FLOW CONTROL AND ROUTING
IN AN INTEGRATED VOICE AND DATA COMMUNICATION NETWORK

by

OLIVER CHUKWUDI IBE
B.Sc., University of Nigeria
(1975)

S.M., Massachusetts Institute of Technology
(1979)

M.B.A., Northeastern University
(1980)

SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF SCIENCE

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May, 1981

Signature of Author Signature redacted
Department of Electrical Engineering and Computer Science May 12, 1981

Certified by Signature redacted
Robert G. Gallager
Thesis Supervisor

Accepted by
Arthur C. Smith
Chairman, Departmental Committee on Graduate Students

FLOW CONTROL AND ROUTING IN AN INTEGRATED VOICE AND DATA COMMUNICATION NETWORK

by

OLIVER CHUKWUDI IBE

Submitted to the Department of Electrical Engineering and Computer Science
on May 12, 1981 in partial fulfillment of the
requirements for the Degree of Doctor of Science

ABSTRACT

We consider a model of an integrated voice and data network. This model, which lends itself to analytic and algorithmic solution, is formulated as a convex optimization problem. The objective function comprises two types of functions: the congestion cost functions and the rate limitation functions. The congestion cost functions act to limit the average traffic entering into the network to values that would not cause network congestion. The rate limitation functions ensure that all conversations are fairly treated. The model can be generalized to solve problems of networks which handle n types of traffic that have different levels of delay sensitivity, where $n \geq 2$.

A joint flow control and routing algorithm is constructed which uses short term average information on the network utilization to set the voice packet lengths and data input rates, and to determine the routes for each conversation. The voice packet lengths and data input rates are set in such a way as to achieve an optimal tradeoff between each user's satisfaction and the cost of network congestion. Additional protocols are specified for dealing with such issues as congestion avoidance and control, and for implementing flow control on a more dynamic basis than the quasi-static joint flow control and routing algorithm can handle.

Thesis Supervisor: Robert G. Gallager

Title: Professor of Electrical Engineering and Computer Science

ACKNOWLEDGEMENTS

My stay at M.I.T. has been a rather brief one, spanning a period of four academic years, and many people have contributed to this success story. The first person is my thesis supervisor, Professor Robert G.Gallager. Bob unwaveringly committed his time to the supervision of this thesis, and ensured that I encountered no financial difficulties right from the first day I came to LIDS. To say that I am grateful to him is an understatement; no amount of "thank you" is enough to portray my indebtedness to him. The next person is my Graduate Counsellor, Professor Al Drake. He explained the modus vivendi of M.I.T. to me, and that I was able to overcome the initial setbacks I encountered here stemmed from his gentle words of encouragement. I sieze this opportunity to thank Al for his services to me.

I have richly benefited from my association with Professor Wilbur B. Davenport, Jr. As one of my thesis readers he provided insightful comments that helped to put the thesis in the present form. I wish to thank him not only for this but also for showing so much interest in my welfare. I also wish to thank Professor Dimitri Bertsekas, who is also my thesis reader, for useful suggestions on the direction of the research. I am grateful to Frantiska Frolik who typed this thesis; in fact she had to cut short her vacation for the sole purpose of doing the typing. I wish to thank Arthur Giordani for drawing the figures. Finally,

I wish to thank my wife, Christina, for her love and understanding which enabled me to complete my studies at M.I.T. on schedule.

This work was supported by a partial assistantship from the Laboratory for Information and Decision Systems(LIDS) with funds provided by the Advanced Research Projects Agency, and by a fellowship from the University of Nigeria, NSUKKA.

TABLE OF CONTENTS

CHAPTER I

INTRODUCTION

1.1 Motivation For The Integrated Network

The development of the integrated voice and data network concept has resulted from a combination of several factors including the following:

(a) the low utilization of network resources when voice, data, fascimile, etc. are transmitted on separate communications networks in which traditional switching techniques are used;

(b) the success of such data communications networks as the Telenet [33], the Tymnet [30], the French Transpac [10], and the Canadian Datapac [7];

(c) the recent technological advances in speech processing, particularly speech digitization.

In a telephone conversation a speaker alternates randomly between the speaking (or talkspurt) mode and the listening (or silence) mode. Measurement studies have shown that in a typical telephone conversation a speaker uses the channel only 35-40% of the time duration of the conversation [4]. This then means that dedicating a channel to a pair of speakers throughout the duration of their conversation is a waste of the network resources.

Many techniques have been proposed to minimize this waste. These techniques attempt to compress n voice conversations onto m channels, where $n > m$. The earliest of these techniques is the Bell System "Time

Assignment Speech Interpolation" (TASI) in which a channel is allocated to a subscriber only when the appropriate hardware detects that he is actively speaking [4]. Once he accesses the channel the speaker uses it without interruption for the duration of his talkspurt. During periods of silence he relinquishes the channel, and it becomes available to other speakers. Any talkspurt that is generated when all channels are busy is "frozen out" and will have to wait on a first-come first-served basis for an available channel. This freeze-out results in the clipping and loss of the initial part of the talkspurt. It is even possible to lose an entire talkspurt if the waiting time is longer than the spurt duration [30]. For a satisfactory speech quality the fraction of the lost speech should be less than 0.5%. The ratio of the number of subscribers supported by the TASI system to the number of channels required to maintain an acceptable fraction of lost speech is called the "TASI Advantage". If p is the fraction of time that a typical speaker is in talkspurt, the maximum potential TASI advantage is $1/p$. Since $p < 0.5$, this advantage is greater than 2. However, at least 40 channels and 80 speakers are required to achieve this TASI advantage [4]. For a smaller number of channels, attempts to achieve a TASI advantage close to $1/p$ will result in unacceptably high lost fractions [32]. The TASI system described above is oriented toward analog speech transmission. Digital variations of TASI that are transmitted in the PCM form have also been implemented. These include Digital Speech Interpolation (DSI) [5], and Speech Predictive Encoding (SPEC) [28].

Speech is bursty in nature, and this accounts for the low utilization of the transmission resources in a voice communications network in which

the traditional frequency division multiplex (FDM) or time division multiplex (TDM) is used. The purpose of the TASI schemes is to achieve a better utilization of the network resources by smoothing out the voice flow. Data are also bursty in nature. An effort to smooth out data flow in a network has led to the concept of statistical multiplexing [8] or store-and-forward networks [1]. Unlike the TDM system traditionally used for data transmission, no slots are dedicated to any data source. Instead messages are merged into a buffer at each network node and the bits are transmitted at a synchronous rate. The messages work their way through the network to their destinations,queueing at each node where they are statistically multiplexed. The messages may remain in complete form (message switching) or be split into shorter blocks called packets (packet switching) [1]. Whether statistical multiplexing exists in the message-switched or packet-switched form, each unit carries a header which specifies among other things the source and destination addresses of that unit. There is basically little difference between packet switching and message switching. We shall use the term packet switching hereafter for both techniques.

That speech is bursty in nature makes it a good candidate for statistical multiplexing. This has been further made possible by the recent advances in speech processing which have led to the packetization of speech. This means that packets of speech, which are generated only when a speaker is in talkspurt, could be stored temporarily at each node in wait for free channels. Thus there would be no loss of any part of the speech as is often the case in the conventional TASI system. And the fact that speech and data can be packetized raises the possibility

of using the same network to transmit them. One then hopes that such an integration of speech and data onto the same network would lead to better utilization of the network resources [17,18].

1.2 Problems With Integration

The integration of voice and data traffics onto one network generates new communications problems. These problems include the choice of an appropriate switching technique, and how to control the network flow in order to meet the different demands made on the network by voice and data conversations.

Three switching strategies have been proposed for use in the integrated network [17]. These are:

(a) Circuit Switching

(b) Packet Switching

(c) Hybrid Switching.

In circuit switching, an end-to-end connection is established for a pair of voice or data users before they commence their conversation. The channel is dedicated to them throughout the duration of the conversation. The channel utilization may be improved by using the TASI system. In packet switching, both voice and data conversations are digitized and segmented into packets and routed through the network in a store-and-forward manner [1,21]. That is, each packet travelling from source node i to destination node j is "stored" in queue at an intermediate node k while awaiting transmission. It is sent "forward" to a selected adjacent node ℓ on the route to node j when link (k,ℓ) is free. The selection of node ℓ is made by some well-defined decision called the

routing algorithm. The process of storing and forwarding is repeated until the packet reaches its destination. Packets of the same conversation usually follow the same path. However, they may follow different paths, possibly arriving at the destination out of order. In this case they would have to be reassembled in their right order before being delivered to the ultimate user. Packet switching thus avoids dedicating the network facilities to users during silent periods since no packets are generated then. In hybrid switching, both circuit-switched and packet-switched traffics exist concurrently on the same channel. Each channel is partitioned into two subchannels; a circuit-switched subchannel and a packet-switched subchannel. Voice traffic uses the circuit-switched subchannel while data traffic uses the packet-switched subchannel. To increase the channel utilization a "movable boundary" feature can be incorporated, which permits the data packets to use any residual circuit-switched capacity that may be momentarily available due to voice traffic variations [9].

A well-designed communications network ensures that the traffic it handles does not exceed the capacity of the network devices in order that congestion may not build up and reduce the throughput of the network. In the extreme case congestion may lead to a deadlock in which case communication becomes impossible. The term flow control is used to describe any mechanism that ensures that the traffic entering into the network is maintained within limits compatible with the capacity of the network devices. A flow control protocol should perform the following functions [16]:

(a) It should prevent throughput degradation and loss of efficiency due to overload.

(b) It should prevent deadlock.

(c) It should ensure a fair allocation of the network resources among competing users.

(d) It should ensure that the rate at which the network accepts data is matched to the rate at which it can transmit them.

Voice and data traffics make different and conflicting demands on the network: They show different tolerances to delay and errors. Voice conversations require continuous and almost real-time delivery; they are very sensitive to delay. Data conversations, on the other hand, are generally intolerant of errors but less sensitive to delay. Data conversations need to be very reliably delivered to their ultimate users. A flow control protocol that is to be applied to an integrated voice and data network must, therefore, perform two additional important functions, namely:

(a) It should ensure that a voice conversation is transmitted with essentially constant delay while trading the speech quality in response to nework fluctuations.

(b) It should ensure that data conversations are transmitted with maximum reliability (that is, no errors or lost information) with communications delay as a secondary considerations.

1.3 Summary of Previous Work

The concept of integrated voice and data communications networks has generated much interest. It has been motivated by the need to exploit

recent technological advances to achieve economies of scale: sharing network resources more efficiently thereby reducing the cost per user. Unfortunately much of the work done in this area so far exists as mere heuristics that depend on simulation for evaluation.

Coviello and Vena [9] propose a type of hybrid-switched network called the slotted envelope network (SENET) which provides some degree of flexibility for a varying mix of traffic. It is a framed TDM scheme in which frames are divided into (V+D) slots. V slots are allocated to voice conversations on a circuit-switched basis, and D slots are allocated to data conversations on a packet-switched basis. Voice calls are blocked when all the V slots reserved for voice conversations are busy. A voice conversation uses its slot throughout the duration of the conversation. In order to increase the channel utilization, a movable boundary scheme is used whereby data traffic could use any voice slots that may be momentarily free due to variations in the number of active voice conversations. Forgie and Nemeth [11] consider a network concept called the packetized virtual circuit (PVC) which combines selected features from the packet and circuit switching technologies. The PVC technique is more or less a packet switching technique in which messages are packetized and travel along virtual connections to their destinations. The authors also suggest some ways to deal with congestion in the network, including discarding some voice packets when queues become excessive. Bially et al. [3] propose a packet-switched network in which a speech digitization concept called "embedded coding" enables the network to respond dynamically to the loading conditions. Here speech is encoded into different packets of different orders of importance. If all the

packets are delivered to the sink, the result would be a high quality speech. When the network loading becomes high the less important packets may be discarded. The delivered packets still produce usable but lower quality speech. Gitman and Frank [17] present an elaborate economic analysis of the integrated voice and data networks. They conclude, on the basis of a number of assumptions, that even when no optimization is performed in selecting the voice packet length, packet switching is still superior to all other switching techniques.

1.4 Synopsis of the Thesis

The hybrid-switched network is basically two distinct networks in one, if the movable boundary feature is not incorporated. Since operational circuit-switched and packet-switched networks exist, designing a pure hybrid-switched integrated voice and data network (i.e. one without movable boundaries) presents no new challenges. When the movable boundary feature is incorporated, the difference between the hybrid-switched and the packet-switched network becomes a matter of semantics rather than technology as the operation of the two techniques are almost identical, but not quite anyway.

We design a packet-switched integrated network that achieves a better resource allocation than the pure hybrid-switched network. We chart a new course by formulating the integrated network problem in a way that lends itself to an analytic and algorithmic solution. Voice packets have a non-preemptive priority over data packets. The reason for practising a non-preemptive priority includes the fact that when a data packet receiving service is preempted by an arriving voice packet,

the part of the data packet which had already been transmitted before preemption might appear as errors in the network. We could avoid this confusion by practising a non-preemptive priority. Also it does not make sense to use network resources to transmit data bits that would later be discarded.

As the title of this thesis suggests, we are primarily interested in the flow control and routing issues of the integrated network. We shall, however, discuss some protocols for congestion avoidance and control, and for admitting new conversations into the network.

The remainder of the thesis is organized as follows. In Chapter II we present the formulation of the network model and the problem statement. In Chapter III we give a solution to the problem of Chapter II. We also formulate a joint flow control and routing algorithm. In Chapter IV we consider protocols for performing miscellaneous functions, including congestion avoidance and control, and the admission of new conversations into the network. In chapter V we discuss how the model can be generalized for a multi-traffic network that carries traffic with different levels of delay sensitivity. In Appendices A and B we give proofs of theorems.

CHAPTER II

MODEL FORMULATION AND PROBLEM STATEMENT

2.1. Introduction

In [12] Gallager formulates a quasi-static routing strategy for a data communication network. He considers a network with slowly varying average input rate for each data conversation, and develops an algorithm for finding minimal delay routing using distributed computation. Golestaani [19] extends Gallager's work by considering a network with controllable input rates. These rates are adjusted periodically on the basis of the information on the network utilization. Also these rates are controlled in a manner that ensures that all users are fairly treated. The algorithm developed in [19] implements this fairness by attempting to establish an optimal tradeoff between each user's satisfaction and the network loading condition. A user's satisfaction is related to the rate at which he is permitted to transmit: the closer the permitted rate is to his desired rate, the more satisfied the user is.

The work done here is a further generalization of the work of Gallager [12], with applications to the integrated voice and data network. As in [19] we consider an integrated network with controllable voice and data input rates. The voice input rate is controlled by adjusting the voice packet lengths. The length of the voice packets determines the quality of the voice conversation: the longer the packets, the higher is the quality of the conversation. The voice packet lengths and the data input rates are also controlled in a manner that ensures that all users

are fairly treated. We effect this fairness by developing an algorithm that attempts to establish an optimal tradeoff between the network conditions and all users' satisfaction. For voice conversations, the longer the packets (and hence the higher the speech quality) the more satisfied the user is. And for data conversations, the closer a user's allowed rate is to his desired rate, the more satisfied he is. We assume that a voice digitizer that can set the voice packets to any desired lengths can be found. We do not concern ourselves with what happens to the speech at the receiver if the voice packet length is continuously varied. This is an aspect of signal processing and has not yet been resolved; it is a topic for future research.

The work done here is not the first to consider a scheme where voice packet lengths are varied in response to network conditions. The work of Bially et al.[3] is similar to ours. The way flow control is effected in [3] is equivalent to varying the voice packet length inside the network, and their scheme is more dynamic than ours. In this thesis we apply flow control at the input to the network by making changes in voice packet lengths before sending the packets into the network. This would then reduce the need to drop any segments of these packets in the network as is equivalently done in [3]. We shall discuss the similarity betweeen the two models more in Chapter IV.

2.2 The Model

2.2.1 Definitions

Consider a store-and-forward network with N nodes represented by $i = 1,2,\ldots,N$. A link that goes from node k to node ℓ is represented

by the symbol (k,ℓ). We assume that link (k,ℓ) is a directed link and is thus different from link (ℓ,k). Since most voice and data links in use are full duplex, we assume that if link (k,ℓ) exists then link (ℓ,k) also exists. Let

$f^v_{ik}(j)$ = the expected voice traffic on link (i,k), in bits/second, destined for node j

$f^d_{ik}(j)$ = the expected data traffic on link (i,k), in bits/second, destined for node j

ℓ_{ij} = the expected length of voice packets, in bits, belonging to conversations which enter the network at node i and are destined for node j

r_{ij} = the expected data input rate at node i, in bits/second, of conversations destined for node j

β = the rate at which voice packets are emitted by the voice digitizer during a talkspurt

$$F^v_{ik} = \sum_{j=1}^{N} f^v_{ik}(j)$$

= the aggregate voice traffic, in bits/second, on link (i,k)

$$F^d_{ik} = \sum_{j=1}^{N} f^d_{ik}(j)$$

= the aggregate data traffic, in bits/second, on link (i,k)

$F_{ik} = F^v_{ik} + F^d_{ik}$, is the total traffic on link (i,k)

C_{ik} = the capacity of link (i,k) in bits/second

$O(i)$ = the set of nodes with links going out from node i

$I(i)$ = the set of nodes with links going into node i

m_{ij} = the number of data conversations entering the network at node i and destined for node j

n_{ij} = the number of "off-hook" speakers with voice conversations entering the network at node i and destined for node j

n_{ij} is made up of two parts:

n_{ij}^T = the number of n_{ij} in talkspurt

n_{ij}^S = the number of n_{ij} in silence.

When a speaker is off-hook (i.e. in the conversational mode) he alternates randomly between the talkspurt and the silence modes. For a large value of n_{ij}, the number of speakers in talkspurt can be modelled by a Poisson arrival process with rate θ. For ease of analysis we assume that the holding times of talkspurts are exponentially distributed with mean ϕ^{-1}. Then the speaker activity for the (i,j) off-hook conversations can be modelled by the Markov chain shown in Fig. 2.1.

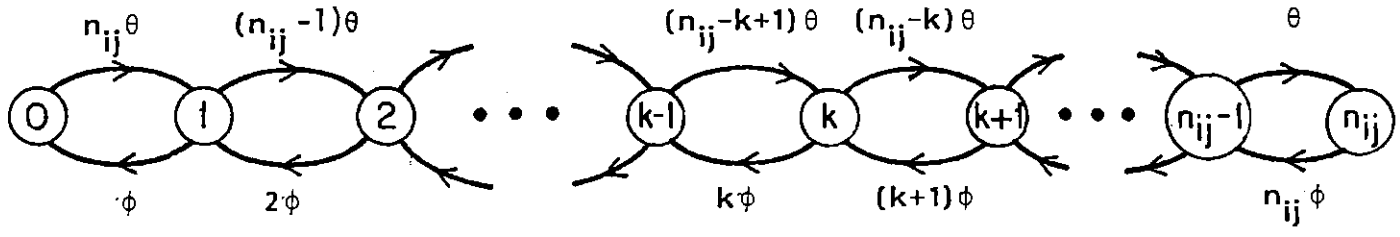

Figure 2.1 Speaker Activity Model for (i,j) Conversations

those of data packets, we grant the voice packets a non-preemptive priority over the data packets. Packets wait on a first-come first-served basis within their priorities. We assume that the queue at each output link can be modelled by an M/G/1 queue with priorities [22].

Consider a tagged packet that arrives at a node for service. Let

λ_v = the Poisson rate at which voice packets arrive at the output queue of the node, in packets/second

λ_d = the Poisson rate at which data packets arrive at the queue, in packets/second

n_v = the number of voice packets already in queue when the tagged packet arrived

n_d = the number of data packets already in queue when the tagged packet arrived

n_v' = the number of voice packets which arrive while the tagged packet is in queue.

Note that if the tagged packet is a voice packet these n_v' packets have no effect on its waiting time.

T_r = the remaining time to complete the service of the packet in service when the tagged packet arrived

T_1 = the time to service all the n_v voice packets

T_1' = the time to service all the n_v' voice packets

T_2 = the time to service all the n_d data packets.

2.2.3 Cost of Link Congestion

Let $d^n_{k\ell}$ = the delay experienced by the n^{th} packet of a talkspurt on link (k,ℓ) including procesing and queueing delays at node k

R = the route of the voice conversation

Then $d^n = \sum_{(k,\ell)\epsilon R} d^n_{k\ell}$, is the path delay of the n^{th} packet of the talkspurt.

In the discussion above the delay experienced by the first packet of a talkspurt, d^1, is crucial in determining whether or not a subsequent packet is discarded. If the first packet experiences little delay while other packets experience larger delays, then many of these packets would be discarded. On the other hand, if the first packet experiences a large delay while other packets experience less delay, then not many of these packets would be discarded. Thus the probability that the n^{th} packet of a talkspurt would be discarded depends not only on d^n but also on d^1. This dependence on d^1 is not a very desirable feature. In order to reduce this dependence (it cannot be totally eliminated!), we will attempt to make all packets of the same talkspurt experience similar delays on each link on their path. This would then translate to their experiencing similar path delays, and hence reducing the chances of their being discarded.

We hope to realize the above objective as follows. We consider a cost to be associated with the link delay of each voice packet. Similarly, we consider a cost to be associated with the link delay of each data packet. Since the delay requirements of voice packets are more stringent than

Thus the activity of each off-hook speaker can be modelled by the two-state Markov chain shown in Fig. 2.2.

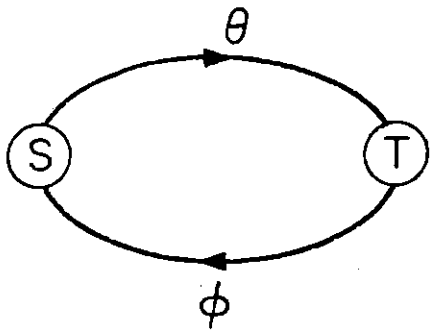

Figure 2.2 A Markov Chain Model of A Speaker

The steady-state probability that the speaker is in talkspurt is:

$$\gamma = \frac{\theta}{\phi + \theta} \tag{2.1}$$

Therefore, if there are n_{ij} off-hook speakers, the steady-state probability that k of them are in talkspurt is:

$$p_k = \binom{n_{ij}}{k} \gamma^k (1 - \gamma)^{n_{ij}-k} \tag{2.2}$$

One problem in the modelling of a voice network is that for any given number of off-hook speakers the number of speakers in talkspurt changes too fast to be tracked by any reasonable algorithm that makes use of global information. We are then left with the choice of estimating n_{ij}^T, given n_{ij}. In this work we initially estimate n_{ij}^T by the mean value of k; that is

$$n_{ij}^T = E(k) = \gamma n_{ij} \tag{2.3}$$

2.2.2 Voice Packet Delay Variations

Traditional circuit switching systems exhibit fixed delay to voice traffic. Packet switching systems, on the other hand, exhibit variable delay due to the possible queueing of the voice packets at the different nodes on the route from the source to the destination. It is assumed that when a speaker is in talkspurt the voice digitizer emits voice packets at a constant rate of β packets per second [3]. These packets experience different queueing delays along their route. If each packet is delivered to the ultimate user (or sink) as soon as it arrives at the destination node, the received message would contain many uneven and annoying gaps. Therefore, it is necessary that we install a smoothing buffer at the destination node, temporarily store the arriving voice packets, and finally release them to the sink at the same rate, β packets/second, at which the voice digitizer sent them into the network. In Chapter IV we shall consider the case where the rate at which the packets are released to the sink is different from β.

