partial t_\ell}{\partial r_\ell} = 1$ and $\frac{\partial D'_{\ell m}}{\partial r_\ell} = D''_{\ell m}\, \phi_{\ell m}$. A similar reasoning shows that

$$\frac{\partial^2 D}{\partial r_\ell \partial r_m} = \frac{\partial}{\partial r_m} \{\sum_n \phi_{\ell n}(D'_{\ell n} + \frac{\partial D}{\partial r_n})\} = \sum_n \phi_{\ell n} \frac{\partial^2 D}{\partial r_m \partial r_n}$$

Combining the above relations we obtain

$$\frac{\partial^2 D}{[\partial r_\ell]^2} = \sum_m \phi^2_{\ell m} D''_{\ell m} + \sum_m \sum_n \phi_{\ell m}\, \phi_{\ell n} \frac{\partial^2 D}{\partial r_m \partial r_n}. \tag{13}$$

Since $\frac{\partial^2 D}{\partial r_m \partial r_n} \geq 0$, by setting $\frac{\partial^2 D}{\partial r_m \partial r_n}$ to zero for $m \neq n$ we obtain the lower bound

$$\sum_m \phi^2_{\ell m}(D''_{\ell m} + \frac{\partial^2 D}{[\partial r_m]^2}).$$

By applying the Cauchy-Schwartz inequality in conjunction with (12) we also obtain

$$\frac{\partial^2 D}{\partial r_m \partial r_n} \leq \sqrt{\frac{\partial^2 D}{[\partial r_m]^2} \frac{\partial^2 D}{[\partial r_n]^2}}.$$

Using this fact in (13) we obtain the upper bound

$$\sum_m \phi_{\ell m}^2 D''_{\ell m} + \Big(\sum_m \phi_{\ell m} \sqrt{\frac{\partial^2 D}{[\partial r_m]^2}}\Big)^2 .$$

It is now easy to see that we have for all ℓ

$$\underline{R}_\ell \leq \frac{\partial^2 D}{[\partial r_\ell]^2} \leq \overline{R}_\ell$$

where $\underline{R}_\ell$ and $\overline{R}_\ell$ are generated by

$$\underline{R}_\ell = \sum_m \phi_{\ell m}^2 (D''_{\ell m} + \underline{R}_m) \tag{14}$$

$$\overline{R}_\ell = \sum_m \phi_{\ell m}^2 D''_{\ell m} + \Big(\sum_m \phi_{\ell m} \sqrt{\overline{R}_m}\Big)^2 \tag{15}$$

$$\underline{R}_N = \overline{R}_N = 0 \tag{16}$$

The computation is carried out by passing $\underline{R}_\ell$ and $\overline{R}_\ell$ upstream together with $\frac{\partial D}{\partial r_\ell}$ and this is well suited for a distributed algorithm. Upper and lower bounds $\underline{\Phi}_{i\ell}, \overline{\Phi}_{i\ell}$ for $\frac{\partial^2 D}{[\partial \phi_{i\ell}]^2}$, $\ell \in B(i;\phi)$ are obtained simultaneoulsy by means of the equation [cf. (11)]

$$\underline{\Phi}_{i\ell} = t_i^2 (D''_{i\ell} + \underline{R}_\ell) \tag{17}$$

$$\overline{\Phi}_{i\ell} = t_i^2 (D''_{i\ell} + \overline{R}_\ell). \tag{18}$$

It is to be noted that in some situations occuring frequently in practice the upper and lower bounds $\underline{\Phi}_{i\ell}$ and $\overline{\Phi}_{i\ell}$ coincide and are equal to the true

second derivative. This will occur if $\phi_{\ell m}\phi_{\ell n}\ \frac{\partial^2 D}{\partial r_m \partial r_n} = 0$ for $m \neq n$. For example if the routing pattern is as shown in Figure 1 (only links that carry flow are shown) then $\overline{\Phi}_{i\ell} = \underline{\Phi}_{i\ell} = \frac{\partial^2 D}{[\partial \phi_{i\ell}]^2}$ for all $(i,\ell) \in L$, $\ell \notin B(i;\phi)$.

