THE INTEGRATION OF ROUTING AND FLOW-CONTROL FOR VOICE AND DATA IN A COMPUTER COMMUNICATION NETWORK

by

ELIEZER MENAHEM GAFNI

Bs.C., Technion-Israel Institute of Technology
(1972)

Ms.C., University of Illinois at Urbana-Champaign
(1979)

SUBMITTED IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF SCIENCE

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

August 1982

Signature of Author _____
Department of Electrical Engineering and Computer Science
August 6, 1982

Certified by _____
Dimitri P. Bertsekas
Thesis Supervisor

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

THE INTEGRATION OF ROUTING AND FLOW-CONTROL FOR VOICE AND DATA IN A COMPUTER COMMUNICATION NETWORK

by

ELIEZER MENAHEM GAFNI

Submitted to the Department of Electrical Engineering and Computer Science on August 6, 1982 in partial fulfillment of the requirements for the Degree of Doctoral of Science in Electrical Engineering and Computer Science

ABSTRACT

A store and forward computer communication network may concurrently support more than one type of user. Because of different user characteristics, the control of each type of user may be done by different mechanisms. Moreover, for each type of user there may be more than one control mechanism operating concurrently. Algorithms previously proposed for user-mechanism pairs were analysed independently ignoring the significant interactions among them.

In this thesis, we consider the voice user and the data user, and the routing mechanism and the flow control mechanism. We propose four algorithms, one for each user-mechanism pair. We show that these algorithms are compatible in the sense that they can be coordinated to achieve some reasonable objective when operating concurrently. Moreover, some of the algorithms are superior in some aspects to existing ones.

Thesis Supervior: Dimitri P. Bertsekas

Title: Professor of Electical Engineering and Computer Science

ACKNOWLEDGEMENT

I would like to thank my Thesis advisor, Professor D.P. Bertsekas for his guidance and support during the last five years. His willingness to spare some good words whenever needed, helped me stand the strains of Graduate School. I consider it my good fortune to have met him.

Thanks are also due to Professor R.G. Gallager and Professor P.A. Humblet who were always available for discussion and served as my Thesis readers. In particular Professor Gallager's approach to communication networks is visible throughout the Thesis.

Last but not least I would like to thank many people who shared an office with me over the years, among them K. Vastola, M. Haimovitch, D. Friedman, I. Castineyra and E. Arikan. Our conversations, ranging from politics to mere gossip and at times "even" to technical matters has made Graduate School enjoyable. I consider my interaction with this diverse group of people an important part of my education. I take their complaints, about the difficulties of studying in my presence, as a complement.

On the more personal level I hope that this Thesis will serve as partial compensation to my wife's parents and mine for our being away from home, and to my wife Anat who has put up with an MIT garduate student, I hope this Thesis will mark the end of such a grueling task.

TABLE OF CONTENTS

PAGE

PAGE

1. Introduction

1.1 Problem Definition

The concept of store and forward operation for communication networks was introduced in order to increase efficiency by sharing network resources, but this efficiency decreases rapidly when the network becomes congested. Because the nodal storage capacity, which is needed for the store and forward operation, is limited, a node cannot accept a packet when its storage area is full. Consequently, a node whose storage area is full may cause a fill up of the storage area in adjacent nodes inhibited from forwarding packets to it. It is not hard to imagine how this fill-up might propagate, causing the network to enter virtual standstill, with each node waiting for the other, as input packets try to enter the network and none succeeds.

Routing and flow control are two mechanisms to avoid congestion. To understand the exact role of each of them, it is helpful to view a communication network as a network of queues in which the server has to devote part of his service capacity to the control of the queue. Assume that the part of the service capacity that goes for the control of the queue is proportional to the number of customers in the queue. Thus as the number of customers in the queue increases the actual service rate decreases. If the number of customers in the queue exceeds some threshold the actual service rate drops to zero and the queue is deadlocked.

The routing in the context of this analogy, tries to avoid a queue deadlock and to increase the actual service rate by distributing the streams of the incoming flow among many queues. In this way the effective arrival rate at each queue within the network is low, allowing a higher fraction of the server time to be devoted to an actual service and reducing

the probability of a queue deadlocking. Moreover, in this way a packet is cleared from the network faster, reducing the total resources consumed by it.

But, routing can only help in reducing the apriori chance of any queue deadlocking. It cannot totally prevent it. This is due to the nondeterministic arrival of packets to the network, and to the nondeterministic manner in which a packet is forwarded in the network.

Prevention of this chance is the task of the flow control mechanism. In particular, end to end and flow-control curtails the flow directed to the congested area until the congestion subsides. However, in many instances end to end flow-control will not be enough and it will be operated in conjunction with another mechanism which controls packet forwarding inside the network.

That routing and flow control have the same role of preventing congestion makes it hard to distinguish where routing ends, and end to end flow-control starts. A routing algorithm which reacts to the network state provides a measure of flow control by aposteriori diverting flow from congested to uncongested areas; a good flow control algorithm provides rerouting of flow by allowing higher flows through uncongested areas while curtailing the flow to congested ones.

Traditionally, in order to understand the exact role of, and improve on, schemes of congestion prevention, routing and flow control have been dealt with separately. When dealing with routing, inputs are usually considered fixed; when dealing with flow control, fixed routing has been assumed. But, in view of the congestion prevention objective that both routing and flow-control try to achieve, they ultimately should be analysed as one mechanism. In making routing changes, the effect of the flow-

control on the inputs should be taken into account; while when changing flow-control parameters the reaction of the routing mechanism to the changing input should be considered.

Many routing and flow-control algorithms that were suggested when viewing each of them separately, turned out to be incompatible when viewed as a single system. The usual problem is that when the flow-control parameters remain fixed, a routing change causes an input change. This routing change which can be shown to achieve a reasonable objective were the inputs to remain fixed, might achieve the opposite objective when the inputs are subject to the flow-control mechanism.

Recently, the interest in providing the store and forward communication network with the capability to accommodate the conversational-voice customer has mounted. Like a computer that waits for a new request after dumping a chunk of information, or like the terminal user who has to type in a new line after transmitting the previous one, so too, a user, who has finished speaking to another, either listens to the other's response or pauses before going on. Inasmuch as the voice user does not utilize the network continually at the same rate, the store and forward concept is also suitable for him. The advent of the technology of packetizing voice has given an incentive to research on the particular problems that may result from incorporating the voice user into the network.

The problem of congestion and the mechanisms of routing and flow control should apply equally well to different types of network users, such as voice and data, since it is only the notion of statistical multiplexing, as in a computer network implemented by the store and forward method, which gives rise to these problems and mechanisms. Nevertheless, different algorithms for routing and flow control are called for because of the

different performance measures that characterize the delivery of data as opposed to voice. While data is permitted to experience long delay, its fidelity cannot be subject to compromise (i.e., errorless delivery); voice, on the other hand, is permitted to undergo relative degradation of fidelity, but it can tolerate only short delay if the continuity of the conversation is not to be impaired.

Thus, on top of the need to design voice flow-control and routing algorithms that are compatible and operate properly as one system, we are faced with the problem of interaction between two different congestion prevention systems, operating in the same network.

In this Thesis we set out to come up with algorithms, and improve on existing algorithms in the four areas of data routing, data flow control, voice routing and voice flow control. Though some of the improvements are quite elaborate, their motivation and the direction they took, were with an eye toward the ultimate goal of operating them concurrently in the same network.

1.2 Summary of Previous Work

1.2.1 Routing

Routing algorithms have been categorized as static, quasi-static or dynamic, and centralized or distributed. Attempts to route dynamically, i.e., a routing based on the instantaneous state of queues in the network, are few and not highly successful [45], [46]. The difficulty stems from the need for almost instantaneous communication and computation on the one hand, and very complex computation and control on the other. The static algorithms, by which inputs are fixed and one deals with flows rather than queues, were applied to communication networks after Kleinrock formulated routing as a multicommodity flow problem [19]. These algorithms are useful

in various applications as well as in communication networks (see [20]). Most of them are centralized and have a low rate of convergence.

Not long ago, Gallager came up with the notion of quasi-static routing [10]. Such a routing is done by an algorithm which operates "on line". It measures the current traffic parameters, carries out one or more iterations of a static algorithm using these parameters, implements the resulting routing variables on the networks, measures the resulting traffic parameters, and so on. If inputs were fixed, the algorithm would converge, while for slowly varying inputs the hope is that the algorithm will be able to "track" the variations and keep routing not far from optimal at all times.

Every static algorithm whose intermediate results improve routing and whose parameters are independent of the iteration number or the input levels, can serve as a quasi-static algorithm. The particular algorithm suggested by Gallager had the feature of distributed computation, and the routing variables used were fractions of flows decomposed by destinations. Gallager's algorithm has the disadvantages of slow convergence, and the need for a stepsize parameter, the proper value of which depends on the input levels.

In Bertsekas et al. [11] and O'Leary [23], improvements to Gallager's algorithm were suggested by introducing second derivative information. These algorithms tried to approximate Newton's method. Still, by upper bounding or neglecting Hessian cross terms, the problem of slow convergence persisted. Similar algorithms which operate in the space of path flows were suggested by Aashtiani [24] and Bertsekas [25].

All of the preceding algorithms tried to minimize an objective function, consisting of average delay or equivalently the number of outstanding

packets in the network. (More realistically, the objective function is some crude measure of congestion.) Another objective function which tries to alleviate bottleneck congestion leads to a formulation of the routing problem as a min-max linear program. Ros Peran [26] investigated the implementation of the simplex algorithm for this problem. Defenderfer [24] and Vastola [29] tried to solve the linear problem by a sequence of non-linear ones, achieving shared computation. But both algorithms are static; both have to carry subproblem optimization to which they have no bound to the number of iterations (i.e., no a priori knowledge of the amount of computation involved in one basic iteration). Computational results by Vastola [29] indicate that the routings resulting form the different objective functions are quite close.

Most of the research in routing has been done in the context of data. When voice is involved, the strict continuity requirements on the packet delivery makes routing on a virtual circuit almost a must. Routing on virtual circuits falls in the domain of integer programming for which analytical techniques are not as well developed as those for noninteger programming. But, when many conversations between each pair of nodes are assumed, one may approximate the problem by a continous one [30].

1.2.2 Flow-Control

Most of the flow control schemes are ad-hoc and hard to analyze. Many of them are buffer management schemes that depend on statistical variables such as queue length and need to be analyzed for a particular buffer length. Since we are interested in results that are general in nature and compatible with quasi-static routing, we focus attention on flow-control

algorithms that will curtail inputs even when nodal buffers are of infinite length.

Three such schemes are all tied to the notion of "permit". The first, the window scheme [21], restricts the number of permits for each origin-destination (OD) pair. The second, the isarithmic scheme [21], restricts the number of permits only for the network as a whole, and the third [22] uses queues of permits for each origin-destination pair.

The window scheme is implemented in today's ARPANET. Each origin-destination pair keeps an account of the packets sent. An origin sending a packet incurs a unit debt until the packet is acknowledged by the destination. Each OD pair is assigned a number, called a "window", of the maximum debt the source may incur. When congestion builds up, the acknowledgement arrival slows down and, as a result, the input rate of the source drops. When the number of conversations is too large, this drop may not be sufficient to alleviate the congestion.

The isarithmic scheme was proposed for the National Physical Laboratory network. A network is allocated a number of "permits", which are just small packets. The permits circulate randomly in the network. A packet may enter the network only if it "captures" a permit. When the packet arrives at the destination, the permit is released back to circulate in the network. Obviously, this scheme restricts the total number of outstanding packets in the network. There are two major difficulties with this scheme: controlling the distribution of permits in the network so that a single OD pair will not capture an "unfair" share of permits, and guarding against unintentional generation or loss of permits.

The third scheme was proposed recently by Kleinrock and Tseng [22]. In this scheme, each origin has a finite, usually small, buffer for permits.

Permits are generated in some prescribed Poisson rate, and are queued in the buffer. A permit which finds the buffer full is destroyed. A packet may enter the network after it captures a permit from the appropriate queue. That captured permit is destroyed. When a source tries to increase its rate too much, many packets will find empty permit queue and the rate will be restricted. This scheme is highly non-adaptive. It reacts to inputs but not to congestion.

In the case of voice, control must be much tighter, because only very small delays can be tolerated. Bially et al. [31] initiated a voice flow control scheme that is based on trading voice quality with packet length. Essentially, a packet can be shortened, maintaining intelligibility, although at a lower quality. In this particular suggestion, packets travelling on a congested link were shortened at the link's input, thus relieving the congestion. The waste in network resources caused by shortening a long packet after traversing some links is apparent.

Jaffe [32] and Hayden [33] independently suggested end to end schemes that overcome this difficulty. In essence, in both schemes the link which is the bottleneck divides its resources equally among all conversation traversing it. Hayden's schemes, unlike Jaffe's, is suitable for quasi-static operation, but, unfortunately, it suffers from bad transient behavior. In order to make this behavior less violent, large "safety coefficients" must be employed, thus wasting resources.

1.2.3 Combined Routing and Flow-Control

In both attempts to analyze flow-control and routing together, discussed below, flow-control was done using the window scheme.

In Gerla and Nilsson [34], minimum delay routing was analyzed in conjunction with fixed windows. It is claimed that minimum delay routing will maximize throughput. An algorithm is proposed to find the optimum, but no convergence proof is given.

Golestaani [35] was the first to view flow control and routing as a combined optimization problem in the space of routing together with inputs. In his scheme, after better routing and inputs are found, flow control variables which will give rise to the desired inputs are to be computed. This obviously assumes the knowledge of the delay function, an assumption that might be objectionable.

Gallager [12] has tried to dispense with the above assumption by using Golestaani's framework, but operating directly in the space of routing and flow control variables. The particular algorithm suggested is not complete in that a descent iteration is given that adjusts only the windown size variables.

Realizing this difficulty Gallager [36] suggested the use of a window for every pair of session-path. In such a case, the windows determine the path flows. This allows us to achieve routing and flow control together using only windows as control variables and thus avoiding the difficulty mentioned above. Still, the resulting algorithm has a poor rate of convergence. A more important side effect that followed from the above realization, was the observation made there [36], that a window for every pair of session-path may be the way to exercise dynamic routing.

1.2.4 Integrated Voice and Data Network

Some suggestions have been made as to the way of achieving this integration [37], [42]. Recently, Ibe [38] proposed the modeling of a

store and forward integrated network with voice packets having priority over data packets. This is done by considering voice and data as different "modes". It essentially transforms the network into one that consists of two duplications of the original network. One duplicate carries the voice, the other the data. Furthermore, the delay on a given link is also affected by the flow on the duplicate one. Ibe then uses Golestaani's scheme for the transformed network.

1.3 Summary of Accomplishments

1.3.1 Introduction

In the chapters that follow we propose four algorithms, two for routing and two for flow control. Which algorithm to choose for which user or network depends mainly on the underlying rules by which a particular network serves a particular user. Such rules for example might require the use of a virtual circuit per session or might require that once a session is in progress no rerouting will be done.

A basic distinction among the algorithms is whether they are system or user oriented. The former takes a global objective function and minimizes it. Put another way, it takes into account the amount of network resources a user utilizes. The latter is more concerned with the resulting service the end user perceives. This resulting service should have a fairness property to it. A generic example which brings forth the difference between the two, is the network depicted in Figure (1.A)

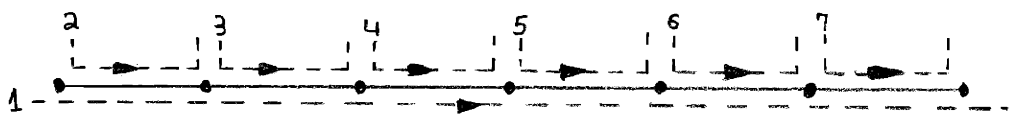


Figure (1.A)

There are seven users in the network. Six of them use only one link each, and user 1 uses all the six links. Let all link capacities be equal. Obviously if the network is highly utilized an increase in the a service to user 1 will come at the expense of the detrioriation of service to the other six users. A system oriented algorithm will take this into account when allocating resources, while a user oriented algorithm will not.

In this thesis we take the point of view that the different performance measures by which the delivery of voice as opposed to date is characterized, as well as the underlying network rules that might be implied, make the user oriented algorithms more suitable for voice and the others more suitable for data.

The reader is encouraged to look in each chapter for the abstract setting in which the chapter is cast, and view the voice or the data framework only as a possible instance of a user for whom this setting is suitable.

1.3.2 Voice Routing

We consider routing each voice session on a virtual circuit. The virtual circuit is fixed throughout the duration of the session. We show that provided the capacity required by each session is relatively small, placing an incoming session on the path of minimal marginal cost is optimal in an asymptotic sense. Moreover we consider the case where the path is not updated upon every new session arrival but is updated periodically. We show that this updating rule is also optimal in an asymptotic sense.

1.3.3 Voice Flow Control

We propose a bottleneck algorithm similar to Hayden's and Jaffe's. We show the difficulty that Hayden's algorithm encounters, and the reason

ours does not. We prove convergence and extend the algorithm beyond Jaffe's, in that we are able to handle nonlinearities.

1.3.4 Data Routing

We exploit the simple structure of the Hessian matrix of the cost function with respect to path flows as opposed to the space of fractions. This structure allows for the distributed multiplication of the Hessian matrix by a vector. Hence we show that we can compute the Newton direction and the Newton stepsize distributedly for the relaxed problem in which the positivity constraints on the path flows are removed.

We propose an algorithm by which we essentially project the relaxed direction on the constrained set. This projection can be done with respect to a metric which matches the simplex constraints involved in the multicommodity flow problem. The algorithm is a generalizaton of the one proposed by Bertsekas [6] for an orthant constraint.

We prove that the algorithm converges superlinearly with a stepsize of unity when started close enough to the optimum, similar to the pure Newton method. Together with the possibility of computing the Newton direction distributedly this makes the algorithm a prime candidate for quasi-static operation.

We state and analyze the algorithm in complete generality and then specialize to the multicommodity problem. In addition, we present computational results based on three sample metworks.

1.3.5 Data Flow-Control

We propose an iterative algorithm to adjust the parameters of the window flow-control scheme to achieve a desired input rate. This

algorithm, based on an observation made by Gallager [12], is guaranteed to converge under mild assumptions concerning the relation between link flow and link delay, and does not require the knowledge of this relation.

Furthermore, we propose a model to investigate the effect of flow control on routing. Using this model, we show that the assertion made in [36] about accomplishing dynamic routing using flow-control mechanisms, is plausible.

1.3.6 Integrated Network

We propose a way to integrate the four algorithms suggested in this Thesis in a network serving voice and data simultaneously. The joint flow-control and routing for data is made in the spirit of the one proposed by Golestaani [35]. The joint flow-control and routing for voice is made in a way that achieves a fair allocation of rates over the capacity allocated to it. We integrate the two joint flow-control and routing algorithms by giving priority to voice and proposing an algorithmic method to dynamically allocate capacity to the voice so as to achieve a tradeoff between voice rate, and data rate and delay. In some sense, this is similar to the algorithm proposed by Ibe [38].

2 Voice Routing*

2.1 Introduction

In this chapter we consider a network in which voice sessions are taking place between various origin-destination (or OD) pairs of nodes. Each actual two-party voice conversation is treated as two sessions. Each session is allocated a virtual-circuit (VC) at the time of setup, which lasts for the whole duration of the session. All sessions communicate at a certain fixed rate. The routing problem we address is to allocate paths to sessions at the time of set up, so as to minimize a certain objective function.

Specifically, we have in mind a large, ARPANET like, network in which an individual voice session is not significant. It is rather the sessions as a group that affect the network. More precisely, we assume that the communication capacity utilized by each session is small relative to the link capacity provided by the network; yet, the number of sessions between each OD pair is large enough to make the cumulative rate of communication between OD pairs significant.

We will analyze a routing rule tailored to this context. By this rule, all incoming sessions in an interval of time are assigned a fixed path. This path is changed periodically in order to adapt to the changing flow pattern in the network, as old sessions terminate and new sessions arrive. Thus this rule avoids long computation or the fast measurements that would be needed by a routing rule which would try to react to sessions on a more "individual" basis. Taking into account the fast rate of arrival we envi-

---

*The problem in this chapter was suggested by A. Segall

sage, it is apparent that doing session by session routing decisions is impractical.