A schematic diagram of what happens to the voice packets as they travel along the route from source node i to destination node j is shown in Fig. 2.3. Figure 2.4 is a diagram of the possible path delays of five packets of a conversation. The first packet is transmitted at time zero; the second packet at time $1/\beta$; and in general, the k^{th} packet is transmitted at time $\frac{k-1}{\beta}$ $k \geq 1$. There are three links on the path.

From the diagram we observe that the first packet suffers a delay d_0, since it arrives at time d_0. This packet is not delivered immediately to the sink but is held back in the buffer for some time $\tau_0 - d_0$, and delivered at time τ_0. The reason for this action will be clear shortly. The second packet arrives at time d_1; it is held back in the buffer and released at time $\tau_0 + 1/\beta$. The third packet arrives at time $d_2 = \tau_0 + 2/\beta$, just in time to be delivered to the sink. The fourth packet arrives later than it is required to be delivered to the sink and so is discarded. A protocol could be formulated for generating fictitious packets which are to be released to the sink when the actual packets arrive late. However, this is a signal processing function which we will not be concerned with here. In general, if the k^{th} packet arrives earlier than $\tau_0 + \frac{k-1}{\beta}$, $k \geq 1$, it is stored in the buffer and released to the sink at time $\tau_0 + \frac{k-1}{\beta}$. If it arrives later than this time, it is thrown away.

The reason for imparting the extra delay on the first packet is now obvious: to help reduce the likelihood of discarding other packets due to their late arrival. Obviously choosing the extra delay $\tau_0 - d_0$ to be very large ensures that, with a probability close to one, all packets arrive in time to be delivered to the sink. However, this is not the solution we seek. We have noted that voice conversations require almost real-time delivery; therefore, the recommended choice of the extra delay is $\tau_0 - d_0 \leq 1/\beta$.

One of our objectives in the above scheme would be to reduce the path delay of each voice packet. This would in turn reduce the likelihood

of the packet being discarded. Another objective would be to increase the quality of each voice conversation. A realization of these objectives resides in making tradeoffs between speech quality and delay. Short packets would travel faster than long packets for two reasons. First, there is less transmission delay per packet on each link. Secondly, short packets generate smaller traffic than long packets, and so there is reduction in congestion. Thus the use of short packets would solve the delay problem. However, short packets imply poorer quality speech. Therefore, we need to use long packets in order to generate higher quality speech; but there is the attendant possibility of incurring larger packet delay. Any attempt to gain in quality, therefore, results in poorer delay performance; and any attempt to improve the delay performance degrades the quality. We need then to specify what would be the point of optimal tradeoff between quality and delay such that once we attain it, we would not be tempted to improve the quality or the delay performance any further. This suggests that we formulate an optimization problem the solution of which would generate the optimal packet lengths (and hence quality) and the optimal delay. Our task then boils down to choosing the appropriate objective function for the problem. Since we are dealing with an integrated voice and data network, we must ensure that the solution to the problem we shall formulate also provides the optimal data input rates and the optimal data packet delay. In the subsequent subsections we discuss the different aspects of the problem, and find appropriate cost functions that will constitute the objective function.

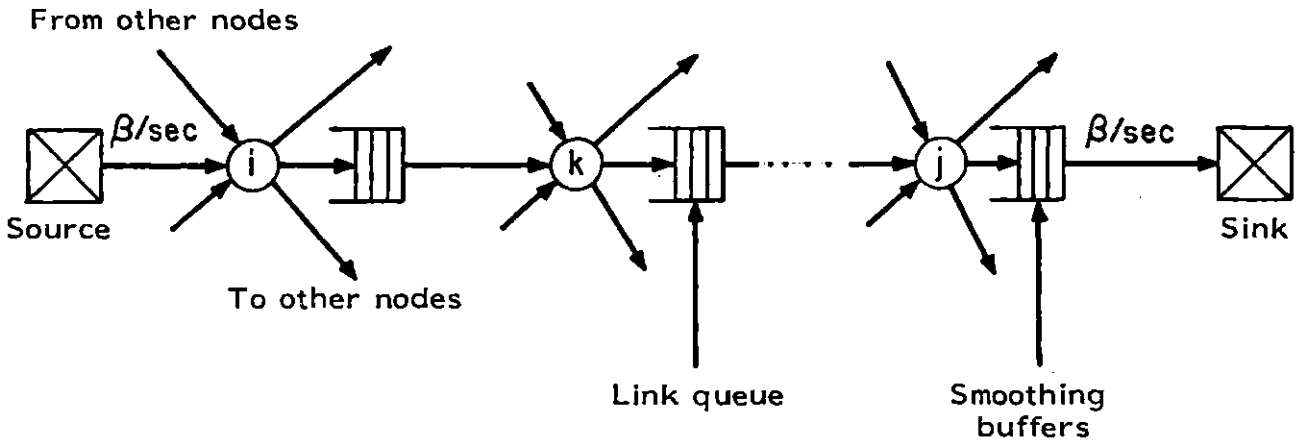

Figure 2.3 Model of A Voice Communication Path

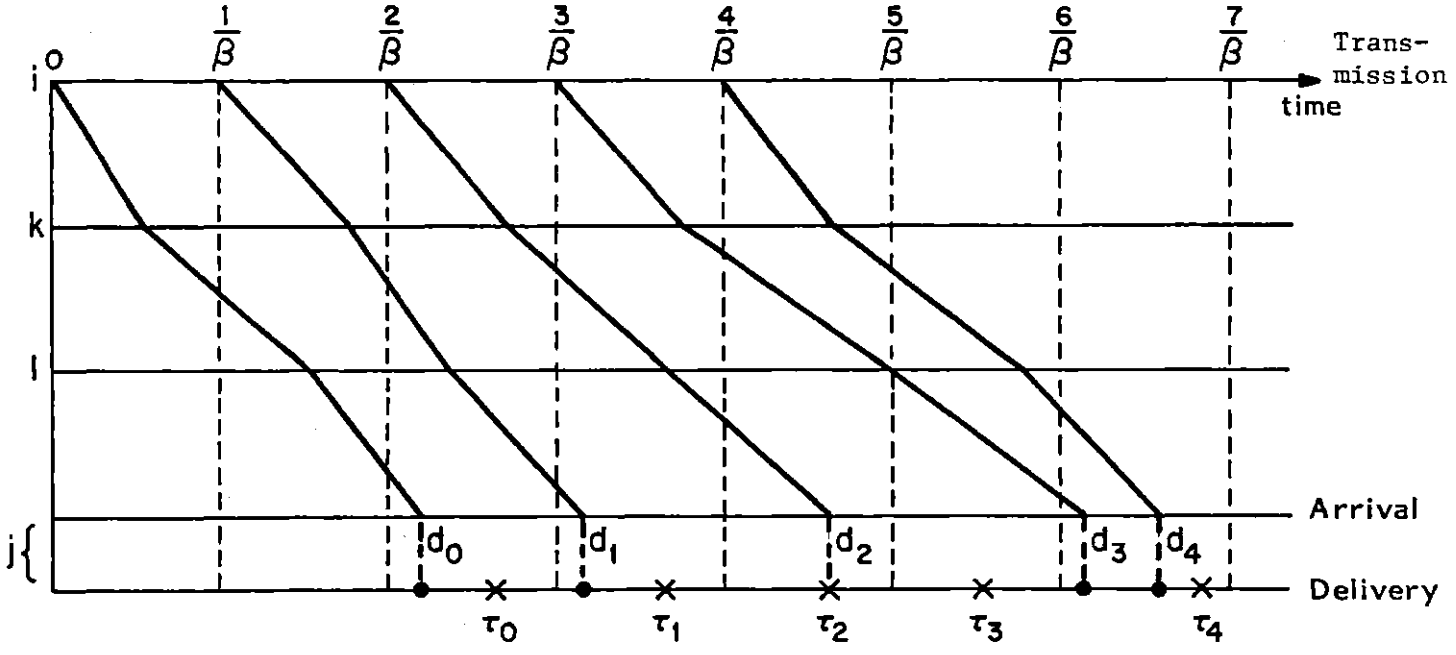

Figure 2.4 Example of Voice Packet Handling in the Network

Then if the tagged packet is a voice packet, its waiting time is:

$$T_v = T_r + T_1$$

Taking expectations we obtain

$$W_v = W_r + W_1 \tag{2.4}$$

where $W_i = E(T_i)$. If the time to service one voice packet is τ_v, then

$$W_1 = E(n_v)\, E(\tau_v) = E(n_v)\, \bar{\tau}_v .$$

By Little's formula [24], $E(n_v) = \lambda_v W_v$. Therefore, from (2.4)

$$W_v = W_r + \lambda_v W_v \bar{\tau}_v$$

Or

$$W_v = \frac{W_r}{1 - \rho_v} \tag{2.5}$$

where $\rho_v \triangleq \lambda_v \bar{\tau}_v$ is the link voice utilization factor. Similarly, if the tagged packet is a data packet, its waiting time is

$$T_d = T_r + T_1 + T_1' + T_2$$

Taking expectations we obtain

$$W_d = W_r + W_1 + W_1' + W_2 \tag{2.6}$$

If the time to service one data packet is τ_d, then

$$W_2 = E(n_d)\, \bar{\tau}_d$$
$$W_1' = E(n_v')\, \bar{\tau}_v$$

By Little's formula [24], $E(n_d) = \lambda_d W_d$. We cannot apply Little's formula directly to $E(n_v')$, but we can establish its value by the following argument. Since we know that a data packet waits for an average time W_d and that voice packets arrive at an average rate λ_v, clearly the average number of

voice packets which arrive over the time interval W_d is $E(n_v') = \lambda_v W_d$. A more rigorous derivation of this formula can be found in [22]. Therefore, from (2.6)

$$W_d = W_r + \lambda_v W_v \bar{\tau}_v + \lambda_v W_d \bar{\tau}_v + \lambda_d W_d \bar{\tau}_d$$

or

$$W_d = \frac{W_r + \rho_v W_v}{1 - \rho_v - \rho_d} = \frac{W_r + \rho_v W_v}{1 - \rho}$$

where $\rho_d \triangleq \lambda_d \bar{\tau}_d$ is the link data utilization factor

$\rho \triangleq \rho_v + \rho_d < 1$, is the link aggregate utilization factor.

Substituting for W_v we obtain

$$W_d = \frac{W_r}{(1-\rho)(1-\rho_v)} \tag{2.7}$$

We may rewrite (2.5) and (2.7) as follows:

$$W_v = \frac{W_r C}{C-\rho_v C} = \frac{W_r C}{C-F_v} \tag{2.8}$$

$$W_d = \frac{W_r C^2}{(C-\rho_v C)(C-\rho C)} = \frac{W_r C^2}{(C-F_v)(C-F)} \tag{2.9}$$

where F_v = the link voice traffic, in bits/second,

F = the aggregate link traffic, in bits/second,

C = the link capacity, in bits/second.

For completeness, the expected remaining time to completion of service

of the packet in service when the tagged packet arrived is given by [22]:

$$W_r = \frac{1}{2}[\lambda_v E(\tau_v^2) + \lambda_d E(\tau_d^2)] \tag{2.10}$$

The delay of a voice packet at a link (ℓ,k) is made up of the waiting time and the service time (including queueing and processing delays at node ℓ). The latter is usually independent of the network status. Hence at each node we bother only about the waiting time at the output link. From (2.8) we observe that the expected waiting time of a voice packet is a function of the link voice traffic F_v. This expected waiting time is also affected by the amount of time the voice packet has to wait for a data packet already in service when the voice packet arrived; but this effect is rather small. Because of these facts we define a cost function $B_{ik}(F_{ik}^v)$ as follows: $B_{ik}(F_{ik}^v)$ is the cost of limiting the voice traffic on link (i,k) to F_{ik}^v. For mathematical tractability we assume that $B_{ik}(F_{ik}^v)$ is a convex increasing and twice differentiable function of F_{ik}^v, with a typical plot as shown in Fig.2.5.

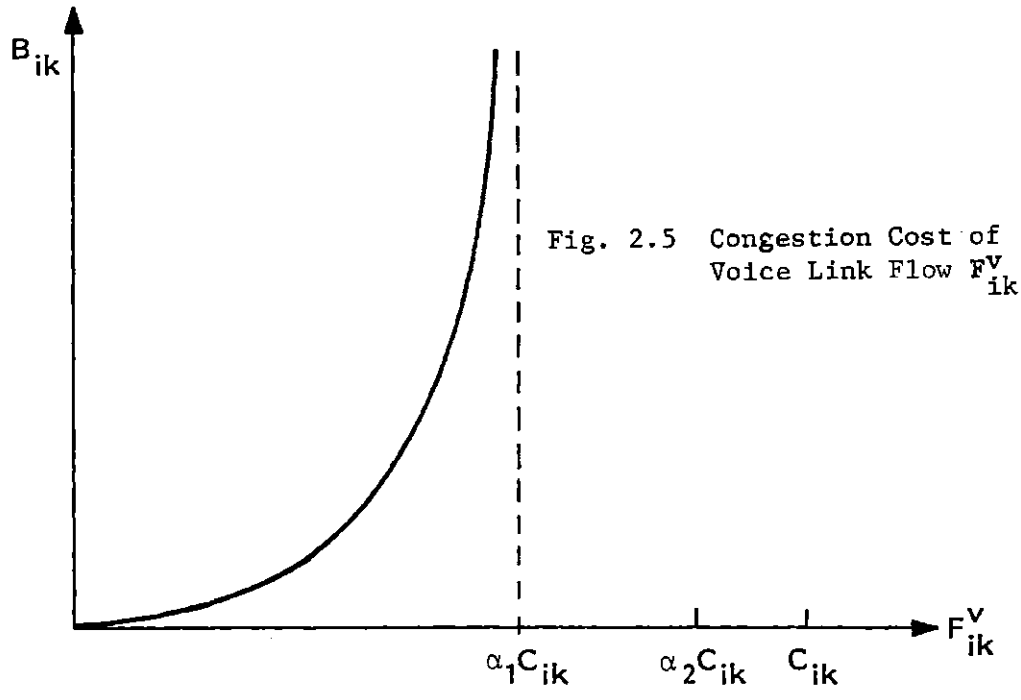

Fig. 2.5 Congestion Cost of Voice Link Flow F_{ik}^v

The reason for the cutoff at $F^v_{ik} = \alpha_1 C_{ik}$ is the following. From (2.8) we observe that as F_v approaches C, the waiting time becomes large. Since voice packet delay is a critical issue, we limit the voice traffic on each link to some predetermined fraction, α_1, of the link capacity. This flow restriction, coupled with the fact that voice traffic has a non-preemptive priority over data traffic, would enable the voice packets to experience less delay than would be the case if the restriction were removed. That is, if we hold the voice traffic below $\alpha_1 C_{ik}$, the link delay would be small.

We could define a cost function for the data traffic in a similar manner. However, we can exploit some characteristics of voice traffic in defining the cost function for the data traffic. First, observe from (2.9) that the expected waiting time of a data packet at the output link is a function of both F_v and F. One major attraction of the integrated voice and data network is the anticipated possibility of exploiting the on-off characteristics of voice traffic to transmit more data when voice traffic is low. A reasonable cost function for data traffic should, therefore, recognize this fact: when $F^v_{ik} = 0$ the data packets should be able to use link (i,k) as if the network were an all-data network. Since data packet delay requirements are not as stringent as those for voice packets, we would permit a higher cutoff point for each link flow. A cutoff is necessary to account for the finite buffer spaces available at the output links.

That data packet delay is more tolerable than voice packet delay means that the marginal cost to a link of a voice packet should be at

least as great as that of a data packet. That is, the cost of congestion for an additional voice packet is greater than that for an additional data packet. When a given voice packet arrives at a node, it holds up the service of all data packets in queue on its arrival. Its delay is only affected by the voice packets which are in queue on its arrival. Data packets affect the voice packet's delay through the possibility that a data packet was receiving service when the voice packet arrived. As we said earlier, this effect is rather small. When a given data packet arrives at a node, its delay is affected not only by all the packets (both voice and data) in queue on its arrival but also by all voice packets which arrive while it is in queue. We then need a model that recognizes this difference in the marginal costs for voice and data conversations.

Now consider the cost function $G_{ik}(F_{ik})$ defined as follows: $G_{ik}(F_{ik})$ is the cost of limiting the aggregate traffic on link (i,k) to F_{ik}. For reason of mathematical tractability we assume that $G_{ik}(F_{ik})$ is a convex increasing and twice differentiable function of F_{ik}, with the typical plot as shown in Fig. 2.6.

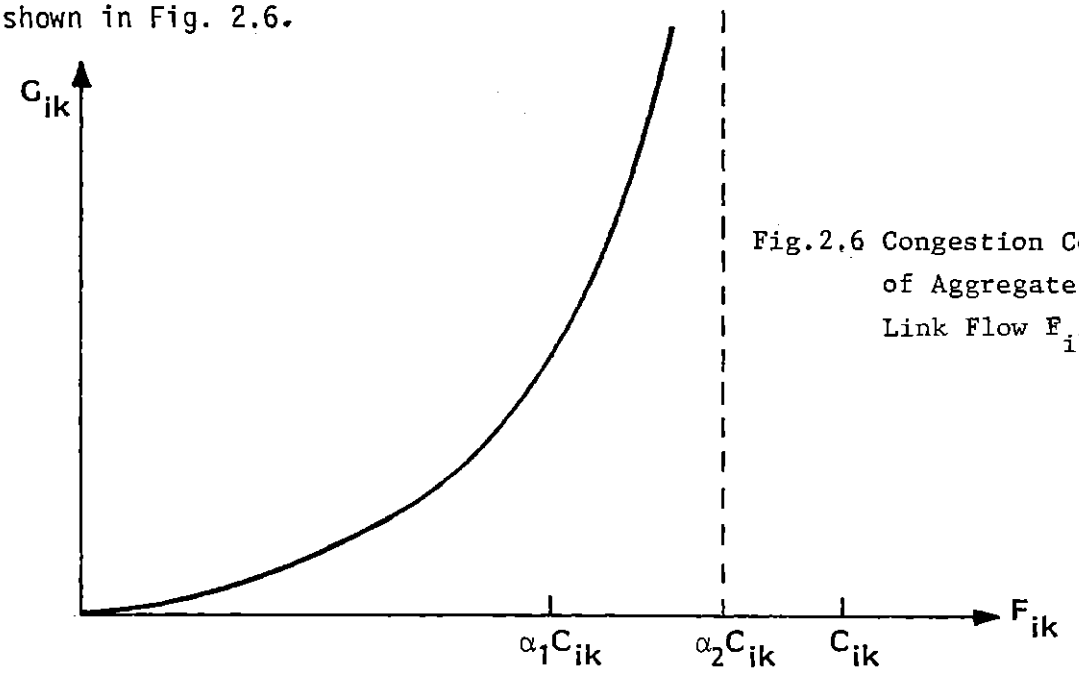

Fig.2.6 Congestion Cost of Aggregate Link Flow F_{ik}

Let the composite function $D_{ik}(F_{ik}, F^v_{ik})$ be defined as follows:

$$D_{ik}(F_{ik}, F^v_{ik}) = B_{ik}(F^v_{ik}) + G_{ik}(F_{ik}) \tag{2.11}$$

That the composite function D_{ik} captures the essence of our design objectives for congestion as listed above can be seen from the following analysis. When $F^v_{ik} = 0$, then from the way we have defined $B_{ik}(F^v_{ik})$ we have that $B_{ik}(0) = 0$ and

$$G_{ik}(F_{ik} | F^v_{ik} = 0) = G_{ik}(F^d_{ik})$$

Thus

$$D_{ik}(F_{ik}, 0) = G_{ik}(F^d_{ik}) \tag{2.12}$$

This is precisely the cost function defined by Golestaani [19] for an all-data network. Also the derivatives

$$\frac{\partial D_{ik}}{\partial F^v_{ik}} = B'_{ik}(F^v_{ik}) + G'_{ik}(F_{ik})$$

$$\frac{\partial D_{ik}}{\partial F_{ik}} = G'_{ik}(F_{ik})$$

[where $B'_{ik}(F^v_{ik}) = \frac{dB_{ik}}{dF^v_{ik}}$, etc.]

are the marginal costs to link (i,k) of voice and data conversations respectively. As can be seen above, the marginal cost of voice to link (i,k) is greater than that of data, as we hoped.

In summary then, our argument is that we need two cost functions that relate to delay (or congestion) in the integrated network. One cost

function deals with the issue of ensuring that we load the network with voice traffic to such a level that would not make the voice packet delay excessive. The other cost function deals with the mix of voice and data traffics. This latter function allows us to exploit the on-off characteristics of voice traffic to transmit more data when voice traffic is low.

2.2.4 Voice Quality Limitation Cost

Having considered the issue of congestion in the network we now deal with the issue of fairness to all voice conversations. We wish to design a network that operates as follows. When a voice conversation is established the network is bound to accept all packets of that conversation as long as the speaker is off-hook. However, the quality of that conversation is not guaranteed to be good at the sink. The quality of different talkspurts of the same conversations may even be different, depending on the status of the network when a talkspurt is generated. As we have mentioned earlier, the quality of any voice conversation is related to the length of the packets of that conversation: the longer the packets, the higher the quality of the conversation.

In order to effect fairness to all voice conversations, then, we make it increasingly costly to degrade the quality of any conversation any further through assigning shorter and shorter packets to that conversation. Let a voice conversation that enters the network at node i and is destined for node j be denoted by voice conversation (i,j). For each voice conversation (i,j) we define a nominal packet length a_{ij}; that is, the maximum length we can assign to packets of conversation (i,j) is a_{ij}. The reasons for this packet length limitation

are as follows. First, above a certain packet length there is no appreciable improvement in the voice quality. Therefore, it would not make sense to assign lengths that would generate more traffic without any extra gain in quality. Secondly, the voice digitizer is limited in the number of bits it can generate over a given time interval. For each voice conversation (i,j) we define a cost function $V_{ij}(\ell_{ij})$ as follows: $V_{ij}(\ell_{ij})$ is the cost of restricting the packet length of voice conversation (i,j) to ℓ_{ij}. For mathematical tractability we assume that $V_{ij}(\ell_{ij})$ is a convex non-increasing and twice differentiable function of ℓ_{ij} with the typical plot as shown in Fig. 2.7. For notational convenience we assume that all voice conversations (i,j) have the same cost function $V_{ij}(\ell_{ij})$.