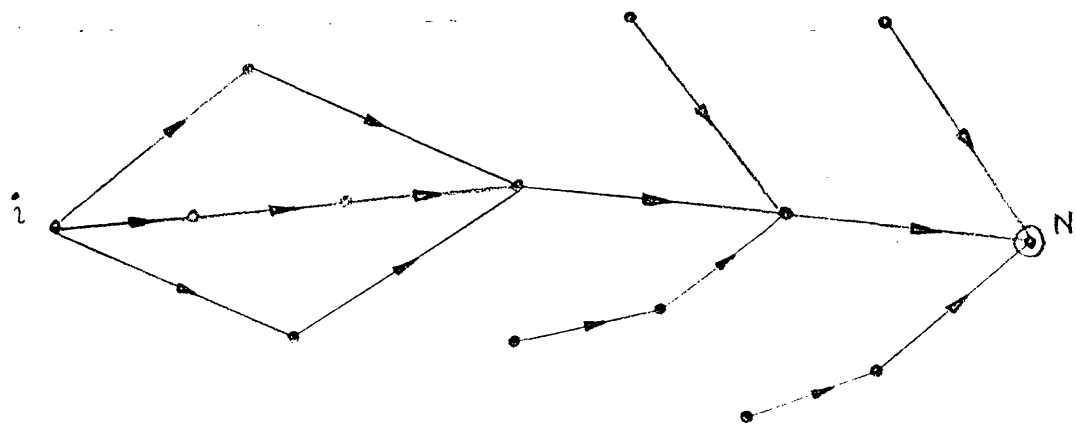

Figure 1

A typical case where $\overline{\Phi}_{i\ell} \neq \underline{\Phi}_{i\ell}$ and the discrepancy affects materially the algorithm to be presented is when flow originating at i splits and joins again twice on its way to N as shown in Figure 2.

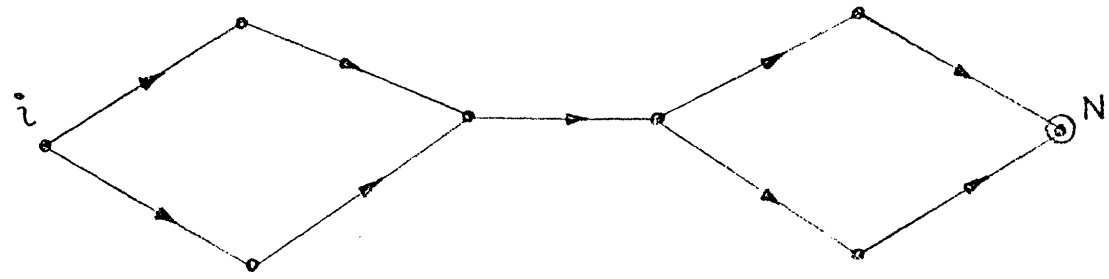

Figure 2

The Algorithm

The following algorithm seems to be a reasonable choice. If $t_i \neq 0$ we take M_i^k in (9) to be the diagonal matrix with $\frac{1}{2t_i}\,\bar{\phi}_{i\ell}$, $\ell\varepsilon 0(i)$ along the diagonal where $\bar{\Phi}_{i\ell}$ is the upper bound computed from (18) and (14)-(16) and α is a positive scalar chosen experimentally. (In most cases $\alpha=1$ is satisfactory.) Convergence of this algorithm can be easily established by verifying that the assumption of Proposition 2 is satisfied. A variation of the method results if we use in place of the upper bound $\bar{\phi}_{i\ell}$ the average of the upper and lower bounds $\frac{\bar{\Phi}_{i\ell} + \underline{\Phi}_{i\ell}}{2}$. This however requires additional computation and communication between modes.

Problem (9) can be written for $t_i \neq 0$ as

$$\text{minimize} \quad \sum_{\ell} \{\delta_{i\ell}\,\Delta\phi_{i\ell} + \frac{\bar{\Phi}_{i\ell}}{2\alpha t_i}\,(\Delta\phi_{i\ell})^2\} \qquad (19)$$

$$\text{subject to} \quad \Delta\phi_{i\ell} \geq -\phi_{i\ell}^k, \sum_{\ell} \Delta\phi_{i\ell} = 0, \ \Delta\phi_{i\ell} = 0 \ \ \forall \ell\varepsilon B(i;\phi^k)$$

and can be solved using a Lagrange multiplier technique. By introducing the expression (18) for $\bar{\Phi}_{i\ell}$ and carrying out the straightforward calculation we can write the corresponding iteration (8) as

$$\phi_{i\ell}^{k+1} = \max\{0, \phi_{i\ell}^{k} - \frac{\alpha(\delta_{i\ell}-\mu_i)}{t_i(D''_{i\ell}+\bar{R}_\ell)}\} \tag{20}$$

where μ_i is a Lagrange multiplier determined from the condition

$$\sum_{\ell \notin B(i;\phi^k)} \max\{0, \phi_{i\ell}^{k} - \frac{\alpha(\delta_{i\ell}-\mu_i)}{t_i(D''_{i\ell}+\bar{R}_\ell)}\} = 1. \tag{21}$$

The equation above is piecewise linear in the single variable μ_i and is nearly trivial computationally. Note from (20) that α plays the role of a stepsize parameter. For $t_i = 0$ the algorithm sets, consistently with problem (9), $\phi_{i\ell}^{k+1} = 1$ for the node $\bar{\ell}$ for which $\delta_{i\bar{\ell}}$ is minimum over all $\delta_{i\ell}$, and sets $\phi_{i\ell}^{k+1} = 0$ for $\ell \neq \bar{\ell}$.

It can be seen that (20) is such that all routing variables $\phi_{i\ell}$ such that $\delta_{i\ell} < \mu_i$ will be increased or stay fixed at unity, while all routing variables $\phi_{i\ell}$ such that $\delta_{i\ell} > \mu_i$ will be decreased or stay fixed at zero. In particular the routing variable with smallest $\delta_{i\ell}$ will either be increased or stay fixed at unity, similarly as in Gallager's algorithm.