The resulting situation is similar to the quasi-static routing of Chapter 4. The hope is that the variation of the flow pattern in the network affected by the terminating and arriving sessions is slow enough, relative to the update period, so that our rule reacts fast enough to keep the flow not far from the optimum at all times.

In the next section we present the objective to be achieved by the routing rule. We discuss the difficulty of analyzing any particular instance of the problem and subsequently motivate the analysis of a qualitative asymptotic behaviour. In the section that follows we present the routing rule, and in the last section we prove the result about its asymptotic behaviour.

2.2 The Voice Routing Problem

We consider a network consisting of N nodes 1,2,...,N and a set of directed links denoted by $\mathcal{L}$. We are given a set W of OD pairs. For each OD pair $w \in W$, we are given a set of directed paths P_w that start at the origin node and terminate at the destination node. Let λ_w denote the mean rate of session arrivals at OD pair $w \in W$, let γ denote the communication rate required by each session, and let each session have an average holding time μ^{-1}. It is assumed that each of the arrival processes is an independent Poisson process, and the holding time of each session is an independent exponentially distributed random variable.

Let the random variable $x^p(t)$ denote the number of sessions between OD pair $w \in W$, traversing path $p \in P_w$ at time t. To describe the evolution of $x^p(t)$ let

$$\{e^w_\lambda(s)\}_{\lambda \in [0,\infty)} \quad \text{and} \quad \overline{e}^p_i(s) \quad i=1,2,\ldots, \qquad \forall\, p \in P_w,\ w \in W,\ \forall\, s \geq 0$$

be a collection of independent stochastic processes taking value in {0,1} and satisfying for all $t_1 \geq t_2 \geq 0$

$$P[e_\lambda^w(t_1)=1/e_\lambda^w(t_2)=1]=P[\overline{e}_i^p(t_1)=1/\overline{e}_i^p(t_2)=1]=e^{-\mu(t_1-t_2)}$$

$$P[e_\lambda^w(t_1)=0/e_\lambda^w(t_2)=0]=P[\overline{e}_i^p(t_1)=0/\overline{e}_i^p(t_2)=0]=1$$

$$\forall\ \lambda\in[0,\infty),\ i=1,\ldots,x^p(0)\quad p\in P_w,\ w\in W.$$

Assume that at time zero the value of all the processes is one. Let $A_w(t)$ be the counting process describing session arrivals at $w\in W$, and let our control be the vector random variable $U(t)$ whose entries $u^p(t)$ $p\in P_w$, $w\in W$ satisfy

$$u^p(t)\in\{0,1\} \qquad \forall\ p\in P_w,\ w\in W,\ \forall\ t\geq 0 \qquad (2.2.1)$$

$$\sum_{p\in P_w} u^p(t) = 1 \qquad \forall\ w\in W\ \ \forall\ t\geq 0. \qquad (2.2.2)$$

Then, for all $p\in P_w$, we define $x^p(t)$ by

$$x^p(t) = \sum_{i=1}^{x^p(0)} \overline{e}_i^p(t) + \int_0^t u^p(s)\cdot e_s^w(t-s)\,d(A_w(s)) \qquad \forall\ w\in W\ \ \forall\ t\geq 0. \qquad (2.2.4)$$

Notice that if X denotes the set defined by (2.2.1) - (2.2.2) then because $U(t)\in X$

$$\sum_{p\in P_w} x^p(t) = \sum_{p\in P_w}\sum_{i=1}^{x^p(0)} \overline{e}_i^p(t) + \int_0^t e_s^w(t-s)\,d(A_w(s)) \qquad \forall\ w\in W,\ \forall\ t\geq 0 \qquad (2.2.3)$$

and, thus, the total number of sessions between any OD pair at any time is independent of our control.

For every link $a\in\mathcal{L}$ we can determine $f^a(t)$, the flow on link $a\in\mathcal{L}$ at time t, by the relation

$$f^a(t) = \sum_{w\in W} \sum_{p\in P_w} 1_p(a)x^p(t)\gamma \qquad \forall\ a\in\mathcal{L} \tag{2.2.5}$$

where $1_p(a)=1$ if path p contains the link a and $1_p(a)=0$ otherise. If we denote by x(t) and f(t) the vectors of the number of path sessions and link flows respectively, at time t, then we can write relation (2.2.5) as

$$f(t) = \overline{E}x(t)\cdot\gamma \tag{2.2.6}$$

where $\overline{E}$ is the arc-chain matrix of the network.

In modeling a packet switched network when μ^{-1} is long enough equation (2.2.6) will hold as an average over an interval of time around t. As will be described shortly, we are concerned with an "average" behaviour objective. In light of this objective writing (2.2.6) as an instantaneous relation is justified.

It is customary to take

$$D(f) = \sum_{a\in\mathcal{L}} D_a[f^a(t)] \qquad \forall\ t\geq 0$$

to be an instantaneous measure of congestion in the network, [35], where $D_a(\cdot)$ is a certain function to be specified later. Thus, a reasonable overall objective to be achieved by the voice routing is to minimize

$$\lim_{T\to\infty} E\ \frac{1}{T}\int_0^T D[f(t)]dt \tag{2.2.7}$$

over all $U(\cdot)$ such that $U(t)\in X$ where f(t) satisfies (2.2.5) and (2.2.3).

Even a very simple instance of problem (2.2.7), in which sessions arrive simultaneously at different OD pairs, one session at a certain OD

pair and all the others at another, with no later arrivals expected, is a very difficult problem. Indeed, by taking an appropriate function for $D_a(\cdot)$, $a \in \mathcal{L}$ it can be shown that the problem we face is equivalent to the two commodity flow problem shown by Even, Itai and Shamir [28] to be an NP-complete problem [42].

Nonsimplified instances of (2.2.7) are more difficult by an order of magnitude. Not only the "interaction" between paths are to be taken into account, but the different rate of arrivals to different OD pairs should play their role, too.

Yet, the simple instance of problem (2.2.7) above has a simple solution if the rate of communication of a session, γ, is taken to be small enough. The optimal routing decision in this case is to allocate the paths which yield minimal marginal costs, i.e. shortest paths with respect to $D_a'(\ )$, $a \in \mathcal{L}$. Since this is exactly the situation we have in mind, it is expected that routing on the marginal delay shortest paths is "robust" in the sense that it will yield a cost close to the optimal, even for the nonsimplified problem.

As for the choice of $D_a(f^a)$, $a \in \mathcal{L}$, in [35] the function

$$D_a(f^a) = \frac{f^a}{C_a - f^a} \tag{2.2.8}$$

is suggested for data. In Chapter 6 we will motivate another function suitable for voice. Noticing from (2.2.4) that we have no control over the total flow emanating from an origin, this choice of $D_a(f^a)$ will result in cost of infinity in (2.2.7) independent of the routing. This is due to the nonzero probability that the number of sessions currently in progress exceeds the network capacity. In practice, when congestion builds up, the

flow control mechanism will block new sessions from entering the network. Thus in order to deal with a meaningful routing problem, and yet, avoid the modeling of the flow control mechanism, we take $D_a(f^a)$, $a\epsilon\mathcal{L}$ to be a quadratic extrapolation to (2.2.8) by making the following assumption concerning $D_a(\cdot)$:

Assumption (2.A): For all $a\epsilon\mathcal{L}$, $D_a(\cdot)$ is convex twice continuously differentiable increasing on $[o,\infty)$ with

$$D_a'(0) \geq \hat{\eta} \qquad \forall\ a\epsilon\mathcal{L}$$

and

$$0 < \underline{\eta} \leq D_a''(f^a) \leq \overline{\eta} < \infty \qquad \forall\ f^a\epsilon[0,\infty),\ \forall\ a\epsilon\mathcal{L}.$$

Evidently, any quadratic extrapolation to (2.2.8) satisfies Assumption (2.A).

2.3 The Algorithm for Voice Routing

Following the observation made in the previous section we propose this routing rule: Every Δt units of time each OD pair updates its shortest path with respect to $D_a'(f^a(t))$ $a\epsilon\mathcal{L}$. In between updates all new incoming sessions are routed on the most recent shortest path, while a session which has arrived before, continue to use the path allocated to it upon arrival, until it terminates.

Mathematically, we choose $\vec{U}(t)\epsilon X$, with $\vec{u}^p(t)$ satisfying for $p\epsilon P_w$

$$\vec{u}^p(t)\epsilon\begin{cases}\{0,1\} & \text{if } p = \arg\min_{p\epsilon P_w}\{d^p(\left\lfloor\frac{t}{\Delta t}\right\rfloor\cdot\Delta t)\}\\ \{0\} & \text{otherwise}\end{cases} \tag{2.3.1}$$

where

$$d^p(t) = \sum_{a\epsilon\mathcal{L}} D_a'(f^a(t))\cdot 1_p(a) \tag{2.3.2}$$

and $\lfloor z \rfloor$ denotes the largest integer smaller than z.

With such a control the pair

$$\{x(k\Delta t + \delta^*), x(k\Delta t)\} \qquad \forall \ 0 \leq \delta^* \leq \Delta t \tag{2.3.3}$$

becomes an imbedded Markov process. Moreover, it is not difficult to see that it is positive recurrent and as a result a steady state distribution of x(t) exists. This implies that with the control $\vec{U}(\cdot)$ the limit in problem (2.2.7) is well defined.

The kind of result we will prove in the next section is that as the arrival and departure of actual flow to and from the network becomes smoother in a sense to be defined shortly, and Δt becomes smaller, the resulting value of expression (2.2.7) approaches the limit infimum that can be achieved by any control $\tilde{U}(\cdot)$ for which $\tilde{U}(t) \varepsilon X$ is a random variable for all $t \geq 0$.

Formally, we let

$$\lambda_w = \bar{\lambda}_w \cdot n \qquad , \qquad \gamma = \bar{\gamma}_w / n \tag{2.3.4}$$

for certain positive constant $\bar{\lambda}_w$ and $\bar{\gamma}_w$, and we state the following proposition below in terms of n. We require n large so that individual session arrivals do not imbalance the routing, and Δt to be small so that large number of session arrivals routed on the same path do not imbalance routing.

Propositon (2.A) Under Assumption (2.A) as $\Delta t \downarrow 0$ and $n \uparrow \infty$ the cost of (2.2.7) resulting from the control defined by (2.3.1) approachs the minimal possible.

2.4 Convergence Proof

We first find a lower bound to (2.2.7) and then show that the cost

resulting from the control $\vec{U}(\cdot)$ as $n\uparrow\infty$ and $\Delta t\downarrow 0$ approaches this bound.

To bound (2.2.7) for any control, we observe that by Jensen's inequality and Fubini's Theorem

$$\liminf_{T\to\infty} E\,\frac{1}{T}\int_0^T D(f(t))dt \geq \liminf_{T\to\infty} D\,(\frac{1}{T}\int_0^T E(f(t))dt). \tag{2.4.1}$$

On the other hand, from (2.2.4) using the facts that

$$\lim_{t\to\infty} \sum_{p\in P_w} \sum_{i=1}^{x^p(0)} \bar{e}_i^p(t) = 0 \ \text{(a.s)} \qquad \forall\, w\in W \tag{2.4.2}$$

and

$$\lim_{t\to\infty} E\int_0^t e_s^w(t-s)d(A_w(s)) = \lambda_w\cdot\mu^{-1} \qquad \forall\, w\in W \tag{2.4.3}$$

where (2.4.3) follows from Little's formula [40], we get that for each initial condition $x^p(0)$, $p\in P_w$, $w\in W$, and $\varepsilon>0$ there exist $\bar{t}$ such that for all $t\geq\bar{t}$

$$\frac{1}{t}\int_0^t [E\gamma\cdot \sum_{p\in P_w} x^p(t)]dt \geq \gamma\cdot\lambda_w\cdot\mu^{-1} - \varepsilon \qquad \forall\, w\in W,\ \forall\, t\geq\bar{t} \tag{2.4.4}$$

$$\frac{\gamma}{t}\int_0^t [Ex^p(t)]dt \geq 0 \qquad \forall\, p\in P_w,\ w\in W,\ \forall\, t\geq\bar{t}. \tag{2.4.5}$$

Consider the problem in $R^{|P|}$ where $|P| = \sum_{\omega\in W} |P_w|$ of

$$\text{minimize } D(\bar{Ey}) \tag{2.4.6}$$

subject to

$$\sum_{p\in P_w} y^p = \lambda_w\cdot\mu^{-1}\cdot\gamma \qquad \forall\, w\in W \tag{2.4.7}$$

$$y^p \geq 0 \qquad \forall\, p\in P_w,\ w\in W. \tag{2.4.8}$$

Denote its solution by $D(\bar{Ey})$. Then by Assumption (2.A) it is not hard to see that the solution $v(\varepsilon)$ to the problem

$$\min D(\bar{E}y)$$

subject to

$$\sum_{p\in P_w} y^p \geq \lambda_w \cdot \mu^{-1} \cdot \gamma - \varepsilon \qquad \forall\, w\in W$$

$$y^p \geq 0 \qquad \forall\, p\in P_w,\ w\in W$$

is "stable" [43] for $\varepsilon=0$ in the sense that the solution to the problem is continuous with respect to perturbations in the problem parameters, and that the equality in (2.4.7) can be replaced by an inequality. As a result we deduce that

$$\lim_{\varepsilon\downarrow 0} v(\varepsilon) = D(\bar{\bar{E}}\bar{y}). \tag{2.4.9}$$

Thus taking

$$\frac{1}{t}\int_0^t E(\gamma x^p(t))dt$$

in (2.4.4), (2.4.5) to play the role of y, we conclude

$$\liminf_{T\to\infty} E\,\frac{1}{T}\int_0^T D(f(t))dt \geq D(\bar{f}) \tag{2.4.10}$$

where

$$\bar{f} = \bar{E}\bar{y}\ .$$

Notice that since $D(\cdot)$ is strictly convex $\bar{f}$ is unique.

We now show that the cost resulting by using the control $\bar{U}(\cdot)$ approaches $D(\bar{f})$ as $n\uparrow\infty$ and $\Delta t\downarrow 0$.

To this end we investigate the behavior of the imbedded Markov process

$$\{x(k\Delta t)\}.$$

We take a convex Lyapunov function $V(x)$, such that $Q(\alpha) = V(\alpha(x_1-x_2))$ is strictly convex provided that

$$\overline{E}(x_1 - x_2) \neq 0$$

and for which

$$V^* = \min_{x \geq 0} V(x) = D(\overline{f}) \tag{2.4.11}$$

where for each x which attains the minimum we have

$$\overline{f} = \overline{E} x\gamma .$$

We then show that for $\overline{E}x(k\Delta t)\gamma$ outside a neighborhood, shrinking to a single point as $n \uparrow 0$ and $\Delta t \downarrow 0$, of $\overline{f}$ we have

$$E\,[V(x[(k+1)\Delta t]) \,/ x(k\Delta t)] \leq V(x(k\Delta t)) - \delta \tag{2.4.12}$$

while inside the neighborhood

$$E\,[V(x[(k+1)\Delta t]) \,/ x(k\Delta t)] \leq V^* + \delta \tag{2.4.13}$$

for some $\delta > 0$ which can be made as small as we wish as $n \uparrow \infty$ and $\Delta t \downarrow 0$.

Taking expected values in (2.4.12) and (2.4.13), and using the covexity of $V(x)$ and the strict convexity of $Q(\alpha)$ we conclude that

$$\lim_{\substack{k\to\infty, n\uparrow\infty, \Delta t\downarrow 0 \\ k\Delta t\to\infty}} f(k\Delta t) = \overline{f} \qquad \text{(in mean)} ,$$

which by (2.4.10) and the fact that similarly to (2.4.12) and (2.4.13) we have outside the neighborhood

$$E[V(x(k\Delta t + \hat{\delta})) \,/ x(k\Delta t)] \leq V(x(k\Delta t)) \qquad \forall\ 0 \leq \hat{\delta} \leq \Delta t,$$

while inside the neighborhood

$$E[V(x(k\Delta t + \hat{\delta})) \,/ x(k\Delta t)] \leq V^* + \delta \qquad \forall\ 0 \leq \hat{\delta} \leq \Delta t,$$

Let η_w, $w\epsilon W$ be the Lagrange multipliers to equations (2.4.7) in problem (2.4.6). It is well known [41] that the function $V(x)$ defined by

$$V(x) = D(\bar{E}x\gamma) + \sum_{w\epsilon W}\left[\eta_w\left(\sum_{p\epsilon P_w}\gamma x^p - \lambda_w\mu^{-1}\gamma\right) + \bar{\eta}\left(\sum_{p\epsilon P_w}\gamma x^p - \lambda_w\mu^{-1}\gamma\right)^2\right] \tag{2.4.14}$$

satisfies (2.4.11). Since in what follows $x^p \geq 0$ $\forall$ $p\epsilon P_w$ $w\epsilon W$, we consider only neighborhoods relative to the positive orthant. With respect to $V(x)$ we have the following lemmas, whose proofs are presented in Appendix (2.A).

Lemma (2.A): For $x(k\Delta t)$ such that $x^p(k\Delta t) \geq 0$ for all $p\epsilon P_w$, $w\epsilon W$ we have

$$E\left[\nabla' V(x(k\Delta t))\cdot\left(x[(k+1)\Delta t] - x(k\Delta t)\right) \Big/ x(k\Delta t)\right]$$

$$= (1 - e^{-\mu\Delta t})\nabla V(x(k\Delta t))'\cdot(\tilde{x} - x(k\Delta t))$$

where $\tilde{x}$ solves

$$\min \nabla V(x(k\Delta t))'(x - x(k\Delta t))$$

subject to

$$\sum_{p\epsilon P_w} x^p = \lambda_w\mu^{-1} \qquad \forall\ w\epsilon W$$

$$x^p \geq 0 \qquad \forall\ p\epsilon P_w,\ w\epsilon W\ .$$

Using the convexity of $V(x)$ we can immediately deduce the corollary that follows by the subdifferential relation:

$$V(y) \geq V(x) + \nabla V(x)'(y - x) \qquad \forall\ y,x.$$

Corollary to Lemma (2.A):

$$E\left[\nabla V(x(k\Delta t))'\cdot\left(x[(k+1)\Delta t] - x(k\Delta t)\right) \Big/ x(k\Delta t)\right] \leq$$

$$(1 - e^{-\mu\Delta t})\cdot[V^* - V(x(k\Delta t))]$$

Lemma (2.B):

$$E\left[\|x[(k+1)\Delta t] - x(k\Delta t)\|^2 \Big/ x(k\Delta t)\right] \leq$$

$$\sum_{w\in W}\sum_{p\in P_w}\{[x^p(k\Delta t)(1-e^{-\mu\Delta t})]^2 +$$

$$x^p(k\Delta t)e^{-\mu\Delta t}(1-e^{-\mu\Delta t})\} +$$

$$\sum_{w\in W}\left\{\lambda_w\mu^{-1}\frac{1-e^{-\mu\Delta t}}{\mu\Delta t}^2 + (\lambda_w\mu^{-1})^2(1-e^{-\mu\Delta t})^2 +\right.$$

$$\left.\lambda_w\mu^{-1}(1-e^{-\mu dt})\left[1-\frac{1-e^{-\mu\Delta t}}{\mu\Delta t}\right]\right\} .$$

Now, by Assumption (2.A) and the definition (2.4.12) it is not difficult to see that $\nabla V(x)$ is Lipschitz continuous with module $\overline{\eta}\gamma^2\|\overline{E}\|^2$, where

$$\|\overline{E}\| = \max\{\|\overline{E}y\| \mid \|y\| \leq 1\}$$

(notice that $\|E\|\geq 1$) and therefore by the Taylor Expansion

$$E\left[V[x((k+1)\Delta t)] - V[x(k\Delta t)]\Big/x(k\Delta t)\right] \leq$$

$$E\left[\nabla' V(x(k\Delta t))\cdot(x((k+1)\Delta t) - x(k\Delta t))\Big/x(k\Delta t)\right] +$$

$$\frac{\overline{\eta}}{2}\|\overline{E}\|^2\gamma^2\cdot E\left[\|x[(k+1)\Delta t] - x(k\Delta t)\|^2\Big/x(k\Delta t)\right].$$