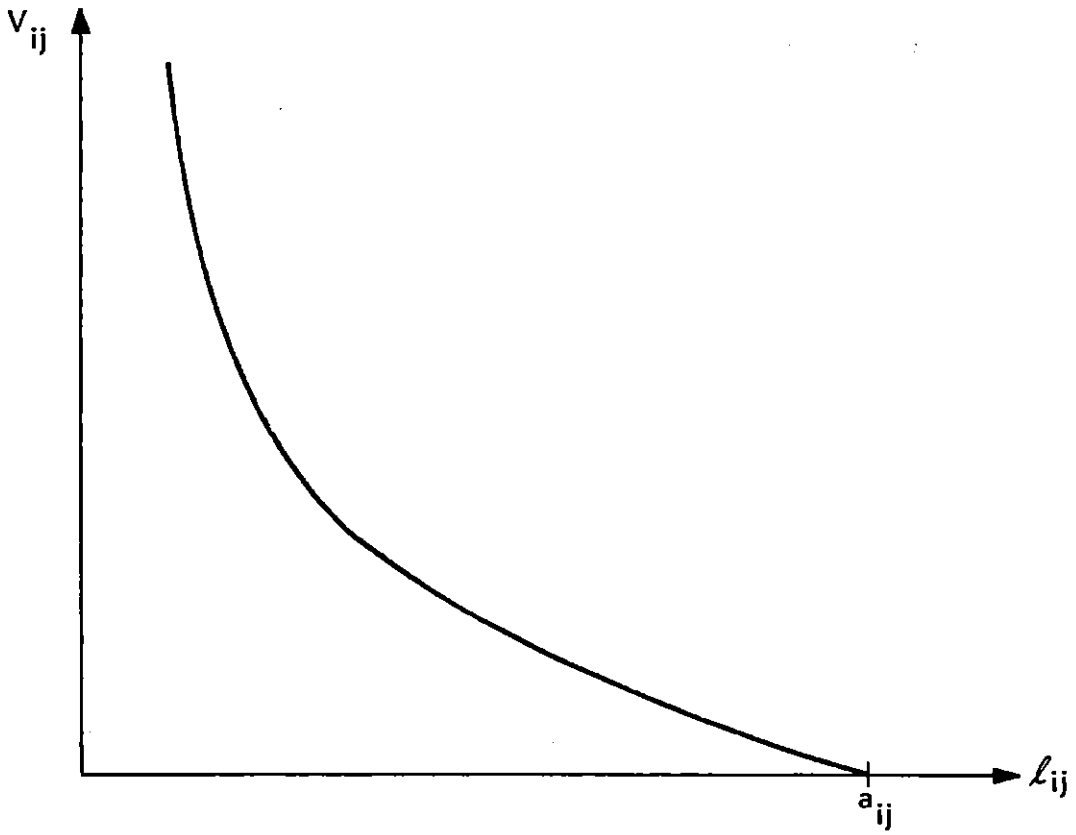

Figure 2.7 Quality Limitation Cost of A Voice Conversation (i,j)

2.2.5 Data Rate Limitation Cost

For data conversations we let the allowable input rates depend on the network loading conditions: when the network is heavily loaded we make the input rates small, and when the network is lightly loaded we permit higher input rates. We thus assume that the lengths of data packets are independent of the network control. In order to effect fairness to all data conversations we make it increasingly costly to cut back any conversation further through assigning it a smaller and smaller input rate. In this regard we define a cost function $E_{ij}(r_{ij})$ as follows: $E_{ij}(r_{ij})$ is the cost of restricting the input rate of data conversation (i,j) to r_{ij}. For reason of mathematical tractability we assume that it is a convex non-increasing and twice differentiable function of r_{ij}, with the typical plot as shown in Fig. 2.8. For simplicity of notation, we take $E_{ij}(r_{ij})$ to be the same for all data conversations (i,j). r_{ij}^d is some desired rate at which data conversation (i,j) would be transmitted if we did not exercise any control over it.

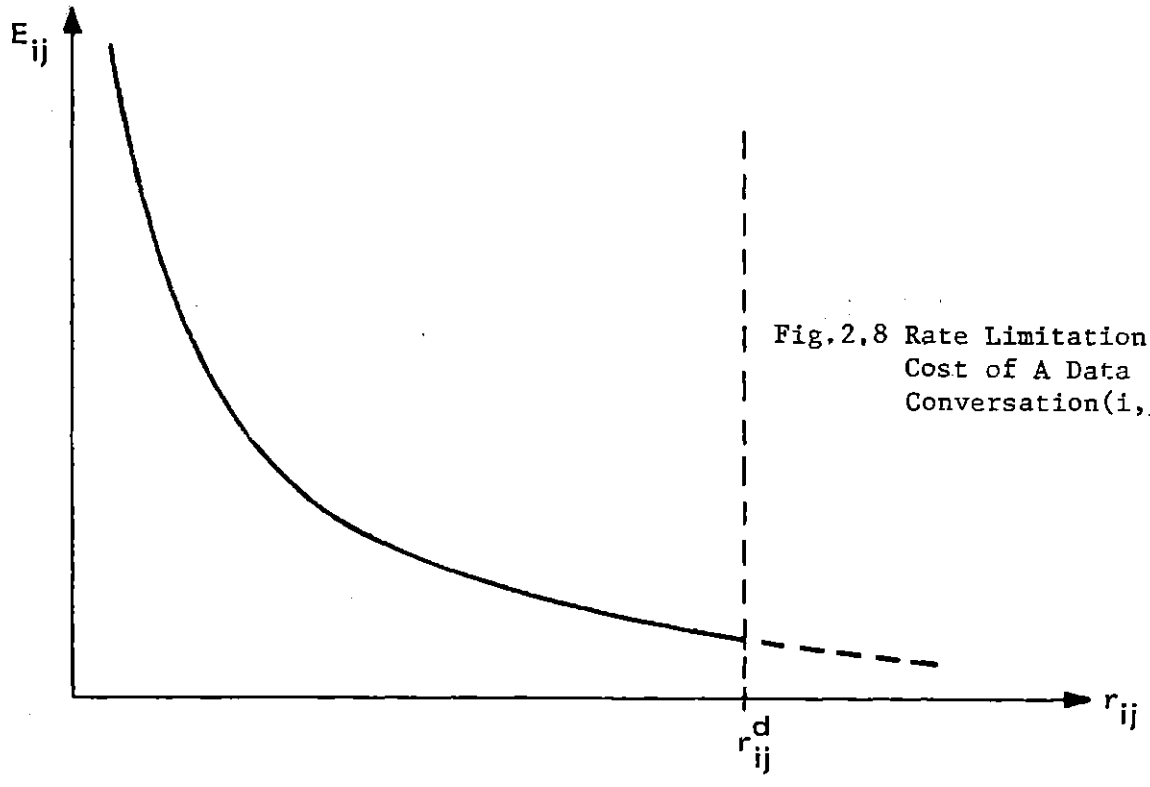

Fig.2.8 Rate Limitation Cost of A Data Conversation(i,j)

2.3 The Problem Statement

Our model formulation is now complete. The objective function of our problem is a sum of four terms:

$$\text{(i)} \quad B_T(F^v) = \sum_{i,k} B_{ik}(F^v_{ik})$$

$$\text{(ii)} \quad G_T(F) = \sum_{i,k} G_{ik}(F_{ik})$$

$$\text{(iii)} \quad V_T(\ell) = \sum_{i,j} \gamma n_{ij} \, V_{ij}(\ell_{ij})$$

$$\text{(iv)} \quad E_T(r) = \sum_{i,j} m_{ij} \, E_{ij}(r_{ij})$$

where γn_{ij} is, as we stated earlier, our estimate of n^T_{ij}, the number of off-hook speakers in talkspurt; and m_{ij} is the number of data conversations (i,j). We can then formulate the problem as follows:

$$\text{Minimize } J = B_T(F^v) + G_T(F) + V_T(\ell) + E_T(r) \tag{2.13}$$

$$\text{subject to} \sum_{k\in O(i)} f^v_{ik}(j) - \sum_{m\in I(i)} f^v_{mi}(j) = \beta\gamma n_{ij} \ell_{ij} \qquad 1 \le i,j \le N \tag{2.14}$$

$$\sum_{k\in O(i)} f^d_{ik}(j) - \sum_{\ell\in I(i)} f^d_{\ell i}(j) = m_{ij} \, r_{ij} \qquad 1 \le i,j \le N \tag{2.15}$$

$$\left.\begin{aligned} f^v_{ik}(j) &\ge 0 \\ f^d_{ik}(j) &\ge 0 \end{aligned}\right\} \quad 1 \le i,j,k \le N \quad i \ne j \tag{2.16}$$

Constraints (2.14) and (2.15) are the so-called continuity (or flow conservation) equations. They state that the total traffic coming into node i and destined for node j is equal to the total traffic going out of node i and destined for node j. Constraint (2.16) states that flows are non-negative. In the next chapter we present a solution to this problem and develop a joint flow control and routing algorithm that sets the voice packet lengths ℓ_{ij} and data input rates r_{ij} to values compatible with the network status, and for routing the voice and data traffics.

CHAPTER III

SOLUTION TO THE PROBLEM

3.1 Introduction

In Chapter II we formulated a model of the integrated voice and data network as a convex optimization problem. In this chapter we present a solution to this problem. We construct a joint flow control and routing algorithm that sets the voice packet lengths and the data input rates, and routes the voice and data traffics, in a manner that attempts to satisfy the optimality conditions we shall derive.

3.2 The solution

Theorem 3.1 Let $u^* = \{\ell_{ij}^*, r_{ij}^*, f_{ik}^{v*}(j), f_{ik}^{d*}(j)\}$ be a feasible point of (2.14) through (2.16). Then u^* minimizes (2.13) if and only if there exist two sets of numbers $\lambda = \{\lambda_{ij}\}$ and $\mu = \{\mu_{ij}\}$, with $\lambda_{jj} = 0$ and $\mu_{jj} = 0$, such that the following Kuhn-Tucker conditions are satisfied.

$$B'_{ik}(F_{ik}^{v*}) + G'_{ik}(F_{ik}^{*}) + \lambda_{kj} \begin{cases} = \lambda_{ij} & \text{if } f_{ik}^{v*}(j) > 0 \\ \geq \lambda_{ij} & \text{if } f_{ik}^{v*}(j) = 0 \end{cases} \tag{3.1}$$

$$G'_{ik}(F_{ik}^{*}) + \mu_{kj} \begin{cases} = \mu_{ij} & \text{if } f_{ik}^{d*}(j) > 0 \\ \geq \mu_{ij} & \text{if } f_{ik}^{d*}(j) = 0 \end{cases} \tag{3.2}$$

$$-\frac{1}{\beta} V'_{ij}(\ell^*_{ij}) \begin{cases} = \lambda_{ij} & \text{if } 0 < \ell^*_{ij} < a_{ij} \\ \leq \lambda_{ij} & \text{if } \ell^*_{ij} = 0 \\ \geq \lambda_{ij} & \text{if } \ell^*_{ij} = a_{ij} \end{cases} \tag{3.3}$$

$$-E'_{ij}(r^*_{ij}) \begin{cases} = \mu_{ij} & \text{if } 0 < r^*_{ij} < r^d_{ij} \\ \leq \mu_{ij} & \text{if } r^*_{ij} = 0 \\ \geq \mu_{ij} & \text{if } r^*_{ij} = r^d_{ij} \end{cases} \tag{3.4}$$

where $B'_{ik}(F^v_{ik})$ denotes the derivative of $B_{ik}(F^v_{ik})$, etc. The proof of this theorem is given in Appendix A.

3.2.1 Definitions. A voice route R^v_{ij} between nodes i and j is a set of links $\{(i,k), (k,\ell),\ldots,(m,j)\}$ which connects nodes i and j and along which voice traffic flows from node i to node j. We define a data route R^d_{ij} in a similar manner.

We can interpret the quantity $B'_{ik}(F^v_{ik}) + G'_{ik}(F_{ik})$ as the marginal cost of voice traffic on link (i,k), or the "voice length" of link (i,k). Similarly, the quantity $G'_{ik}(F_{ik})$ can be interpreted as the marginal cost of data traffic on link (i,k) or the "data length" of link (i,k). Since $\lambda_{jj} = 0$ and $\mu_{jj} = 0$, we can solve for λ_{ij} and μ_{ij} recursively to obtain

$$\lambda_{ij} = \sum_{(\ell,k)\in R^v_{ij}} \left\{ B'_{\ell k}(F^{v*}_{\ell k}) + G'_{\ell k}(F^*_{\ell k}) \right\} \tag{3.5}$$

$$\mu_{ij} = \sum_{(\ell,k)\in R^d_{ij}} G'_{\ell k}(F^*_{\ell k}) \tag{3.6}$$

Then we can interpret λ_{ij} as the marginal voice cost of congestion on a path R^v_{ij} of optimal flow, or the voice length of path R^v_{ij} when flow is optimal. Similarly, we can interpret μ_{ij} as the marginal data cost of congestion on a path R^d_{ij} of optimal flow, or the data length of path R^d_{ij} when flow is optimal. Equations (3.1) and (3.2) then state that all traffic flows along paths of minimum marginal cost, i.e. "shortest" paths. In (3.5) and (3.6) λ_{ij} and μ_{ij} are defined as properties of the optimal flow and are difficult to find without solving the optimization problem. We shall need a form of λ_{ij} and μ_{ij} in our joint flow control and routing algorithm. Therefore, we shall redefine λ_{ij} and μ_{ij} in terms of measurable quantities, as functions of an arbitrary flow as follows:

$$\lambda_{ij} = \underset{R^v_{ij}}{\text{Min}} \sum_{(\ell,k)\in R^v_{ij}} [B'_{\ell k}(F^v_{\ell k}) + G'_{\ell k}(F_{\ell k})] \tag{3.7}$$

$$\mu_{ij} = \underset{R^d_{ij}}{\text{Min}} \sum_{(\ell,k)\in R^d_{ij}} G'_{\ell k}(F_{\ell k}) \tag{3.8}$$

When flow is optimal, the λ_{ij} obtained from (3.5) is equal to that obtained from (3.7); and the μ_{ij}'s obtained from (3.6) and (3.8) are equal.

3.2.2 The Priority Functions [19]

We define the voice priority function, $p_{ij}(\ell_{ij})$, for voice conversation (i,j) as follows:

$$p_{ij}(\ell_{ij}) = -\frac{1}{\beta} V'_{ij}(\ell_{ij}) \tag{3.9}$$

Similarly, we define the data priority function, $q_{ij}(r_{ij})$, for data

conversation (i,j) as

$$q_{ij}(r_{ij}) = - E'_{ij}(r_{ij}) \tag{3.10}$$

Then we can restate the optimality conditions (3.3) and (3.4) as follows:

$$p_{ij}(\ell^*_{ij}) \begin{cases} = \lambda_{ij} & \text{if } 0 < \ell^*_{ij} < a_{ij} \\ \leq \lambda_{ij} & \text{if } \ell^*_{ij} = 0 \\ \geq \lambda_{ij} & \text{if } \ell^*_{ij} = a_{ij} \end{cases} \tag{3.11}$$

$$q_{ij}(r^*_{ij}) \begin{cases} = \mu_{ij} & \text{if } 0 < r^*_{ij} < r^d_{ij} \\ \leq \mu_{ij} & \text{if } r^*_{ij} = 0 \\ \geq \mu_{ij} & \text{if } r^*_{ij} = r^d_{ij} \end{cases} \tag{3.12}$$

Recall that in section 2.2.2 we stated that the solution to the integrated network problem resides in finding an optimum compromise between speech quality and voice delay, and between data input rate and network congestion. The priority functions have an interesting interpretation in this regard. First, the priority functions $p_{ij}(\ell_{ij})$ are the marginal gain in voice quality for an additional voice packet length allocation. What (3.11) states is that optimality occurs when the marginal gain in voice quality is as close as possible to the marginal voice cost of congestion λ_{ij}, subject to $0 < \ell^*_{ij} < a_{ij}$. The equilibrium point (or point of optimal tradeoff) for any voice conversation is attained when its marginal gain in quality equals its marginal cost of congestion. At any other point the marginal cost of congestion of that conversation will be either greater

than or less than its marginal gain in quality. This means that any improvement in quality at the non-optimal point would be obtained at the expense of voice packet delay, and vice versa. The same explanation holds for data conversations. A desirable flow control algorithm would then be one that attempts to equalize these two sets of marginal values for each conversation. In section 3.2.3 we shall exploit the above fact in setting the voice packet lengths and the data input rates. In section 3.4 we shall consider a class of priority functions.

3.2.3 A Joint Flow Control and Routing Algorithm

The joint flow control and routing algorithm we propose performs four different network functions: it effects voice flow control by setting the voice packet lengths to values appropriate for the network conditions; it effects data flow control by setting data input rates to appropriate values; and it routes voice and data traffics along the shortest paths. There are two approaches to defining the algorithm. We call these approaches the "all-at-once protocol" and the "cyclic coordinate protocol".

In the all-at-once protocol, the algorithm performs all the four network functions simultaneously during each iteration. In the cyclic coordinate protocol, the algorithm performs the four network functions in a step-by-step manner, changing one parameter (or coordinate) at a time. Specifically, after an update of the network condition is made, voice flow control is effected; another update is made after a steady state has been reached, and voice routing changes are made. Then after another steady state is reached, another update is made and data flow control is effected. Then again another update is made after another steady state

is reached, and data routing changes are made. The process is repeated starting with another update followed by voice flow control. Thus there are four updates to complete one iteration of the algorithm. The convergence of the cyclic coordinate algorithm appears to be slower than that of the all-at-once algorithm. We shall consider only the all-at-once algorithm here.

Our joint flow control and routing algorithm belongs to the class of algorithms proposed by Bertsekas [2]. This class of algorithms operates in the space of path flows rather than link flows in which our problem is defined. However, the correspondence between a problem defined in the space of link flows and one defined in the space of path flows can be easily established [2].

An iteration of the algorithm starts with each node i computing the shortest voice and data distances from itself to every node j for which node i's conversations are destined. We shall explain how distance computation is carried out later. The metric for distance computation is the marginal cost of congestion on each link summed over all links on the path from node i to node j. Generally there may be two or more paths that qualify for the shortest path for each (i,j) pair. Voice packet lengths and data input rates are increased or decreased according to whether the priority functions are greater than or less than the shortest distances. Routing changes are made as follows. The traffic on each non-shortest path is decreased by an amount proportional to the difference between the length of that path and the length of the shortest path. The traffic on the shortest path is then the total traffic generated by the (i,j) conversations less the traffic routed along the other paths.

If two or more paths qualify for the shortest path, we arbitrarily choose one of them to perform the function described above.

Let the state of node i at iteration n be denoted by

$$u^n = \{\ell_{ij}^n,\ r_{ij}^n,\ f_{ik}^{vn}(j),\ f_{ik}^{dn}(j)\} \tag{3.13}$$

Since there may be more than one shortest path for voice and/or data conversations (i,j), we define the joint flow control and routing algorithm H as follows. Let

U = the set of feasible points of (2.14) through (2.16).

Then the algorithm H is a point-to-set mapping: it maps a point $u \in U$ into a set of points contained in U. Thus for $u^n \in U$, the algorithm yields $H(u^n) \subset U$ from which an arbitrary element u^{n+1} is selected. Thus given an initial point u^0, the algorithm generates sequences through the iteration [25]:

$$u^{n+1} \in H(u^n)\ . \tag{3.14}$$

Let $s_{ij}^{vn} = \beta\, \gamma\, n_{ij}\, \ell_{ij}^n$

= the total expected rate of all the voice conversation entering the network at node i and destined for node j at iteration n,

$s_{ij}^{dn} = m_{ij}\, r_{ij}^n$

= the total expected rate of all data conversations entering the network at node i and destined for node j at iteration n

$A_v = \{(i,j) \mid s_{ij}^{vn} > 0,\ \ i,j = 1,\ldots,N\}$

$A_d = \{(i,j) \mid s_{ij}^{dn} > 0,\ \ i,j = 1,\ldots,N\}$

For each $a = (i,j) \in A_v$, denote s_{ij}^{vn} by $s^{vn}(a)$; and

for each $a = (i,j) \in A_d$, denote s_{ij}^{dn} by $s^{dn}(a)$

P_a = the set of directed paths, with no repeated nodes, originating at node i and terminating at node j

S_p^{vn} = the total voice flow for $a \in A_v$ along path $p \in P_a$ at iteration n

S_p^{dn} = the total data flow for $a \in A_d$ along path $p \in P_a$ at iteration n

Then the relationship between the link flows F_{ik}^{vn} and F_{ik}^{dn} and the path flows is given by

$$F_{ik}^{vn} = \sum_{a \in A_v} \sum_{p \in P_a} \delta_p(i,k)\, S_p^{vn} \tag{3.15}$$

$$F_{ik}^{dn} = \sum_{a \in A_d} \sum_{p \in P_a} \delta_p(i,k)\, S_p^{dn} \tag{3.16}$$

where the incidence term $\delta_p(i,k)$ is defined as follows:

$$\delta_p(i,k) = \begin{cases} 1 & \text{if link } (i,k) \in p \\ 0 & \text{otherwise} \end{cases}$$

Let $d_{ik}^{vn} = B'_{ik}(F_{ik}^{vn}) + G'_{ik}(F_{ik}^{n})$

= the voice length of link (i,k) at iteration n

$d_{ik}^{dn} = G'_{ik}(F_{ik}^{n})$

= the data length of link (i,k) at iteration n.

Then for each $a \in A_v$ and $p \in P_a$ we define the voice length of path p at

only in the fact that we have two additional cost functions for voice traffic.

3.4 A Class of Priority Functions

We have defined the voice and data priority functions as

$$p_m(\ell_m) = -\frac{1}{\beta} V_m'(\ell_m), \qquad m \in A_v$$

$$q_m(r_m) = -E_m'(r_m), \qquad m \in A_d$$

respectively. Golestaani [19] introduced the concept of priority functions in data networks. He also considered a class of priority functions which enables us to prioritize the different conversations according to how severely they are to be cut back when the network loading becomes high. We shall consider this class of priority functions in this section together with a modified form of this class of priority functions for data conversations. Consider priority functions of the form:

$$p_m(\ell_m) = \left(\frac{a_m}{\ell_m}\right)^{\nu_m} \qquad \nu_m > 0 \qquad (3.31)$$

$$q_m(r_m) = \left(\frac{b_m}{r_m}\right)^{\delta_m} \qquad \delta_m > 0 \qquad (3.32)$$

We define ν_m and δ_m as the voice priority index for voice conversation m, and the data priority index for data conversation m, respectively. The quantity a_m is the nominal voice packet length defined earlier. The quantity b_m is the nominal input rate for data conversation m. Note that we do not know r_m^d *a priori*, but we still assign the nominal rate b_m to conversation m. Figure 3.1 is a plot of $p_m(\ell_m)$.

iteration n as

$$d_p^{vn} = \sum_{(\ell,k)\epsilon p} d_{\ell k}^{vn} \tag{3.17}$$

Similarly, for each $a \epsilon A_d$ and $p \epsilon P_a$ we define the data length of path p at iteration n as

$$d_p^{dn} = \sum_{(\ell,k)\epsilon p} d_{\ell k}^{dn} \tag{3.18}$$

Finally, we define the shortest voice and data distances from node i to node j at iteration n as

$$d^{vn}(a) = \underset{p\epsilon P_a}{\text{Min}}\ d_p^{vn} \tag{3.19}$$

$$d^{dn}(a) = \underset{p\epsilon P_a}{\text{Min}}\ d_p^{dn} \tag{3.20}$$

respectively. Many efficient methods exist for finding shortest paths in a network; some of these methods are given in [23]. Then the joint flow control and routing algorithm can be formally given as follows:

A Updating At Iteration n

1. Each node i broadcasts $B'_{ik}(F_{ik}^{vn})$ and $G'_{ik}(F_{ik}^{n})$ for each $k\epsilon O(i)$

2. After receiving the above information from all nodes, each node i computes the path lengths d_p^{vn} and d_p^{dn} for each $p \epsilon P_a$ such that $S_p^{vn} > 0$ and $S_p^{dn} > 0$:

$$d_p^{vn} = \sum_{(\ell,k)\epsilon p: S_p^{vn}>0} \{B'_{\ell k}(F_{\ell k}^{vn}) + G'_{\ell k}(F_{\ell k}^{n})\}$$

$$d_p^{dn} = \sum_{(\ell,k)\epsilon p: S_p^{dn}>0} G'_{\ell k}\,(F^n_{\ell k})$$

3. The minimum voice and data distances between nodes i and j are then computed as follows:

$$d^{vn}(a) = \min_{p\epsilon P_a} d_p^{vn}$$

$$d^{dn}(a) = \min_{p\epsilon P_a} d_p^{dn}$$

Usually there are many paths from node i to node j, and it is not computationally feasible to find the length of every path. This is why in step 2 we require that only the lengths of those paths with $S_p^{vn} > 0$ and $S_p^{dn} > 0$ be computed for voice and data conversations, respectively.