4. An Algorithm Based on an Upper Bound to Newton's Method

While the introduction of a diagonal scaling based on second derivatives alleviates substantially the problem of stepsize selection, it is still possible that in some iterations a unity stepsize will not lead to a reduction of the objective function and may even cause divergence of the algorithm of the previous section. This can be corrected by using a smaller stepsize as shown in Proposition 2 but the proper range of stepsize magnitude depends on the network topology and may not be easy to determine. This dependence stems from the replacement of the Hessian matrix of D by a diagonal approximation which in turn facilitates the computation of upper bounds to second derivatives in a distributed manner. Neglecting the off-diagonal terms of the Hessian has two types of effects. First, while operating the algorithm for one destination, we ignore changes which are caused by other destinations. The potential difficulties resulting from this can be alleviated (and for most practical problems eliminated) by operating the algorithm in a "one-at-a-time" version as discussed in Section 2. Second, the effect of neglecting off-diagonal terms can be detrimental in situations such as the one depicted by Figure 3. Here,

$r_1 = r_2 = r_3 = r_4 > 0$, $r_5 = r_6 = 0$ and node 7

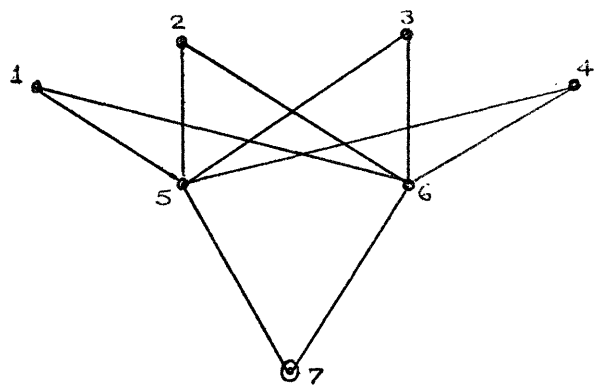

Figure 3

is the only destination. If the algorithm of the previous section is applied to this example with $\alpha=1$, then it can be verified that each of the nodes 1,2,3 and 4 will adjust its routing variables according to what would be Newton's method if all other variables remained unchanged. If we assume symmetric initial conditions and that the first and second derivatives D'_{57}, D''_{57} and D'_{67}, D''_{67} are much larger than the corresponding derivatives of all other links, then the algorithm would lead to a change of flow about four times larger than appropriate. Thus for example a stepsize $\alpha = 1/4$ is appropriate, while $\alpha=1$ can lead to divergence.

The algorithm proposed in this section bypasses these difficulties at the expense of additional computation per iteration. We show that if the initial flow vector is near optimal then the algorithm is guaranteed to reduce the value of the objective function at each iteration and to converge to the optimum with a unity stepsize. The algorithm "upper bounds" a quadratic approximation to the objective function D. This is done by first making a trial change $\Delta\phi^*$ in the routing variables using algorithm (8),(9). The link flows that would result from this change are then calculated going from the "most upstream" nodes downstream towards the destination. Based on the calculated trial flows the algorithm "senses" situations like the one in Figure 3 and finds a new change $\Delta\tilde{\phi}$. We describe the algorithm for the case of a single destination (node N). The algorithm for the case of more than one destination consists of sequences of single destination iterations whereby all destinations are taken up cyclically (i.e. the one-at-a-time mode of operation).

The Upper Bound

At the typical iteration of the algorithm, we have a vector of loop-free routing variables ϕ and a corresponding flow vector f. Let Δf denote an increment of flow such that $f + \Delta f$ is feasible. A constrained version of Newton's method [3] is obtained if Δf is chosen to minimize the quadratic objective function

$$N(\Delta f) = \sum_{i,\ell} D'_{i\ell}\Delta f_{i\ell} + \frac{1}{2} \sum_{i,\ell} D''_{i\ell}(\Delta f_{i\ell})^2 \tag{22}$$

subject to $f + \Delta f \in F$ where F is the set of feasible flow vectors. Let $\Delta\phi$ be a change in ϕ corresponding to Δf and let

$$\bar{\phi} = \phi + \Delta\phi \tag{23}$$

Let t be the vector of total traffic at the network nodes (cf. (1)), and let Δt be the corresponding change in t. Then

$$\Delta t_\ell = \sum_i \Delta f_{i\ell} \tag{24}$$

$$\Delta f_{i\ell} = \Delta t_i \, \bar{\phi}_{i\ell} + t_i \Delta\phi_{i\ell} \tag{25}$$

Substituting (25) in (22), we can express $N(\Delta f)$ in terms of $\Delta\phi$.

$$N(\Delta f) = \sum_{i,\ell} D'_{i\ell}\Delta t_i \, \bar{\phi}_{i\ell} + \sum_{i,\ell} D'_{i\ell} \, t_i \, \Delta\phi_{i\ell} \tag{26}$$

$$+ \frac{1}{2} \sum_{i,\ell} D''_{i\ell}[(\Delta t_i \bar{\phi}_{i\ell})^2 + 2\Delta t_i \bar{\phi}_{i\ell} \, t_i \, \Delta\phi_{i\ell} + (t_i \Delta\phi_{i\ell})^2]$$

We would like to minimize this expression by a distributed algorithm in which each node i selects $\Delta\phi_{i\ell}$ for each outgoing link (i,ℓ). The difficulty

here is that the nodes are all coupled through the vector Δt; a change $\Delta\phi_{i\ell}$ generates a change Δt_n at each node n downstream of the link (i,ℓ).