Let $\overline{M}$ be given by

$$\overline{M} = \min_y \sum_{w\in W}\{\eta_w\left(\sum_{p\in P_w}\gamma y^p - \lambda_w\cdot\mu^{-1}\gamma\right) + \overline{\eta}\left(\sum_{p\in P_w}\gamma y^p - \lambda_w\mu^{-1}\gamma\right)^2\}$$

then by Assumption (2.A) and (2.4.14)

$$\sum_{w\in W}\sum_{p\in P_w}\gamma\cdot x^p(k\Delta t) \leq \frac{1}{\underline{\eta}}V(x(k\Delta t)) - \overline{M} \qquad \forall\ k\geq 0 \qquad (2.4.15)$$

$$\sum_{w\in W}\sum_{p\in P_w}\gamma^2\cdot x^p(k\Delta t)^2 \leq \frac{1}{\underline{\eta}}V(x(k\Delta t)) - \overline{M} \qquad \forall\ k\geq 0. \qquad (2.4.16)$$

Using the Corollary to Lemma (2.A), Lemma (2.B) and (2.4.15)-(2.4.16) in the expansion above we get

$$E\{V[x([k+1]\Delta t)] - V(x(k\Delta t))/x(k\Delta t)\} \qquad (2.4.17)$$

$$\leq (1 - e^{-\mu\Delta t})[V^* - V(x(k\Delta t))] +$$

$$\frac{\overline{\eta}}{2}\|\overline{E}\|^2 \quad \frac{(1 - e^{-\mu\Delta t})^2}{\underline{\eta}} + \frac{(1 - e^{-\mu\Delta t})e^{-\mu\Delta t}\cdot\gamma}{\hat{\underline{\eta}}}\ (V(x(k\Delta t)) - \overline{M})$$

$$+ \sum_{w\in W}\left[(\lambda_w\gamma\mu^{-1})\frac{(1 - e^{-\mu\Delta t})^2}{\mu\Delta t}\cdot\gamma + (\lambda_w\gamma\mu^{-1})^2(1 - e^{-\mu\Delta t})^2\right.$$

$$\left.+ (\lambda_w\gamma\mu^{-1})(1 - e^{-\mu\Delta t})(1 - \frac{1 - e^{-\mu\Delta t}}{\mu\Delta t})\cdot\gamma\right]\Bigg\}$$

$$= (1 - e^{-\mu\Delta t})\left\{V^* - V(x(k\Delta t))\cdot\left[1 - \frac{\overline{\eta}}{2}\|\overline{E}\|^2\left(\frac{1 - e^{-\mu\Delta t}}{\underline{\eta}}\right.\right.\right.$$

$$\left.\left.\left.+ \frac{\gamma\cdot e^{-\mu\Delta t}}{\eta}\right)\right] - \frac{\overline{\eta}}{2}\|\overline{E}\|^2\left\{\left[\frac{1 - e^{-\mu\Delta t}}{\underline{\eta}} + \frac{\gamma\cdot e^{-\mu\Delta t}}{\underline{\eta}}\right]\overline{M}\right.\right.$$

$$+ \sum_{w\in W}\left[(\lambda_w\gamma\mu^{-1})\cdot\gamma + (\lambda_w\gamma\mu^{-1})^2(1 - e^{-\mu\Delta t})\right.$$

$$\left.\left.\left.\left.+ (\lambda_w\gamma\mu^{-1})(1 - \frac{1 - e^{-\mu\Delta t}}{\mu\Delta t})\cdot\gamma\right]\right\}\right\}$$

$$\leq (1 - e^{-\mu\Delta t})\left[V^* - V(x(k\Delta t))(1 - c_1(n,\Delta t)) + c_2(n,\Delta t)\right]$$

where, using $\lambda_w\gamma = \overline{\lambda}_w\cdot\overline{\gamma}$ for all n,

$$\lim_{\substack{\Delta t \downarrow 0 \\ n \uparrow \infty}} c_1(n,\Delta t) = \lim_{\substack{\Delta t \downarrow 0 \\ n \uparrow \infty}} c_2(n,\Delta t) = 0.$$

This follows because

$$\lim_{n \uparrow \infty} \gamma = 0, \quad \lim_{\Delta t \downarrow 0} (1 - e^{-\mu \Delta t}) = 0.$$

Consider the set

$$F(\varepsilon) = \{f \mid f = \bar{E} x \gamma \quad V(x) \leq V^* + \varepsilon \quad x^p \geq 0 \quad \forall\, p \in P_w \quad w \in W\}.$$

By the construction of V we have that for any $\delta > 0$ there exist $\bar{\varepsilon}$ such that for all $\varepsilon \leq \bar{\varepsilon}$

$$\{f \mid \| \bar{f} - f \| < \delta\} \supset F(\varepsilon).$$

Now, if we take $\varepsilon \leq \bar{\varepsilon}$, n large enough and Δt small enough such that

$$\frac{V^* + c_2(n,\Delta t)}{1 - c_1(n,\Delta t)} \leq V^* + \frac{\varepsilon}{2},$$

we conclude that for all $x(k\Delta t)$ such that $f \notin F(\varepsilon)$, $f = \bar{E} x(k\Delta t)\gamma$ we have

$$E\,[V(x([k+1]\Delta t) - V(x(k\Delta t))/x(k\Delta t)] \leq -(1 - e^{-\mu dt}) \cdot \frac{\varepsilon}{2} \qquad (2.4.18)$$

Also, relation (2.4.17) implies that

$$\lim_{\substack{\Delta t \downarrow 0 \\ n \uparrow \infty}} E\,[V(x([k+1]\Delta t)) - V(x(k\Delta t))/x(k\Delta t)] = 0 \qquad (2.4.19)$$

uniformly in $x(k\Delta t)$ in a bounded set. Relations (2.4.18) and (2.4.19) prove the propositon. Q.E.D

3. Voice Flow-Control

3.1 Introduction

Recently a group at Lincoln Laboratory [31] has introduced an interesting Voice Coder (Vocoder) scheme. The proposed Vocoder uses a digitization method, dubbed "embedded coding", which was first proposed by the Naval Research Laboratory [47]. Essentially, a segment of talkspurt is coded into packets of different "priority" levels. The higher "priority packets contain the "core" of the speech while the lower priority packets contain the information that "fine tunes" it. This coding schemes allows for the implementation of a sophisticated flow control mechanism.

While traditional voice flow control mechanisms use blocking either by preventing the initiation of a call or by discarding small segments of it when the call is already in progress, the "embedded coding" scheme allows for the alleviation or prevention of congestion by dynamically trading off between voice quality and congestion, by discarding the lower "priority" packets either at the point of congestion or the point of entry. Thus, in contrast with the traditional schemes which convey only part of the message, each part at high quality, the "embedded coding" scheme preserves the continuity of the speech with some overall degradation. The level of congestion at which the gaps between the segments, delivered by the traditional schemes, render the speech unintelligible is much lower than the one at which the "embedded coding" scheme delivers unintelligible information. This flexibility in exercising flow control makes the embedded coding scheme attractive.

Alleviation and prevention of congestion by discarding lower priority packets at the point of entry seems to be superior to discarding them at

the point of congestion. The later amounts to a waste of network resources. But, it would not be advisable to forgo the capability of discarding lower priority packets at the point of congestion, because of the time delay involved in making the entry points aware of the congestion build-up situation. As a result, we advocate the use of the two capabilities in complementary roles. The rates at the entry points will be determined upon longer time averages of congestion levels while the capability of discarding packets at the point of congestion will serve to alleviate intolerable momentary congestion. The rates at the entry point will be adjusted so that the capability of discarding packets at the point of congestion will not be exercised too often. This is analogous to the complementary roles of quasi-static routing and window flow-control of data.

In this chapter we discuss a method of determining the input rates at the entry point. To this end, we will ignore the capability of discarding packets within the network in order to avoid the "interaction" issues. These issues will be taken up in the final chapter.

As in quasi-static routing, we are looking for an algorithm that will adapt the input rate to the changing flows in the network resulting from the initiation and termination of sessions. As in quasi-static routing, we employ an "on line" iterative algorithm that will solve a static problem. The hope is that the algorithm converges fast enough relative to the sessions initiation and termination process, and as a result will be able to "track" its variation, keeping the rates in the ballpark of the optimal rates at all times.

The criterion used to determine input rates is one of the central issues in this chapter. In Section 3.2 we introduce the notion of "fair

allocation". In Section 3.3 we present two previous algorithms after which our algorithm is fashioned. In the following section we present and motivate the exact mathematical problem we intend to solve. Next, we introduce the algorithm, prove its convergence, and finally discuss issues arising from the need to implement the algorithm in a distributed manner.

Keeping in the back of our mind the compatibility issues among the algorithms presented in this Thesis, we discuss the input rate allocation problem in the context whereby routing for each session is done on a virtual circuit, fixed for the duration of the session. A heuristic virtual circuit allocation rule was discussed in the previous chapter.

3.2 Fair Allocation

In this section we discuss the criterion for determining input rates. Although the total of the input rates that should be allocated might be debatable, being a tradeoff between throughput and delay, it is undebatable that the individual session rate within the generally limited total rate, should be allocated in a fair manner.

It is customary to consider, as one of the characteristics of a fair allocation, the feature that it is indifferent to the geographical separation of the session's origin and destination. Although there might be different priorities assigned to sessions, these priorities are not assigned on the basis of geographical separation. Moreover, two sessions of the same priority should obtain the same rate, if the rate of one can be traded for the rate of the other. This is in the spirit of making the network "transparent" to the user. The user should have no idea of the length of the path assigned to him through the rate allocated to him.

To capture the notions of fairness and priority as presented above, we define the notion of "fair allocation":

Let Q be a totally ordered set (i.e. for all aeQ, beQ, $a \neq b$ we have $a<b$ or $a>b$) and let X be a given subset of Q^n. A vector $b = (b^1,\ldots,b^n)'$ is said to be lexicographically less than or equal to the vector $d = (d^1,\ldots,d^n)'$ if $b^i > d^i$ implies the existence of $j < i$ such that $b^j < d^j$. The vector $x = (x^1,\ldots x^n)' \in X$ is called a fair allocation of x over X if for each $y \in X$ there exists a permutation $\tilde{x}$ of x which is lexicographically greater than or equal to all permutations $\tilde{y}$ of y.

If we consider the set X as a "feasible" set, a fair allocation vector x over X solves an hierarchy of nested problems. The first one maximizes the minimal entry of vectors in X. The second maximizes the second minimal entry of all vectors which solve the first problem, etc.

The usual difficulty with such a problem is that in order to solve the $j^{\underline{th}}$ subproblem in the hierarchy, the solutions to the preceding subproblems have to be available. This was the case in [26] where a fair allocation was sought by solving a nested sequence of linear programs. It turns out in our case, that an iterative algorithm has an advantage in the sense that all the subproblems in the hierarchy can be solved simultaneously. This can be explained by the continuity of the solution of the $j^{\underline{th}}$ subproblem in all the preceding solutions. A more detailed explanation will be given within the convergence proof to an algorithm we will present in Section 3.4.

3.3 The Problem and Previous Work

Let N denote a network with nodes 1, 2,...,N and let $\mathcal{L}$ be a set of directed links connecting the nodes. With each link $a \in \mathcal{L}$ we associate a number c_a, called the capacity of link a. Let S denote a set of sessions taking place between nodes. Each session $s \in S$ has an origin node, a

destination node and a simple path p_s leading from the origin node to the destination node. Define

$$1_{p_s}(a) = \begin{cases} 1 \text{ if } a \text{ belongs to } p_s \\ 0 \qquad \text{otherwise.} \end{cases} \tag{3.3.1}$$

The kind of problem we deal with in this chapter is to allocate to each session $s\varepsilon S$ a rate γ^s, such that the allocation satisfies a given criterion.

Hayden [33] proposed a quasi-static distributed algorithm which results in a vector $\overline{\gamma} = (\ldots,\overline{\gamma^s},\ldots)'$ which is a fair allocation over the set of all vectors $(\ldots,\gamma^s,\ldots)'$ such that

$$\sum_{s\varepsilon S} \gamma^s \cdot 1_{p_s}(a) \leq \rho \cdot c_a \qquad \forall\, s\varepsilon S,\ \forall\, a\varepsilon \mathcal{L} \tag{3.3.2}$$

and where $0<\rho<1$ is a certain constant, usually taken to be 0.8. Jaffe [32] proposed a distributed nonquasi-static algorithm resulting in a vector $\overline{\gamma}$ such that the vector $(\ldots,\overline{\gamma^s}/\beta^s,\ldots)'$ is a fair allocation over the set of all vectors $(\ldots,\gamma^s/\beta^s,\ldots)'$ such that

$$\gamma^s/\beta^s \cdot 1_{p_s}(a) \leq c_a - \sum_{t\varepsilon S} \gamma^t 1_{p_t}(a) \qquad \forall\, s\varepsilon S,\ \forall\, a\varepsilon \mathcal{L} \tag{3.3.3}$$

and where β^s is some constant associated with session $s\varepsilon S$.

The rationale behind (3.3.2) is quite simple: we do not allow the total flow on each link to occupy more than some fraction of the total capacity. The rationale behind (3.3.3) is more sophisticated. Primarily, it allows us to accommodate fluctuations of a session rate which is a function of the rate, and in addition, it enables us to establish preferences among sessions.

While Jaffe's algorithm is not iterative and not suitable for quasi-static operation, Hayden's may result in transient flows that are much much larger than the capacity available to accommodate them. Since the last point may not be obvious at first glance we show it by an example. We first state his algorithm.

Let n_a be the number of sessions traversing link $a \in \mathcal{L}$. Then γ_k^s is determined by the iterative algorithm

$$R_a^k = R_a^{k-1} + \frac{1}{n_a}\left[c_a - \sum_{t \in S} \gamma_{k-1} 1_{p_t}(a)\right] \qquad \forall\, a \in \mathcal{L} \qquad (3.3.4)$$

$$\gamma_k = \min_{a:1_{p_s}(a)=1} R_a^k \qquad (3.3.5)$$

where R_a^o for all $a \in \mathcal{L}$ is an arbitrary number.

Consider the Figure (3.1):

There are 7 sessions in the network. Five of them originate at nodes 1 to 5 and terminate at node 12. One originates at node 1, traverses the zig-zag path, and terminates at node 6, and the last one originates at node 11 and terminates at node 12,

Let all capacities be c aside from link 11-12 with capacity 3.5c. A fair allocation $\bar{\gamma} = (\ldots, \bar{\gamma}^s, \ldots)'$ over the set defined by (3.3.2), with $\rho = 1$ is c/2 to all sessions, aside from session (11-12) which gets a rate of c. The steady state R's are

$$R_{(11,10)} = R_{(2,9)} = R_{(3,8)} = R_{(4,7)} = R_{(5.6)} = c/2$$

all other R's equal c. Immediately following the withdrawal of session (1-6) all R's will be c, according to (3.3.4). By (3.3.5) this will result in a total rate of 6c over link (11,12) which can accomodate only half of it.

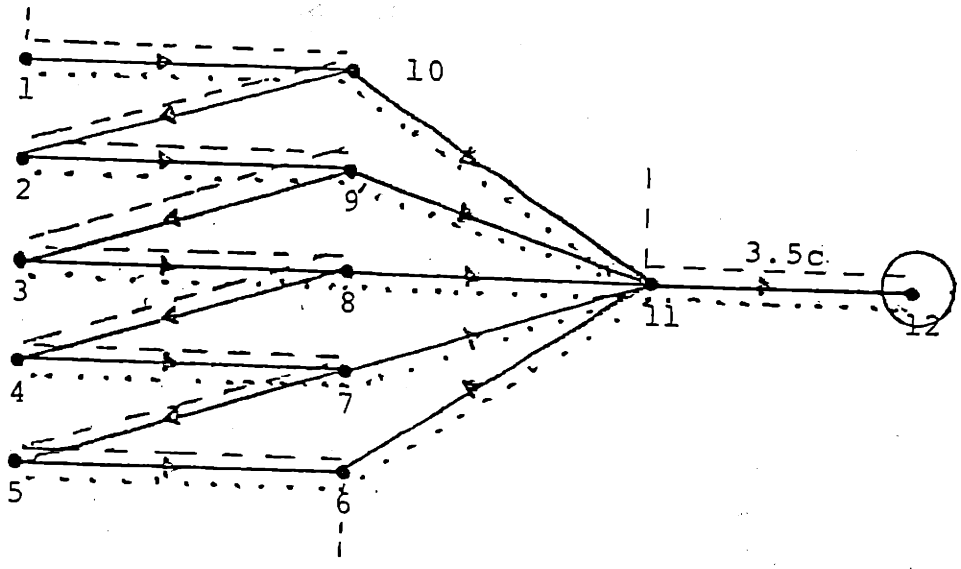


Figure 3.1

No reasonable correction factor such as ρ in (3.3.2) can help here. The problem stems from determining only one number for each link. When a session changes its rate, it changes it with no regard to the particular rate it had before.

We generalize the set defined by (3.3.3) in the following way:

Let $g_a: R^+ \to R^+$ be a function associated with link $a \in \mathcal{L}$. Let $f_s: R^+ \to R^+$ be a function associated with session $s \in S$ and which possesses an inverse f_s^{-1}. We are interested in a quasi-static algorithm which will result in a vector $\bar{\gamma}$ such that the vector $(\ldots, f_s^{-1}(\bar{\gamma}^s), \ldots)'$ is a fair allocation over the set of all vectors $(\ldots, f_s^{-1}(\gamma^s), \ldots)'$ such that

$$f_s^{-1}(\gamma^s) \cdot 1_{p_s}(a) \leq g_a(c_a - \sum_{t \in S} \gamma^t \cdot 1_{p_t}(a)) \cdot 1_{p_s}(a) \qquad (3.3.6)$$

$$\forall\, s \in S,\ \forall\, a \in \mathcal{L},$$

$$\sum_{t \in S} \gamma^t \cdot 1_{p_t}(a) \leq c_a \qquad \forall\, a \in \mathcal{L}, \qquad (3.3.7)$$

$$\gamma^s \geq 0 \qquad \forall\, s \in S. \qquad (3.3.8)$$

To see the role of g_a, assume that f_s is the identity function. Depending on the length of time over which the rate of a session is measured, we can have two interpretations of the role of our algorithm. Both interpretations suggest the same type of form for the function g_a.

In the first interpretation, the length of time over which the rate is averaged is relatively short with respect to the "time constant" of the counting process of the number of off-hook speakers in talkspurt mode. In this case we just deal with and worry about accommodating the speakers which are currently in the talkspurt mode. Since about 30% of a talkspurt is silence and some segments of the talkspurt need more encoding than others, we view the bit rate generated by the Vocoder for session $s \in S$ as a

stochastic process with mean γ^s - the rate assigned to user $s\varepsilon S$. This amounts to the assumption that the Vocoder has the means of dynamically reconfiguring to the demands of the voice to achieve the desired average rate.

Suppose that we want to reserve excess capacity on each link so as to be able to accommodate at least a variation as large as the standard deviation of the flow on the link. Assume that the standard deviation of the rate of a session $s\varepsilon S$ which was allocated an average rate γ^s is $\beta\cdot\gamma^s$ for some $0<\beta<1$. Let $s'\varepsilon S$ be such that

$$s' = \arg\max_{t\varepsilon S} \left\{ \gamma^t \cdot 1_{p_t}(a) \right\}, \tag{3.3.9}$$

then, by the independence of rates of different sessions we get (abusing notations)

$$\begin{aligned} \sigma\left(\sum_{t\varepsilon S} \gamma^t \cdot 1_{p_t}(a) \right) &\leq (c_a/\gamma^{s'})^{1/2}\ \sigma(\gamma^{s'}) \qquad (3.3.10)\\ &\leq (c_a/\gamma^{s'})^{1/2} \beta\cdot\gamma^{s'} \leq (c_a)^{1/2}\beta(\gamma^{s'})^{1/2} \\ &\leq \beta\cdot(c_a)^{1/2}\cdot\ g_a^{1/2}(c_a - \sum_{t\varepsilon S} \gamma^t \cdot 1_{p_t}(a)) \end{aligned}$$

where the last inequality follows from (3.3.6) with $f_s = 1$. Thus if we take

$$g_a(\cdot) = \frac{1}{\beta^2 c_a}(\cdot)^2 \qquad \forall\ a\varepsilon\mathcal{L} \tag{3.3.11}$$

we are guaranteed to accommodate the standard deviation of the flow resulting from the fair allocation.