4. For each a = (i,j) node i computes the voice priority function $p_a(\ell_a^n)$ if $S^{vn}(a) > 0$, and the data priority function $q_a(r_a^n)$ if $S^{dn}(a) > 0$.

B. Voice Flow Control

5. Let $Z_a^n = d^{vn}(a) - p_a(\ell_a^n)$ (3.21)

6. $$\ell_a^{n+1} = \begin{cases} \ell_a^n - Z_a^n & \text{if } 0 < \ell_a^n - \eta_\ell Z_a^n < a_{ij} \\ a_{ij} & \text{if } \ell_a^n - \eta_\ell Z_a^n \geq a_{ij} \\ 0 & \text{if } \ell_a^n - \eta_\ell Z_a^n \leq 0 \end{cases} \quad (3.22)$$

where η_ℓ is some positive scale factor, and a_{ij} is the nominal voice packet length defined earlier. What this algorithm does is the following:

(i) if $p_a(\ell_a^n) > d^{vn}(a)$, increase ℓ_a^n, $\ell_a^n < a_{ij}$

(ii) if $p_a(\ell_a^n) < d^{vn}(a)$, decrease ℓ_a^n, $\ell_a^n > 0$

(iv) if $p_a(\ell_a^n) = d^{vn}(a)$, leave ℓ_a^n as it is.

That is, the algorithm attempts to equalize the two marginal values: $p_a(\ell_a^n)$ and $d^{vn}(a)$.

C. Data Flow Control

7. Let $x_a^n = d^{dn}(a) - q_a(r_a^n)$ (3.23)

8. $$r_a^{n+1} = \begin{cases} r_a^n - \eta_r x_a^n & \text{if } 0 < r_a^n - \eta_r x_a^n < r_a^d \\ r_a^d & \text{if } r_a^n - \eta_r x_a^n \geq r_a^d \\ 0 & \text{if } r_a^n - \eta_r x_a^n \leq 0 \end{cases} \quad (3.24)$$

where η_r is some positive scale factor, and r_a^d is the desired input rate defined earlier. The operation of this algorithm is similar to that of B.

D. Voice Traffic Routing

9. Let $e_p^{vn} = d_p^{vn} - d^{vn}(a)$ (3.25)

$$\Delta_p^{vn} = \min [S_p^{vn}, \eta_v e_p^{vn}] \quad (3.26)$$

where η_v is some positive scale factor, and $S_p^{vn} > 0$.

10. $$S_p^{v(n+1)} \begin{cases} S_p^{vn} - \Delta_p^{vn}, & p \neq p_v^* \\ S^{v(n+1)}(a) - \sum_{p \neq p_v^*} S_p^{v(n+1)}, & p = p_v^* \end{cases} \quad (3.27)$$

where p_v^* is a voice shortest path. What this algorithm does is the following. It measures the deviation e_p^{vn} of the length of each path $p \varepsilon P_a$ from the shortest distance. Then it decreases the flow on path p by an amount proportional to this deviation, and routes the remaining voice traffic from voice conversation $a \varepsilon A_v$ along a shortest path. Note that $S^{v(n+1)}(a)$, the total voice traffic entering the network at node i and destined for node j at iteration (n+1), is determined by the voice flow control algorithm.

E. Data Traffic Routing

11. Let $$e_p^{dn} = d_p^{dn} - d^{dn}(a) \tag{3.28}$$

$$\Delta_p^{dn} = \min [S_p^{dn}, \eta_d e_p^{dn}] \tag{3.29}$$

12. $$S_p^{d(n+1)} = \begin{cases} S_p^{dn} - \Delta_p^{dn} & p \neq p_a^* \\ S^{d(n+1)}(a) - \sum_{p \neq p_d}^{*} S_p^{d(n+1)}, & p = p_d^* \end{cases} \tag{3.30}$$

where η_d is some positive scale factor, $s_p^{dn} > 0$, and p_d^* is a data shortest path. The operation of this algorithm is similar to that of D. Note that we assume that at each node, voice packets would be transmitted before the data packets, in the spirit of the non-preemptive priority voice packets have over data packets. The convergence properties of the joint flow control and routing algorithm are described by the following

Theorem 3.2 Let u^o be any feasible point of (2.14) through (2.16), and let $J(u^o) \leq J_o$. Then for every positive number J_o, there exist scale

factors η_ℓ, η_r, η_v and η_d such that

$$\lim_{n\to 0} J(u^n) = \min_u J(u),$$

where

$$u^n \in H(u^{n-1}), \quad \text{for all } n \geq 1$$

This theorem states that there exist η_ℓ, η_r, η_v and η_d such that if we start at any feasible point u^o and apply the point-to-set mapping H repeatedly, the set of points $\{u^n\}$ we obtain converges to a point which minimizes J. The proof of this theorem is given in Appendix B.

3.3 Discussion

Our routing algorithm is similar to the new ARPANET routing algorithm [26]: both are shortest path routing algorithms. However, each uses a different metric for distance computation. In [26] the distances are absolute delays averaged over 10 seconds, but in our work the distances are the marginal delays on the paths. Also in our work we gradually decrease the traffic on the paths that are no longer the shortest paths at each iteration, thereby ensuring that the algorithm converges. But in [26], at each iteration an old path that is no longer the shortest path is entirely cancelled and all traffic is shifted to the new shortest path. This type of algorithm is not likely to converge [2].

Our routing algorithm is also similar to the algorithm proposed by Segall [29]. Both are shortest path algorithms and use the marginal delay for distance computation. Also in both algorithms the computation is distributed. However, the two algorithms differ in a number of ways.

First, in [29] the algorithm is defined in the space of link flows. Secondly, the update information in [29] is propagated upstream, starting from the destination node as follows: Each node, upon receiving the update information from its downstream neighbor, performs the necessary computation and forwards the result to its upstream neighbors who in turn perform similar operations and pass the information further upstream, until the source node is reached. This mode of update information transmission has the potential for generating loops that could prevent the upstream nodes from ever receiving the update information. Specifically, if there is a link (k,ℓ) such that $f_{k\ell}(j) > 0$ and $f_{\ell k}(j) > 0$, then during an update node k needs the value of the distance $\lambda_{\ell j}$ to compute λ_{kj} and node ℓ needs the value of λ_{kj} to compute $\lambda_{\ell j}$. Then a deadlock ensues and no further upstream forwarding of the update information can take place. This necessitates the introduction, as was done in [29], of the concept of blocking which ensures that loop-freedom is maintained. In our algorithm the update information is broadcast and so we do not worry about loop-freedom.

Our routing algorithm is also similar to the algorithm proposed by Gallager [12]. However, that algorithm is defined in the space of fractions of link flows rather than path flows in which our algorithm is defined. Also, because of the way the update information is transmitted (same as for [29]),the issue of blocking is introduced to maintain loop-freedom. The original idea of blocking as a means of maintaining loop-freedom was formulated in [12].

Finally, our flow control algorithm is similar to that of Golestaani [19]. This is because the two models are similar, differing

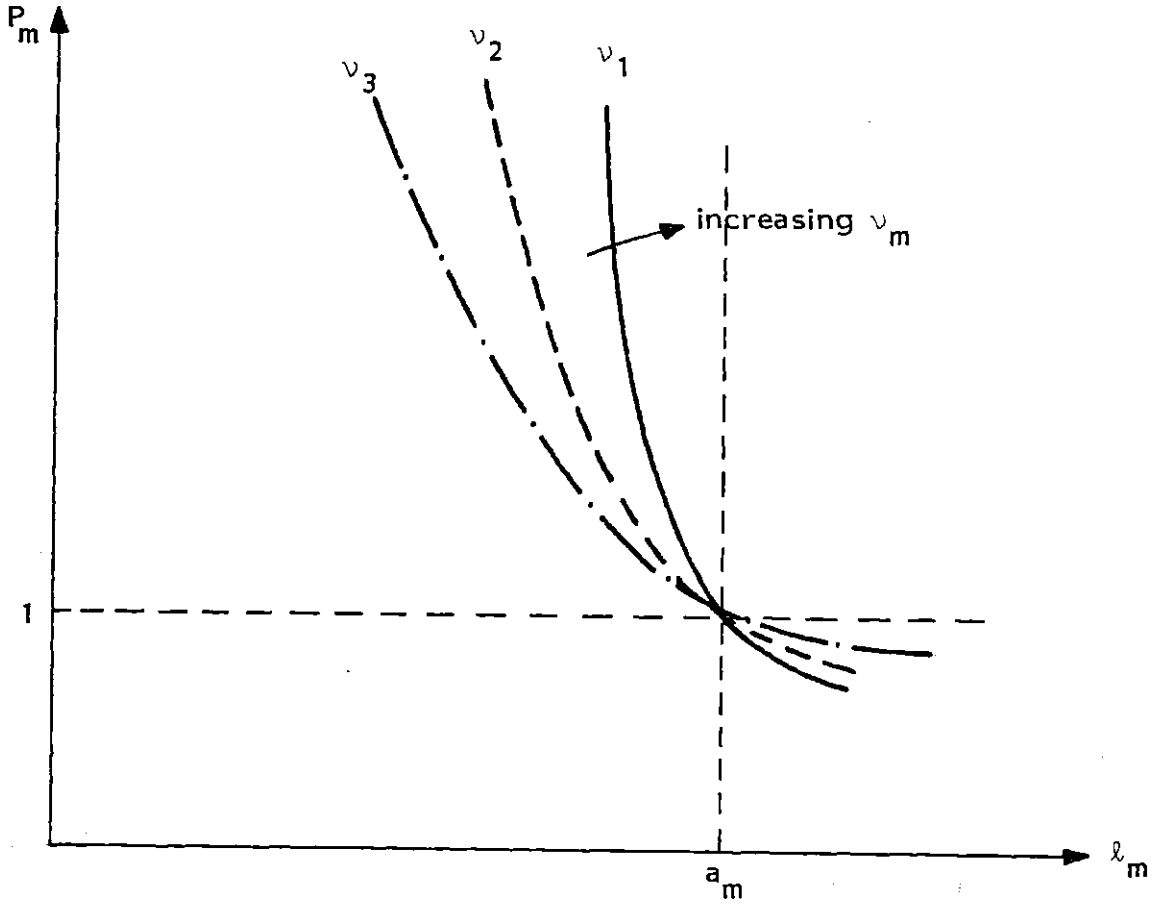

Figure 3.1 Voice Priority Function

The priority indices relate to the extent to which conversations get cut back when the network loading becomes high. Conversations with small priority indices are cut back more severely than those with higher priority indices. As can be seen in Fig. 3.1, packet lengths of voice conversations with high ν_m remain fairly constant over a wide range of values of $p_m(\ell_m)$. But packet lengths of voice conversations with lower values of ν_m vary wildly as $p_m(\ell_m)$ changes. When all conversations have high priority indices it is no longer true that packet lengths remain fairly constant. This is because when congestion sets in all packets would have to be shortened to reduce network traffic. And since all the

conversations have the same order of importance, they would all then experience severe cutback in packet length.

As we noted in chapter II, the nominal voice packet length a_m relates to the length of packets required to produce good quality speech. However, for fixed values of $p_m(\ell_m)$ and ν_m, the higher the value of a_m the higher is the value of ℓ_m. Since the amount of traffic generated by a voice packet is directly proportional to its length, care must be taken not to choose such values of a_m that could indirectly induce congestion. Similar arguments hold for the choice of b_m.

We mentioned earlier that we do not know r_m^d a priori but still assign a nominal rate b_m. It may turn out that $r_m^d > b_m$ and an assigned rate $r_m > b_m$. Observe, however, that when $r_m > b_m$ some inversion takes place: For any given value of $q_m(r_m) < 1$, the data conversations with lower priority indices get higher rates than those with higher priority indices. If this is the desired objective of the network designer, then the definition of $q_m(r_m)$ given in (3.32) is enough. However, it may be desirable to allow the higher priority conversations (i.e. those with high priority indices) to have higher input rates at all times. For this purpose the definition of $q_m(r_m)$ could be modified as follows:

$$q_m(r_m) = \begin{cases} \left(\frac{b_m}{r_m}\right)^{\delta_m} & 0 < r_m \leq b_m \\ \left(\frac{b_m}{r_m}\right)^{1/\delta_m} & r_m \geq b_m \end{cases} \qquad (3.33)$$

The modified priority function is sketched in Fig. 3.2.

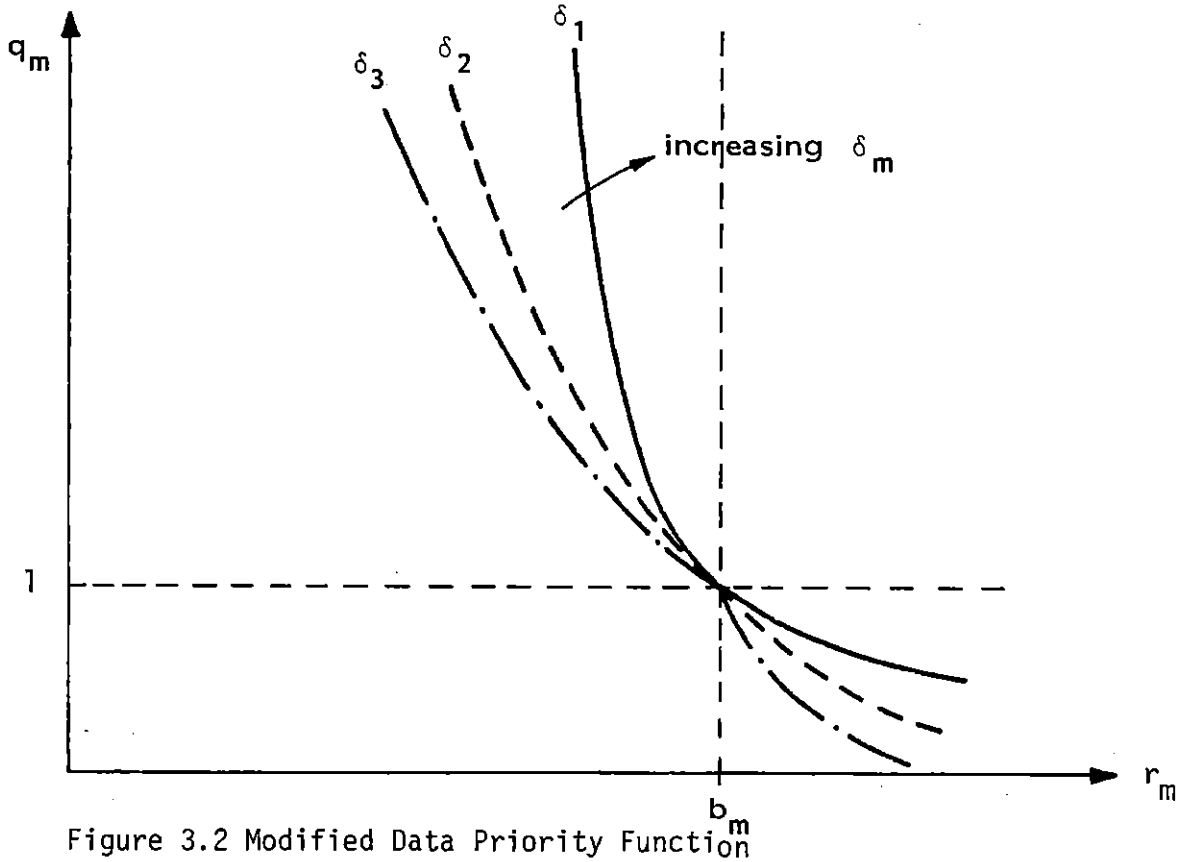

Figure 3.2 Modified Data Priority Function

For the class of priority functions defined by (3.31) and (3.32) we have that for voice conversations $m \in A_v$, when $\ell_m = a_m$, $p_m(\ell_m) = 1$. When this happens we would not like the network to be congested, and we would not like to unnecessarily restrict the flows to some arbitrarily small values since this is one way of avoiding congestion. Recall that the voice flow control algorithm operates as follows:

(i) if $p_m(\ell_m^n) > d^{vn}(m)$, it increases ℓ_m^n if $\ell_m^n < a_m$;

(ii) if $p_m(\ell_m^n) < d^{vn}(m)$, it decreases ℓ_m^n if $\ell_m^n > 0$;

and (iii) if $p_m(\ell_m^n) = d^{vn}(m)$, it leaves ℓ_m^n unchanged.

If our network is properly designed (and we have no reason to believe it is not!) , it will operate at the equilibrium point most of the time. That is, $p_m(\ell_m^n) = d^{vn}(m)$ most of the time. We would like to know what happens when $p_m(\ell_m^n) = 1$ and the network operates at the equilibrium point. Recall that

$$d_{ik}^{vn} = B'_{ik}(F_{ik}^{vn}) + G'_{ik}(F_{ik}^{n}), \qquad \text{and}$$

$$d^{vn}(m) = \min_{p \in P_m} \sum_{(i,k) \in p} d_{ik}^{vn}$$

Assume that all links $(i,k) \in p$ have the same flow, and hence the same value of d_{ik}^{vn} ; and let σ_m be the number of links on the path $p \in P_m$. Then we require that at the equilibrium point when $p_m(\ell_m^n) = 1$, $\sigma_m\, d_{ik}^{vn} = 1$. That is, $d_{ik}^{vn} = 1/\sigma_m$. Thus when the network operates at the equilibrium point with $\ell_m^n = a_m$, the voice length of each link $(i,k) \in p$ is determined by the number of links on the path p. Furthermore, as the number of links σ_m increases, the voice length d_{ik}^{vn} decreases. For example, when $\sigma_m = 2$, $d_{ik}^{vn} = 0.5$; and when $\sigma_m = 5$, $d_{ik}^{vn} = 0.2$. This means that for a multi-hop conversation m, the condition that $p_m(\ell_m) = 1$ at the equilibrium point implies that the traffic on its path be very light. The above argument for voice conversations hold for data conversations. In sum then, when we choose a class of priority functions $p_m(\ell_m)$, we indirectly define the nature of the class of congestion cost functions that are compatible with it. This is so because the optimality conditions demand that there be a proper scaling between the congestion cost functions and the priority functions.

CHAPTER IV

EXTENSIONS TO THE BASIC ALGORITHM

4.1 Introduction

The functions of the basic algorithm, the joint flow control and routing algorithm, are the setting of the voice packet lengths and data input rates to values compatible with the network utilization, and the routing of the voice and data traffics. In this chapter we consider a group of protocols for extending the functions of the algorithm. In particular, we consider a quasi-static protocol that is used to reduce the likelihood of discarding many voice packets which may arrive late at the smoothing buffers due to high network loading. We also consider protocols for dealing with congestion on a more dynamic basis than the joint flow control and routing algorithm can deal with. Finally, we consider the place of the cut-through routing protocol, proposed by Kermani and Kleinrock [20], in the integrated network.

Despite the flow control we practice, the network can still get congested. This can be caused by several factors including the following. The joint flow control and routing algorithm is intended for quasi-static applications where the network input statistics change very slowly. However, in practice the network has varying input statistics. Between updates, because of the inherent burstiness of data conversations, some data conversations may decrease their desired rates to values less than the assigned rates, while others may terminate their conversations entirely. Some inactive data conversations, and voice conversations in silence may become active. If voice packet lengths and data input rates are assigned on the basis of low network utilization, then the new active sources may generate such additional traffic that could drive the network into con-

capable of producing an acceptable speech quality, but the quality is lower than would be the case if the packet was received with all segments in place.

The difference between our proposed scheme and that in [3] is that we first practise flow control before practising congestion control. In [3] no flow control is practised. Also we have no need for a mechamism for informing the source node of the dropped segment because the flow control algorithm will take care of that by setting the voice packet lengths to compatible values at the next iteration.

4.3 The Variable Voice Packet Delivery Rate Strategy

We require that voice packets be delivered to the sink from the smoothing buffers at the same rate β at which the voice digitizer emits them into the network. However, when the network gets congested, voice packets arrive very late and so the smoothing buffers would be delivering packets to the sink faster than new packets arrive at the buffers. In this case many of these packets would be discarded because of late arrival.

In order to reduce the number of voice packets that are discarded, we would have to vary the rate at which the smoothing buffers deliver these packets to the sink in response to the network conditions. Specifically, when the network loading becomes high and packets arrive later than before, we decrease the packet delivery to a value β'. When the network loading becomes light again, we increase β' back to β. In order to fully understand the effect of changing the packet delivery rate to β', consider the delay of a packet that enters the network at time τ and leaves the network (i.e. is delivered to the sink) at time d; see Figure 4.1.

dropped so that it would generate only the more important packets. We emphasize at this point that this scheme is equivalent to using one packet that has different segments of different orders of importance. When the network loading becomes high, the less important segments of the packet could be discarded anywhere in the network. Flow control is practised by setting the voice packet length to a value compatible with the network conditions before sending it into the network. This corresponds to dropping the less important segments of the packet at the source.

We would like to make a distinction between flow control and congestion control, even though the two terms are often used interchangeably in the literature. For our purpose we define flow control as any mechanism that regulates the entry of traffic into the network in order to avoid network overload. In other words, flow control is applied at the entry points of the traffic. Thus, voice packet length setting and data input rate setting are examples of flow control. By congestion control we mean any mechanism that alleviates high network loading by acting on the traffic already inside the network. Thus by this definition, the embedded coding scheme of Bially et al. [3] is a congestion control scheme and not a flow control scheme.

In addition to effecting flow control we propose to practise congestion control in the integrated network. Our approach is similar to the embedded coding scheme [3]. After setting the voice packet lengths, each voice packet would be encoded into different segments of different orders of importance. As in [3], when the network loading becomes high, the less important segments of the packet could be discarded anywhere in the network. The more important segments reaching the destination are considered

gestion. Therefore, the joint flow control and routing algorithm, while reducing the likelihood of network congestion, is not a perfect safeguard against congestion because it will not be able to track the changing input statistics. To do this, the algorithm would have to be updated more frequently, which requires more updating protocols and hence a reduction in the effective link capacities available for the voice and data traffics.

4.2 A Direct Method of Congestion Control

There are differing views on what to do when the network gets congested. Some researchers propose that voice packets be dropped when network congestion occurs; and if congestion persists, then some data packets be dropped next [11]. The reason for this is that it is easier to drop voice packets because the speaker can easily repeat the message if he does not get a response in time. Moreover, since the voice packets are likely to arrive late at their destinations and hence be discarded, it is better to get rid of them earlier rather than waste the network resources in transmitting them. The problem with dropping voice packets as a means of fighting congestion is that if the algorithm that determines when voice packets are to be dropped is not dynamic enough, it would not stop dropping packets after relieving congestion.