In what follows, we will first eliminate the dependence of $N(\Delta f)$ on the linear terms in Δt; we then proceed to upper bound $N(\Delta f)$ in such a way as to eliminate the quadratic terms in Δt. Finally then, we show how each node i can select $\Delta\phi_{i\ell}$ for its outgoing links so as to approximately minimize the upper bound to $N(\Delta f)$. We start by combining the two terms in (26) that are linear in Δt,

$$N(\Delta f) = \sum_{i,\ell} \bar{D}'_{i\ell}\Delta t_i \bar{\phi}_{i\ell} + \sum_{i,\ell} D'_{i\ell} t_i \Delta\phi_{i\ell} \tag{27}$$

$$+ \frac{1}{2} \sum_{i,\ell} D''_{i\ell}[(\Delta t_i \bar{\phi}_{i\ell})^2 + (t_i \Delta\phi_{i\ell})^2]$$

where

$$\bar{D}'_{i\ell} = D'_{i\ell} + D''_{i\ell}\, t_i\, \Delta\phi_{i\ell} \tag{28}$$

We can interpret $\bar{D}'_{i\ell}$ to first order as the derivative of $D_{i\ell}$ evaluated at the flow $t_i\, \bar{\phi}_{i\ell}$. The following simple lemma will eliminate Δt from the first term in (27); we state it in greater generality than needed here since we will use it again on the quadratic term in Δt.

Lemma 1: Let $\mu_{i\ell}$ be real for each $(i,\ell)\varepsilon L$, and for each node i, let T_i, $\tilde{T}_i$, d_i, $\tilde{d}_i$ be variables related by

$$T_\ell = \sum_i T_i\, \mu_{i\ell} + \tilde{T}_\ell \; ; \; 1 \le \ell \le N \tag{29}$$

$$d_i = \sum_\ell d_\ell\, \mu_{i\ell} + \tilde{d}_i \quad ; \; 1 \le i \le N \tag{30}$$

Then

$$\sum_i \tilde{d}_i T_i = \sum_i d_i \tilde{T}_i \tag{31}$$

Proof: Using (30) and then (29), we have

$$\sum_i \tilde{d}_i T_i = \sum_i d_i T_i - \sum_\ell d_\ell \mu_{i\ell} T_i$$

$$= \sum_i d_i T_i - \sum_\ell d_\ell (T_\ell - \tilde{T}_\ell)$$

$$= \sum_\ell d_\ell \tilde{T}_\ell \qquad \text{Q.E.D.}$$

To use this lemma on the first term of (27), associate $\bar{\phi}_{i\ell}$ with $\mu_{i\ell}$, Δt_i with T_i and $\sum_m t_m \Delta\phi_{mi}$ with $\tilde{T}_i$. Then (24) and (25) are equivalent to (29) in the lemma. Defining $\bar{D}'_i$ by

$$\bar{D}'_i = \sum_\ell \bar{D}'_\ell \bar{\phi}_{i\ell} + \sum_\ell \bar{D}'_{i\ell} \bar{\phi}_{i\ell} \; ; \; \bar{D}'_N = 0 \tag{32}$$

and associating $\bar{D}'_i$ with d_i and $\sum_\ell \bar{D}'_{i\ell} \bar{\phi}_{i\ell}$ with $\tilde{d}_i$, the lemma asserts that

$$\sum_{i,\ell} \bar{D}'_{i\ell} \Delta t_i \bar{\phi}_{i\ell} = \sum_{i,\ell} \bar{D}'_\ell t_i \Delta\phi_{i\ell} \tag{33}$$

It can be seen that D'_i can be calculated in a distributed fashion starting from the destination and proceeding upstream similarly as in algorithm (8), (9). Using (33) in (27), we have

$$N(\Delta f) = \sum_{i,\ell} \bar{D}'_\ell \; t_i \; \Delta\phi_{i\ell} + \sum_{i,\ell} D'_{i\ell} \; t_i \; \Delta\phi_{i\ell} \tag{34}$$

$$+ \frac{1}{2} \sum_{i,\ell} D''_{i\ell} (\Delta t_i \bar{\phi}_{i\ell})^2 + \frac{1}{2} \sum_{i,\ell} D''_{i\ell} (t_i \Delta\phi_{i\ell})^2$$

All of the terms in (34) except for $(\Delta t_i)^2$ can be calculated in a distributed fashion, moving upstream as in (8),(9). We recall now that the algorithm is going to use the algorithm of (8),(9) first to calculate a trial change $\Delta\phi^*$. We next show how $\Delta\phi^*$ will be used to upper bound $(\Delta t_i)^2$ in such a way that lemma 1 can be employed on the result. For all $(i,\ell)\in L$, define

$$\Delta\phi^{*+}_{i\ell} = \max(0,\Delta\phi^*_{i\ell});\ \Delta\phi^{*-}_{i\ell} = |\min(0,\Delta\phi^*_{i\ell})| \tag{35}$$

$$\Delta t^{*+}_{\ell} = \sum_i [t_i\Delta\phi^{*+}_{i\ell} + \Delta t^{*+}_i(\phi_{i\ell}+\Delta\phi^{*+}_{i\ell})] \tag{36}$$

$$\Delta t^{*-} = \sum_i [t_i\Delta\phi^{*-}_{i\ell} + \Delta t^{*-}_i\ \phi^-_{i\ell}] \tag{37}$$