In the second interpretation, the length of time over which the rate is averaged is relatively long with respect to the "time constant" of the

counting process of the number of off-hook speakers in talkspurt mode. In this case we deal concurrently with all the off-hook sessions and want to be able to accommodate the standard deviation around the mean of the process (i.e. the instantaneous effect of the number of speakers at the talkspurt mode is washed out by the long time average). Let q be the fraction of time a speaker is in the talkspurt mode and assume his rate while in the talkspurt mode is constant. Then using notations as before

$$\sigma\left(\sum_{t\in S}\gamma^t\cdot 1_{p_t}(a)\right) \leq \sum_{t\in S}\left[\left(\gamma^t/q\right)^2\cdot q(1-q)\cdot 1_{p_t}(a)\right]^{1/2} \qquad (3.3.12)$$

$$\leq \left(c_a/\gamma^{s'}\right)^{1/2}\cdot\gamma^{s'}\,\frac{[q(1-q)]^{1/2}}{q}$$

$$\leq \left(\frac{1-q}{q}\,c_a\right)^{1/2} g_a^{1/2}\left(c_a - \sum_{t\in S}\gamma^t 1_{p_t}(a)\right)$$

and we get a similar result as before for $g_a(\ )$.

The point we want to make by the above arguments is the need to allow g_a to be a nonlinear function, which may depend on c_a, rather than only on the excess capacity as (3.3.3) implies. The exact role of g_a is up to the network manager to decide, and our formulation allows him a great deal of flexibility in this regard.

To understand the role of f_s, notice that if $f_{s_1}^{-1}(\gamma^{s1})$ and $f_{s_2}^{-1}(\gamma^{s2})$ are entries of the vector of fair allocation and γ^{s1} can be traded for γ^{s2} then the ratio of γ^{s2}/γ^{s1} will be governed by $f_{s_1}^{-1}$ and $f_{s_2}^{-1}$. Taking these functions to be nonlinear allows changing the ratio of the rates as a function of the actual rates γ^{s1} and γ^{s2}. Without advocating a particular preference whether to get the ratio toward unity as one session approaches the threshold of intelligibility, or not, having the possibility of using a nonlinear function in this regard too, may be convenient in

various applications.

As will be explained in the final section, in order to carry out the algorithm of the next section we have to store in link $a\varepsilon\mathcal{L}$, the functions g_a, and f_s for all s traversing a. This is not too difficult if there are few "priority" classes of sessions and correspondingly few possibilities for f_s. All that a link has to know in this case is merely the class number of each session traversing it.

3.4 The Algorithm

We will state and prove the convergence of the algorithm in a centralized context. Distributed implementation will be discussed in the final section.

Assume that γ_k^s is given for all $s\varepsilon S$ and that

$$0 \leq \gamma_k^s, \qquad \sum_{t\varepsilon S} \gamma_k^t \cdot 1_{p_t}(a) < c_a \qquad \forall\, a\varepsilon\mathcal{L},\ \forall\, s\varepsilon S \tag{3.4.1}$$

then γ_{k+1}^s is determined by

$$\gamma_{k+1}^s = \min_{a:1_{p_s}(a)=1} \left[\gamma_k^s + \alpha_k^a \left(f_s g_a [c_a - \sum_{t\varepsilon S} \gamma_k^t \cdot 1_{p_t}(a)] - \gamma_k^s \right) \right] \qquad \forall\, s\varepsilon S \tag{3.4.2}$$

where α_k^a is to be specified later and $f_s g_a(\cdot)$ denotes $f_s(g_a(\cdot))$.

We make the following assumptions concerning g_a and f_s:

Assumption (3.A): $g_a(\cdot)$ and $f_s(\cdot)$ for all $a\varepsilon\mathcal{L}$ and $s\varepsilon S$ are monotonically non-decreasing.

Assumption (3.B): $f_s g_a(\cdot)$ is convex (concave is possible too, but will not be pursued) differentiable with

$$f_s g_a(0) = 0$$
$$0 < f_s g_a(c_a) \triangleq m_{sa} < \infty$$

for all $s\varepsilon S$ and $a\varepsilon\mathcal{L}$.

There are two options for choosing α_k^a.

The first

$$\alpha_k^a = \frac{1}{1 + \frac{\sum_{t\in S}[m_{ta} - f_t g_a(c_a - \sum_{u\in S}\gamma_k^u 1_{p_u}(a))]\cdot 1_{p_t}(a)}{\sum_{t\in S}\gamma_k^t 1_{p_t}(a)}} \qquad (3.4.3)$$

$$\forall\, a\in\mathcal{A},\ k=1,2,\ldots$$

and the second

$$\alpha_k^a = \frac{1}{1 + \sum_{t\in S}(f_t g_a)'[c_a - \sum_{u\in S}\gamma_k^u 1_{p_u}(a)]\cdot 1_{p_t}(a)} \quad . \qquad (3.4.4)$$

The step size (3.4.3) is not defined for

$$\sum_{t\in S}\gamma_k^t 1_{p_t}(a) = 0.$$

In such a case we use the step size (3.4.4). It will be evident later that this amounts to defining (3.4.3) by taking limits.

For the two options of step sizes (3.4.3), (3.4.4) we have the following lemma whose proof is relegated to Appendix (3.A).

Lemma (3.A): Let γ_k satisfy (3.4.1) and let γ_{k+1} be determined from γ_k by (3.4.2) with α_k^a as in (3.4.3) or (3.4.4). Then under Assumptions (3.A) - (3.B)

$$0 < \gamma_{k+1}^t,\ \sum_{t\in S}\gamma_{k+1}^t\cdot 1_{p_t}(a) < c_a \qquad \forall\, t\ S,\ \forall\, a\in\mathcal{A} \qquad (3.4.5)$$

and

$$\limsup_{k\to\infty}\sum_{t\in S}\gamma_k^t 1_{p_t}(a) < \sum_{t\in S} f_t g_a\left[c_a - \limsup_{k\to\infty}\sum_{u\in S}\gamma_k^u\cdot 1_{p_u}(a)\right]\cdot 1_{p_t}(a) \qquad (3.4.6)$$

$$\forall\, a\in\mathcal{A}.$$

The continuity of the R.H.S. of (3.4.6) in c_a implies the following corollary.

Corollary to Lemma (3.A): Under the conditions of Lemma (3.A) if c_a is replaced by a sequence $\{(c_a)_k\}$ such that (3.4.5) is satisfied at each iteration and

$$\lim_{k\to\infty} (c_a)_k = \bar{c}_a$$

then (3.4.6) holds with c_a replaced by $\bar{c}_a$.

The idea behind the choice of the expressions (3.4.3) and (3.4.4) as well as the simple intuition behind Lemma (3.A) can be best explained by the use of Figures (3.2) and (3.3). Let F_k^a denote

$$\sum_{s\in S} \gamma_k^s 1_{p_s}(a)$$

and let the function $G_a(\cdot):R^+\to R^+$ denote

$$\sum_{s\in S} f_s g_a(c_a - (\cdot))1_{p_s}(a).$$

The two figures depict the relations between F_k^a and F_{k+1}^a, when the network consisted of the single link a. When this is not the case, (3.4.2) implies that we have at most over estimated F_{k+1}^a. In Fig. (3.2), which corresponds to α_k^a as in (3.4.3), F_{k+1}^a is determined by intersecting the line connecting the point $(o,G_a(o))$ with the point $(F_k^a,G_a(F_k^a))$, with the line $y = F^a$. In Fig. (3.3), which corresponds to α_k^a as in (3.4.4), F_{k+1}^a s determined by intersecting the tangent to the graph of $y = G_a(F_k^a)$ at the point $(F_k^a, G(F_k^a))$, with the line $y=F^a$. The reader can easily convince himself the $\limsup_{k\to\infty} F_k^a$ must, in both cases, lie in the area where

$$F^a \leq G_a(F^a)$$

which gives rise to the lemma.

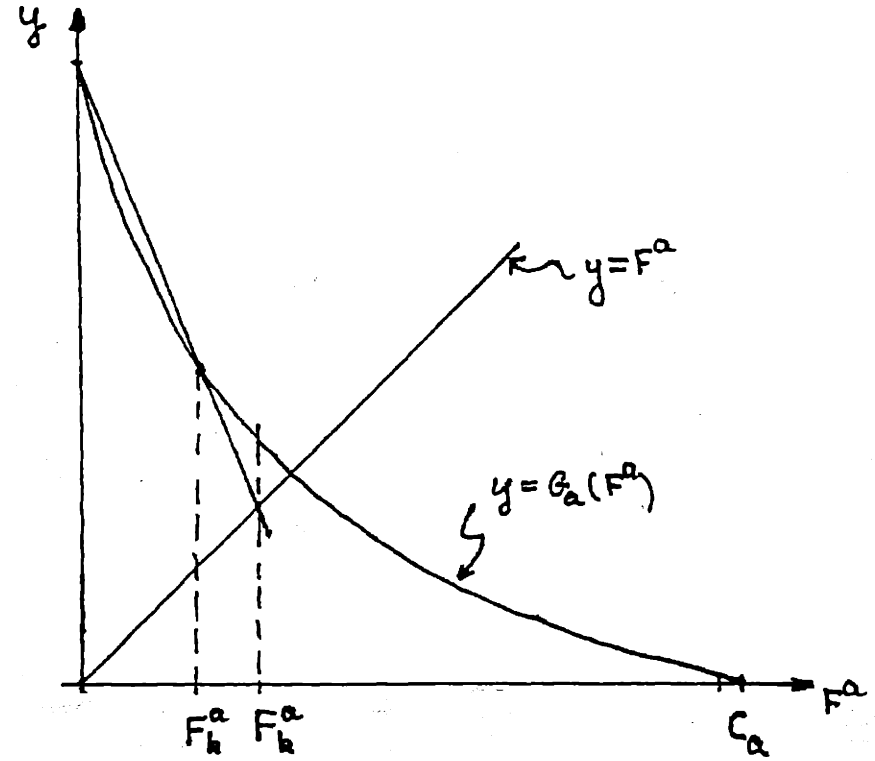


Figure 3.2

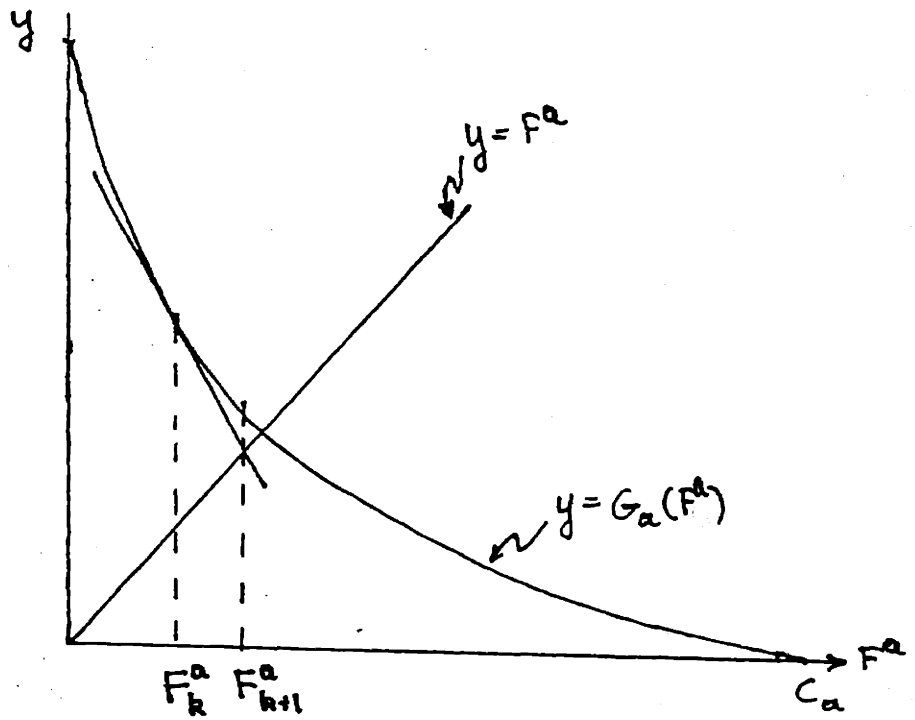


Figure 3.3

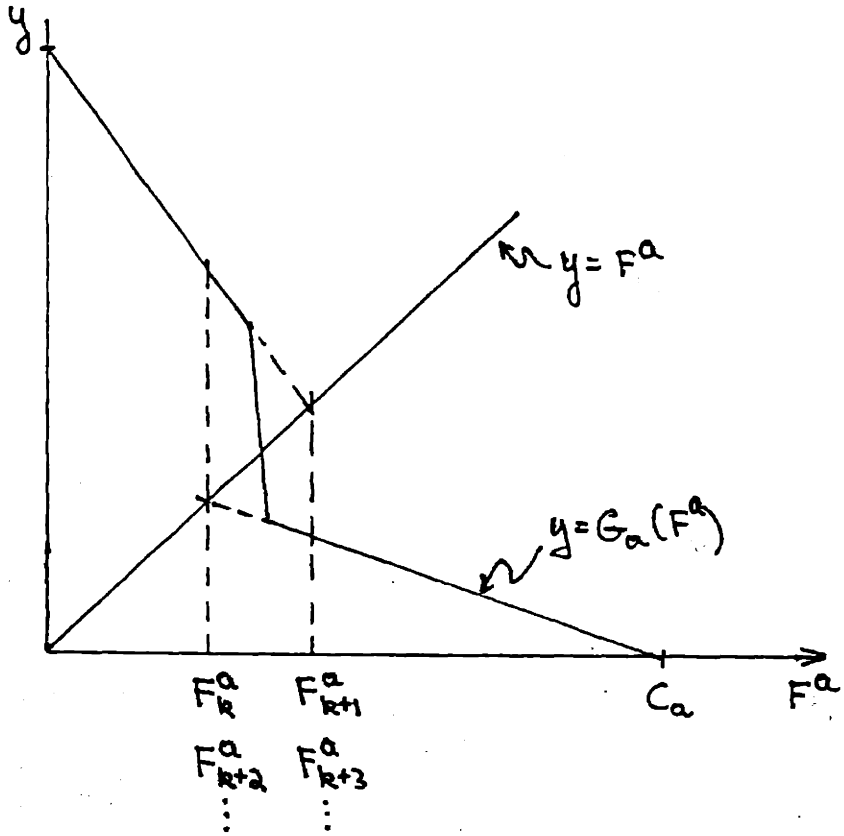


Figure 3.4

As for the possible functions for f_s and g_a, Assumption (3.B) is not too restrictive since it will be satisfied for instance when both f_s and g_a are convex increasing on (o, ∞). Figure (3.4) shows why just monotonicity of $G_a(\ )$ is not sufficient for the lemma to hold.

We can now state the main result of this chapter.

Proposition (3.A): Under Assumptions (3.A)-(3.B), with α_k^a as in (3.4.3) or (3.4.4), the sequence $\{\gamma_k\}$, generated by (3.4.2) with γ_o satisfying (3.4.1), converges to a vector $\bar{\gamma}$. Moreover the vector $(\ldots, f^{-1}(\bar{\gamma}^s), \ldots)'$ is a fair allocation over the set of all vectors $(\ldots, f_s^{-1}(\gamma^s), \ldots)'$ where γ^s for all $s \varepsilon S$ satisfy (3.3.6) - (3.3.8).

3.5 Convergence Proof:

Let K_s be a subsequence such that

$$\lim_{\substack{k\varepsilon K_s \\ k\to\infty}} \gamma_{k+1}^s = \lim_{k\to\infty} \inf\ \gamma_k^s \quad . \tag{3.5.1}$$

By (3.4.2) we can now write

$$\lim_{\substack{k\varepsilon K \\ k\to\infty^s}} \gamma_{k+1}^s = \lim_{\substack{k\varepsilon K \\ k\to\infty^s}} \min_{\substack{a:1 \\ p_s(a)=1}} \gamma_k^s + \alpha_k^a [f_s g_a(c_a - F_k^a) - \gamma_k^s] \quad .$$

Since the path p_s consists of a finite number of links, then w.l.o.g. possibly by taking a subsequence, there exists a link $\bar{a}\varepsilon\mathcal{L}$, $1_{p_s}(\bar{a})=1$ such that

$$\lim_{\substack{k\varepsilon K \\ k\to\infty^s}} \gamma_{k+1}^s = \lim_{\substack{k\varepsilon K \\ k\to\infty^s}} [\gamma_k^s + \alpha_k^{\bar{a}}(f_s g_{\bar{a}}[c_{\bar{a}} - F_k^{\bar{a}}] - \gamma_k^s)]. \tag{3.5.2}$$

By relation (3.4.5), Assumptions (3.A)-(3.B) and (3.4.3)-(3.4.4) it can be easily verified that there exist $\underline{\beta}>0$ and $\bar{\beta}>0$ such that

$$0 < \underline{\beta} < \alpha_k^a < \bar{\beta} < 1 \qquad \forall\ a\varepsilon\mathcal{L} \quad k=0,1,\ldots.$$

and therefore w.l.o.g., possibly by taking a subsequence of K_s we have

$$\lim_{\substack{k\in K\\ k\to\infty^s}} \alpha_k^{\bar{a}} = \bar{\alpha}^{\bar{a}} > 0. \tag{3.5.3}$$

Thus, we get

$$\begin{aligned}\lim_{\substack{k\in K\\ k\to\infty^s}} \gamma_{k+1}^s &\geq \lim_{\substack{k\in K\\ k\to\infty^s}} \inf(1-\alpha_k^{\bar{a}})\gamma_k^s + \lim_{\substack{k\in K\\ k\to\infty^s}} \inf\, \alpha_k^{\bar{a}} \cdot f_s g_{\bar{a}}(c_{\bar{a}} - F_k^{\bar{a}}) \\ &\geq (1-\bar{\alpha}^{\bar{a}}) \lim_{k\to\infty} \inf\, \gamma_k^s + \bar{\alpha}^{\bar{a}} \lim_{k\to\infty} \inf\, f_s g_{\bar{a}}(c_{\bar{a}} - F_k^{\bar{a}}).\end{aligned} \tag{3.5.4}$$

Using (3.5.1) and Assumption (3.B) we obtain from (3.5.4)

$$\lim_{k\to\infty} \inf\, \gamma_k^s \geq f_s g_{\bar{a}}(c_{\bar{a}} - \lim_{k\to\infty} \sup\, F_k^{\bar{a}}). \tag{3.5.5}$$

Since the derivation of(3.5.5) was independent of s we can conclude that for all $t\in S$ there exist a(t) such that (3.5.5) holds with t substituting s, and a(t) substituting $\bar{a}$.

Let $a_1 \in \mathcal{A}$ satisfy

$$a_1 = \arg\min_{a\in\mathcal{A}} g_a(c_a - \lim_{k\to\infty} \sup\, F_k^a). \tag{3.5.6}$$

Using the monotonicity of f_s (Assumption (3.A)) we have

$$f_t g_{a(t)}[c_{a(t)} - \lim_{k\to\infty} \sup\, F_k^{a(t)}] \geq f_t g_{a_1}[c_{a_1} - \lim_{k\to\infty} \sup\, F_k^{a_1}] \quad \forall\, t\in S, \tag{3.5.6}$$

and therefore by (3.5.5)

$$\lim_{k\to\infty} \inf\, \gamma_k^s \geq f_s g_{a_1}(c_{a_1} - \lim_{k\to\infty} \sup\, F_k^{a_1}) \qquad \forall\, s\in S. \tag{3.5.8}$$

Summing (3.5.8) over all $s\in S$ such that $1_{p_s}(a_1) = 1$ we conclude that

$$\liminf_{k\to\infty} F_k^{a_1} \geq \sum_{t\in S} \liminf_{k\to\infty} \gamma_k^t \cdot 1_{p_t}(a_1) \geq \tag{3.5.9}$$

$$\geq \sum_{t\in S} f_t g_{a_1} [c_{a_1} - \limsup_{k\to\infty} F_k^{a_1}] \cdot 1_{p_t}(a_1),$$

and therefore by Lemma (3.A)

$$\liminf_{k\to\infty} F_k^{a_1} \geq \limsup_{k\to\infty} F_k^{a_1}$$

implying that

$$\lim_{k\to\infty} F_k^{a_1}$$

exists. Moreover we now have

$$\sum_{t\in S} \liminf_{k\to\infty} \gamma_k^t \cdot 1_{p_t}(a_1) \geq \lim_{k\to\infty} \sum_{t\in S} \gamma_k^t \cdot 1_{p_t}(a_1) \tag{3.5.10}$$

which implies the existence of

$$\lim_{k\to\infty} \gamma_k^t 1_{p_t}(a_1) \qquad \forall\ t\in S, \tag{3.5.11}$$

and therefore

$$\lim_{k\to\infty} \gamma_k^t 1_{p_t}(a_1) = f_t g_{a_1} \left[c_{a_1} - \sum_{u\in S} \lim_{k\to\infty} \gamma_k^u \cdot 1_{p_u}(a_1)\right] \cdot 1_{p_t}(a_1) \tag{3.5.12}$$

$$\forall\ t\in S.$$

Thus we have proved the convergence of the rates of all sessions traversing link a_1, as defined in (3.5.6).