The embedded coding scheme of Bially et al. [3] controls congestion in a dynamic manner, and is adaptable to our model. Recall that in that scheme voice is encoded into the more important packets and the less important packets. The less important packets may be dropped anywhere in the network on a dynamic basis, according to the network loading conditions. The source node would have to be informed of any packets that have been

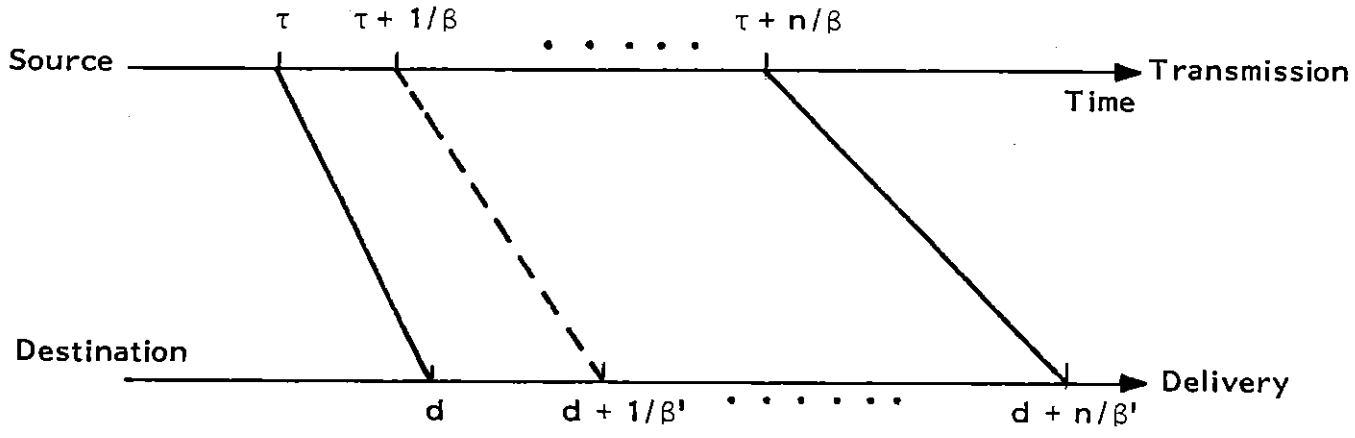

Figure 4.1 Additional Delay Caused By Change In β

As can be seen from the figure, a packet that enters the network at time $\tau + n/\beta$ is expected to be at the smoothing buffers no later than $d + n/\beta'$, sustaining a maximum delay of

$$(d + \frac{n}{\beta'}) - (\tau + n/\beta) = (d-\tau) + \frac{n(\beta-\beta')}{\beta\beta'}$$

without being thrown away. If $\beta = \beta'$, then this packet would be expected to suffer a maximum delay of $(d-\tau)$. But with $\beta \neq \beta'$, we permit the packet to suffer an additional delay of $\frac{n(\beta-\beta')}{\beta\beta'}$ without being thrown away when it arrives. Note that this additional delay increases with time. Thus the packets that enter the network long after we have switched the delivery rate from β to β' have less chances of late arrival than those that enter immediately after the switch. Since the variation of voice delivery rate affects the voice quality, the value of $n(\beta-\beta')/\beta\beta'$ should be such that is a compromise between the probability of discarding fewer packets and the amount of quality degradation the sink can tolerate. We can define a priori how large we would make the additional delay

$n(\beta-\beta')/\beta\beta'$. When this predetermined value is reached, we revert to β whether congestion still persists or not.

4.4 A Congestion Avoidance Protocol

The joint flow control and routing algorithm uses γn_{ij} as its estimate of n_{ij}^T in setting the voice packet lengths. This, as we have noted, is due to the fact that it is difficult to use the instantaneous values of n_{ij}^T on a global basis. This difficulty results in occasional incidence of heavy network loading, which in turn necessitates such actions as the shortening of voice packets and the varying of the voice packet delivery rate, as we recommended.

It is possible to use the n_{ij}^T on a local basis, especially in a manner that would enable each source node i, which sends traffic to destination node j, to permit some maximum variation of the (i,j) voice traffic from the expected rate. The total expected rate of all voice (i,j) conversations is $\beta\gamma n_{ij}\ell_{ij}^*$, where ℓ_{ij}^* is the value of the voice packet length determined by the algorithm. If at any time n_{ij}^T greatly exceeds γn_{ij}, then the network may be tending to congestion because of the increased rate over and above the expected rate. We propose that the total instantaneous voice rate for the (i,j) conversations be limited to a value no greater than $\beta[\gamma n_{ij} + \varepsilon(n_{ij})]\ell_{ij}^*$, for some choice of $\varepsilon(n_{ij})$ to be explained later. Specifically, let $\ell_{ij}(t)$ be the instantaneous value of ℓ_{ij}. At each instant that voice packets are to be constructed, note i makes the following decision:

$$\ell_{ij}(t) = \begin{cases} \ell_{ij}^*, \text{if } n_{ij}^T \leq \gamma n_{ij} + \varepsilon(n_{ij}) \\ \dfrac{[\gamma n_{ij} + \varepsilon(n_{ij})]\ell_{ij}^*}{n_{ij}^T}, \text{ otherwise} \end{cases} \qquad (4.1)$$

This strategy can be explained as follows. That we are dealing with average rates means that we recognize that fluctuations about the mean value exist. But since voice traffic is delay-sensitive, we cannot tolerate large upswings in voice traffic. Therefore, we limit the maximum upswing to $\beta[\gamma n_{ij} + \varepsilon(n_{ij})]\ell^*_{ij}$. When traffic exceeds this value, we degrade the voice quality of all the (i,j) voice conservations through assigning shorter packets that would force the traffic down to where we can tolerate. The choice of $\varepsilon(n_{ij})$, the maximum amount by which n^T_{ij} exceeds γn_{ij} before action is taken, depends on the value of n_{ij}. Specifically, since the standard deviation of n^T_{ij} is proportional to $\sqrt{n_{ij}}$, we permit the instantaneous values of n^T_{ij} to exceed some standard deviations (or fraction of a standard deviation) before we take any action on the traffic. Thus functions of the form

$$\varepsilon(n_{ij}) = k\sqrt{n_{ij}}$$

where $k > 0$ is a constant, are appropriate for this purpose.

4.5 A Strategy For Admitting New Conversations

As the joint flow control and routing algorithm shows, we make small routing changes at each iteration in order to ensure that the algorithm converges. Therefore, any decision to admit new conversations into the network should be made in such a way that large changes are not made in the flow pattern. For all conversations in the network let

ℓ_o = the minimum acceptable voice packet length

r_o = the minimum acceptable data input rate

4.7 Implementation of Data Flow Control

We have discussed how flow control can be effected in the integrated voice and data network: Voice packet lengths and data input rates are set in response to the network conditions. For voice conversations we assume that we can find a voice digitizer which generates packets of appropriate lengths at a constant rate of β per second whenever a speaker is in talkspurt [3]. Thus if such a hardware is found, voice flow control can be easily implemented. For data conversations the input rates are controlled and the packet lengths are assumed to be independent of network control. Assume the expected data packet length is Γ bits and the optimal input rate of data conversation m is r_m^*. Then in order to implement this rate of r_m^* bits/sec., the packets of this conversation would be sent into the network at an average rate of r_m^*/Γ per second. As we mentioned earlier, that we have effected flow control does not guarantee a congestion-free operation. Often statistical fluctuations arise that cause the flow-controlled network to be congested. Thus a strict adherence to sending packets into the network at an average rate of r_m^*/Γ may not be a good strategy, especially when congestion sets in. What is required then is a scheme that will not only restrict the input rates to the optimal values r_m^* but also respond dynamically to the network conditions between updates.

A common approach to effecting flow control in a data network is the so-called window strategy [6,15]. In this strategy each source node i keeps the number of unacknowledged (or outstanding) packets in the network, of each conversation destined for node j, below a given number w_{ij} called the window size (or window width). Thus a new packet cannot be sent into the network if the number of outstanding packets of

$$L = \{0, \ell_0, \ell_1, \ldots, a_{ij}\} \tag{4.2}$$

where $0 < \ell_0 < \ell_1 < \ldots < a_{ij}$. Let the "raw packet length" obtained from the flow control algorithm be ℓ_a; and let ℓ_a satisfy the condition $\ell_{k-1} < \ell_a < \ell_k$, where $\ell_{k-1}, \ell_k \in L$. Then we are required to set ℓ_a to some value $\hat{\ell}_a \in L$.

The problem with this type of extra constraint is that it could upset the convergence of our algorithm. One way to get around this problem is to divide the voice (i,j) conversations arbitrarily into two groups, A and B. Then we use the following assignment rule:

$$\hat{\ell}_a = \begin{cases} \ell_k & \text{for group A} \\ \ell_{k-1} & \text{for group B} \end{cases} \tag{4.3}$$

where the dividing line for group membership should be drawn in such a way as to make the net traffic remain close to the level given by the flow control algorithm. That is,

$$\gamma n_a \ell_a \approx |A|\ell_k + |B|\ell_{k-1} \tag{4.4}$$

where $|X|$ is the cardinality of set X. In this way we avoid the overload caused by letting all conversations be assigned $\hat{\ell}_a = \ell_k$, and the under-utilization caused by assigning $\hat{\ell}_a = \ell_{k-1}$ to all conversations.

conversation (i,j) at that time is equal to w_{ij}. This is a dynamic scheme because when the network is heavily loaded, packet acknowledgements (ACKs) arrive late and hence the frequency of packet entry into the network decreases. On the other hand, when the network loading is light, ACKs arrive faster and the frequency of packet entry into the network increases.

One question we may ask at this point is: Given an optimal input rate r_m^*, can we set a window size w_m such that, by ensuring that the number of outstanding data packets in the network is no greater than w_m, we force the data conversation m to transmit at rate r_m^*? It turns out that under certain conditions, which we shall give shortly, the answer to this question is yes. Golestaani [19] shows that if for each active conversation (i,j) (i.e. one that is not silent) there are packets waiting to enter the network so that as soon as a new ACK arrives at node i a new packet enters the network, there is a unique correspondence between the set of outstanding packets in the network and the set of input rates of these conversations. The implication of this statement is the following. Since the window strategy is a dynamic scheme, we can practise it as an indirect enforcement of the flow control algorithm for those conversations that meet the necessary conditions.

The motivations for practising the window strategy in a data network carry over to the integrated network. However, the window strategy cannot be applied to voice conversations. This is not only because we set the voice packets to the desired lengths and strictly send them into the network at a constant rate of β per second but also because the assigned window size may be less than the number of voice packets generated during a talkspurt. This would lead to loss of continuity in the conver-

old conversations have packet lengths that are very close to the minimum acceptable value ℓ_o. Therefore, we must ensure that both the delay performance and quality of all old a conversations are of accepable standards (which can be set a priori) before we admit any new a conversation. And the reason for restricting the admission to at most one voice conversation per source-destination pair per period is to make small changes to voice traffic at a time as much as possible. The same arguments hold for the data conversations.

An indication of how to choose $\bar{\ell}_a$ can be seen from Figure 4.2. As the figure shows, $\bar{\ell}_a$ is close to the nominal packet length a_{ij}.

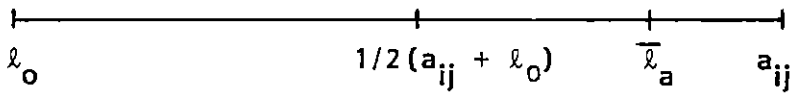

Figure 4.2 Choice of Threshold Packet Length $\bar{\ell}_a$

Note that the newly admitted voice conversation would be assigned the current value of packet length and routed along the shortest path. Similarly, for data conversations, the new a conversation would be assigned the current input rate of the a conversations. If the rates are translated into window sizes (see section 4.7), then the new data conversation would be assigned the current window size, and routed along the shorest path.

4.6 Constrained Voice Packet Lengths

As mentioned in the previous section, assume we have a minimum acceptable voice packet length ℓ_o. Furthermore, let ℓ_o belong to a set L of admissible voice packet lengths:

That is, if a voice conversation is assigned a packet length less than ℓ_o, it opts out of the network rather than transmit at the assigned rate. A similar situation holds for r_o. For each voice conversation a, let

$\overline{\ell}_a$ = some threshold packet length (whose meaning we shall explain shortly)

$d_o^v(a)$ = some threshold voice path length

$d_{p_*}^{vn}(a)$ = the path length of the longest path that voice conversations a are using at iteration n.

Similarly, for each data conversation a, let

$\overline{r}_a$ = some threshold input rate

$d_o^d(a)$ = some threshold data path length

$d_{p_*}^{dn}(a)$ = the path length of the longest path that data conversations a are using at iteration n.

Then the admission policy we propose is the following:

1. If $d_{p_*}^{vn}(a) \leq d_o^v(a)$ and $\ell_a^n \geq \overline{\ell}_a$ at iteration n, then and only then will node i admit one new voice (i,j) conversation, where a = (i,j), during period n.
2. If $d_{p_*}^{dn}(a) \leq d_o^d(a)$ and $r_a^n \geq \overline{r}_a$ at iteration n, then and only then will node i admit one new data a conversation during period n.

This admission policy can be explained as follows. We would not admit any new voice (i,j) conversation when the old (i,j) conversations (i.e. those already in the network) have a delay greater than some threshold $d_o^v(a)$. Also we would not admit any new voice (i,j) conversations when the

$q_{m\ell}$ = the fraction of conversation m on link ℓ. For unifilar flow, such as in virtual line-switched systems, $q_{m\ell}$ is either 1 or 0, depending on whether $\ell \varepsilon R_m^d$ or not.

Then from Little's formula [24] the number of packets of conversation m outstanding in the network at any time is given by

$$u_m = \frac{r_m}{\Gamma} \left\{ \alpha_m + \sum_{\ell \varepsilon R_m^d} q_{m\ell}\, t_\ell\, (F_\ell,\, F_\ell^v) \right\} \tag{4.4}$$

For those conversations with $r_m^* < r_m^d$, (4.4) offers a nice way to set the window size. By setting w_m to

$$w_m = \frac{r_m^*}{\Gamma} \left\{ \alpha_m + \sum_{\ell \varepsilon R_m^d} q_{m\ell}\, t_\ell (F_\ell,\, F_\ell^v) \right\} \tag{4.5}$$

we force conversation m to buffer the remaining data so that whenever an ACK arrives there is a packet waiting to be sent into the network. However, for those conversations with $r_m^* > r_m^d$, the allocated window size $w_m > u_m$; that is, the allocated widow size is greater than the number of outstanding packets in the network at any time. For such conversations, arriving packets would tend to enter the network earlier than those of the previous category because there is a great likelihood that at any time the number of packets outstanding is less than the window size.

To summarize, the window strategy enables us to partition the data conversations into two groups: those with $r_m^* < r_m^d$ and for whom $w_m = u_m$, and those with $r_m^* > r_m^d$ and for whom $w_m > u_m$. In the former group packets are likely to be buffered in wait for ACK arrival because the window is likely to be closed (i.e. the number of outstanding packets equals the

sation, an undesirable feature in any voice conversation.

In deriving the relationship between r_m and w_m Golestaani [19] defined a function $t_\ell(F_\ell)$, which is the average delay per packet on link ℓ including processing and queueing delays at the input to link ℓ. He assumed that $t_\ell(F_\ell)$ is a convex increasing and differentiable function of F_ℓ, the aggregate flow on link ℓ. In our model the link delay of a data packet is not only dependent on the aggregate link flow F_ℓ but also on what proportion of that link flow is voice traffic. For example, consider a link flow of one unit. If the voice traffic is 0.3 units, then the data packet delay would be smaller than the case where the voice traffic is 0.7 units. In the latter case a data packet is likely to wait a long time as newly arriving voice packets receive service before it. Thus in our analysis of the window strategy for the integrated network we define $t_\ell(F_\ell, F_\ell^v)$ to be an increasing and differentiable function of both F_ℓ and F_ℓ^v, the voice traffic on link ℓ. The effect of the presence of voice packets on the data packet delay is to modulate the link delay according to what proportion of the link traffic is voice traffic. The complete relationship between r_m and w_m can be obtained as follows. Let

r_m = the input rate of data conversation m

Γ = the expected data packet length

w_m = the window size of conversation m

α_m = the average delay of ACKs for conversation m. α_m is assumed to be independent of the network conditions if the ACK traffic has priority over other traffics.

window size) when they arrive. In the latter group packets are likely to enter the network as they arrive because the window is likely to be open when they arrive.

If $r_m^* < r_m^d$ for all m, we show below that when the window size is set by (4.4) there is a unique correspondence between r_m^* and w_m [19]. First, we note that

$$F_\ell = \sum_m q_{m\ell}\, r_m + F_\ell^v$$

Therefore

$$\Gamma \frac{\partial w_m}{\partial r_k} = \begin{cases} \alpha_m + \sum\limits_{\ell \in R_m^d} q_{m\ell}\, t_\ell + r_m \sum\limits_{\ell \in R_m^d} q_{m\ell}^2 \dfrac{\partial t_\ell}{\partial F_\ell}, & k = m \\ r_m \sum\limits_{\ell \in R_m^d} q_{m\ell}\, q_{k\ell} \dfrac{\partial t_\ell}{\partial F_\ell}, & k \neq m \end{cases} \qquad (4.6)$$

Let T be a diagonal matrix with elements T_{mm} given by

$$T_{mm} = \alpha_m + \sum_{\ell \in R_m^d} q_{m\ell}\, t_\ell\, (F_\ell, F_\ell^v)$$

Q = a matrix with elements $q_{m\ell}$

R = a diagonal matrix with entries $R_{mm} = r_m$

M = a diagonal matrix with entries $M_{\ell\ell} = \dfrac{\partial t_\ell}{\partial F_\ell}$

P = a matrix with entries $P_{mk} = \Gamma \dfrac{\partial W_m}{\partial r_k}$

Then (4.6) can be written as

$$P = T + RQMQ^T \tag{4.7}$$

Post-multiplying by R we obtain

$$PR = TR + RQMQ^TR^T \qquad (\text{since } R^T = R)$$

If we assume that $r_m > 0$ for all m, then since T, R, and M are positive diagonal matrices, PR is a positive definite matrix, and hence is invertible. Since $(PR)^{-1} = R^{-1} P^{-1}$, and since R^{-1} exists, then P^{-1} exists. This means that incremental changes in window sizes lead to uniquely defined changes in data input rates.

In the way window strategies are generally used in practice, the window size of each conversation remains fixed throughout the duraton of the conversation. However, the unique correspondence between r_m and w_m suggests that w_m could be adjusted in response to the network conditions. Specifically, when $\frac{\partial J^*}{\partial r_m} > 0$ we need to decrease w_m; and when $\frac{\partial J^*}{\partial r_m} < 0$ we need to increase w_m [13]. One problem is how to effect this window size adjustment. A possible method is to change w_m by an amount corresponding to the desired change in r_m [14]; namely

$$\Delta w_m = \frac{\Delta r_m}{\Gamma} \left\{ \alpha_m + \sum_{\ell \in R_m^d} q_{m\ell}\, t_\ell\, (F_\ell, F_\ell^v) \right\} \tag{4.8}$$

The problem with this approach can be seen in the following example.† Consider a simple network consisting of one link and two nodes n_1 and n_2.

† This example is due to Professor Gallager

Two sources transmit at rates r_1 and r_2 from node n_1 as shown in Figure 4.3. If we assume that $F^v = 0$, then

$$w_1 = \frac{r_1}{\Gamma} \quad t(r_1 + r_2)$$

$$w_2 = \frac{r_2}{\Gamma} \quad t(r_1 + r_2) \qquad \text{(i)}$$

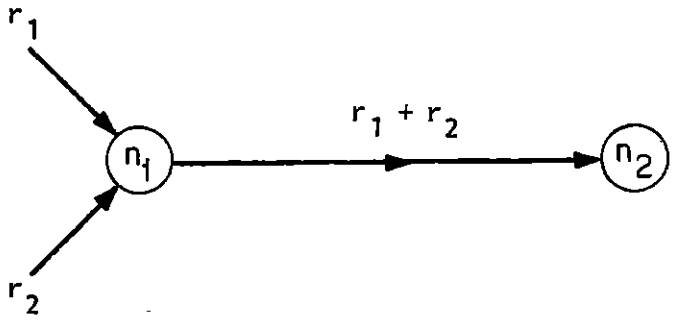

Figure 4.3

To first order

$$\Delta w_1 = \frac{\Delta r_1}{\Gamma} \, t + \frac{r_1 t'}{\Gamma} \, [\Delta r_1 + \Delta r_2] \qquad \text{(ii)}$$

$$\Delta w_2 = \frac{\Delta r_2}{\Gamma} \, t + \frac{r_2 t'}{\Gamma} \, [\Delta r_1 + \Delta r_2] \qquad \text{(iii)}$$

Assume that $r_1 = 0$, $r_2 > 0$; and $\frac{\partial J^*}{\partial r_1} = -1$, $\frac{\partial J^*}{\partial r_2} = -2$

so that $\Delta w_1 = \frac{1}{2} \Delta w_2$. Then from (ii) and (iii) we have that

$$\Delta w_1 = \frac{\Delta r_1 t}{\Gamma} \qquad \text{(iv)}$$

$$\Delta w_2 = \frac{\Delta r_2}{\Gamma} \, [t + r_2 t'] + \frac{r_2 t'}{\Gamma} \, \Delta r_1 \qquad \text{(v)}$$

From (iv) $\Delta r_1 = \frac{\Gamma \Delta w_1}{t} = \frac{\Gamma \Delta w_2}{2t}$

since $\Delta w_1 = \frac{1}{2} \Delta w_2$. Therefore, from (v) we obtain

$$\Delta w_2 (2t - r_2 t') = \frac{2t(t + r_2 t')}{\Gamma} \Delta r_2$$

Or

$$\Delta w_2 = \frac{2t(t + r_2 t')}{\Gamma(2t - r_2 t')} \Delta r_2$$

If we assume $r_2 t' \gg t$, then $r_2 t' > 2t$ and

$$\Delta w_2 = -k\ t\ \Delta r_2 , \quad \text{when } k > 0$$

That is, as we increase the window size we decrease the rate. Thus this method is good for adjusting the window size of inactive conversations but not so for active conversations, when it is applied simultaneously to the two groups. Increase in the window size of the former is accompanied by an increase in the input rate; but in the latter it is accompanied by a decrease in the rate.

Consider the following approach. Let

$$\Delta w_m = -\eta r_m \frac{\partial J^*}{\partial r_m} \tag{4.9}$$

where η is some positive scale factor. As pointed out in [13], this method of window size adjustment has some nice properties. When the window size changes by Δw_m then to first order the change in J^* is

$$\Delta J^* = \sum_m \frac{\partial J^*}{\partial w_m} \Delta w_m$$

$$= -\eta \sum_m \frac{\partial J^*}{\partial w_m} r_m \frac{\partial J^*}{\partial r_m} \quad , \quad \text{from (4.9)}$$

$$= -\eta \sum_m \frac{\partial J^*}{\partial w_m} r_m \frac{\partial J^*}{\partial w_m} \frac{\partial w_m}{\partial r_m}$$

$$= -\frac{\eta}{\Gamma} (\nabla_w \cdot J^*) [TR + RQMQ^T R] (\nabla_w \cdot J^*)^T$$

$$= -\frac{\eta}{\Gamma} (\nabla_w \cdot J^*) PR (\nabla_w \cdot J^*)^T \qquad (4.10)$$

where $\nabla_w \cdot J^*$ is the gradient of J^* with respect to w. Since PR is positive definite, $\Delta J^* < 0$ if $J^* \neq 0$. Thus this method leads to a descent algorithm.