The quantities Δt^{*+}_{ℓ}, Δt^{*-}_{ℓ} are well defined by virtue of the fact that the set of links

$$L^* = \{(i,\ell)\in L|\ \phi_{i\ell} > 0, \text{ or } \phi_{i\ell}+\Delta\phi^*_{i\ell} > 0\}$$

forms an acyclic network [in view of the manner that the sets of blocked nodes $B(\phi;i)$ are defined in algorithm (8),(9)]. As a result Δt^{*+}_{ℓ} and Δt^{*-}_{ℓ} are zero for all nodes ℓ which are the "most upstream" in this acyclic network. Starting from these nodes and proceeding downstream the computation of Δt^{*+}_{ℓ} and Δt^{*-}_{ℓ} can be carried out in a distributed manner for each ℓ using (36) and (37).

We next define the same positive and negative parts for $\Delta\phi$,

$$\Delta\phi^+_{i\ell} = \max(0,\Delta\phi_{i\ell});\ \Delta\phi^-_{i\ell} = |\min(0,\Delta\phi_{i\ell})| \tag{38}$$

$$\Delta t^+_{\ell} = \sum_i [t_i\Delta\phi^+_{i\ell} + \Delta t^+_i(\phi_{i\ell}+\Delta\phi^+_{i\ell})] \tag{39}$$

$$\Delta t_{\ell}^{-} = \sum_{i}[t_i \Delta\phi_{i\ell}^{-} + \Delta t_i^{-} \phi_{i\ell}] \tag{40}$$

The following constraints are now placed on $\Delta\phi$:

$$\Delta\phi_{i\ell} > 0 \text{ only if } \Delta\phi_{i\ell}^{*} > 0 \tag{41a}$$

$$\Delta\phi_{i\ell} < 0 \text{ only if } \Delta\phi_{i\ell}^{*} < 0 \tag{41b}$$

$$\sum_{\ell} \Delta\phi_{i\ell} = 0 \; ; \quad \phi_{i\ell} + \Delta\phi_{i\ell} \geq 0 \tag{41c}$$

With these constraints Δt_{ℓ}^{+}, Δt_{ℓ}^{-} are also well defined; Δt_{ℓ}^{+} is interpreted as the increase in flow at ℓ due to increases in $\Delta\phi$, omitting the effects of decreases in $\Delta\phi$. Similarly Δt_{ℓ}^{-} is an upper bound to the magnitude of the decrease in flow at ℓ due to decreases in $\Delta\phi$. It follows easily that

$$(\Delta t_{\ell})^2 \leq (\Delta t_{\ell}^{+})^2 + (\Delta t_{\ell}^{-})^2 \tag{42}$$

With the constraint (41), it is also easy to see, from (35)-(40), that for each ℓ

$$\Delta t_{\ell}^{+} > 0 \quad \text{only if } \Delta t_{\ell}^{*+} > 0 \tag{43a}$$

$$\Delta t_{\ell}^{-} > 0 \quad \text{only if } \Delta t_{\ell}^{*-} > 0 \tag{43b}$$

Finally, using the Cauchy-Schwartz inequality on (39), we obtain

$$(\Delta t_{\ell}^{+})^2 = \Big[\sum_i \Big(\frac{t_i \Delta\phi_{i\ell}^{+}}{\sqrt{t_i \Delta\phi_{i\ell}^{*+}}} \sqrt{(t_i \Delta\phi_{i\ell}^{*+})} \Big) + \sum_i \frac{\Delta t_i^{+}(\phi_{i\ell}+\Delta\phi_{i\ell}^{+})}{\sqrt{\Delta t_i^{*+}(\phi_{i\ell}+\Delta\phi_{i\ell}^{*+})}} \sqrt{\Delta t_i^{*+}(\phi_{i\ell}+\Delta\phi_{i\ell}^{*+})} \Big]^2$$

$$\leq \left[\sum_i \frac{t_i (\Delta\phi_{i\ell}^+)^2}{\Delta\phi_{i\ell}^{*+}} + \sum_i \frac{(\Delta t_i^+)^2 (\phi_{i\ell}+\Delta\phi_{i\ell}^+)^2}{\Delta t_i^{*+}(\phi_{i\ell}+\Delta\phi_{i\ell}^{*+})} \right] \Delta t_\ell^{*+} \tag{44}$$

when we have used (36), and where from each summation we exclude all nodes i for which the denominator (and hence the numerator by (41) and (43)) is 0. Similarly,

$$(\Delta t_\ell^-)^2 \leq \left[\sum_i \frac{t_i (\Delta\phi_{i\ell}^-)^2}{\Delta\phi_{i\ell}^{*-}} + \sum_i \frac{(\Delta t_i^-)^2}{\Delta t_i^{*-}} \phi_{i\ell} \right] \cdot \Delta t_\ell^{*-} \tag{45}$$

The following proposition yields the desired upper bound.