Consider now a new network derived from the previous one by deleting link a_1, and all the sessions traversing it. We now only update the capacity of link $a\in\mathcal{L}$ at iteration k to be

$$c_a - \sum_{t\in S} \gamma_k^t \cdot 1_{p_t}(a) \cdot 1_{p_t}(a_1).$$

Using this new network and the Corollary to Lemma (3.A), we can show the convergence of the rates of session traversing link a_2 which satisfies

$$a_2 = \arg \min_{a \in \mathcal{L},\, a \neq a_1} g_a(c_a - \limsup_{k \to \infty} F_k^a).$$

This procedure will exhaust all the links in the network and since each session traverses at least one link, we have therefore proved that the rates of all sessions converge.

To see that the vector $(\ldots, f_s^{-1}(\bar{\gamma}^s), \ldots)'$ is a fair allocation over the set defined by (3.3.6)-(3.3.8), notice that by the existence of the limit in (3.5.11) we get equality in (3.5.5) and therefore by (3.5.7) we get

$$f_t^{-1}(\bar{\gamma}^t) \geq g_{a_1}[c_{a_1} - \sum_{u \in S} \bar{\gamma}^u 1_p(a_1)], \tag{3.5.13}$$

with equality for $t \in S$ such that $1_{p_t}(a_1)=1$. Thus $f_t^{-1}(\bar{\gamma}^t)$, for $t \in S$ such that $1_{p_t}(a_1)=1$, maximizes the minimal value of $f_t^{-1}(\gamma^t)$ for all $t \in S$ when γ satisfies (3.3.6). Since $\bar{\gamma}$ is easily seen to belong to the set defined by (3.3.6)-(3.3.8), it therefore solves the first problem in the hierarchy of problems that results in a vector $(\ldots, f_s^{-1}(\bar{\gamma}^s), \ldots)'$ of fair allocation over this set. Likewise inductively we can show that it solves the second problem in the hierarchy etc., and the Proposition is proved.

Q.E.D.

3.6 Implementation Issues

The iteration (3.4.2) is stated in a centralized context, whereby all the needed quantities are simultaneously available everywhere. Certainly this is not the case in a geographically distributed network. In this section we modify the iteration in a way that renders it suitable for distributed implementation.

Pictorially we look at each link and each origin of a session as a processor. These processors implement the algorithm by exchanging messages between themselves. We assume that each link can send messages to all origins of sessions that traverse it. The order of messages on a path is preserved. To avoid the question of how the rate is measured, we assume we are dealing with a computational algorithm whereby the rate γ^s is a variable determined by the origin-processor responsible for session $s \in S$. The distributed version of iteration (3.4.2) will determine how this variable is changed.

The key to the distributed version of the algorithm is the observation that Lemma (3.A) holds for each link without any regard to whether other links exist at all or participate in the algorithm. With this in mind, it is easy to rework the proof of Proposition (3.A) to see that a sufficient condition for the convergence of the algorithm is that each link "performs" (in a sense that will be clear shortly) step (3.4.2) infinitely often. Thus we propose the following model.

Each link $a \in \mathcal{L}$ keeps a list of local variables $\tilde{\gamma}_a^s$ for all sessions $s \in S$ that traverse it (together with the functions f_s and g_a). An origin of session s keeps a list of variables $\hat{\gamma}_a^s$ for all links a on the path p_s. We assume that by a certain internal rule a link $a \in \mathcal{L}$ "awakes" in certain times such that the "awake" occurs infinitely often. Each time a link $a \in \mathcal{L}$ "awakes" it performs, subject to conditions permitting it, a "link message" procedure. Similarly, an origin to session $s \in S$ "awakes" once in a while and performs a "node message" procedure. A link which receives a message from an origin node performs a "link update" procedure and an origin to a session which receives a message from a link performs a "node update" procedure. The details of the procedures are:

Link Message: (link "awakes")

1. To each session s traversing a, link a sends a message containing the quantity

$$\tilde{\gamma}_a^s + \alpha^a[f_s g_a(c_a - \sum_{u\in S} \tilde{\gamma}_a^u 1_{p_u}(a)) - \tilde{\gamma}_a^s]$$

where $\tilde{\gamma}_a^s$ is the value of the corresponding local variable at the time of the "awake" and α^a is as in (3.4.3) or (3.4.4) with $\tilde{\gamma}_a^s$ replacing γ_k^s.

2. A new "awake" is aborted until a "node message" is received from all origins traversing a. For each origin the "node message" should come after an acknowledgement to the message sent in 1 is received.

Node Message: (origin of session s "awakes")

1. $\gamma^s \leftarrow \min_{a \text{ s.t. } 1_{p_s}(a)=1} \hat{\gamma}_a^s$

2. γ^s is sent to all links $a\varepsilon\mathcal{L}$ on the path p_s.

Link Update: (message from origin s arrives at link a)

1. $\tilde{\gamma}_a^s \leftarrow$ value of the message.

2. Send an acknowledgement.

Node Update: (message from link a arrives at origin of session s)

1. $\hat{\gamma}_a^s \leftarrow$ value of the message.

2. Send an acknowledgement.

It is not difficult to ascertain from the observation made above that this distributed procedure will result in values of γ^s, that will converge to the value that would have been produced had a centralized algorithm been used.

A careful examination of all possible sequencing of events that may occur in the network during the implementation of the algorithm reveals

that the sum

$$\sum_{s \in S} \tilde{\gamma}_a^s 1_{p_s}(a)$$

is not guaranteed not to exceed c_a at all times. In the way the algorithm is stated the above sum can be proved not to exceed capacity only at moments of unaborted "awakes". The problem is that after a "link message" is performed, those sessions that are allowed by the link to increase might respond before those that are instructed to decrease. This by itself will not cause a problem since it can be shown that any combination of $\tilde{\gamma}_a^s$ at "awake" time and the values sent to the origins by the link does not exceed the capacity. But in between the link's "awake" and the response of the origins, sessions that are instructed to decrease, can in the meantime increase on the behalf of actions taken by other links. This is a very rare but still possible occurance.

The remedy is to modify the "link message" procedure by sequencing the messages sent in step 1 properly. Notice that at the beginning of each "awake" the values $\tilde{\gamma}_a^s$ are upper bounded by the values sent at the previous "awake". If we first send the messages whose values are below the corresponding values above and then wait for the appropriate node "awake", we preclude the possibility mentioned in the previous paragraph. The exact statement of the modified "link message" procedure can be easily worked out.

An alternative remedy is to abort a "node message" procedure until a node which is an origin to session s, has heard from all the links on p_s, since it last performed the procedure.

The first remedy is cumbersome; the second remedy might slow down the algorithm.

4 Data Routing

4.1 Introduction

In this chapter we consider an iterative method for the solution of the multicommodity flow routing problem. We are mainly interested in an iterative method which converges fast and facilitates distributed computation. These two characteristics are the key to a successful quasi-static algorithm for routing messages in a computer communication network.

The fast convergence allows a static routing algorithm, aimed at achieving an optimal flow pattern when inputs are fixed, to be used when inputs are changing. If the convergence is fast and the variations in the inputs are slow, the algorithm will result in an instantaneous flow pattern which is close to the optimal flow pattern with respect to the instantaneous inputs.

The distributed computation is particularly appealing in computer communication networks where each node is a computer. It facilitates a speedup of the algorithm, it increases the survivability of the algorithm when nodes are subject to failure and it increases the modularity of the network. In particular when distributing computation in an environment of changing inputs, the algorithm should not utilize parameters which are sensitive to those changes. This means for instance the use of a constant stepsize which preserves the fast rate of convergence for a broad spectrum of inputs.

The need to achieve fast convergence, coupled with the need to utilize constant stepsize applicable to a broad spectrum of inputs, make Newton's method a top candidate, since it converges superlinearly and employs stepsize of unity for any input. The superlinear convergence when unity step is employed, is guaranteed only when the algorithm starts near the optimum,

but this is exactly the situation in which the quasi static algorithm is operating.

The above consideration led to a research effort ([11], [23]) aimed at imitating Newton's method while facilitating distributed computation. Unfortunately, in all imitations the distributed computation was achieved at the expense of the deterioration of the convergence rate from superlinear to linear.

Most of the algorithms suggested in ([11], [23]) can be cast as variations of a projected Newton step in which the Hessian of the objective function was approximated by a diagonal matrix. Two reasons accounted for the need of a diagonal approximation. The first was the difficulty of computing the off diagonal terms. The second, and the more serious difficulty, was that of distributing the computation associated with the projection, once the approximation to the Hessian was not diagonal.

In the sections that follow we propose a way to overcome the above two difficulties. In Sections 4.2 - 4.6 we take up the problem of approximating the Hessian by a nondiagonal approximation while at the same time projecting with respect to a diagonal matrix, thus facilitating the distributed computation of the projection. It is shown that the nondiagonal approximation can be taken in such a way as to preserve the superlinear rate of convergence in spite of the fact that the projection is made with respect to a diagonal matrix. These sections are cast within a rather general framework and by themselves constitute a contribution to the nonlinear programming methodology.

In Section 4.7, the first difficulty, that of computing the nondiagonal elements of the Hessian is considered. The difficulty encountered with those elements stems from the space of routing variables used in

[11],[23]. This space, of fractions decomposed by destinations as presented first in [10], gives rise to a nonlinear transformation from the routing variables, namely, the fractions, to the link flows. We propose the space of path flows in which the transformation from the routing variables, namely, the path flows, to the link flows is linear. In this space the Hessian obtains a particularly simple structure which facilitates distributed computation.

The distributed computation facilitated is that of a solution of a system of linear equation built around the Hessian. We use the Conjugate-Gradient method to solve this system and show that all the operations involved can be decomposed according to the nodes in the network, in a way that does not require any node to know the global topology of the network.

The advantages gained by considering the routing problem in the space of path flows may by itself be a good reason for the use of these variables in a computer communication network, instead of fractions. Moreover, in Chapter 5, which deals with data flow control we give an additional impetus to this change of variables. This is in the spirit of this thesis, that of employing compatible algorithms for the various tasks associated with the delivery of a packet from an origin to a destination.

We conclude in Section 4.7 with a computational example of the multicommodity flow problem in the space of path flows.

4.2 Two Metric Projection Method

Projection methods stemming from the original proposal of Goldstein [1], and Levitin and Poljak [2] are often very useful for solving the problem

$$\begin{aligned} &\text{minimize } f(x) \\ &\text{subject to } x\in X \end{aligned} \tag{4.2.1}$$

where $f:H\to R$ and X is a convex subset of a linear space H. These methods take the form

$$x_{k+1} = P_k(x_k - \alpha_k g_k) \tag{4.2.2}$$

where α_k is a positive scalar stepsize, $P_k(\cdot)$ denotes projection on X with respect to some Hilbert space norm $\|\cdot\|_k$ on H and g_k denotes the Frechet derivative of f with respect to $\|\cdot\|_k$, i.e., g_k is the vector in H satisfying

$$f(x) = f(x_k) + \langle g_k, x-x_k\rangle_k + o(\|x-x_k\|), \tag{4.2.3}$$

where $\langle\cdot,\cdot\rangle_k$ denotes the inner product corresponding to $\|\cdot\|_k$.

As an example let $H=R^n$, and B_k be an nxn positive definite symmetric matrix. Consider the inner product and norm corresponding to B_k

$$\langle x,y\rangle_k = x'B_k y, \qquad \|x\|_k = (\langle x,x\rangle)^{1/2}, \quad \forall\ x,y\in H, \tag{4.2.4}$$

where all vectors above are considered to be column vectors and prime denotes transposition. With respect to this norm we have [cf.(4.2.3)]

$$g_k = B_k^{-1}\nabla f(x_k), \tag{4.2.5}$$

where $\nabla f(x_k)$ is the vector of first partial derivatives of f

$$\nabla f(x_k) = \begin{bmatrix} \dfrac{\partial f(x_k)}{\partial x^1} \\ \vdots \\ \dfrac{\partial f(x_k)}{\partial x^n} \end{bmatrix} . \qquad (4.2.6)$$

When problem (4.2.1) is unconstrained (X = H), iteration (4.2.2) takes the familiar form

$$x_{k+1} = x_k - \alpha_k B_k^{-1} \nabla f(x_k).$$

Otherwise the vector

$$x_{k+1} = P_k(x_k - \alpha_k g_k)$$

is the solution of the problem

$$\text{minimize } \|x - x_k + \alpha_k g_k\|_k^2$$
$$\text{subject to } x \in X.$$

A straightforward computation using (4.2.4) and (4.2.5) shows that the problem above is equivalent to the problem

$$\text{minimize } \nabla f(x_k)'(x - x_k) + \frac{1}{2\alpha_k}(x-x_k)'B_k(x-x_k) \qquad (4.2.7)$$
$$\text{subject to } x \in X.$$

When X is a polyhedral set and B_k is a Quasi-Newton approximation of the Hessian of f, the resulting method is closely related to recursive quadratic programming methods which currently enjoy a great deal of popularity (e.g., Garcia-Palomares [3], Gill et al [4]).

It is generally recognized that in order for methods of this type to be effective it is essential that the computational overhead for solving the

quadratic programming problem (4.2.7) should not be excessive. For large-scale problems this will be so only if the matrix B_k is chosen in a way that matches the structure of the constraint set. For example if X is the Cartesian product $\prod_{i=1}^{m} X_i$ of m simpler sets X_i, the matrix B_k should be block diagonal with one block corresponding to each set X_i, in which case the projection problem (4.2.7) decomposes naturally. Unfortunately such a choice of B_k precludes the possibility of superlinear convergence of the algorithm which typically cannot be achieved unless B_k is chosen to be a suitable approximation of the Hessian matrix of f ([3], [5]).

In this Chapter we propose projection methods of the form

$$x_{k+1} = P(x_k - \alpha_k g_k) \tag{4.2.8}$$

where the norms $\|\cdot\|$ and $\|\cdot\|_k$ corresponding to the projection and the differentiation operators respectively can be different. This allows the option to choose $\|\cdot\|$ to match the structure of X, thereby making the projection operation computationally efficient, while reserving the option to choose $\|\cdot\|_k$ on the basis of second derivatives of f thereby making the algorithm capable of superlinear convergence. When $H = R^n$, the projection norm $\|\cdot\|$ is the standard Euclidean norm

$$\|x\| = (x'x)^{1/2} = |x|, \tag{4.2.9}$$

and the derivative norm $\|\cdot\|_k$ is specified by an nxn positive definite symmetric matrix B_k

$$\|x\|_k = (x'B_k x)^{1/2}, \tag{4.2.10}$$

the vector x_{k+1} of (4.2.8) is obtained by solving the quadratic programming subproblem

$$\text{minimize } g_k'(x - x_k) + \frac{1}{2\alpha_k}|x - x_k|^2$$
$$\text{subject to } x \in X \tag{4.2.11}$$

where

$$g_k = B_k^{-1}\nabla f(x_k). \tag{4.2.12}$$

The quadratic programming problem (4.2.11) may be very easy to solve if X has special structure. As an example consider the case of an orthant constraint

$$X = \{x \mid 0 \leq x^i,\ i = 1,\ldots,n\}. \tag{4.2.13}$$

Then, the iteration takes the form

$$x_{k+1} = [x_k - \alpha_k B_k^{-1}\nabla f(x_k)]^+ \tag{4.2.14}$$

where for any vector $v \in R^n$ with coordinates v^i, $i = 1,\ldots,n$ we denote by v^+ the vector with coordinates

$$(v^i)^+ = \max\{0, v^i\}.$$

Iteration (4.2.14) was first proposed in Bertsekas [6], and served as the starting point for the present paper. It is not true in general that for an arbitrary positive definite choice B_k, iteration (4.2.14) is a descent iteration [in the sense that if x_k is not a critical point of f over the set X of (4.2.13) then for α_k sufficiently small we have $f(x_{k+1}) < f(x_k)$]. Indeed this is the main difficulty in constructing two-metric extensions of the Goldstein-Levitin-Poljak method. It was shown, however in [6] that if B_k is chosen to be partially diagonal with respect to a suitable subset of coordinates then (4.2.14) becomes a descent iteration. We give a nontrivial extension of this result in the next section (Proposition (4.A)).

The construction of the "scaled gradient" g_k satisfying the descent condition

$$<g_k, \nabla f(x_k)> \quad 0 \tag{4.2.15}$$

is based on a decomposition of the negative gradient into two orthogonal components by projection on an appropriate pair of cones that are dual to each other. One of the two components is then "scaled" by multiplication with a positive definite self-adjoint operator (which may incorporate second derivative information) and added to the first component to yield g_k. The method of construction is such that g_k, in addition to (4.2.15), also satisfies

$$f[P(x_k - \alpha g_k)] < f(x_k)$$

for all α in an interval $(0, \overline{\alpha}_k]$, $\overline{\alpha}_k > 0$.

Section 4.4 describes the main algorithm and proves its convergence. While other stepsize rules are possible, we restrict attention to an Armijo-like stepsize rule for selecting α_k on the arc

$$\{z \mid z = P(x_k - \alpha g_k),\ \alpha > 0\}$$

which is patterned after similar rules proposed in Bertsekas [6], [7]. Variations of the basic algorithm are considered in Section 4.6, while in Section 4.5 we consider rate of convergence aspects of algorithm (4.2.8), (4.2.11), (4.2.12) as applied to finite dimensional problems. We show that the descent direction g_k can be constructed on the basis of second derivatives of f so that the method has a typically superlinear rate of convergence. Here we restrict attention to Newton-like versions of the algorithm. Quasi-Newton, and approximate Newton implementations based on

successive overrelaxation or conjugate gradient methods are possible. In fact a superlinearly convergent conjugate gradient-based implementation of the method is applied to large-scale multicommodity flow problems in Section 4.7.

While the algorithm is stated and analyzed in general terms we pay special attention to the case where X is a finite dimensional polyhedral set with a decomposable structure since we believe that this is the case where the algorithm is most likely to find application.

4.3 The Algorithmic Map and Its Descent Properties

Consider the problem

$$\text{minimize } f(x) \qquad (4.3.1)$$
$$\text{subject to } x \in X$$

where f is a real-valued function on a Hilbert space H, and X is a non-empty, closed, convex subset of H. The inner product and norm on H will be denoted by $\langle .,. \rangle$ and $\|.\|$ respectively. We say that two vectors $x, y \in H$ are orthogonal if $\langle x,y \rangle = 0$. For any $z \in H$ we denote by P(z) the unique projection of z on X, i.e.,

$$P(z) = \arg\min \{\|x - z\| \mid x \in X\}. \qquad (4.3.2)$$

We assume that f is continuously Frechet differentiable on H. The Frechet derivative at a vector $x \in H$ will be denoted by $\nabla f(x)$. It is the unique vector in H satisfying

$$f(z) = f(x) + \langle \nabla f(x), z - x \rangle + o(\|z - x\|)$$

where $o(\|z - x\|)/\|z - x\| \to 0$ as $z \to x$. We say that a vector $x^* \in X$ is <u>critical</u> with respect to problem (16) if

$$\langle \nabla f(x^*), x - x^* \rangle \geq 0, \qquad \forall\, x \in X, \qquad (4.3.3)$$

or equivalently, if $x^* = P[x^* - \nabla f(x^*)]$ for all $\alpha > 0$.

It will be convenient for our purposes to represent the set X as an intersection of half spaces

$$X = \{x \mid \langle a_i, x \rangle \leq b_i, \ \forall\, i \in I\}, \qquad (4.3.4)$$

where I is a, possibly infinite, index set and, for each $i \in I$, a_i is a

nonzero vector in H and b_i is a scalar. For each closed convex set X there exists at least one such representation. We will assume that the set I is nonempty - the case where I is empty corresponds to an unconstrained problem which is not the subject of this paper. Our algorithm will be defined in terms of a specific collection $\{(a_i,b_i) \mid i\in I\}$ satisfying (4.3.4) which will be assumed given. This is not an important restriction for many problems of interest including, of course, the case where X is a polyhedron in R^n.