Note, however, that this method cannot work for inactive conversations (whose $r_m = 0$). And since the condition for PR to be positive definite is that all conversations have $w_m = u_m$, or equivalently $r_m^* < r_m^d$, this method is also not applicable to conversations with $r_m^* > r_m^d$. We can then make window size adjustments in the following manner: For inactive conversations which may demand a larger window size when they become active,

$$\Delta w_m = \frac{\Delta r_m}{\Gamma} \left\{ \alpha_m + \sum_{\ell \in R_m^d} q_{m\ell} \, t_\ell \, (F_\ell, F_\ell^v) \right\}$$

where Δr_m is their desired change in rate. And for active sources whose allocated rates $r_m^* < r_m^d$, use

$$\Delta w_m = -\eta r_m \frac{\partial J^*}{\partial r_m}$$

Generally the window size obtained from (4.5) is not an integer. In practice, however, fractional window sizes cannot be easily implemented. One task then is to construct integer window sizes $\hat{w}_m$ given raw values w_m. It has been suggested [19] that the parameter α_m be adjusted in order that the value of w_m in (4.5) is an integer. Thus if an ACK encounters a delay α_m, it is further delayed by an amount $\alpha_m' - \alpha_m$, where α_m' is the value of α_m that makes (4.5) an integer. The manipulations of the ACK delay in the manner we have described above to achieve integer window sizes means rounding w_m off to the next higher integer. This in turn means permitting more traffic into the network. This is thus likely to upset the convergence of our flow control algorithm. To avert this potential problem we make the following proposal. For each source-destination pair (i,j), node i will partition the data conversations into two groups, A and B. Members of group A will be the most important data conversations while members of group B will be the less important conversations. Members of group A will get the window size allocation $\hat{w}_m = \lceil w_m \rceil$, and members of group B will get the window size $\hat{w}_m = \lfloor w_m \rfloor$; where the partition is performed in such a way that

$$m_{ij}\, w_m \simeq |A|\, \lceil w_m \rceil + |B|\, \lfloor w_m \rfloor$$

where $\lceil x \rceil$ = the smallest integer greater than or equal to x;

$\lfloor x \rfloor$ = the largest integer less than or equal to x.

The exact demarcation line for group membership would depend on where the above approximate equality is best achieved.

4.8 Cut-Through Routing of Voice Conversations

We have emphasized the need to route the voice conversations fast enough for them to be available at the destination node when they are required for delivery to the sinks. We have also noted that voice conversations are not particularly sensitive to errors. These two factors combine to make the cut through routing [20] a good strategy for the voice conversations. The cut-through routing protocol was originally defined for data networks and works as follows. When a packet arrives at node i and an output link (i,k) is selected for it, then after receiving the packet header the packet transmission starts immediately without link (i,k) waiting to receive the entire packet and, therefore, without checking for errors, if link (i,k) is free. Only if link (i,k) is busy would it be necessary to buffer an arriving packet, as in normal packet switching operation. In this way the delay associated with buffering to receive the entire packet and check for errors in front of an idle link is eliminated. A cut is said to have been made if the packet successfully goes through the link without waiting to receive the packet before transmission starts.

The aim of this section is to point out the possible gains that could be obtained from applying the cut-through routing protocol to voice conversations in the integrated network. We will not go into detailed analysis of the scheme; the interested reader is referred to [20]. We will merely make a somewhat qualitative comparison between the cut-through routing and the normal packet-switched operation. We distinguish between two types of cuts: a perfect cut and a partial cut. A perfect cut is the type we described above; where a packet arrives at a link which is idle and

"cuts through" without having to wait. When a packet arrives at an output link that is busy, it waits until current service is completed whence it receives service if there is no other packet in queue. If the remaining time to completion of the service of the current packet is less than the time to receive the entire packet and there is no other packet in queue, then the transmission of the packet can start while part of the packet is still being received. This type of cut-through is called a partial cut since some waiting is involved, but the waiting time is less than the time to receive the entire packet. This virtual waiting time may also be equal to the time to receive the entire packet if the current service ends just as the last bit of the packet is received. In [20] only perfect cut-through routing is considered.

We present an approximate analysis of the cut-through routing to show the type of saving in packet delay that can be derived from practising the routing strategy. Assume there is negligible propagation delay on each link and let

s_{ik} = the total service time of a tagged packet on link (i,k)

Then s_{ik} is made up of two parts:

$$s_{ik} = w_{ik} + \tau_{ik}$$

where w_{ik} = the waiting time of the packet on link (i,k)

τ_{ik} = the transmission time of the packet on link (i,k)

Without cut-through routing, each packet encounters both a waiting time and a transmission time on each link: Even when no other packet is in service, the tagged packet must be completely received before transmission

can start. Thus the total system time of the tagged packet when cut-through routing is not practised is

$$s_p = \sum_{(i,k)\varepsilon p} (w_{ik} + \tau_{ik}) \tag{4.11}$$

where p is the path of the tagged packet. With cut-through routing, for links (i,k), $(k,\ell)\,\varepsilon\, p$, the waiting time at link (k,ℓ) starts when transmission starts at link (i,k). Thus except at the output link of the destination node where the entire packet may possibly be completely received before being delivered to the sink, the transmission time at any intermediate link is absorbed by the waiting time at the successor link on the path. Therefore, the system time of the tagged packet is

$$s_p = \sum_{(i,k)\varepsilon p} w_{ik} + \tau_f \tag{4.12}$$

where τ_f = the transmission time at the final link. From (4.11) and (4.12) we see that the system time is reduced by the transmission times on all but the final output link. Note that in practice the values of the w_{ik} used in (4.11) may not be equal to those used in (4.12) because of the fact that perfect cuts might be made in some links. Under such conditions the reduction in delay would be greater than we stated above.

The tradeoff in the cut-through routing strategy is between delay and errors. A packet that makes a cut at each link could arrive at the destination with such a serious error that would necessitate its being discarded. In this situation, we would have wasted the network resources in transmitting that packet. Without cut-through routing, such a packet

would have been discarded earlier, if the error proved to be uncorrectable, without the network resources being wasted on the packet. However, there are certain errors that are tolerable in voice communication. This means that we can practice cut-through routing of voice traffic in the integrated network. We hope that most of the errors we would encounter in the network would be the tolerable type. Under this condition then the cut-through routing scheme would provide some assistance in making the voice packets meet the stringent delay requirements of the integrated network.

CHAPTER V

A CLASS OF INTEGRATED NETWORK MODELS

Our network model belongs to a broader class of models for integrating n different types of traffics that have different degrees of delay sensitivity onto one network. In this type of network problem it is very necessary to ensure that the network is not heavily loaded with traffic from the very delay-sensitive traffic type. Assume we index the traffic types in a decreasing order of delay sensitivity. That is, traffic type 1 is the most delay-sensitive, followed by traffic type 2, and so on with traffic type n the least delay-sensitive. Then we assign the highest (non-preemptive) priority to traffic type 1, followed by traffic type 2, etc. The reasons for practising non-preemptive priority are the same as those advanced in chapter I; namely, to avoid the confusion that could arise if the part of an item that had been serviced before a preemption occurs is interpreted as network errors, and to avoid using the network resources to transmit something that would later be discarded. Let

F_{ik}^{j} = the expected total traffic on link(i,k) of traffic type j, j = 1,...,n.

$$F_{ik}^{1} = F_{ik}^{1}$$

$$F_{ik}^{2} = F_{ik}^{1} + F_{ik}^{2}$$

$$F_{ik}^{3} = F_{ik}^{1} + F_{ik}^{2} + F_{ik}^{3}$$

$$F_{ik}^{4} = F_{ik}^{1} + F_{ik}^{2} + F_{ik}^{3} + F_{ik}^{4}$$

$$\vdots$$

$$F_{ik}^{n-1} = \sum_{j=1}^{n-1} F_{ik}^{j}$$

$$F_{ik} = \sum_{j=1}^{n} F_{ik}^{j} = \text{the aggregate traffic on link } (i,k)$$

We define the cost functions $B_{ik}^{j}(F_{ik}^{j})$ and $G_{ik}(F_{ik})$ as follows:

$B_{ik}^{j}(F_{ik}^{j})$ = the congestion cost to link (i,k) of limiting the traffic from traffic types 1 through j to F_{ik}^{j}, j = 1,2, ...n-1.

$G_{ik}(F_{ik})$ = the congestion cost to link (i,k) of limiting the aggregate traffic to F_{ik}.

For ease of analysis we assume that these cost functions are convex increasing and twice differentiable functions of their respective arguments. For the purpose of ensuring that the network is not heavily loaded with the very delay-sensitive traffics, we require that B_{ik}^{1} cuts off earlier than B_{ik}^{2}, which in turn cuts off earlier than B_{ik}^{3}, and so on with G_{ik} cutting off the last but before the link capacity C_{ik}. By defining appropriate rate limitation functions as we did in Chapter II, we ensure that no traffic type is completely excluded from using the network. For the purpose of illustrating how the congestion cost functions cut off, we present typical plots of these functions on the same set of axes in Figure 5.1. In the plot, traffic types 1 and 2 are the very delay-sensitive types, while types 3 through n are less delay-sensitive.

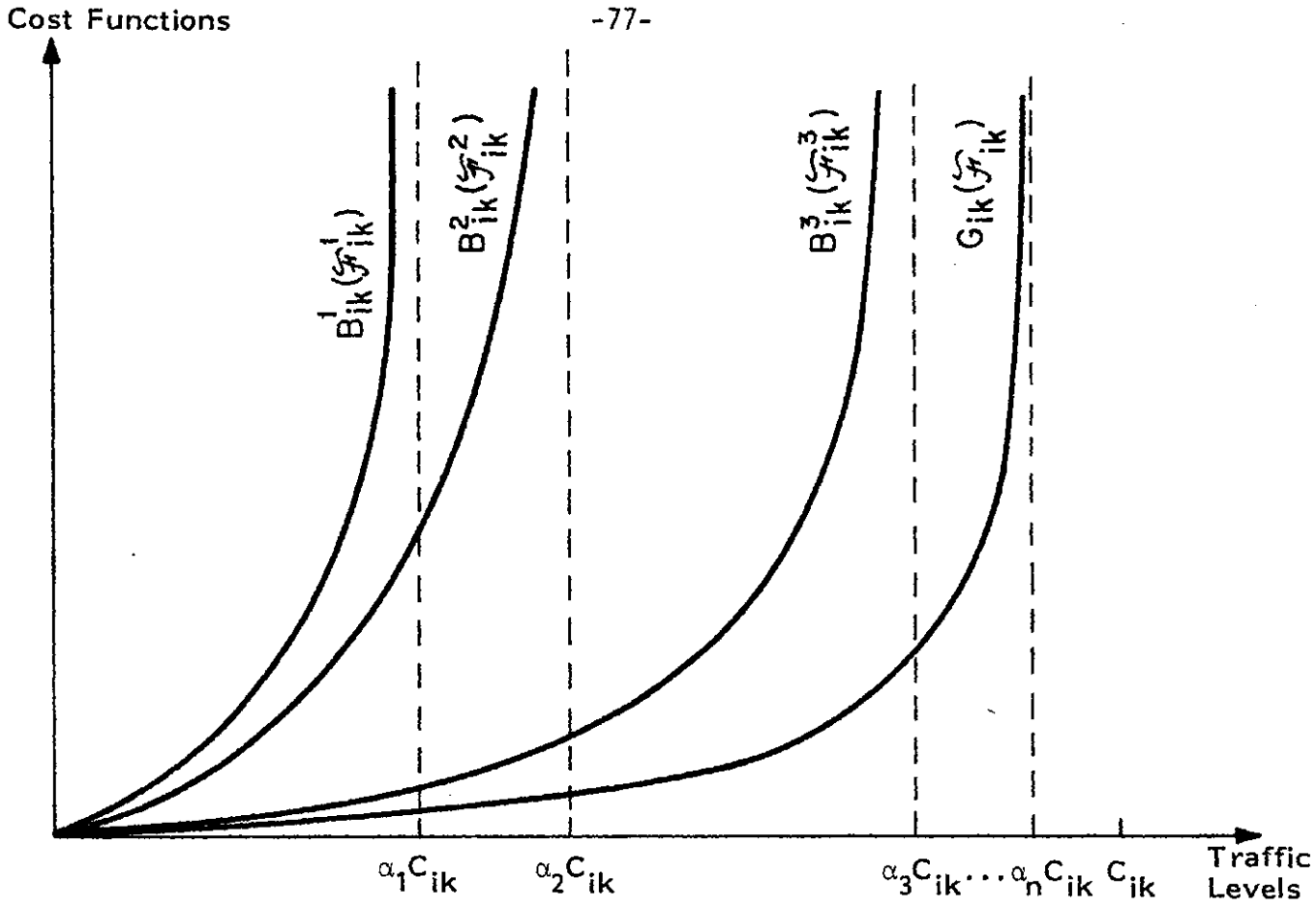

Figure 5.1 Congestion Costs of Link Traffics

A more general communication network application of this generalized model than the type we consider in this thesis is the following. We can partition all communication network traffic into three different classes [9]:

Class I : characterized by long messages that require continuous real-time delivery; e.g. voice facsimile, video.

Class II: characterized by short discrete messages that require near real-time delivery; e.g. interactive data.

Class III: characterized by long messages that require neither continuity nor immediate delivery; e.g. bulk data.

To integrate these three traffic types onto one network, we have

to recognize the need for classes I and II traffics to suffer less delay than class III traffic. This necessitates assigning the highest (non-preemptive) priority to class I traffic, followed by class II traffic, and class III traffic is the low priority traffic. Let

$F^1_{ik} = F^1_{ik}$ = the expected total class I traffic on link (i,k)

F^2_{ik} = the expected total class II traffic on link (i,k)

F^3_{ik} = the expected total class III traffic on link (i,k)

$F^2_{ik} = F^1_{ik} + F^2_{ik}$

$F_{ik} = F^1_{ik} + F^2_{ik} + F^3_{ik}$

Then applying our argument on the nature of the congestion cost functions we define $B^1_{ik}(F^1_{ik})$, $B^2_{ik}(F^2_{ik})$ and $G_{ik}(F_{ik})$ as shown in Figure 5.2.

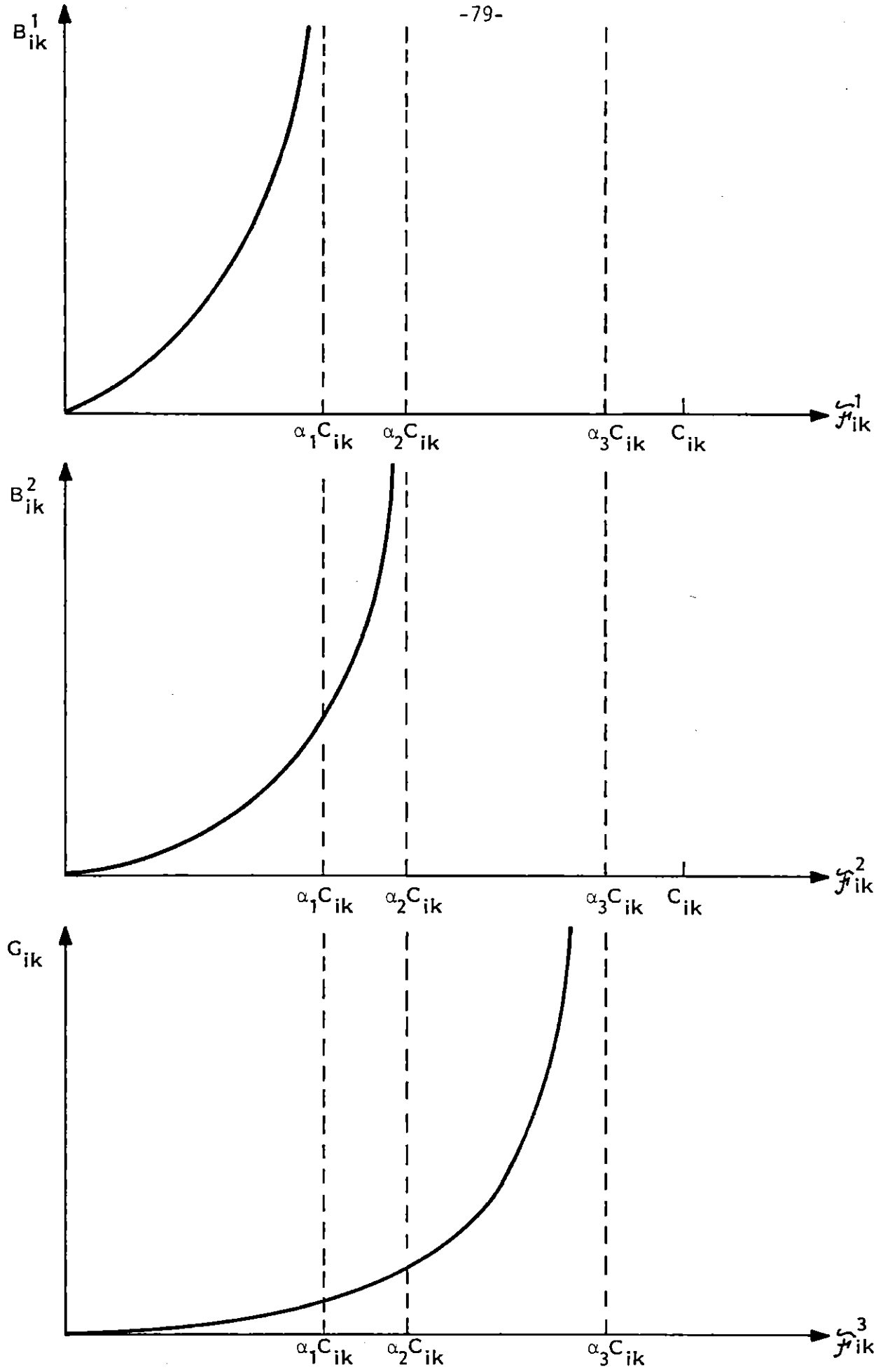

Figure 5.2 Congestion Cost of Link Traffic For A 3-Commodity Network

APPENDIX A

In this appendix we give the proof of theorem 3.1 which deals with the necessary and sufficient conditions for optimality of the solution to (2.13) subject to (2.14) through (2.16). We prove the sufficiency part of the theorem here; the proof of the necessity part of the theorem is given in Appendix B.

Lemma A.1 For any feasible point $P^* = \{f_{ik}^{v*}(j),\ f_{ik}^{d*}(j),\ \ell_{ij}^*,\ r_{ij}^*\}$ which satisfies the conditions of theorem 3.1, and any other feasible point $P = \{f_{ik}^{v}(j),\ f_{ik}^{d}(j),\ \ell_{ij},\ r_{ij}\}$ the following inequalities hold:

$$\text{(i)} \quad \sum_{i,j,k} [B_{ik}'(F_{ik}^{v*}) + G_{ik}'(F_{ik}^*)]\, f_{ik}^{v}(j) \geq \sum_{i,j} \beta^{\gamma n_{ij}}\, \ell_{ij}\, \lambda_{ij}$$

$$\text{(ii)} \quad \sum_{i,j,k} G_{ik}'(F_{ik}^*)\, f_{ik}^{d}(j) \geq \sum_{i,j} m_{ij}\, r_{ij}\, \mu_{ij}$$

with equality if $P = P^*$

Proof From (3.1) we have that

$$B_{ik}'(F_{ik}^{v*}) + G_{ik}'(F_{ik}^*) + \lambda_{kj} \geq \lambda_{ij} \quad \ldots\ldots (*)$$ with equality

if $f_{ik}^{v*}(j) > 0$. Multiply both sides of (*) by $f_{ik}^{v}(j)$ and sum over i,j,k:

$$\sum_{i,j,k} [B_{ik}'(F_{ik}^{v*}) + G_{ik}'(F_{ik}^*)] f_{ik}^{v}(j) + \sum_{i,j,k} \lambda_{kj}\, f_{ik}^{v}(j) \geq \sum_{i,j,k} \lambda_{ij}\, f_{ik}^{v}(j)$$

with equality if $f^{v}_{ik}(j) = f^{v*}_{ik}(j)$. Thus

$$\sum_{i,j,k} [B'_{ik}(F^{v*}_{ik}) + G'_{ik}(F^{*}_{ik})]\, f^{v}_{ik}(j) \geq \sum_{i,j,k} \lambda_{ij}\, f^{v}_{ik}(j) - \sum_{i,j,k} \lambda_{kj}\, f^{v}_{ik}(j)$$

$$= \sum_{i,j} \lambda_{ij} \sum_{k} f^{v}_{ik}(j) - \sum_{k,j} \lambda_{kj} \sum_{\ell} f^{v}_{\ell k}(j)$$

$$= \sum_{i,j} \lambda_{ij} \left[\sum_{k} f^{v}_{ik}(j) - \sum_{\ell} f^{v}_{\ell i}(j)\right]$$

$$= \sum_{i,j} \lambda_{ij}\, \beta\gamma n_{ij}\, \ell_{ij}$$

where the last equality follows from (2.14). Similarly, from (3.2) we can show that inequality (ii) holds, thus completing the proof of the lemma.

Now let there be λ_{ij} and μ_{ij} satisfying (3.1) through (3.4) at a feasible point $P^{*} = \{f^{v}_{ik}(j), f^{d*}_{ik}(j), \ell^{*}_{ij}, r^{*}_{ij}\}$. For any other point $P = \{f^{v}_{ik}(j), f^{d}_{ik}(j), \ell_{ij}, r_{ij}\}$ and α, where $0 \leq \alpha \leq 1$, define

$$f^{v\alpha}_{ik}(j) = (1 - \alpha)\, f^{v*}_{ik}(j) + \alpha\, f^{v}_{ik}(j)$$

$$f^{d\alpha}_{ik}(j) = (1 - \alpha)\, f^{d*}_{ik}(j) + \alpha\, f^{d}_{ik}(j)$$

$$\ell^{\alpha}_{ij} = (1 - \alpha)\, \ell^{*}_{ij} + \alpha \ell_{ij}$$

$$r^{\alpha}_{ij} = (1 - \alpha)\, r^{*}_{ij} + \alpha r_{ij}$$

Then $J(\alpha) = \sum_{i,k} [B_{ik}(F^{v\alpha}_{ik}) + G_{ik}(F^{\alpha}_{ik})] + \sum_{i,j} [\gamma n_{ij}\ V_{ij}(\ell^{\alpha}_{ij}) + m_{ij}\, E_{ij}(r^{\alpha}_{ij})]$

is a convex function of α. Therefore,

$$J(1) - J(0) \geq \left.\frac{dJ(\alpha)}{d\alpha}\right|_{\alpha = 0}$$

$$= \sum_{i,j,k} [B'_{ik}(F^{v*}_{ik}) + G'_{ik}(F^{*}_{ik})] [f^{v}_{ik}(j) - f^{v*}_{ik}(j)]$$

$$+ \sum_{i,j} \gamma n_{ij} V'_{ij}(\ell^{*}_{ij}) [\ell_{ij} - \ell^{*}_{ij}]$$

$$+ \sum_{i,j,k} G'_{ik}(F^{*}_{ik}) [f^{d}_{ik}(j) - f^{d*}_{ik}(j)]$$

$$+ \sum_{i,j} m_{ij} E'_{ij}(r^{*}_{ij}) [r_{ij} - r^{*}_{ij}]$$

$$\geq \sum_{i,j} \lambda_{ij} \beta\gamma n_{ij} [\ell_{ij} - \ell^{*}_{ij}] + \sum_{i,j} \beta\gamma n_{ij} p_{ij}(\ell^{*}_{ij}) [\ell^{*}_{ij} - \ell_{ij}]$$

$$+ \sum_{i,j} \mu_{ij} m_{ij} [r_{ij} - r^{*}_{ij}] + \sum_{i,j} m_{ij} q_{ij}(r^{*}_{ij}) [r^{*}_{ij} - r_{ij}]$$

$$= \sum_{i,j} \beta\gamma n_{ij} [p_{ij}(\ell^{*}_{ij}) - \lambda_{ij}] [\ell^{*}_{ij} - \ell_{ij}]$$

$$+ \sum_{i,j} m_{ij} [q_{ij}(r^{*}_{ij}) - \mu_{ij}] [r^{*}_{ij} - r_{ij}]$$

where the second inequality follows from lemma A.1.