Proposition 3: Under the constraint (41),

$$N(\Delta f) \leq \sum_i t_i Q_i(\Delta\phi_i) \tag{46}$$

where

$$Q_i(\Delta\phi_i) = \sum_\ell [(D'_{i\ell}+\bar{D}'_\ell)\Delta\phi_{i\ell} + \frac{1}{2}(t_i D''_{i\ell}+\beta_{i\ell})(\Delta\phi_{i\ell})^2] \tag{47}$$

$$\beta_{i\ell} = \begin{cases} \dfrac{D''^+_\ell}{\Delta\phi^{*+}_{i\ell}} & \text{if} \quad \Delta\phi^*_{i\ell} > 0 \\ \dfrac{D''^-_\ell}{\Delta\phi^{*-}_{i\ell}} & \text{if} \quad \Delta\phi^*_{i\ell} < 0 \\ 0 & \text{if} \quad \Delta\phi^*_{i\ell} = 0 \end{cases} \tag{48}$$

and

$$D''^+_i = \sum_\ell [D''_{i\ell}\, \bar{\phi}^2_{i\ell}\, \Delta t^{*+}_i + D''^+_\ell \frac{(\phi_{i\ell}+\Delta\phi^+_{i\ell})^2}{\phi_{i\ell}+\Delta\phi^{*+}_{i\ell}}] \; ; \; i \neq N \tag{49}$$

$$D''^-_i = \sum_\ell [D''_{i\ell}\, \bar{\phi}^2_{i\ell}\, \Delta t^{*-}_i + D''^-_i \phi_{i\ell}] \; ; \; i \neq N \tag{50}$$

$$D''^+_N = D''^-_N = 0$$

Proof: In view of (34), it will suffice to show that

$$\sum_{i,\ell} D''_{i\ell} \left(\Delta t_i \bar{\phi}_{i\ell}\right)^2 \leq \sum_{i,\ell} t_i \beta_{i\ell} (\Delta\phi_{i\ell})^2 \tag{51}$$

From (42),

$$\sum_{i,\ell} D''_{i\ell} (\Delta t_i \bar{\phi}_{i\ell})^2 \leq \sum_{i,\ell} D''_{i\ell} \bar{\phi}_{i\ell}^2 (\Delta t_i^+)^2 + \sum_{i,\ell} D''_{i\ell} \bar{\phi}_{i\ell}^2 (\Delta t_i^-)^2 \tag{52}$$

Define T_ℓ, for all nodes ℓ, to satisfy

$$T_\ell = \sum_i t_i \frac{(\Delta\phi_{i\ell}^+)^2}{\Delta\phi_{i\ell}^{*+}} + \sum_i T_i \frac{(\phi_{i\ell}+\Delta\phi_{i\ell}^+)^2}{(\phi_{i\ell}^*+\Delta\phi_{i\ell}^{*+})} \tag{53}$$

By comparing with (44), we see that $(\Delta t_\ell^+)^2/\Delta t_\ell^{*+} \leq T_\ell$. Thus

$$\sum_{i,\ell} D''_{i\ell} \bar{\phi}_{i\ell}^2 (\Delta t_i^+)^2 \leq \sum_{i,\ell} D''_{i\ell} \bar{\phi}_{i\ell}^2 \Delta t_i^{*+} T_i = \sum_i \tilde{d}_i T_i$$

where

$$\tilde{d}_i = \sum_\ell D''_{i\ell} \bar{\phi}_{i\ell}^2 \Delta t_i^{*+} \tag{54}$$

Applying lemma 1, associating $\tilde{T}_\ell$ with the first term of (53) and d_i with $D_i''^+$ in (49), we have

$$\sum_{i,\ell} D''_{i\ell} \bar{\phi}_{i\ell}^2 (\Delta t_i^+)^2 \leq \sum_{i,\ell} D_\ell''^+ t_i \frac{(\Delta\phi_{i\ell}^+)^2}{\Delta\phi_{i\ell}^{*+}} \tag{55}$$

The second term can be considered as a sum over just those terms for which $\Delta\phi_{i\ell}^{*+} > 0$. Handling the Δt_i^- term in the same way,

$$\sum_{i,\ell} D''_{i\ell}\, \bar{\phi}_{i\ell}^{\,2} (\Delta t_i^-)^2 \;\leq\; \sum_{i,\ell} D''^{-}_{\ell}\, t_i \, \frac{(\Delta\phi^-_{i\ell})^2}{\Delta\phi^{*-}_{i\ell}} \tag{56}$$

Combining (55), (56) with (51), (52) completes the proof. Q.E.D.