We now describe the algorithmic mapping on which our method is based. For a given vector $x\in X$ we will define an arc of points $\{x(\alpha) \mid \alpha \geq 0\}$ which depends on an index set $I_x \subset I$ and an operator D_x which will be described further shortly. The index set I_x is required to satisfy

$$I_x \supset \{i\in I \mid \langle a_i,x\rangle \geq b_i - \varepsilon\|a_i\|\} \tag{4.3.5}$$

where ε is some positive scalar. Let C_x be the cone defined by

$$C_x = \{z \mid \langle a_i,z\rangle \leq 0, \ \forall\ i\in I_x\} \tag{4.3.6}$$

and C_x^+ be the dual cone of C_x

$$C_x^+ = \{z \mid \langle y,z\rangle \leq 0, \ \forall\ y\in C_x\}. \tag{4.3.7}$$

For orientation purposes we mention that if X is a polyhedral subset of R^n (or more generally if the index set I is finite), and ε is sufficiently small, then I_x can consist of the indexes of the active constraints at x, i.e., we may take $I_x = \{i \mid \langle a_i,x\rangle = b_i,\ i\in I\}$. In that case C_x is the cone of feasible directions at x, while C_x^+ is the cone generated by the vectors a_i correspoding to the active constraints at x. More generally C_x is a

(possibly empty) subset of the set of feasible directions at x, and for any $\Delta x \in C_x$ with $\|\Delta x\| < \varepsilon$ the vector $x + \Delta x$ belongs to X.

Let d_x be the projection of $[-\nabla f(x)]$ on C_x, i.e.,

$$d_x = \arg\min \{\|z + \nabla f(x)\| \mid z \in C_x\}. \tag{4.3.8}$$

Define

$$d_x^+ = -[\nabla f(x) + d_x]. \tag{4.3.9}$$

It can be easily seen that the vectors d_x and d_x^+ are orthogonal and that d_x^+ is the projection of $[-\nabla f(x)]$ on C_x^+, i.e.,

$$d_x^+ = \arg\min \{\|z + \nabla f(x)\| \mid z \in C_x^+\}. \tag{4.3.10}$$

Note that if the norm $\|\cdot\|$ on H is such that projection on the set X is relatively simple then typically the same is true for the projection (4.3.8), required to compute d_x and d_x^+.

Let Γ_x be the subspace spanned by the elements of C_x which are orthogonal to d_x^+, i.e.,

$$\Gamma_x = \text{span}\{C_x \cap \{z \mid \langle z, d_x^+ \rangle = 0\}\}. \tag{4.3.11}$$

Note that

$$d_x \in \Gamma_x \tag{4.3.12}$$

since d_x belongs to C_x and is orthogonal to d_x^+. Let $D_x: \Gamma_x \to \Gamma_x$ be a positive definite self-adjoint operator mapping Γ_x into itself. Consider the projection $\tilde{d}_x$ of $D_x d_x$ on the closed cone $C_x \cap \{z \mid \langle z, d^+ \rangle = 0\}$, i.e.

$$\tilde{d}_x = \arg\min \{\|z - D_x d_x\| \mid z \in C_x, \langle z, d_x^+ \rangle = 0\}. \tag{4.3.13}$$

Consider also the "direction" vector

$$g = -(d_X^+ + \tilde{d}_X). \tag{4.3.14}$$

Given x, I_x, and D_x, our algorithm chooses the next point along the arc

$$x(\alpha) = P(x - \alpha g), \qquad \alpha > 0. \tag{4.3.15}$$

The stepsize α will be chosen by an Armijo-like stepsize rule that will be described in the next section.

The process by means of which the direction g is obtained is illustrated in Figures 4.1 - 4.4 The crucial fact that will be shown in Proposition (4.A) below is that, if x is not critical, then for sufficiently small $\alpha > 0$ we have $f[x(\alpha)] < f(x)$, i.e., by moving along the arc $x(\alpha)$ of (4.3.15) we can decrease the value of the objective. Furthermore we have $\langle \nabla f(x), g \rangle$ 0 which means that g can be viewed as a "scaled" gradient, i.e., the product of $\nabla f(x)$ with a positive definite self-adjoint operator.
We now demonstrate the process of calculating the direction g for some interesting specially structured constraint sets.

Example (4.A): Let $H = R^n$, $\langle x,y \rangle = x'y$, and X be the positive orthant

$$X = \{x \mid x^i \geq 0, \ i = 1,\ldots,n\}.$$

Then X consists of the intersection of the n halfspaces $\{x \mid x^i \geq 0\}$ $i = 1,\ldots,n$ and is of the form (4.3.4). The set I_x must contain all indices i such that $0 \leq x^i \leq \varepsilon$ [cf. (4.3.5)]. The cones C_x and C_x^+ are given by

$$C_x = \{z \mid z^i \geq 0, \ \forall\, i \in I_x\}, \quad C_x^+ = \{z \mid z^i \leq 0, \ \forall\, i \in I_x, \ z^j = 0 \ \ \forall\, j \notin I_x\}$$

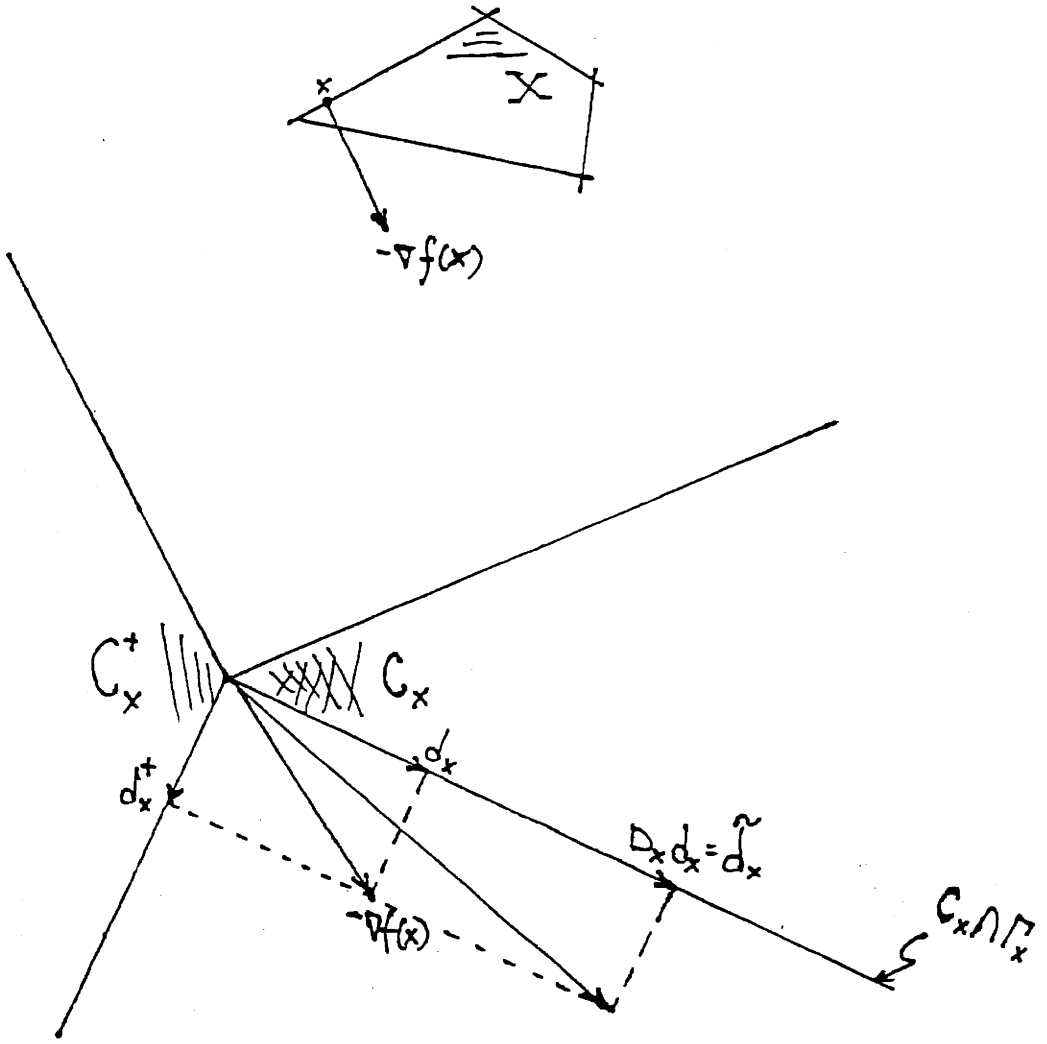


Figure 4.1: A case where both C_x and C_x^+ have non-empty interior in R^2 and $-\nabla f$ lies outside C_x

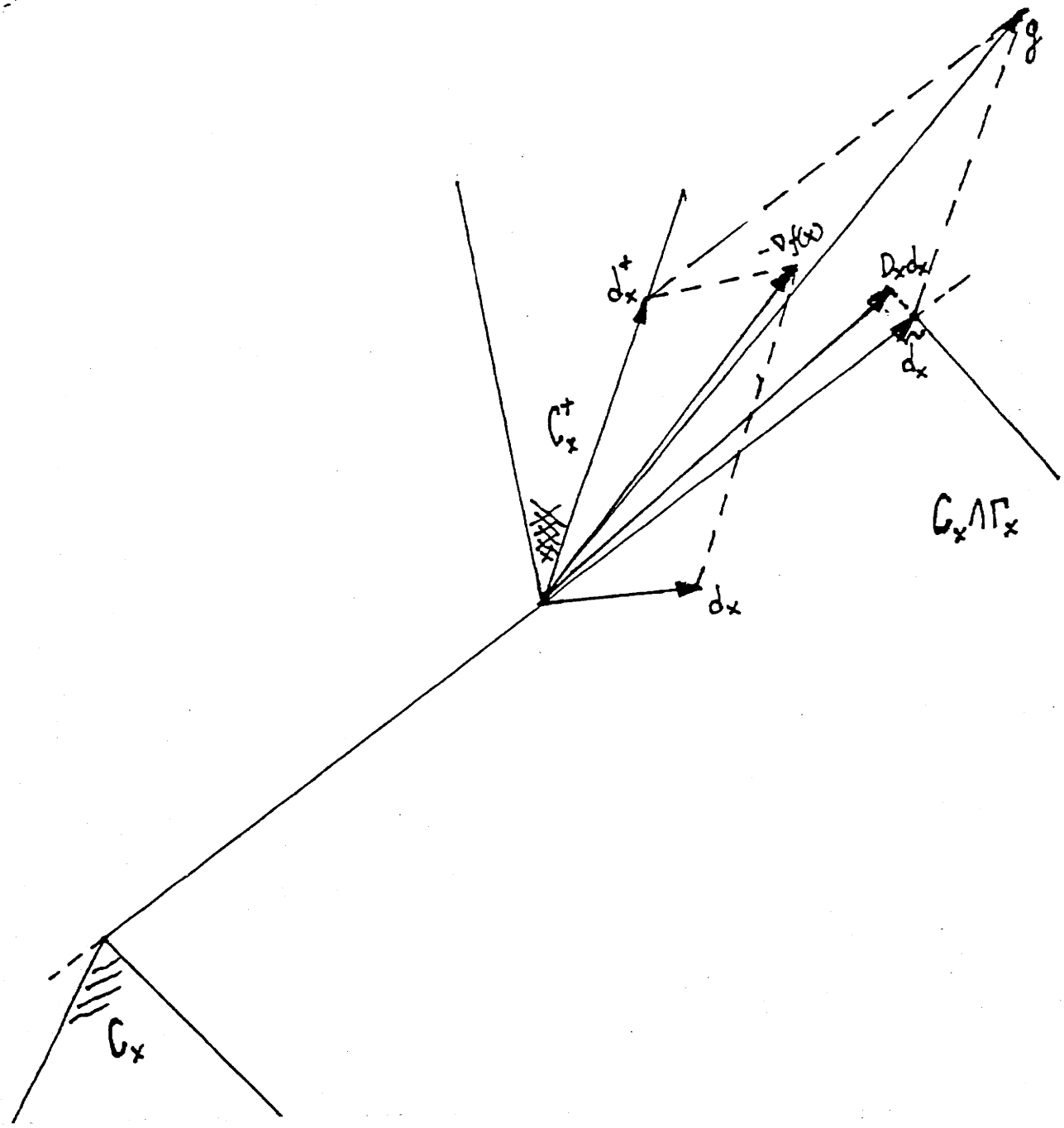


Figure 4.2: Obtaining g for a case where C_x^+ lies on a 2-dimensional manifold in R^3

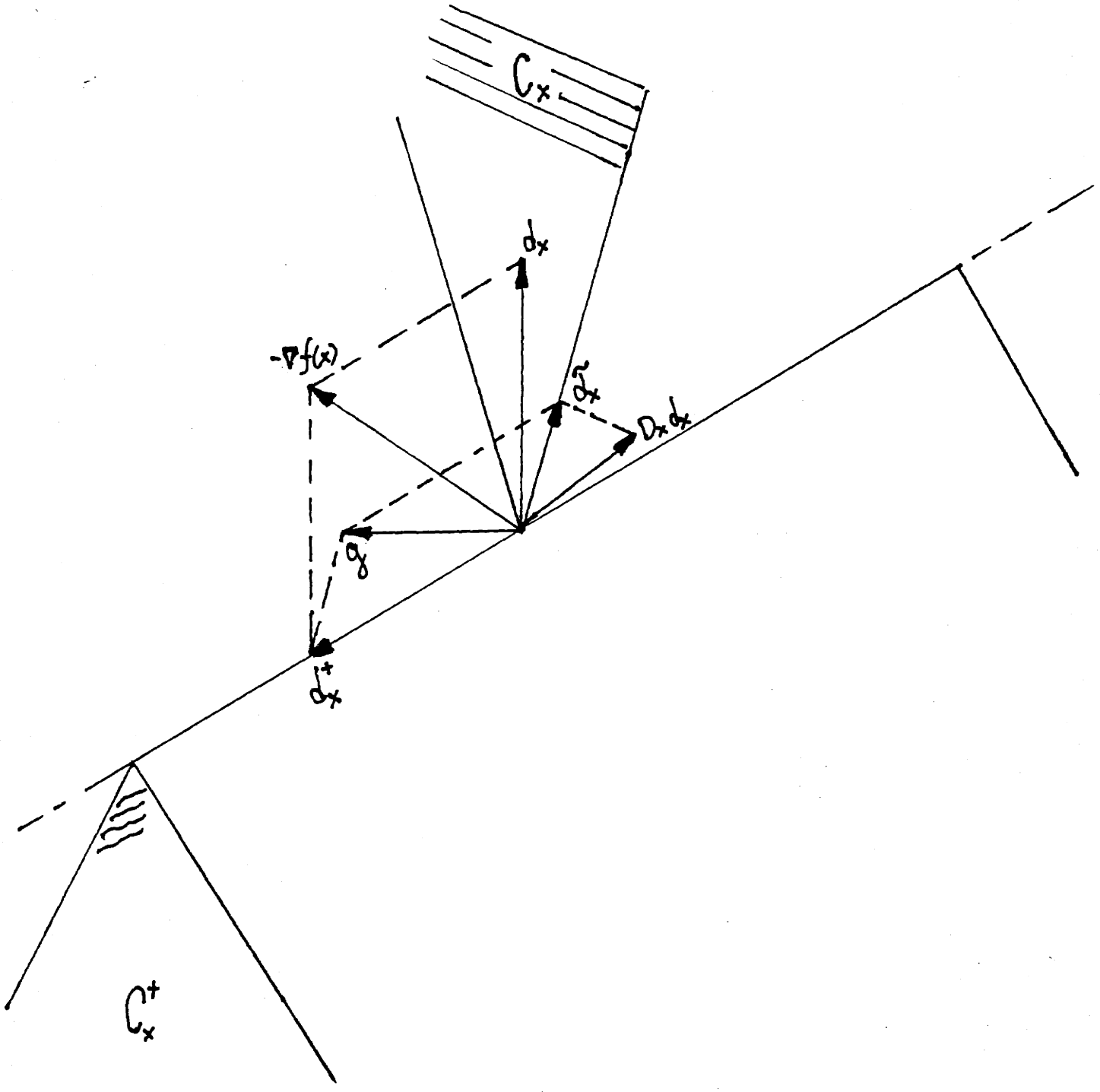


Figure 4.3: Obtaining g for a case where C_x lies on a 2-dimensional manifold in R^3

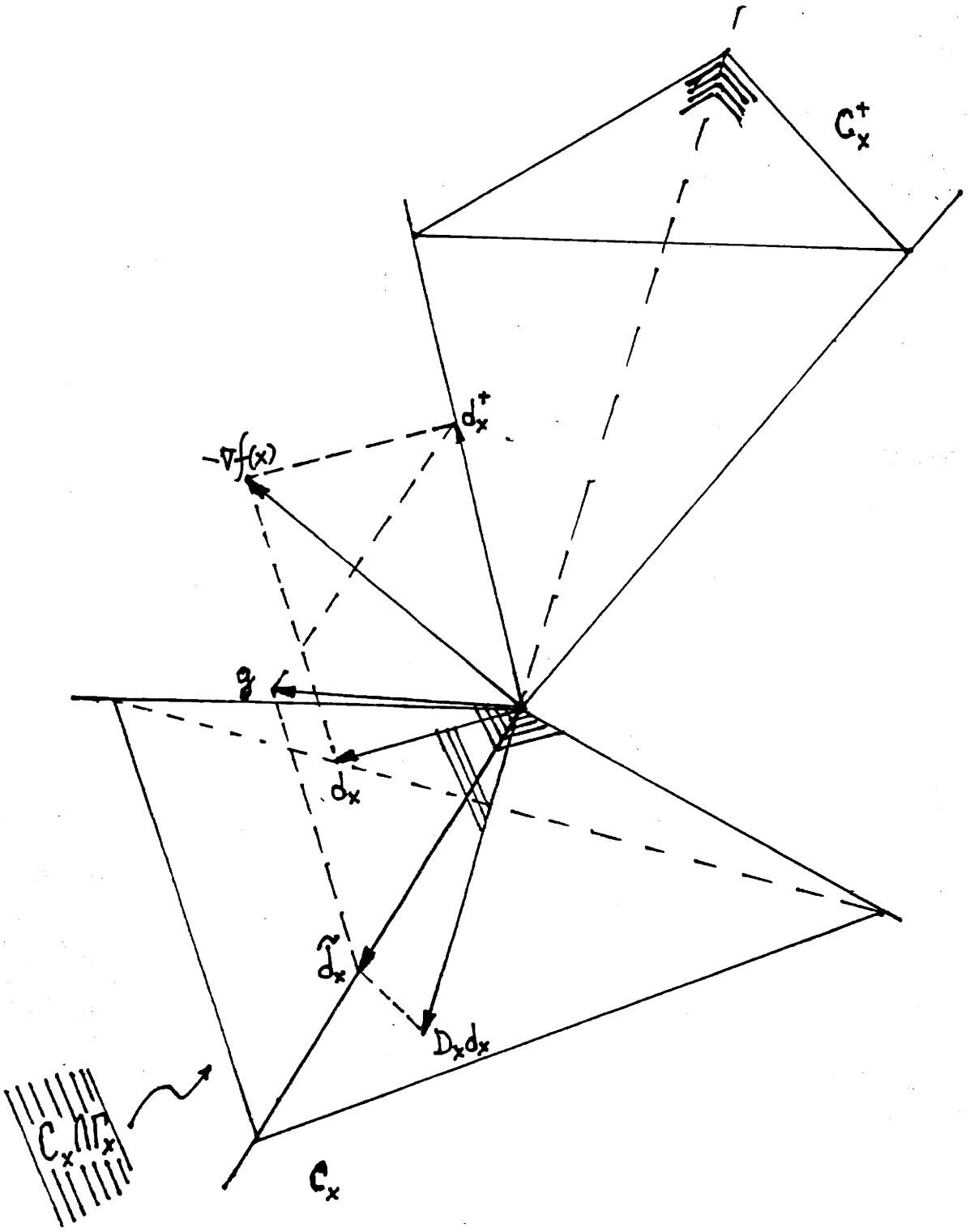


Figure 4.4: Obtaining g for a case where both C_x^+ C_x have nonempty interior in R^3

The vector d_x, and d_x^+ [cf. (4.3.8), (4.3.9)] have coordinates given by

$$d_x^i = \begin{cases} -\dfrac{\partial f(x)}{\partial x^i} & \text{if } i \notin I_x \\ 0 & \text{if } i \in I_x \end{cases}$$

$$d_x^{i+} = \begin{cases} 0 & \text{if } i \notin I_x \\ -\dfrac{\partial f(x)}{\partial x^i} & \text{if } i \in I_x \end{cases}$$

where

$$I_x = \{i \mid i \in I_x \text{ and } \frac{\partial f(x)}{\partial x^i} > 0\}.$$

If I_x is empty then $\Gamma_x = R^n$ and we have $d_x = -\nabla f(x)$, $d_x^+ = 0$. In this case $g = -D_x d_x = D_x \nabla f(x)$ where D_x is any nxn positive definite symmetric matrix. If I_x is not empty, by rearranging indices if necessary assume that for some integer p with $0 \leq p \leq n-1$ we have $I_x = \{p+1,\ldots,n\}$. Partition $\nabla f(x)$ as

$$\nabla f(x) = \begin{bmatrix} \tilde{w} \\ w \end{bmatrix}$$

where $\tilde{w} \in R^p$ and $w \in R^{n-p}$. The vector g is given by

$$g = \begin{bmatrix} (D_x \tilde{w})^{\#} \\ w \end{bmatrix}$$

where D_x is a pxp positive definite symmetric matrix, $(D_x\tilde{w})^{\#}$ denotes projection of $D_x\tilde{w}$ on C_x, i.e. $(D_x\tilde{w})^{\#}$ is obtained from $D_x\tilde{w}$ by setting to zero those coordinates of $D_x\tilde{w}$ which are negative and their indices belong to I_x.