Thus

$$J(1) - J(0) \geq \sum_{\substack{i,j: \\ 0<\ell^*_{ij}<a_{ij}}} \beta\,\gamma\, n_{ij}[p_{ij}(\ell^*_{ij}) - \lambda_{ij}]\,[\ell^*_{ij} - \ell_{ij}]$$

$$+ \sum_{i,j:\ell^*_j=0} \beta\,\gamma n_{ij}\,[p_{ij}(\ell^*_{ij}) - \lambda_{ij}]\,[\ell^*_{ij} - \ell_{ij}]$$

$$+ \sum_{i,j:\ell^*_{ij}=a_{ij}} \beta\,\gamma\, n_{ij}\,[p_{ij}(\ell^*_{ij}) - \lambda_{ij}]\,[\ell^*_{ij} - \ell_{ij}]$$

$$+ \sum_{\substack{i,j: \\ 0<r^*_{ij}<r^d_{ij}}} m_{ij}\,[q_{ij}\,(r^*_{ij}) - \mu_{ij}]\,[r^*_{ij} - r_{ij}]$$

$$+ \sum_{i,j:r^*_{ij}=0} m_{ij}\,[q_{ij}\,(r^*_{ij}) - \mu_{ij}]\,[r^*_{ij} - r_{ij}]$$

$$+ \sum_{i,j:r^*_{ij}=r^d_{ij}} m_{ij}\,[q_{ij}\,(r^*_{ij}) - \mu_{ij}]\,[r^*_{ij} - r_{ij}]$$

Now from (3.1) and (3.2) the first and fourth terms are zero. All other terms are greater than or equal to zero. Hence

$$J(1) - J(0) = J_p - J_{p^*} \geq 0$$

That is, J at any feasible point P is greater than or equal to J at P^*. This completes the proof of sufficiency of (3.1) through (3.4).

APPENDIX B

In this appendix we give the proof of theorem 3.2, which deals with the convergence properties of the joint flow control and routing algorithm H. Recall that the objective function J is given by

$$J = \sum_{i,k} B_{ik}(F^v_{ik}) + \sum_{i,k} G_{ik}(F_{ik}) + \sum_a \gamma n_a V_a(\ell_a) + \sum_a m_a E_a(r_a)$$

Define the vector u by the tuple $u = (\ell, r, F^v, F^d)$, where ℓ is the vector of voice packet lengths, r is the vector of data input rates, F^v is the vector of link voice traffics, and F^d is the vector of link data traffics. Let u^n denote the vector u at iteration n. Assume the algorithm is started with a feasible u^0 such that $J(u^0) \leq J_0$, for some positive J_0. Let the set U_0 be defined as follows:

$$U_0 = \{u | J(u) \leq J_0\} \tag{B.1}$$

Then $u^0 \in U_0$. Define $\Delta(\theta)$ as follows:

$$\Delta(\theta) = J(u^\theta) - J(u^n) \tag{B.2}$$

where $u^\theta = \theta u^{n+1} + (1-\theta)u^n$, $0 \leq \theta \leq 1$. Then for all $u^\theta \in U_0$, $\Delta(\theta)$ is continuous in θ, and

$$\Delta(0) = 0 \tag{B.3}$$

$$\Delta(1) = J(u^{n+1}) - J(u^n) \tag{B.4}$$

The Taylor expansion of $\Delta(\theta)$ about zero is

$$\Delta(\theta) = \theta \sum_i \frac{\partial J(u^n)}{\partial u_i} [u_i^{n+1} - u_i^n] + \frac{1}{2} \theta^2 \sum_i \sum_j \frac{\partial^2 J(u)}{\partial u_i \partial u_j} [u_i^{n+1} - u_i^n] [u_j^{n+1} - u_j^n] \Big|_{u=u^{\theta'}} :$$

$$0 \leq \theta' \leq \theta$$

Thus

$$\left.\frac{d\Delta(\theta)}{d\theta}\right|_{\theta=0} = \sum_i \frac{\partial J(u^n)}{\partial u_i} [u_i^{n+1} - u_i^n]$$

$$= \sum_{i,k} [B'_{ik} (F_{ik}^{vn}) + G'_{ik} (F_{ik}^{n})] [F_{ik}^{v(n+1)} - F_{ik}^{vn}]$$

$$+ \sum_{i,k} G'_{ik} (F_{ik}^{n}) [F_{ik}^{d(n+1)} - F_{ik}^{dn}]$$

$$+ \sum_a \gamma n_a V'_a (\ell_a^n) [\ell_a^{n+1} - \ell_a^n]$$

$$+ \sum_a m_a E'_a (r_a^n) [r_a^{n+1} - r_a^n]$$

Lemma B.1 Assume that $u^n \in U_o$. Then $\left.\frac{d\Delta(\theta)}{d\theta}\right|_{\theta=0} \leq 0$.

Proof From (3.25) and (3.26)

$$\sum_{p \in P_a} e_p^{vn} \Delta_p^{vn} = \sum_{p \neq P_v^*} [d_p^{vn} - d^{vn}(a)] [S_p^{vn} - S_p^{v(n+1)}]$$

(since $e_{p_v^*}^{vn} \equiv 0$)

That is,

$$\sum_{p \in P_a} e_p^{vn} \Delta_p^{vn} = \sum_{p \neq p_v^*} d_p^{vn} [S_p^{vn} - S_p^{v(n+1)}] - d^{vn}(a) \sum_{p \neq P_v^*} [S_p^{vn} - S_p^{v(n+1)}]$$

$$= \sum_p d_p^{vn} [S_p^{vn} - S_p^{v(n+1)}] - d_{p_v^*}^{vn} [S_{p_v^*}^{vn} - S_{p_v^*}^{v(n+1)}]$$

$$- d^{vn}(a) \sum_{p \neq p_v}^{*} [S_p^{vn} - S_p^{v(n+1)}]$$

Now $d_{p_v^*}^{vn} = d^{vn}(a)$, and $S_{p_v}^{vn*} = S^{vn}(a) - \sum_{p \neq p_v^*}^{*} S_p^{vn}$

Thus $\sum_{p \in P_a} e_p^{vn} \Delta_p^{vn} = - \sum_p d_p^{vn} [S_p^{v(n+1)} - S_p^{vn}]$

$$+ d^{vn}(a) [S^{v(n+1)}(a) - S^{vn}(a)]$$

But $d_p^{vn} = \sum_{i,k} \delta_p(i,k) [B'_{ik} (F_{ik}^{vn}) + G'_{ik} (F_{ik}^{n})]$

Therefore,

$$\sum_a \sum_{p \in P_a} e_p^{vn} \Delta_p^{vn} = - \sum_{i,k} [B'_{ik} (F_{ik}^{vn}) + G'_{ik}(F_{ik}^{n})] \sum_a \sum_{p \in P_a} \delta_p(i,k) [S_p^{v(n+1)} - S_p^{vn}]$$

$$+ \sum_a d^{vn}(a) [S^{v(n+1)}(a) - S^{vn}(a)]$$

$$= - \sum_{i,k} [B'_{ik} (F_{ik}^{vn}) + G'_{ik} (F_{ik}^{n})] [F_{ik}^{v(n+1)} - F_{ik}^{vn}]$$

$$+ \beta\gamma \sum_a n_a d^{vn}(a) [\ell_a^{n+1} - \ell_a^{n}]$$

where the last equality follows from (3.15) and the definition of $S^{vn}(a)$. From (3.21)

$$d^{vn}(a) = Z_a^n + p_a(\ell_a^n)$$

$$\text{Thus} \sum_a \sum_{p \in P_a} e_p^{vn} \Delta_p^{vn} = - \sum_{i,k} [B'_{ik}(F_{ik}^{vn}) + G'_{ik}(F_{ik}^n)] [F_{ik}^{v(n+1)} - F_{ik}^{vn}]$$

$$+ \beta\gamma \sum_a n_a p_a(\ell_a^n) [\ell_a^{n+1} - \ell_a^n]$$

$$- \beta\gamma \sum_a n_a Z_a^n [\ell_a^n - \ell_a^{n+1}]$$

$$= - \sum_{i,k} [B'_{ik}(F_{ik}^{vn}) + G'_{ik}(F_{ik}^n)] [F_{ik}^{v(n+1)} - F_{ik}^{vn}]$$

$$- \sum_a \gamma n_a V'_a(\ell_a^n) [\ell_a^{n+1} - \ell_a^n]$$

$$- \beta\gamma \sum_a n_a Z_a^n [\ell_a^n - \ell_a^{n+1}] \qquad \text{(B.5)}$$

where the last equality follows from the definition of $p_a(\ell_a^n)$; see (3.9). Now from (3.22), $|\ell_a^n - \ell_a^{n+1}| \leq n_\ell |z_a^n|$. Moreover, when $Z_a^n \geq 0$, $\ell_a^n - \ell_a^{n+1} \geq 0$; and when $Z_a^n \leq 0$, $\ell_a^n - \ell_a^{n+1} \leq 0$. Thus

$$Z_a^n(\ell_a^n - \ell_a^{n+1}) \equiv |Z_a^n| \, |\ell_a^n - \ell_a^{n+1}| \geq n_\ell^{-1} [\ell_a^n - \ell_a^{n+1}]^2$$

Therefore, $\beta\gamma \sum_a n_a Z_a^n [\ell_a^n - \ell_a^{n+1}] \geq \beta\gamma n_\ell^{-1} \sum_a n_a [\ell_a^n - \ell_a^{n+1}]^2$

and $- \beta\gamma \sum_a n_a Z_a^n [\ell_a^n - \ell_a^n] \leq - \beta\gamma n_\ell^{-1} \sum_a [\ell_a^n - \ell_a^{n+1}]^2 n_a$

Thus from (B.5)

$$\sum_a \sum_{p \epsilon P_a} e_p^{vn} \Delta_p^{vn} \leq - \sum_{i,k} [B'_{ik}(F_{ik}^{vn}) + G'_{ik}(F_{ik}^{n})][F_{ik}^{v(n+1)} - F_{ik}^{vn}]$$

$$- \sum_a \gamma n_a V'_a(\ell_a^n)[\ell_a^{n+1} - \ell_a^n]$$

$$- \beta \gamma \eta^{-1} \sum_a n_a [\ell_a^{n+1} - \ell_a^n]^2 \qquad (B.6)$$

Also from (3.26) $\Delta_p^{vn} \leq \eta_v e_p^{vn}$. Thus $\Delta_p^{vn} e_p^{vn} \geq \eta_v^{-1} [\Delta_p^{vn}]^2$

Therefore, from (B.6) we have that

$$\eta_v^{-1} \sum_a \sum_{p \epsilon P_a} [\Delta_p^{vn}]^2 \leq - \sum_{i,k} [B'_{ik}(F_{ik}^{vn}) + G'_{ik}(F_{ik}^{n})][F_{ik}^{v(n+1)} - F_{ik}^{vn}]$$

$$- \sum_a \gamma n_a V'_a(\ell_a^n)[\ell_a^{n+1} - \ell_a^n]$$

$$- \beta \gamma \eta^{-1} \sum_a n_a [\ell_a^{n+1} - \ell_a^n]^2 \qquad (B.6a)$$

$$\text{Or } \sum_{i,k} [B'_{ik}(F^{vn}_{ik}) + G'_{ik}(F^{n}_{ik})][F^{v(n+1)}_{ik} - F^{vn}_{ik}] + \sum_a \gamma n_a V'_a(\ell^n_a)[\ell^{n+1}_a - \ell^n_a]$$

$$\leq -\beta\gamma\eta_\ell^{-1} \sum_a n_a [\ell^{n+1}_a - \ell^n_a]^2 - \eta_v^{-1} \sum_a \sum_{p\in P_a} [\Delta^{vn}_p]^2$$

Similarly, it can be shown that

$$\sum_{i,k} G'_{ik}(F^n_{ik})[F^{d(n+1)}_{ik} - F^{dn}_{ik}] + \sum_a m_a E'_a(r^n_a)[r^{n+1}_a - r^n_a]$$

$$\leq -\eta_r^{-1} \sum_a m_a [r^{n+1}_a - r^n_a] - \eta_d^{-1} \sum_a \sum_{p\in P_a} [\Delta^{dn}_p]^2$$

Therefore, $\left.\frac{d\Delta(\theta)}{d\theta}\right|_{\theta=0}$

$$\leq -\eta_\ell^{-1}\beta\gamma \sum_a n_a [\ell^{n+1}_a - \ell^n_a]^2 - \eta_v^{-1} \sum_a \sum_{p\in P_a} [\Delta^{vn}_p]^2 - \eta_d^{-1} \sum_a \sum_{p\in P_a} [\Delta^{dn}_p]^2$$

$$- \eta_r^{-1} \sum_a m_a [r^{n+1}_a - r^n_a]^2 \qquad \text{(B.7)}$$

$$\leq 0$$

This completes the proof of lemma B.1.

Lemma B.2. Assume $u^n \in U_o$. Then $\left.\frac{d\Delta(\theta)}{d\theta}\right|_{\theta=0} < 0$

if u^n is not an optimal point.

Proof From the result of lemma B.1 we have that $\left.\frac{d\Delta(\theta)}{d\theta}\right|_{\theta=0} \leq 0$. To show that the inequality is strict it suffices to show that if u^n is not an optimal point then at least one of the sums in (B.7) is strictly positive. Assume that none of the four sums in (B.7) is strictly positive.

Then for voice we have the following situation:

(i) $\ell_a^{n+1} = \ell_a^n$. If $0 < \ell_a^n < a_{ij}$, then this situation implies that $Z_a^n = d^{vn}(a) - p_a(\ell_a^n) = 0$, and the marginal gain in voice quality is equal to the marginal voice cost of congestion. If $\ell_a^n = 0$, then $Z_a^n \geq 0$ and $d^{vn}(a) \geq p_a(\ell_a^n)$. And if $\ell_a^n = a_{ij}$, then $Z_a^n \leq 0$ and $d_a^{vn} \leq p_a(\ell_a^n)$.

(ii) $\Delta_p^{vn} = 0$, which means that $e_p^{vn} = 0$ since by assumption $S_p^{vn} > 0$.

That is, $e_p^{vn} = d_p^{vn} - d^{vn}(a) = 0$, and all voice traffic flows on shortest paths.

By a similar argument for data we have that

(iii) $q_a(r_a^n)$
$$\begin{cases} = d^{dn}(a) & \text{if } 0 < r_a^n < r_a^d \\ \geq d^{dn}(a) & \text{if } r_a^n = r_a^d \\ \leq d^{dn}(a) & \text{if } r_a^n = 0 \end{cases}$$

(iv) All data traffic flows on shortest paths.

But results (i) through (iv) are precisely the Kuhn-Tucker conditions for optimality given in (3.1) through (3.4). Thus by the sufficiency part of theorem 3.1, proved in Appendix A, u^n is an optimal point. Therefore, if u^n is not an optimal point at least one sum in (B.7) must be strictly positive, thus proving the lemma.

The implication of lemma B.2 is that for small enough θ, $\Delta(\theta) < 0$. Since $\Delta(\theta)$ is continuous in θ for $u^\theta \in U_0$, either $\Delta(\theta) < 0$ for all

$\theta \leq 1$ or there exists a $\theta_1 < 1$ such that $\Delta(\theta) < 0$ for $\theta < \theta_1$ and $\Delta(\theta_1) = 0$. That is, the plot of $\Delta(\theta)$ takes the form of either curve (i) or curve (ii) of Fig. B.1. Let θ_1 be the smallest θ, $0 < \theta \leq 1$, for which $\Delta(\theta) = 0$. If no such θ exists, then let $\Delta(\theta_1) = \Delta(1)$.

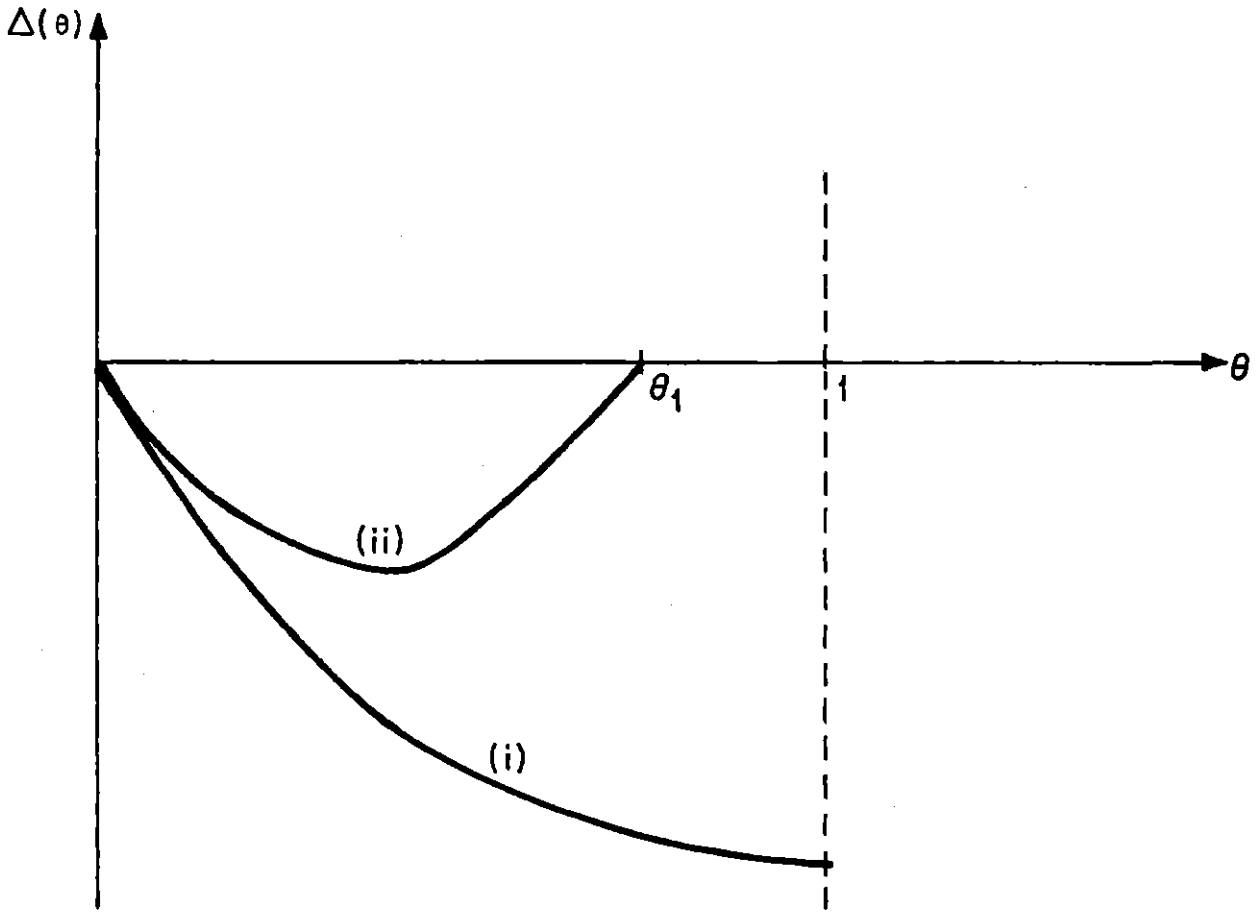

Figure B.1 Possible Forms of $\Delta(\theta)$

Assume that there exists $\theta_1 \leq 1$, such that $\Delta(\theta_1) = 0$. The Taylor expansion of $\Delta(\theta_1)$ (about zero) is

$$\Delta(\theta_1) = \theta_1 \sum_i \frac{\partial J(u^n)}{\partial u_i} [u_i^{n+1} - u_i^n]$$

$$+ \frac{1}{2} \theta_1^2 \sum_i \sum_j \frac{\partial^2 J(u)}{\partial u_i \partial u_j} [u_i^{n+1} - u_i^n] [u_j^{n+1} - u_j^n] \Big|_{u=u^{\theta'} : \theta' < \theta_1}$$

From lemma B.1 $\sum_i \frac{\partial J(u^n)}{\partial u_i} [u_i^{n+1} - u_i^n] = \frac{d\Delta(\theta)}{d\theta}\Big|_{\theta = 0}$

$$\leq - \eta_\ell^{-1} \beta \gamma \sum_a n_a [\ell_a^{n+1} - \ell_a^n]^2$$

$$- \eta_v^{-1} \sum_a \sum_{p \epsilon P_a} [\Delta_p^{vn}]^2 - \eta_r^{-1} \sum_a m_a [r_a^{n+1} - r_a^n]^2$$

$$- \eta_d^{-1} \sum_a \sum_{p \epsilon P_a} [\Delta_p^{dn}]^2$$

One of the terms in the sum $\sum_i \sum_j \frac{\partial^2 J(u)}{\partial u_i \partial u_j} [u_i^{n+1} - u_i^n] [u_j^{n+1} - u_j^n]$

is $\sum_a \sum_{i,k} \frac{\partial^2 J(u)}{\partial \ell_a \partial F_{ik}^v} [\ell_a^{n+1} - \ell_a^n] [F_{ik}^{v(n+1)} - F_{ik}^{vn}]$. Now

$$\frac{\partial^2 J(u)}{\partial \ell_a \partial F_{ik}^v} = \frac{\partial}{\partial \ell_a} \left[\frac{\partial J(u)}{\partial F_{ik}^v} \right]$$

$$= \frac{\partial}{\partial \ell_a} [B_{ik}'(F_{ik}^v) + G_{ik}'(F_{ik})]$$

$$= 0$$

The last equality follows from the fact that we are taking partial and not total derivatives. By the same argument we have that all terms $\frac{\partial^2 J(u)}{\partial u_i \partial u_j}$, in which $u_i \neq u_j$, are zero except that $\frac{\partial^2 J(u)}{\partial F^v_{ik} \partial F^d_{ik}} \neq 0$. Thus

$$\sum_i \sum_j \frac{\partial^2 J(u)}{\partial u_i \partial u_j} [u_i^{n+1} - u_i^n] [u_j^{n+1} - u_j^n] \Big|_{u = u^{\theta'}}$$

$$= \sum_{i,k} [B''_{ik} (F^{v\theta'}_{ik}) + G''_{ik} (F^{\theta'}_{ik})] [F^{v(n+1)}_{ik} - F^{vn}_{ik}]^2$$

$$+ \sum_{i,k} G''_{ik} (F^{\theta'}_{ik}) [F^{d(n+1)}_{ik} - F^{dn}_{ik}]^2$$

$$+ 2 \sum_{i,k} G''_{ik} (F^{\theta'}_{ik}) [F^{v(n+1)}_{ik} - F^{vn}_{ik}] [F^{d(n+1)}_{ik} - F^{dn}_{ik}]$$

$$+ \sum_a \gamma n_a V''_a (\ell^{\theta'}_a) [\ell^{n+1}_a - \ell^n_a]^2$$

$$+ \sum_a m_a E''_a (r^{\theta'}_a) [r^{n+1}_a - r^n_a]^2 \qquad \text{(B.8)}$$

Since $\Delta(\theta)$ is continuous in θ for $u^\theta \in U_o$, the second derivatives exist over the compact region $0 \leq \theta \leq \theta_1$.

Lemma B.3 Let $M_\ell = \max_a \max_{u:J(u)\leq J_o} V''_a(\ell_a)$

for some positive J_o. Then

$$\sum_a n_a V''_a (\ell^{\theta'}_a) [\ell^{n+1}_a - \ell^n_a]^2 \leq M_\ell \sum_a n_a [\ell^{n+1}_a - \ell^n_a]^2$$

Proof Recall that $U_o = \{u | J(u) \leq J_o\}$ for some positive J_o. Since we assume $u^\theta \in U_o$ for $\theta \in [0,1]$, then $V_a''(\ell_a^\theta) \leq M_\ell$. The proof of the lemma follows then from the assumption of the lemma.