The Algorithm

The algorithm can now be completely defined. After the routing increment $\Delta\phi^*$ is calculated in a distributed manner by means of algorithm (8),(9), each node i computes the quantities Δt_i^{*+} and Δt_i^{*-}. This is done recursively and in a distributed manner by means of equations (36), (37) starting from the "most upstream" nodes and proceeding downstream towards the destination. When this downstream propagation of information reaches the destination indicating that all nodes have completed the computation of Δt_i^{*+} and Δt_i^{*-}, the destination gives the signal for initiation of the second phase of the iteration which consists of computation of the actual routing increments $\Delta\tilde{\phi}_i$. To do this each node i must receive the values of $\bar{D}'_\ell$, D''^{+}_ℓ, and D''^{-}_ℓ from its downstream neighbors ℓ and then determine the increments $\Delta\tilde{\phi}_{i\ell}$ which minimize $Q_i(\Delta\phi_i)$ subject to the constraint (41) and the new routing variables

$$\bar{\phi}_{i\ell} \;=\; \phi_{i\ell} + \Delta\tilde{\phi}_{i\ell}.$$

Then node i proceeds to compute $\bar{D}'_i$, D''^{+}_i, and D''^{-}_i via (32), (49), and (50) and broadcasts these values to all upstream neighbors. Thus proceeding recursively upstream from the destination each node computes the actual routing increments $\Delta\tilde{\phi}_i$ in much the same way as the trial routing increments $\Delta\phi^*_i$ were computed earlier.

We now analyze the descent properties of the algorithm. We assume a single destination but the proof extends trivially to the case where we have multiple destinations and the algorithm is operated in the one destination at a time mode. In view of the fact that each function $D_{i\ell}$ is strictly convex it follows that there is a unique optimal set of total link flows $\{f^*_{i\ell} | (i,\ell)\in L\}$. It is clear that given any $\varepsilon>0$ there exists a scalar γ_ε such that for all feasible total link flow vectors f satisfying

$$|f_{i\ell} - f^*_{i\ell}| \leq \gamma_\varepsilon \quad , \quad \forall (i,\ell)\in L \tag{57}$$

we have

$$\frac{1}{1+\varepsilon} D''_{i\ell}(f^*_{i\ell}) \leq D''_{i\ell}(f_{i\ell}) \leq (1+\varepsilon) D''_{i\ell}(f^*_{i\ell}), \quad \forall (i,\ell)\in L. \tag{58}$$

The strict positivity assumption on $D''_{i\ell}$ also implies that for each $\gamma_\varepsilon>0$ there exists a scalar $\delta(\gamma_\varepsilon)$ such that every feasible f satisfying $\sum_{i,\ell} D_{i\ell}(f_{i\ell}) \leq \delta(\gamma_\varepsilon)$ also satisfies (57) and hence also (58). Furthermore $\delta(\gamma_\varepsilon)$ can be taken arbitrarily large provided γ_ε is sufficiently large. We will make use of this fact in the proof of the following result the proof of which may be found in [14].

Proposition 4: Let ϕ and $\bar{\phi}$ be two successive vectors of routing variables generated by the algorithm of this section (with stepsize $\alpha=1$) and let f and $\bar{f}$ be the corresponding vectors of link flows. Assume that for some ε with $0 < \varepsilon < \sqrt{\frac{3}{2}} - 1$ we have

$$\sum_{i,\ell} D_{i\ell}(f_{i\ell}) \leq \delta(\gamma_\varepsilon) \tag{59}$$

where γ_ε is the scalar corresponding to ε as in (57), (58), and $\delta(\gamma_\varepsilon)$ is such that (57) [and hence also (58)] holds for all feasible f satisfying (59). Then

$$D(\bar{\phi},r) - D(\phi,r) \leq -\rho(\varepsilon) \sum_{(i,\ell)\in L} t_i (D''_{i\ell}+\beta_{i\ell})(\bar{\phi}_{i\ell}-\phi_{i\ell})^2 \qquad (60)$$

where $\rho(\varepsilon) = \frac{1-4\varepsilon-2\varepsilon^2}{2} > 0$ for all ε with $0 < \varepsilon < \sqrt{\frac{3}{2}} - 1$.

The preceding proposition shows that the algorithm of this section does not increase the value of the objective function once the flow vector f enters a region of the form $\{f \mid \sum_{i,\ell} D_{i\ell}(f_{i\ell}) \leq \delta(\gamma_\varepsilon)\}$, and that the size of this region increases as the third derivative of $D_{i\ell}$ becomes smaller. Indeed if each function $D_{i\ell}$ is quadratic then (58) is satisfied for all $\varepsilon > 0$ and the algorithm will not increase the value of the objective for all f. These facts can be used to show that <u>if the starting total flow vector f is sufficiently close to the optimal the algorithm of this section will converge to the optimal solution</u>. The proof is similar to the one of Proposition 2 as given in [11] and is omitted.