Example 2. Let $H = R^n$, and X be the unit simplex

$$X = \{x \mid \sum_{i=1}^{n} x_i = 1,\ x^i \geq 0,\quad i = 1,\ldots,n\}. \tag{4.3.16}$$

Suppose the inner product on R^n is taken to be

$$\langle x,y\rangle = \sum_{i=1}^{n} s^i x^i y^i \tag{4.3.17}$$

where s^i, $i = 1,\ldots,n$ are some positive scalars. Let I_x be a set of indices including those indices i such that $0 \leq x^i \leq \varepsilon/\sqrt{s^i}$. Then the cone C_x can be taken to be

$$C_x = \{z \mid \sum_{i=1}^{n} z_i = 0,\ z^i \geq 0,\ \forall\ i\in\hat{I}_x\}. \tag{4.3.18}$$

The vector d_x is obtained as the solution of the projection problem

$$\text{minimize } \frac{1}{2}\sum_{i=1}^{n} s^i\left[z^i + \frac{1}{s^i}\frac{\partial f(x)}{\partial x^i}\right]^2 \tag{4.3.19}$$

$$\text{subject to } \sum_{i=1}^{m} z^i = 0,\ z^i \geq 0,\ i\in\hat{I}_x.$$

The solution of this problem is very simple. By introducing a Lagrange multiplier λ for the equality constraint $\sum_{i=1}^{n} z^i = 0$ we obtain that λ is the solution of the simple piecewise linear equation

$$\sum_{i\in\hat{I}_x}\frac{1}{s^i}\left[\lambda - \frac{\partial f(x)}{\partial x^i}\right]^+ + \sum_{i\notin\hat{I}_x}\frac{1}{s^i}\left[\lambda - \frac{\partial f(x)}{\partial x^i}\right] = 0\ . \tag{4.3.20}$$

Once λ is obtained the coordinates of d_x are given by

$$d_x^i = \begin{cases} \frac{1}{s^i}\left[\lambda - \frac{\partial f(x)}{\partial x^i}\right]^+ & \text{if } i \in \hat{I}_x \\ \frac{1}{s^i}\left[\lambda - \frac{\partial f(x)}{\partial x^i}\right] & \text{if } i \notin \hat{I}_x. \end{cases} \tag{4.3.21}$$

The vector d_x^+ is then obtained from the equation

$$d_x^+ = -\left[\nabla f(x) + d_x\right].$$

Let

$$\tilde{I}_x = \left\{ i \,\middle|\, i \in \hat{I}_x \text{ and } \lambda < \frac{\partial f(x)}{\partial x^i} \right\}. \tag{4.3.22}$$

It is easily verified that the subspace Γ_x is given by

$$\Gamma_x = \left\{ z \,\middle|\, \sum_{i=1}^{n} z^i = 0, \quad z^i = 0, \quad \forall\, i \in \tilde{I}_x \right\}. \tag{4.3.23}$$

The vector $\tilde{d}_x$ is obtained as the solution of the simple projection problem

$$\text{minimize} \quad \frac{1}{2} \sum_{i=1}^{n} s^i \left[z^i - (D_x d_x)^i\right]^2 \tag{4.3.24}$$

$$\text{subject to} \quad \sum_{i=1}^{n} z^i = 0, \quad z^i \geq 0, \quad \forall\, i \in \hat{I}_x, \quad z^j = 0 \quad \forall\, j \in \tilde{I}_x$$

where $(D_x d_x)^i$ is the ith coordinate of the vector $D_x d_x$ obtained by multiplying d_x with an nxn symmetric matrix D_x which maps Γ_x into Γ_x and is positive definite on Γ_x. We will comment further on the choice of D_x in the last section of the paper. The vector g is given now by $g = -(\tilde{d}_x + d_x^+)$. Note that the solution of both projection problems (4.3.19) and (4.3.24), as well as the problem of projection on the simplex X of

(4.3.16) is greatly simplified by the choice of the "diagonal" metric specified by (4.3.17).

Proposition (4.A) below is the main result regarding the algorithmic map specified by (4.3.5) - (4.3.9), (4.3.13) - (4.3.15). For its proof we will need the following lemma the proof of which is given in Appendix (4.A).

Lemma (4.A): Let Ω be a closed convex subset of a Hilbert space H, and let $P_\Omega(\cdot)$ denote projection on Ω. For every $x\varepsilon\Omega$ and $z\varepsilon H$:

a) The function $h:(0,\infty) \to R$ defined by

$$h(\alpha) = \frac{\|P_\Omega(x + \alpha z) - x\|}{\alpha} \qquad \forall\ \alpha > 0$$

is monotonically nonincreasing.

b) If y is any direction of recession of Ω [i.e., $(x + \alpha y)\varepsilon\Omega$ for all $\alpha \geq 0$], then

$$\langle y, x + z\rangle \leq \langle y, P_\Omega(x + z)\rangle. \tag{4.3.5}$$

Proposition (4.A): For $x\varepsilon X$, let $\varepsilon > 0$ and I_x satisfy (4.3.5), and let $D_x:\Gamma_x \to \Gamma_x$ be a positive definite self-adjoint operator on the subspace Γ_x defined by (4.3.6) - (4.3.11). Consider the arc $\{x(\alpha) \mid \alpha \geq 0\}$ defined by (4.3.8), (4.3.9), (4.3.13) - (4.3.15).

a) If x is critical, then

$$x(\alpha) = x, \qquad \forall\ \alpha > 0.$$

b) If x is not critical, then

$$\langle \nabla f(x), g\rangle > 0, \tag{4.3.26}$$

and

$$\langle \nabla f(x), x - x(\alpha)\rangle \geq \alpha \langle d_x, D_x d_x\rangle + \frac{1}{\alpha}\|x(\alpha) - (x + \alpha\tilde{d}_x)\|^2 \geq 0 \quad \forall\ \alpha\varepsilon\left(0, \frac{\varepsilon}{\|g\|}\right). \tag{4.3.27}$$

Furthermore there exists $\bar{\alpha} > 0$ such that

$$f(x) > f[x(\alpha)], \qquad \forall\ \alpha\in(0,\bar{\alpha}]. \tag{4.3.28}$$

Proof: a) It is easily seen that for every $z\in C_x$ we have

$$(x + \frac{\varepsilon}{\|z\|}z)\in X \tag{4.3.29}$$

in view of the definitions (4.3.4) - (4.3.6). Since x is critical we have $\langle\nabla f(x),y-x\rangle \geq 0$ for all $y\in X$. Therefore using (4.3.29) we have

$$\langle\nabla f(x),z\rangle \geq 0, \qquad \forall\ z\in C_x. \tag{4.3.30}$$

From the definitions of C_x^+, d_x and d_x^+ [cf. (4.3.6) - (4.3.9)] and (4.3.30) it follows that

$$-\nabla f(x)\in C_x^+$$

and

$$d_x^+ = -\nabla f(x), \quad d_x = 0.$$

Using (4.3.13) - (4.3.15) we obtain $x(\alpha) = P[x - \alpha\nabla f(x)]$. Since x is critical we have that $x = P[x - \alpha\nabla f(x)]$ for all $\alpha \geq 0$ and the conclusion follows.

b) We have using the facts $\nabla f(x) = -(d_x + d_x^+)$ and $\langle\tilde{d}_x,d_x^+\rangle = 0$

$$\langle\tilde{d}_x,\nabla f(x)\rangle = -\langle\tilde{d}_x,d_x + d_x^+\rangle = -\langle\tilde{d}_x,d_x\rangle. \tag{4.3.31}$$

Now $\tilde{d}_x$ is the projection of D_xd_x on the cone $C_x \cap \{z \mid \langle z,d_x^+\rangle = 0\}$, and d_x is a direction of recession of this cone since it belongs to it. Therefore from Lemma (4.A) part b) we obtain

$$\langle d_x,\tilde{d}_x\rangle \geq \langle d_x,D_xd_x\rangle. \tag{4.3.32}$$

Combining (4.3.31) and (4.3.32) we obtain

$$\langle \tilde{d}_x, \nabla f(x) \rangle \leq - \langle d_x, D_x d_x \rangle \leq 0 \tag{4.3.33}$$

where the second inequality is strict if and only if $d_x \neq 0$. Also d_x^+ is the projection of $- \nabla f(x)$ on C_x^+, so

$$\langle d_x^+, \nabla f(x) \rangle \leq 0 \tag{4.3.34}$$

with strict inequality if and only if $d_x^+ \neq 0$. Combining (4.3.33) and (4.3.34) and using the fact $g = - (d_x^+ + \tilde{d}_x)$, we obtain

$$\langle g, \nabla f(x) \rangle \geq 0 \tag{4.3.35}$$

with equality if and only if $d_x = 0$ and $d_x^+ = 0$, or, equivalently $\nabla f(x) = 0$. Since x is not critical we must have $\nabla f(x) \neq 0$, so strict inequality holds in (4.3.35) and (4.3.26) is proved.

Take any $\alpha \in (0, \frac{\varepsilon}{\|g\|})$. Since projection on a closed convex set is a nonexpansive operator (see e.g. [8]), we have

$$\|x(\alpha) - x\| \leq \|x - \alpha g - x\| = \alpha \|g\| < \varepsilon \tag{4.3.36}$$

Therefore we have

$$\langle a_i, x \rangle \leq b_i - \varepsilon \|a_i\| \leq b_i - \langle a_i, x(\alpha) - x \rangle, \quad \forall\, i \notin I_x$$

and

$$\langle a_i, x(\alpha) \rangle \leq b_i, \quad \forall\, i \notin I_x.$$

It follows that $x(\alpha)$ is also the projection of the vector $x - \alpha g$ on the set $\Omega_x \supset X$ given by

$$\Omega_x = \{ z \mid \langle a_i, z \rangle \leq b_i,\ i \in I_x \},$$

i.e.,

$$x(\alpha) = \arg\min \{\|z - (x - \alpha g)\| \mid z \in \Omega_X\}. \tag{4.3.37}$$

Now the vector d_X is easily seen to be a direction of recession of the set Ω_X, so by Lemma (4.A) part b) we have

$$\langle d_X, x(\alpha)\rangle \geq \langle d_X, x - \alpha g\rangle = \langle d_X, x + \alpha d_X^+ + \alpha \tilde{d}_X\rangle.$$

Since $\langle d_X, d_X^+\rangle = 0$, the relation above is written by using also (4.3.32)

$$-\langle d_X, x - x(\alpha)\rangle \geq \alpha \langle d_X, D_X d_X\rangle. \tag{4.3.38}$$

In view of the fact $\tilde{d}_X \in C_X$ we have $(x + \alpha \tilde{d}_X) \in \Omega_X$, and since $x(\alpha)$ is the projection on Ω_X of $(x + \alpha d_X^+ + \alpha \tilde{d}_X)$ [cf. (4.3.37)] we have

$$\langle x + \alpha d_X^+ + \alpha \tilde{d}_X - x(\alpha), x + \alpha \tilde{d}_X - x(\alpha)\rangle \leq 0.$$

Equivalently, using the fact $\langle d_X^+, \tilde{d}_X\rangle = 0$,

$$-\langle d_X^+, x - x(\alpha)\rangle \geq \frac{\|x(\alpha) - (x + \alpha \tilde{d}_X)\|^2}{\alpha}. \tag{4.3.39}$$

By combining (4.3.38) and (4.3.39) and using the fact $\nabla f(x) = -(d_X + d_X^+)$ we obtain

$$\langle \nabla f(x), x - x(\alpha)\rangle \geq \alpha \langle d_X, D_X d_X\rangle + \frac{\|x(\alpha) - (x + \alpha \tilde{d}_X)\|^2}{\alpha} \tag{4.3.40}$$

which is the left inequality in (4.3.27). To show that the right side of (4.3.40) cannot be zero, note that if it were then we would have both $d_X = 0$ (implying $\tilde{d}_X = 0$, $x(\alpha) = x - \alpha \nabla f(x)$) and $x(\alpha) = x + \alpha \tilde{d}_X$ [implying $p(x - \alpha \nabla f(x)) = x$]. Since x is not critical, we arrive at a contradiction. Therefore the right inequality in (4.3.27) is also proved.

By using the mean value theorem we have

$$f(x) - f[x(\alpha)] = \langle \nabla f(x), x - x(\alpha) \rangle + \langle \nabla f(\zeta_\alpha) - \nabla f(x), x - x(\alpha) \rangle \qquad (4.3.41)$$

where ζ_α lies on the line segment joining x and $x(\alpha)$. Using (4.3.40) and (4.3.41) we obtain for all $\alpha \in (0, \frac{\varepsilon}{\|g\|})$

$$\frac{1}{\alpha}\{f(x) - f[x(\alpha)]\} \geq \langle d_x, D_x d_x \rangle + \frac{\|x(\alpha) - (x + \alpha \tilde{d}_x)\|^2}{\alpha^2} \qquad (4.3.42)$$

$$+ \langle f(\zeta_\alpha) - \nabla f(x), \frac{x - x(\alpha)}{\alpha} \rangle .$$

Using (4.3.36) and the Cauchy-Schwartz inequality we see that

$$\langle \nabla f(\zeta_\alpha) - \nabla f(x), \frac{x - x(\alpha)}{\alpha} \rangle \geq - \|\nabla f(\zeta_\alpha) - \nabla f(x)\| \cdot \|g\| . \qquad (4.3.43)$$

Since $\|\nabla f(\zeta_\alpha) - \nabla f(x)\| \to 0$ as $\alpha \to 0$ we see from (4.3.42) and (4.3.43) that if $d_x \neq 0$ then for all positive but sufficiently small α we have $f(x) > f[x(\alpha)]$. If $d_x = 0$ then $\tilde{d}_x = 0$ and using Lemma (4.A) part a)

$$\frac{\|x(\alpha) - (x + \alpha \tilde{d}_x)\|^2}{\alpha^2} = \frac{\|x(\alpha) - x\|^2}{\alpha^2} \geq \|x(1) - x\|^2, \ \forall \ \alpha \in (0,1]. \qquad (4.3.44)$$

From (4.3.42), (4.3.43) and (4.3.44) we see again that when $d_x = 0$ then for all positive but sufficiently small α we have $f(x) > f[x(\alpha)]$. Therefore, there exists $\bar{\alpha} > 0$ such that (4.3.28) holds in both cases where $d_x = 0$ and $d_x \neq 0$.

Q.E.D.

4.4 Convergence Analysis

The previous section has shown how a vector $x\varepsilon X$, a scalar $\varepsilon>0$, an index set I_x satisfying

$$I_x \supset \{i\varepsilon I \mid \langle a_i, x\rangle \geq b_i - \varepsilon\|a_i\|\},$$

and a positive definite self-adjoint operator $D_x: \Gamma_x \to \Gamma_x$ where Γ_x is the subspace defined by (4.3.6) - (4.3.11), uniquely define an arc of points $x(\alpha)\varepsilon X$, $\alpha \geq 0$ where

$$x(\alpha) = P(x - \alpha g), \quad \alpha \geq 0$$

and g is defined via (4.3.8), (4.3.9), (4.3.13) - (4.3.15). Furthermore for each $x\varepsilon X$ which is not critical, Proposition 1b) shows that by choosing α sufficiently small we can obtain a point of lower cost on this arc. Therefore any procedure that, for any given $x\varepsilon X$, chooses I_x, ε, and D_x satisfying the above requirements, coupled with a rule for selecting a point of lower cost on the corresponding arc $x(\alpha)$ leads to a descent algorithm. There is a large variety of possibilities along these lines but we will focus attention on the following broad class of methods:

We assume that we are given a continuous function $\varepsilon: X \to R$ such that

$$\varepsilon(x) \geq 0, \qquad \forall x\varepsilon X \tag{4.4.1}$$

$$\varepsilon(x) = 0 \implies x \text{ is critical} \tag{4.4.2}$$

(for example $\varepsilon(x) = \min\{\varepsilon, \|x - P[x - \nabla f(x)]\|\}$ where $\varepsilon > 0$ is a given constant). We are also given scalars $\beta\varepsilon(0,1)$, $\sigma\varepsilon(0,1/2)$, $\lambda_1 > 0$ and $\lambda_2 > 0$ with $\lambda_1 \leq \lambda_2$.

At the beginning of the kth iteration of the algorithm we have a vector $x_k \in X$. If x_k is stationary we set $x_{k+1} = x_k$. Else we obtain the next vector x_{k+1} as follows:

Step 1: Choose an index set $I_k \subset I$ satisfying

$$I_k \quad \{i \in I \mid \langle a_i, x_k \rangle \geq b_i - \epsilon(x_k)\|a_i\|\}, \tag{4.4.3}$$

and compute

$$d_k = \arg\min \{\|z + \nabla f(x_k)\| \mid z \in C_k\} \tag{4.4.4}$$

$$d_x^+ = -[\nabla f(x_k) + d_k] \tag{4.4.5}$$

where

$$C_k = \{z \mid \langle a_i, z \rangle \leq 0, \quad i \in I_k\}. \tag{4.4.6}$$

Step 2: Choose a positive definite self-adjoint operator $D_k : \Gamma_k \to \Gamma_k$, where

$$\Gamma_k = \text{span}\{C_k \cap \{z \mid \langle z, d_k^+ \rangle = 0\}\}. \tag{4.4.7}$$

and D_k satisfies

$$\|D_k\| \leq \lambda_2, \quad \text{and} \quad \lambda_1\|z\|^2 \leq \langle z, D_k z \rangle, \quad \forall\, z \in \Gamma_k \tag{4.4.8}$$

Compute $\tilde{d}_k$ given by

$$\tilde{d}_k = \arg\min \{\|z - D_k d_k\| \mid z \in C_k, \langle z, d_k^+ \rangle = 0\}. \tag{4.4.9}$$

Define

$$g_k = -(d_k^+ + \tilde{d}_k) \tag{4.4.10}$$

and

$$x_k(\alpha) = P(x_k - \alpha g_k), \quad \forall\, \alpha > 0. \tag{4.4.11}$$

Step 3: Set

$$x_{k+1} = x_k(\alpha_k) \tag{4.4.12}$$

where

$$\alpha_k = \beta^{m_k} \tag{4.4.13}$$

and m_k is the first nonnegative integer m satisfying

$$f(x_k) - f[x_k(\beta^m)] \geq \sigma \{\beta^m \langle d_k, D_k d_k \rangle + \frac{\|x_k(\beta^m) - (x_k + \beta^m \tilde{d}_k)\|^2}{\beta^m}\} \tag{4.4.14}$$

Proposition (4.A) part b) shows that x_{k+1} is well defined via the stepsize rule (4.4.12) - (4.4.14) in the sense that m_k is a (finite) integer and furthermore

$$f(x_k) > f(x_{k+1})$$

for all k for which x_k is not critical. The following proposition is our main convergence result.