Lemma B.4. Let $M_r = \max_a \max_{u:J(u)\leq J_o} E_a''(r_a)$ for some positive J_o. Then

$$\sum_a m_a E_a''(r_a^{\theta'}) [r_a^{n+1} - r_a^n]^2 \leq M_r \sum_a m_a [r_a^{n+1} - r_a^n]^2$$

Proof Similar to the proof of lemma B.3.

Lemma B.5 Let $M_v = \max_{i,k} \max_{u:J(u)\leq J_o} [B_{ik}''(F_{ik}^v) + G_{ik}''(F_{ik})]$

for some J_o. Furthermore, let the cardinalities of the sets A_v and P_a be $|A_v|$ and $|P_a|$, respectively. Then

$$\sum_{i,k} [B_{ik}''(F_{ik}^{v\theta'}) + G_{ik}''(F_{ik}^{\theta'})] [F_{ik}^{v(n+1)} - F_{ik}^{vn}]^2$$

$$\leq M_v N^2 |A_v|^2 |P_a|^3 \sum_a \sum_{p\in P_a} [\Delta_p^{vn}]^2$$

Proof For the same reason given in the proof of lemma B.3, we have that $B_{ik}''(F_{ik}^{v\theta'}) + G_{ik}''(F_{ik}^{\theta'}) \leq M_v$. Thus

$$\sum_{i,k} [B_{ik}''(F_{ik}^{v\theta'}) + G_{ik}''(F_{ik}^{\theta'})] [F_{ik}^{v(n+1)} - F_{ik}^{vn}]^2 \leq M_v \sum_{i,k} [F_{ik}^{v(n+1)} - F_{ik}^{vn}]^2$$

From (3.15) we have that

$$[F_{ik}^{v(n+1)} - F_{ik}^{vn}]^2 = \left[\sum_a \sum_{p\in P_a} \delta_p(i,k) [S_p^{v(n+1)} - S_p^{vn}]\right]^2$$

$$\leq |A_v| |P_a| \sum_a \sum_{p\in P_a} \delta_p(i,k) [S_p^{v(n+1)} - S_p^{vn}]^2$$

where the inequality is the Cauchy-Schwarz inequality. Since we make incremental changes at each iteration, the largest change in any path flow cannot exceed the sum of all changes in path flows for the same source - destination pair. Therefore,

$$|S_p^{v(n+1)} - S_p^{vn}| \leq \sum_p \Delta_p^{vn} \qquad , \qquad p \epsilon P_a$$

Thus $$[S_p^{v(n+1)} - S_p^{vn}]^2 \leq [\sum_{p \epsilon P_a} \Delta_p^{vn}]^2$$

$$\leq |P_a| \sum_{p \epsilon P_a} [\Delta_p^{vn}]^2$$

where the last inequality is the Cauchy-Schwarz inequality. Therefore,

$$|A_v| \; |P_a| \sum_a \sum_{p \epsilon P_a} \delta_p(i,k) \; [S_p^{v(n+1)} - S_p^{vn}]^2$$

$$\leq |A_v|^2 \; |P_a|^3 \sum_a \sum_{p \epsilon P_a} \delta_p(i,k) \; [\Delta_p^{vn}]^2$$

That is, $[F_{ik}^{v(n+1)} - F_{ik}^{vn}]^2 \leq |A_v|^2 \; |P_a|^3 \sum_a \sum_{p \epsilon P_a} \delta_p(i,k) \; [\Delta_p^{vn}]^2$

And $$\sum_{i,k} [F_{ik}^{v(n+1)} - F_{ik}^{vn}]^2 \leq N^2 \; |A_v|^2 \; |P_a|^3 \sum_a \sum_{p \epsilon P_a} [\Delta_p^{vn}]^2$$

Multiplying both sides by M_v completes the proof of the lemma.

<u>Lemma B.6</u> Let $M_d = \max_{i,k} \max_{u:J(u) \leq J_o} G''_{ik}(F_{ik})$

for some J_o. Furthermore let $|P_a|$ be as defined in lemma B.5, and let $|A_d|$ be the cardinality of the set A_d. Then

$$\sum_{i,k} G''_{ik}(F^{\theta'}_{ik})[F^{d(n+1)}_{ik} - F^{dn}_{ik}]^2 \leq M_d N^2 |A_d|^2 |P_a|^3 \sum_a \sum_{p\epsilon P_a} [\Delta^{dn}_p]^2$$

Proof The proof is similar to the proof of lemma B.5.

Lemma B.7 Let M_d and $|A_d|$ be as defined in lemma B.6, and let $|A_v|$ and $|P_a|$ be as defined in lemma B.5.

Then

$$\sum_{i,k} G''_{ik}(F^{\theta'}_{ik})[F^{v(n+1)}_{ik} - F^{vn}_{ik}][F^{d(n+1)}_{ik} - F^{dn}_{ik}]$$

$$\leq \frac{1}{2} M_d N^2 |A_v|^2 |P_a|^3 \sum_a \sum_{p\epsilon P_a} [\Delta^{vn}_p]^2$$

$$+ \frac{1}{2} M_d N^2 |A_d|^2 |P_a|^3 \sum_a \sum_{p\epsilon P_a} [\Delta^{dn}_p]^2$$

Proof We know that $\sum_i (x_i - y_i)^2 = \sum_i (x_i^2 + y_i^2 - 2x_i y_i) \geq 0$

Thus $\sum_i x_i y_i \leq \frac{1}{2} \sum_i x_i^2 + \frac{1}{2} \sum_i y_i^2$ (B.9)

Now from the assumptions of the lemma,

$$\sum_{i,k} G''_{ik}(F^{\theta'}_{ik})[F^{v(n+1)}_{ik} - F^{vn}_{ik}][F^{d(n+1)}_{ik} - F^{dn}_{ik}]$$

$$\leq M_d \sum_{i,k} [F^{v(n+1)}_{ik} - F^{vn}_{ik}][F^{d(n+1)}_{ik} - F^{dn}_{ik}]$$

If we let $x_i = F_{ik}^{v(n+1)} - F_{ik}^{vn}$, and

$$y_i = F_{ik}^{d(n+1)} - F_{ik}^{dn}$$

then from (B.9) we obtain

$$\sum_{i,k} G''_{ik}(F_{ik}^{\theta'})[F_{ik}^{v(n+1)} - F_{ik}^{vn}][F_{ik}^{d(n+1)} - F_{ik}^{dn}]$$

$$\leq \frac{1}{2} M_d \sum_{i,k} [F_{ik}^{v(n+1)} - F_{ik}^{vn}]^2$$

$$+ \frac{1}{2} M_d \sum_{i,k} [F_{ik}^{d(n+1)} - F_{ik}^{dn}]^2$$

$$\leq \frac{1}{2} M_d N^2 |A_v|^2 |P_a|^3 \sum_a \sum_{p\in P_a} [\Delta_p^{vn}]^2$$

$$+ \frac{1}{2} M_d N^2 |A_d|^2 |P_a|^3 \sum_a \sum_{p\in P_a} [\Delta_p^{dn}]^2$$

where the last inequality follows from lemmas B.5 and B.6.

From the results of lemma B.1, and lemmas B.3 through B.7 we obtain

$$\begin{aligned}\Delta(\theta_1) \leq\ & \theta_1[\tfrac{1}{2}\theta_1(M_v + M_d) N^2 |A_v|^2 |P_a|^3 - n_v^{-1}] \sum_a \sum_{p\in P_a} [\Delta_p^{vn}]^2 \\ &+ \theta_1[\theta_1 M_d N^2 |A_d|^2 |P_a|^3 - n_d^{-1}] \sum_a \sum_{p\in P_a} [\Delta_p^{dn}]^2 \\ &+ \theta_1[\tfrac{1}{2}\theta_1 M_\ell - n_\ell^{-1}\beta\gamma] \sum_a n_a [\ell_a^{n+1} - \ell_a^n]^2 \\ &+ \theta_1[\tfrac{1}{2}\theta_1 M_r - n_r^{-1}] \sum_a m_a [r_a^{n+1} + r_a^n]^2 \end{aligned} \qquad \text{(B.10)}$$

Lemma B.8 Let the scale factors be chosen as follows:

$$\eta_v = [\tfrac{1}{2}(M_v + M_d)\, N^2\, |A_v|^2\, |P_a|^3]^{-1}$$

$$\eta_d = [M_d\, N^2\, |A_d|^2\, |P_a|^3]^{-1}$$

$$\eta_\ell = \beta\gamma[\tfrac{1}{2} M_\ell]^{-1}$$

$$\eta_r = [\tfrac{1}{2} M_r]^{-1}$$

Then if u^n is not an optimal point, $\Delta(\theta_1) < 0$.

Proof Substituting the η's, as defined in the lemma, in (B.10) we obtain

$$\Delta(\theta_1) \leq \tfrac{1}{2}\theta_1(M_v + M_d)\, N^2\, |A_v|^2\, |P_a|^3\, [\theta_1 - 1] \sum_a \sum_{p\in P_a} [\Delta_p^{vn}]^2$$

$$+ \theta_1\, M_d\, N^2\, |A_d|^2\, |P_a|^3\, [\theta_1 - 1] \sum_a \sum_{p\in P_a} [\Delta_p^{dn}]^2$$

$$+ \tfrac{1}{2}\, \theta_1\, M_\ell\, [\theta_1 - 1] \sum_a \eta_a\, [\ell_a^{n+1} - \ell_a^n]^2$$

$$+ \tfrac{1}{2}\, \theta_1\, M_r\, [\theta_1 - 1] \sum_a m_a\, [r_a^{n+1} - r_a^n]^2$$

Since we assume $\theta_1 \leq 1$, we have that $\Delta(\theta_1) \leq 0$. To show that $\Delta(\theta_1) < 0$ if the η's are selected as stated in the lemma, it suffices to show that at least one of the sums above is strictly nonzero if u^n is not an optimal point. But this is what we proved in lemma B.1. Thus the proof of the lemma is completed.

The result of this lemma shows that there exists a choice of scale factors such that $\Delta(\theta_1) < 0$. Therefore, with the scale factors chosen as in lemma B.8, the assumption that $\Delta(\theta_1) = 0$ for $\theta_1 \leq 1$ is contradicted. Thus if u^n is not an optimal point and the scale factors are chosen as in lemma B.8, $\Delta(\theta_1) < 0$ for $0 < \theta_1 \leq 1$. And in particular, $\Delta(1) < 0$. That is, $J(n^{n+1}) - J(u^n) < 0$. By induction, $J(u^n) < J(n^{n-1}) < \ldots < J(u^0)$. But we started with a feasible u^0 such that $J(u^0) \leq J_0$. Therefore $J(u^n) < J_0$. Thus the algorithm is a descent algorithm; and if $u^0 \in U_0$, the $u^n \in U_0$ for all $n \geq 1$.

Lemma B.9 The mapping H is closed.

Proof We define closedness of a point-to-set mapping in the same way it is defined by Luenberger [25]. That is, let H be a point-to-set mapping which maps points in a space X into subsets of another space Y. Then H is said to be closed at $\hat{x} \in X$ if the assumptions that

(i) $\{x^n\} \to \hat{x}$, for $x^n \in X$

(ii) $\{y^n\} \to \hat{y}$, for $y^n \in H(x^n)$

imply that $\hat{y} \in H(\hat{x})$.

Let $\{w^n\} = \{S_p^{vn}, S_p^{dn}, \ell_a^n, r_a^n\}$ be a sequence which converges to $w = (S_p^v, S_p^d, \ell_a, r_a)$. Then

$$S^{vn}(a) = \sum_{p \in P_a} S_p^{vn} \to S^v(a)$$

Since $F_{ik}^{vn} = \sum_{a \in A_v} \sum_{p \in P_a} \delta_p(i,k)\, S_p^{vn}$,

and since $S_p^{vn} \to S_p^v$, the link flows $F_{ik}^{vn} \to F_{ik}^v$ if we started with $u^o \in U_o$ (since by the previous result all subsequent u^n are also elements of U_o).

Thus $d_p^{vn} = \sum_{(i,k) \in p} [B'_{ik}(F_{ik}^{vn}) + G'_{ik}(F_{ik}^n)] \to d_p^v$ since

B_{ik} and G_{ik} are continuous in F_{ik}^{vn}. And

$$d^{vn}(a) = \min_{p \in P_a} d_p^{vn} \to d^v(a)$$

$$e_p^{vn} = d_p^{vn} - d^{vn}(a) \to e_p^v .$$

Therefore, $\Delta_p^{vn} = \min [S_p^{vn}, \eta_v e_p^{vn}] \to \Delta_p^v$.

Let the sequence $\{\hat{w}_n\} = \{S_p^{vn}, S_p^{dn}, \ell_a^n, r_a^n\}$ be another sequence that converges to $\hat{w} = (\hat{S}_p^v, \hat{S}_p^d, \hat{\ell}_a, \hat{r}_a)$. Furthermore, let the sequences $\{w^n\}$ and $\{\hat{w}^n\}$ be related as follows:

$$\hat{S}_p^{vn} = \begin{cases} S_p^{vn} - \Delta_p^{vn} & \text{if } p \neq p_v^{*n} \\ \hat{S}^{vn}(a) - \sum_{p \neq p_v^{*n}} (S_p^{vn} - \Delta_p^{vn}) & \text{if } p = p_v^{*n} \end{cases}$$

where p_v^{*n} is an optimal voice path at iteration n.

$$\hat{S}_p^{dn} = \begin{cases} S_p^{dn} - \Delta_p^{dn} & \text{if } p \neq p_d^{*n} \\ \hat{S}^{dn}(a) - \sum_{p \neq p_d^{*n}} (S_p^{dn} - \Delta_p^{dn}) & \text{if } p = p_d^{*n} \end{cases}$$

where p_d^{*n} is an optimal data path at iteration n.

$$\hat{\ell}_a^n = \begin{cases} \hat{\ell}_a^n - \eta_\ell Z_a^n & \text{if } 0 < \ell_a^n - \eta_\ell Z_a^n < a_{ij} \\ a_{ij} & \text{if } \ell_a^n - \eta_\ell Z_a^n \geq a_{ij} \\ 0 & \text{if } \ell_a^n - \eta_\ell Z_a^n \leq 0 \end{cases}$$

$$\hat{r}_a^n = \begin{cases} r_a^n - \eta_r x_a^n & \text{if } 0 < r_a^n - \eta_r x_a^n < r_a^d \\ r_a^d & \text{if } r_a^n - \eta_r x_a^n \geq r_a^d \\ 0 & \text{if } r_a^n - \eta_r x_a^n \leq 0 \end{cases}$$

where Z_a^n and x_a^n are as previously defined. Note that Z_a^n and x_a^n converge to Z_a and x_a, respectively, since these terms (i.e. Z_a^n and x_a^n) are continuous in w^n.

Now we have that $S_p^{vn} - \Delta_p^{vn}$ converges to $S_p^v - \Delta_p^v$ for all p and, by assumption, $\hat{S}_p^{vn}$ converges to $\hat{S}_p^v$ for all p. Suppose $\hat{S}_p^v \neq S_p^v - \Delta_p^v$ for some p. Then there exists some $\epsilon > 0$ such that $|\hat{S}_p^v - (S_p^v - \Delta_p^v)| = \epsilon$. Also there exists some n_o such that for $n \geq n_o$ and $p \neq p_v^{*n}$, $|\hat{S}_p^{vn} - (S_p^{vn} - \Delta_p^{vn})| < \epsilon/2$. Thus for $n > n_o$, $p = p_v^{*n}$. That is,

$$\hat{S}_p^{vn} \to \hat{S}_p^v = \begin{cases} S_p^v - \Delta_p^v & \text{if } p \neq p_v^* \\ \hat{S}^v(a) - \sum_{*}^{p \neq p_v} (S_p^v - \Delta_p^v) & \text{if } p = p_v^* \end{cases}$$

where p_v^* is some limiting optimal path. By the definition of closedness [25], we conclude that H is closed at S_p^v since $\hat{S}_p^v \in H(S_p^v)$. By a similar argument we also conclude that H is closed at S_p^d, since $\hat{S}_p^d \in H(S_p^d)$. Since $\ell_a^n - \eta_\ell Z_a^n$ converges to $\ell_a - \eta_\ell Z_a$ for all a, and since by assumption $\hat{\ell}_a^n$ converges to $\hat{\ell}_a$, we can apply the above argument to conclude that

$$\hat{\ell}_a^n \to \hat{\ell}_a = \begin{cases} \ell_a - \eta_\ell Z_a & \text{if } 0 < \ell_a - \eta_\ell Z_a < a_{ij} \\ a_{ij} & \text{if } \ell_a - \eta_\ell Z_a \geq a_{ij} \\ 0 & \text{if } \ell_a - \eta_\ell Z_a \leq 0 \end{cases}$$

Thus H is closed at ℓ_a since $\hat{\ell}_a \in H(\ell_a)$. And by a similar argument we conclude that H is also closed at r_a, since $\hat{r}_a \in H(r_a)$. Thus H is closed at w since $\hat{w} \in H(w)$, and this completes the proof of the lemma.

Proof of Theorem 3.2 Let U be the following compact set:

$$U = \{u \mid J(u) \leq J_0, \text{ u is a feasible point of (2.14) through (2.16)}\}$$

Then H is a closed point-to-set mapping of points in U into sets of points in U. Also by lemma B.8, the sequence $\{J(u^n)\}$ is a strictly decreasing sequence. That is, if u^n is not an optimal point then $J(u^{n+1}) < J(u^n)$, where $u^{n+1} \in H(u^n)$. Therefore, by the global convergence theorem [25, p. 125], the limit of any convergent subsequence of the sequence $\{u^n\}$ generated by H is a solution to (2.13). This completes the proof of theorem 3.2.

Proof of Necessity of Theorem 3.1

From lemma B.8, we observe that J strictly decreases if equations (3.1) through (3.4) are not satisfied. Thus at any point u that these equations are not satisfied the algorithm produces a better point, which means that u cannot be an optimal point. This then proves the necessity part of theorem 3.1.

References

1. Baran, P., "On Distributed Communications Networks", IEEE Trans. Comm. Systems, Vol. CS-13, pp. 1-9, March 1964.

2. Bertsekas, D.P., "A Class of Optimal Routing Algorithms for Communication Networks", Proc. 5th International Conference on Computer Communication, Atlanta, GA, pp. 71-75, October 1980.

3. Bially, T. et al., "A Technique For Adaptive Voice Flow Control In Integrated Packet Network", IEEE Trans. Comm., Vol. COM-28, No.3. pp. 325-333, March 1980.

4. Bullington, K. and J.M. Frazer, "Engineering Aspects of TASI", Bell System Technical Journal, Vol. 38, pp. 353-364, March 1959.

5. Campanella, S.J. and H. Suyerhoud, "Performance of Digital Speech Interpolation Systems for Telecommunications", NTC Conference Records, pp. 46-1 to 46-3, December 1975.

6. Cerf, V.G. and R.E. Kahn, "A Protocol For Packet Network Intercommunication", IEEE Trans. Comm., Vol. COM-22, No. 5, pp. 637-648, May 1974.

7. Clipsham, W.W. et al., "Datapac Network Overview", Proc. ICCC, pp. 131-136, Toronto, 1976.

8. Chu, W.W., "A Study of the Technique of Asynchronous Time Division Multiplexing for Time-Sharing Computer Communications", Proc. 1969 Hawaii Int. Conf. System Science, pp. 607-610.

9. Coviello, G. and P. Vena, "Integration of Circuit/Packet Switching in a SENET (Slotted Envelope Network) Concept", NTC Conference Records, pp. 42-12 to 42-17, December 1975.

10. Daret, A. et al."The French Public Packet Switching Service: The Transpac Network", Proc. ICCC, pp. 251-260, Toronto, 1976.

11. Forgie, J.W. and A.G. Nemeth, "An Efficient Packetized Voice/Data Network Using Statistical Flow Control", ICC Conference Records, Vol. III, pp. 44-48, June 1977.

12. Gallager, R.G., "A Minimum Delay Routing Algorithm Using Distributed Computation", IEEE Trans. Comm.,Vol.,COM-25, No. 1,pp. 73-85, January 1977.

13. Gallager, R.G. and S.J. Golestaani, "Flow Control and Routing Algorithms for Data Networks", Proc. 5th Int. Conf. Comp. Comm., Atlanta, GA, October 1980.

14. Gallager, R. G., Priyate Communication.

15. Gerla, M. and W. Chou, "Flow Control Strategies In Packet-switched Computer Networks", NTC Conference Records, December 1974.

16. Gerla, M. and L. Kleinrock, "Flow Control: A Comparative Survey", IEEE Trans. Comm., Vol. COM-28, No.4, pp. 553-574, April 1980.

17. Gitman, I. and H. Frank, "Economic Analysis of Integrated Voice and Data Networks: A Case Study", Proc. IEEE, Vol. 66, No. 11, pp. 1549-1570, November 1978.

18. Gold, B., "Digital Speech Networks", Proc. IEEE, Vol. 65, No. 12, pp. 1636-1658, December 1977.

19. 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, M.I.T., Cambridge, MA., January 1980.

20. Kermani, P. and L. Kleinrock, "Virtual Cut-Through: A New Computer Communication Switching Technique", Computer Networks, Vol. 3, pp. 267-286, 1979.

21. Kleinrock, L., Communication Nets: Stochastic Message Delay, New York: McGraw-Hill Book Co., 1964.

22. Kleinrock, L., Queueing Systems, Vol. 2, New York: John Wiley and Sons, Inc., 1976

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

24. Little, J.D.C., "A Proof for the Queueing Formula: $L = \lambda W$", Operations Research, Vol. 9, pp. 383-387, 1961.

25. Luenberger, D., Introduction to Linear and Nonlinear Programming, Reading, MA: Addison-Wesley, 1973.

26. McQuillan, J.M. et al., "The New Routing Algorithm for the ARPANET", IEEE Trans. Comm. , Vol. COM-28, No. 5, pp. 711-719, May 1980.

27. Roberts, L.G. and B.D. Wessler, "Computer Network Development to Achieve Resource Sharing", Proc. AFIPS Spring Joint Comp. Conf., Vol. 36, pp. 543-549, May 1970.

28. Sciulli, J.A. and S.J. Campanella, "A Speech Predictiye Encoding Communication System for Multichannel Telephony", IEEE Trans. Comm., Vol. COM-21, No. 7, pp. 827-835, July 1973.

29. Segall, A., "Optimal Distributed Routing for Virtual Line-Switched Data Networks", IEEE Trans. Comm., Vol. COM-27, No. 1, pp. 201-209, January 1979.

30. Tymes, L., "TYMNET - A Terminal-Oriented Communications Network," AFIPS Conf. Proc., Vol. 38, pp. 211-216, 1971.

31. Weinstein, C.J., "Fractional Speech Loss and Talker Activity Model for A TASI and Packet-Switched Speech", IEEE Trans. Comm., Vol.COM-26, No. 8, pp. 1253-1257, August 1978.

32. Weinstein, C.J. and E.M. Hofstetter, "The Tradeoff Between Delay and TASI Advantage in a Packetized Speech Multiplexer", IEEE Trans. Comm., Vol. COM-27, No.11, pp. 1716-1720, November 1979.

33. Wessler, B.D. and R.B. Hovey, "Public Packet-Switched Networks", Datamation, pp. 85-87, July 1974.