We cannot expect to be able to guarantee theoretical convergence when the starting routing variables are far from optimal since this is not a generic property of Newton's method which the algorithm attempts to approximate. However in a large number of computational experiments with objective functions typically arising in communication networks and starting flow vectors which were far from optimal [5] we have never observed divergence or an increase of the value of the objective function in a single iteration. In any case it is possible to prove a global convergence result for the version of the algorithm whereby the expression $Q_i(\Delta\phi_i)$ is replaced by

$$Q_i^{\alpha}(\Delta\phi_i) = \sum_{\ell} \left[(D'_{i\ell} + \overline{D}'_{\ell}) \Delta\phi_{i\ell} + \frac{1}{2\alpha} (t_i D''_{i\ell} + \beta_{i\ell}) (\Delta\phi_{i\ell})^2 \right] \tag{61}$$

where α is a sufficiently small positive scalar stepsize. The preceding analysis can be easily modified to show that if we introduce a stepsize α as in (61) then the algorithm of this section is a descent algorithm at all flows in the region $\{f \mid \sum_{i,\ell} D_{i\ell}(f_{i\ell}) \leq \delta(\gamma_\varepsilon)\}$ where

$$0 < \varepsilon < \sqrt{\frac{2 + \alpha}{2\alpha}} - 1.$$

From this it follows that <u>given any starting point $\phi^o \varepsilon \Phi$, there exists a scalar $\overline{\alpha} > 0$ such that for all stepsizes $\alpha \varepsilon (0, \overline{\alpha}]$ the algorithm of this section does not increase the value of the objective function at each subsequent iteration</u>. This fact can be used to prove a convergence result similar to the one of Proposition 2.

References

[1] Gallager, R.G., "A Minimum Delay Routing Algorithm Using Distributed Computation", IEEE Trans. on Communication, Vol. COM-25, 1977, pp. 73-85.

[2] Goldstein, A.A., "Convex Programming in Hilbert Space", Bull. Amer. Math. Soc., Vol. 70, 1964, pp. 709-710.

[3] Levitin, E.S. and B.T. Polyak, "Constrained Minimization Problems", USSR Comput. Math. Math. Phys., Vol. 6, 1966, pp. 1-50.

[4] Bertsekas, D.P., "Algorithms for Nonlinear Multicommodity Network Flow Problems", in Proc. of International Symposium on Systems Optimization and Analysis, A. Bensoussan and J.L. Lions (eds.), Springer-Verlag, 1979, pp. 210-224.

[5] Bertsekas, D.P., Gafni, E., and Vastola, K.S., "Validation of Algorithms for Optimal Routing of Flow in Networks", Proc. of IEEE Conf. on Decision and Control, San Diego, Calif., Jan. 1979, pp. 220-227.

[6] Vastola, K.S., "A Numerical Study of Two Measures of Delay for Network Routing", M.S. Thesis, Dept. of Electrical Engineering, Univ. of Ill, Urbana, Ill., Sept. 1979.

[7] Bertsekas, D.P., "A Class of Optimal Routing Algorithms for Communication Networks", Proc. of the Fifth International Conference on Computer Communication (ICCC-80), Atlanta, Ga., Oct. 1980, pp. 71-76.

[8] Bertsekas, D.P., and Gafni, E., "Projection Methods for Variational Inequalities with Application to the Traffic Assignment Problem", Report LIDS-P-1043, Mass. Institute of Technology, Cambridge, Mass., June 1980, Math. Programming Study 17, 1982, pp. 139-159.

[9] Golestaani, S.J., "A Unified Theory of Flow Control and Routing in Data Communication Networks", Report LIDS-TH-963, Laboratory for Information and Decision Systems, Mass. Institute of Technology, Cambridge, Mass., Jan. 1980.

[10] Gallager, R.G., and Golestaani, S.J., "Flow Control and Routing Algorithms for Data Networks", Proc. of the Fifth International Conference on Computer Communication (ICCC-80), Atlanta, Ga., Oct., 1980, pp. 779-784.

[11] Gafni, E.M., "Convergence of a Routing Algorithm", Report LIDS-TH-907, Laboratory for Information and Decision Systems, Mass. Institute of Technology, Cambridge, Mass., May 1979.

[12] Bertsekas, D.P., and Gafni, E.M., "Projected Newton Methods and Optimization of Multicommodity Flows", Report LIDS-P-1140, Laboratory for Information and Decision Systems, Mass. Institute of Technology, Cambridge, Mass., August 1981, IEEE Trans. on Aut. Control, Dec., 1983.

[13] Gafni, E.M., "The Integration of Routing and Flow Control for Voice and Data in Computer Communication Networks", Ph.D. Dissertation, Dept. of Electrical Engineering and Computer Science, Mass. Institute of Technology, Cambridge, Mass., August 1982.

[14] Bertsekas, D.P., Gafni, E.M., and Gallager, R.G., "Second Derivative Algorithms for Minimum Delay Distributed Routing in Networks", M.I.T., LIDS Report P-1082-A, Laboratory for Information and Decision Systems, Mass. Institute of Technology, August 1982.

[15] Bertsekas, D.P., "Optimal Routing and Flow Control Methods for Communication Networks", in Analysis and Optimization of Systems, A. Bensoussan and J. L. Lions, eds., Springer-Verlag, Berlin and N.Y., 1982, pp. 615-643.

[16] Bertsekas, D.P., Gendron, R., and Tsai, W.K., "Implementation of an Optimal Multicommodity Network Flow Algorithm Based on Gradient Projection and a Path Flow Formulation", Lab. for Information and Decision Systems Report LIDS-P-1364, January 1984.