Proposition (4.B): Every limit point of a sequence $\{x_k\}$ generated by the algorithm above is a critical point.

Proof: Let $\{x_k\}_K$ be a subsequence of $\{x_k\}$ converging to a point $\bar{x}$ which is not critical. We will arrive at a contradiction. Since $\{\alpha_k\}$ is bounded we assume without loss of generality that

$$\lim_{\substack{k\to\infty \\ k\in K}} \alpha_k = \bar{\alpha}$$

where $\bar{\alpha} \in [0,1]$. Since $\{f(x_k)\}$ decreases monotonically to $f(\bar{x})$ it follows from the form of the stepsize rule that

$$\lim_{\substack{k\to\infty \\ k\in K}} \alpha_k \langle d_k, D_x d_x \rangle = 0 \tag{4.4.15}$$

$$\lim_{\substack{k\to\infty \\ k\in K}} \frac{\| x_k(\alpha_k) - (x_k + \alpha_k \tilde{d}_k) \|^2}{\alpha_k} = 0 \tag{4.4.16}$$

We consider two cases:

Case 1 ($\bar{\alpha} > 0$): It follows from (4.4.15) and the fact $\langle d_k, D_k d_k \rangle \geq \lambda_1 \| d_k \|^2$ (cf. (4.4.8)) that $\lim_{\substack{k\to\infty \\ k\in K}} d_k = 0$, and therefore also $\lim_{\substack{k\to\infty \\ k\in K}} \tilde{d}_k = 0$, $\lim_{\substack{k\to\infty \\ k\in K}} d_k^+ = \nabla f(\bar{x})$.

By taking limit as $k\to\infty$, $k\in K$, in the equation $x_k(\alpha_k) = P(x_k + \alpha_k d_k^+ + \alpha_k \tilde{d}_k)$, using the coninuity of the P operator which follows because P is nonexpansive we obtain

$$\lim_{\substack{k\to\infty \\ k\in K}} x_k(\alpha_k) = P[\bar{x} - \bar{\alpha} f(\bar{x})].$$

Therefore (75) yields

$$\bar{x} = P[\bar{x} - \bar{\alpha} \nabla f(\bar{x})].$$

Since $\bar{\alpha} > 0$ this implies that $\bar{x}$ is critical thereby contradicting our earlier assumption.

Case 2 ($\bar{\alpha}=0$): It follows that for all $k\in K$ which are sufficiently large

$$f(x_k) - f[x_k(\tfrac{\alpha_k}{\beta})] < \sigma \{ \frac{\alpha_k}{\beta} \langle d_k, D_k d_k \rangle + \frac{\| x_k(\frac{\alpha_k}{\beta}) - (x_k + \frac{\alpha_k}{\beta} \tilde{d}_k) \|^2}{\frac{\alpha_k}{\beta}} \}, \tag{4.4.17}$$

i.e., the test (4.4.14) of the stepsize rule will be failed at least once for all $k\in K$ sufficiently large.

Since $g_k = -(d_k^+ + \tilde{d}_k)$, $\langle d_k^+, \tilde{d}_k \rangle = 0$, we have

$$\|g_k\|^2 = \|d_k^+\|^2 + \|\tilde{d}_k\|^2. \tag{4.4.18}$$

Since $\tilde{d}_k$ is the projection of $D_k d_k$ on $C_x \cap \{z \mid \langle z, d_k^+ \rangle = 0\}$ we must have $\|\tilde{d}_k\| \leq \|D_k d_k\|$ and, using (4.4.8), $\|\tilde{d}_k\| \leq \lambda_2 \|d_k\|$. Therefore from (4.4.18) and the fact $\|d_k^+\| \leq \|\nabla f(x_k)\|$, $\|d_k\| \leq \|\nabla f(x_k)\|$ we obtain

$$\|g_k\|^2 \leq (1 + \lambda_2^2)\|\nabla f(x_k)\|^2.$$

It follows that

$$\limsup_{\substack{k \to \infty \\ k \in K}} \|g_k\| < \infty. \tag{4.4.19}$$

We also have

$$\lim_{\substack{k \to \infty \\ k \in K}} \varepsilon(x_k) = \varepsilon(\bar{x}) > 0. \tag{4.4.20}$$

It follows from (4.4.19), (4.14.20) and the fact $\bar{\alpha} = 0$ that for all $k \in K$ sufficiently large $\frac{\alpha_k}{\beta} \in (0, \frac{\varepsilon(x_k)}{\|g_k\|})$ and therefore using Proposition (4.A) part b) [cf. (4.3.27)] we obtain

$$\langle \nabla f(x_k), x_k - x_k(\tfrac{\alpha_k}{\beta}) \rangle \geq \frac{\alpha_k}{\beta} \langle d_k, D_k d_k \rangle + \frac{\|x_k(\frac{\alpha_k}{\beta}) - (x_k + \frac{\alpha_k}{\beta} d_k)\|^2}{\frac{\alpha_k}{\beta}}. \tag{4.4.21}$$

Using the mean value theorem we have

$$f(x_k) - f[x_k(\tfrac{\alpha_k}{\beta})] = \langle \nabla f(x_k), x_k - x_k(\tfrac{\alpha_k}{\beta}) \rangle + \tag{4.4.22}$$

$$+ \langle \nabla f(\zeta_k) - \nabla f(x_k), x_k - x_k(\tfrac{\alpha_k}{\beta}) \rangle$$

where ζ_k lies on the line segment connecting x_k and $x_k(\frac{\alpha_k}{\beta})$. From (4.4.17), (4.4.21), and (4.4.22) we obtain for all $k \in K$ sufficiently large

$$(1-\sigma)\left\{\langle d_k, D_k d_k\rangle + \frac{\| x_k(\frac{\alpha_k}{\beta}) - (x_k + \frac{\alpha_k}{\beta}\tilde{d}_k)\|^2}{(\frac{\alpha_k}{\beta})^2}\right\}$$

$$\leq \langle \nabla f(x_k) - \nabla f(\zeta_k), \frac{x_k - x_k(\frac{\alpha_k}{\beta})}{\frac{\alpha_k}{\beta}} \rangle. \tag{4.4.23}$$

Since [cf. (4.3.36),(4.4.19)] we have

$$\limsup_{\substack{k\to\infty \\ k\in K}} \frac{\| x_k - x_k(\frac{\alpha_k}{\beta})\|}{\frac{\alpha_k}{\beta}} \leq \limsup_{\substack{k\to\infty \\ k\in K}} \| g_k \| < \infty$$

and $\lim_{\substack{k\to\infty \\ k\in K}} \| \nabla f(x_k) - \nabla f(\zeta_k)\| = 0$ it follows that the right side of (4.4.23) tends to zero as $k\to\infty$, $k\in K$. Therefore so does the left side which implies that

$$\lim_{\substack{k\to\infty \\ k\in K}} d_k = 0, \qquad \lim_{\substack{k\to\infty \\ k\in K}} \tilde{d}_k = 0 \tag{4.4.24}$$

and

$$\lim_{\substack{k\to\infty \\ k\in K}} \frac{\| x_k(\frac{\alpha_k}{\beta}) - (x_k + \frac{\alpha_k}{\beta}\tilde{d}_k)\|^2}{(\frac{\alpha_k}{\beta})^2} = 0. \tag{4.4.25}$$

Since it follows from (4.4.20) and (4.4.24) that there exists $\bar{k}$ such that

$$x_k + \frac{\alpha_k}{\beta}\tilde{d}_k \in X \qquad \forall\, k \geq \bar{k}$$

we obtain using Lemma 1a)

$$\frac{\|x_k(\frac{\alpha_k}{\beta}) - (x_k + \frac{\alpha_k}{\beta}\tilde{d}_k)\|^2}{(\frac{\alpha_k}{\beta})^2} \geq \|P[(x_k + \frac{\alpha_k}{\beta}\tilde{d}_k) + d_k^+] - (x_k + \frac{\alpha_k}{\beta}\tilde{d}_k)\|^2. \qquad (4.4.26)$$

From (4.4.25) and (4.4.26) it follows that

$$\lim_{\substack{k\to\infty \\ k\in K}} \|P[(x_k + \frac{\alpha_k}{\beta}\tilde{d}_k) - (\nabla f(x_k) + d_k)] - (x_k + \frac{\alpha_k}{\beta}\tilde{d}_k)\|^2 = 0.$$

Using (4.4.24) we obtain

$$\|P[\bar{x} - \nabla f(\bar{x})] - \bar{x}\| = 0$$

which contradicts the assumption that $\bar{x}$ is not critical. Q.E.D. .

We mention that some of the requirements on the sequences $\{\varepsilon(x_k)\}$ and $\{D_k\}$ can be relaxed without affecting the result of Proposition (4.B). In place of continuity of $\varepsilon(\cdot)$ and assumption (4.4.8) it is sufficient to require that if $\{x_k\}_K$ is a subsequence converging to a noncritical point $\bar{x}$, then

$$\liminf_{\substack{k\to\infty \\ k\in K}} \varepsilon(x_k) > 0,$$

$$\liminf_{\substack{k\to\infty \\ k\in K}} \inf \{\langle z, D_k z\rangle \mid \|z\| = 1,\ z\in\Gamma_k\} > 0$$

$$\limsup_{\substack{k\to\infty \\ k\in K}} \|D_k\| < \infty.$$

This can be verified by inspection of the proof of Proposition (4.B).

A practically important generalization of the algorithm results if we allow the norm on the Hilbert space H to change from one iteration to the next. By this we mean that at each iteration k a new inner product $\langle .,. \rangle_k$ and corresponding norm $\|\cdot\|_k$ on H are considered. The statement of the algorithm and corresponding assumptions must be modified as follows:

a) The gradient $\nabla f(x_k)$ will be with respect to the current inner product $\langle .,. \rangle_k$ [cf. (4.2.3)].

b) The projection defining d_k, d_k^+, $\tilde{d}_k$ and the arc $x_k(\cdot)$ should be with respect to the current norm $\|\cdot\|_k$.

c) The assumptions on I_k, and D_k, and the stepsize rule should be restated in terms of the current inner product and norm.

There is no difficulty in reworking the proof of Proposition (4.B) for this generalized version of the algorithm provided we assume that all the norms $\|\cdot\|_k$, k = 0, 1,... are "equivalent" to the original norm $\|\cdot\|$ on H in the sense that for some m > 0 and M > 0 we have

$$m\|z\| \leq \|z\|_k \leq M\|z\|, \quad \forall\, z \in H,\ k = 0,1,\ldots \ .$$

Naturally the norms $\|\cdot\|_k$ should be such that projection on X with respect to any one of them is relatively easy for otherwise the purpose of the methodology of this paper is defeated. The motivation for considering a different inner product at each iteration stems from the fact that it is often desirable in nonlinear programming algorithms to introduce iteration-dependent scaling on the optimization variables. This is sometimes referred to as "preconditioning". The use of the operator D_k fulfills that need to a great extent but while this operator scales the component d_x of the negative gradient, it does not affect at all the second component d_x^+.

The role of an iteration-dependent norm can be understood by considering situations where the index set I_k is so large that the cone C_k is empty. In this case $d_k^+ = - \nabla f(x_k)$, $\tilde{d}_k = 0$ and the kth iteration reduces to an iteration of the original Goldstein-Levitin-Poljak method, for which practical experience shows that simple, for example diagonal, scaling at each iteration can sometimes result in spectacular computational savings.

4.5 Rate of Convergence

In this section we will analyze the rate of convergence of algortihm (4.4.3) - (4.4.14) for the case where X is polyhedral and H is finite dimensional. An important property of the Goldstein-Levitin-Poljak method [cf. (4.1.7)] is that if it generates a sequence $\{x_k\}$ converging to a strict local minimum $\bar{x}$ satisfying certain sufficiency conditions (compare with [7]), then after some index $\bar{k}$ the vectors x_k lie on the manifold of active constraints at $\bar{x}$, i.e., $x_k \in \bar{x} + N_{\bar{x}}$ where

$$N_{\bar{x}} = \{z \mid \langle a_i, z\rangle = 0, \ \forall\, i \in A_{\bar{x}}\} \tag{4.5.1}$$

and where

$$A_{\bar{x}} = \{i \mid i \in I, \ \langle a_i, \bar{x}\rangle = b_i\}. \tag{4.5.2}$$

Our algorithm preserves this important characteristic. Indeed, we will see that, under mild assumptions, our algorithm "identifies" the set of active constraints at the limit point in a finite number of iterations, and subsequently reduces to an unconstrained optimization method on this subspace. This brings to bear the rate of convergence results available from unconstrained optimization.

The rate of convergence analysis will be carried out under the following assumptions:

(4.A) H is finite dimensional, X is polyhedral, f is continuously Frechet differentiable, and ∇f is Lipschitz continuous on bounded sets, i.e., for every bounded set there exists $L > 0$ such that for every x and y in the set we have

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|. \tag{4.5.3}$$

(4.B) $\bar{x}$ is a strict local minimum and there exists $\delta>0$ such that

$$P(y)\epsilon x + N_{\bar{x}} \qquad \forall\, y \text{ such that } \|\bar{x} - \nabla f(\bar{x}) - y\| \leq \delta \tag{4.5.4}$$

(4.C) the function $\epsilon(x)$ in the algorithm has the form

$$\epsilon(x) = \min \{\epsilon, \|x - P[x - \nabla f(x)]\|\} \tag{4.5.5}$$

where $\epsilon>0$ is a given scalar. Furhtermore the set I_k in the algorithm is chosen to be [cf. (4.4.3)]

$$I_k = \{i\epsilon I | \langle a_i, x_k\rangle \geq b_i - \epsilon(x_k)\, \|a_i\|\}. \tag{4.5.6}$$

The Lipschitz condition (4.5.3) is satisfied in particular if f is twice continously differentiable. Condition (4.5.4) is a weakened version of an often employed regularity and strict complementarity assumption which requires that the set of vectors $\{a_i | i\epsilon A\_\$$ is linearly independent and all Lagrange multipliers corresponding to the active constraints are strictly positive. The form (4.5.5) for $\epsilon(x)$ is required for technical purposes in our subsequent proof. The reader can verify that there are other forms of $\epsilon(x)$ that are equally suitable. Finally the choice (4.5.5) for the set I_k is natural and is ordinarily the one that is best for alrgorithmic purposes.

the follwoing proposition allows us to transfer rate of convergence results from unconstrained minimization to alforithm (4.4.2) - (4.4.14).

Proposition (4.C): Let x be a limit point of the sequence $\{x_k\}$ generated by iteration (4.4.3) - (4.4.14), and let Assumptions (4.A) - (4.C) hold. Then

$$\lim_{k\to\infty} x_k = x$$

and there exists k such that for all $k \geq k$ we have

$$x_k \in x + N_{\bar{x}} \tag{4.5.7}$$

$$\Gamma_k = \text{span}\{C_k \quad \{z \mid \langle z, d_k^+ \rangle = 0\}\} = N_{\bar{x}}\,, \tag{4.5.8}$$

$$d_k = \arg\min\{\|\nabla f(x_k) + z\| \mid z \in N_{\bar{x}}\}, \tag{4.5.9}$$

$$x_{k+1} = x_k + \alpha_k D_k d_k, \tag{4.5.10}$$

where $\alpha_k = \beta^{m_k}$ and m_k is the first nonnegative integer m for which

$$f(x_k) - f[x_k(\beta^m)] \geq \sigma\beta^m \langle d_k, D_k d_k \rangle. \tag{4.5.11}$$

The proof of Proposition (4.C) is given in Appendix (4.B). From (4.5.10) and (4.5.11) we see that eventually the method reduces to an unconstrained minimization method on the manifold $\bar{x} + N_{\bar{x}}$. The proposition shows that if the matrix D_k is chosen so that for all k sufficiently large it is equal to the inverse Hessian of f restricted on the manifold $\bar{x} + N_{\bar{x}}$ then the method essentially reduces to the unconstrained Newton method and attains a superlinear rate of convergence.

4.6 Algorithmic Variations

Many variations on iteration (4.4.3) - (4.4.14) are possible. One of them, changing the metric on the Hilbert space H from iteration to iteration, was discussed at the end of Section 4.4. In this section we discuss other variations. These will include the use, in various cases, of a pseudometric on H instead of a metric, variations on the step size rules and finally variations on the various projections in (4.4.3) - (4.4.14). We will state the variations without a convergence proof. In each case, the reworking of the proofs of Sections 4.3 - 4.4 to show that the variation is valid, poses no difficulty.

4.6.1 Singular Transformation of Variables through a Pseudometric

Here we address the case where X is not a solid body in H, i.e. for some linear manifold M we have $X \subset M \neq H$. In this case we observe that (4.3.27) is the only place where a metric as opposed to a pseudometric is needed. Noticing that if $X \subset M$, then all quantities in (4.3.27) belong to M, one can conclude that all that is necessary is to have a metric on M. This leads us to consider the use of pseudometric on H provided it induces a metric on M. Furthermore, we can change the pseudometric on H from iteration to iteration, as we can change the metric, provided that the metrics induced on M are equivalent in the sense described in Section 4.4.

The introduction of a pseudometric serves to facilitate the projection further. In Example (4.C) below we rework example (4.B) using a pseudometric. Specifically we assume that for some $\bar{i} \in \{1,\ldots,n\}$ [cf. (4.3.17)] $s^{\bar{i}}$ is zero. The resulting pseudometric (4.3.17) restricted to $M = \{x \mid \sum_{i=1}^{n} x^i = 0\}$ induces a metric on M. We define projections on M taking limits in Example (4.B) as $s^{\bar{i}}$ approaches zero. Although the gradient of f with respect to the pseudometric does not exist, the limit of the projection of $\nabla f(x)$ on M as $s^{\bar{i}}$ approaches zero does exit and it is the unique vector $q(x) \in M$ satisfying

$$f(z) = f(x) + q(x)'S(z - x) + o\|z - x\| \qquad \forall\, z \in M,\ \forall\, x \in M.$$

where S is a diagonal matrix with s^i in its (i,i) position.

Example 4.C: Let s^i in Example (4.B) be zero. By taking limit as $s^i \to 0$ we derive that

$$I_X = \{i \mid \bar{i},\ \text{or}\ 0 \leq x^i \leq \varepsilon / \sqrt{s^i}\ \ \forall\, i \neq i\}.$$

In order to find d_X we have to take the limit in (4.3.20). There is one

case, and for our purposes an important one, in which this limit is readily obtained. Assume without loss of generality that $\bar{i}$ satisfies

$$\bar{i} = \underset{i\in\{1,\ldots,n\}}{\arg\min} \left\{\frac{\partial f(x)}{\partial x^i}\right\}.$$

Then

$$d_x^i = \begin{cases} \dfrac{1}{s^i}\left(\dfrac{\partial f(x)}{\partial x^{\bar{i}}} - \dfrac{\partial f(x)}{\partial x^i}\right) & i\neq\bar{i} \\ -\sum\limits_{j\neq 1}\dfrac{1}{s^j}\left(\dfrac{\partial f(x)}{\partial x^{\bar{i}}} - \dfrac{\partial f}{\partial x^j}\right) & i=\bar{i}, \end{cases}$$

and

$$\tilde{I}_x = \left\{ i \,\middle|\, i\in\hat{I}_x \text{ and } \frac{\partial f(x)}{\partial x^i} < \frac{\partial f(x)}{\partial x^{\bar{i}}} \right\}.$$

It can be seen using the definitions that

$$\tilde{I}_x = \hat{I}_x - \bar{i}$$

and therefore among all choices of $\bar{i}$, the one made above results in the maximality of $\tilde{I}_x$.

The advantage of such a pseudometric is easily manifested, once it is realized that by using it we dispense with the need to solve the piecewise linear equations in example (4.B) [cf.(4.3.19), (4.3.24)]. On the other hand examples involving a quadratic objective can be constructed in which, when using a metric the algorithm converges in one step, while with the use of a pseudometric it takes several steps. This will happen when $\nabla^2 f$ is diagonal positive definite.

4.6.2 Step Size Rules

The Armijo-like rule (4.4.14) can be viewed as combination of the Armijo rule used in unconstained minimization [9], and an Armijo like rule for constrained optimization proposed by Bertsekas in [[7], cf. eq (12)]. Corresponding to an alternate suggestion made there [[7], cf. eq (22)] we can replace (4.4.14) by

$$f(x_k) - f(x_k(\beta^m)) \geq \sigma\{\beta^m \langle d_k, D_k d_k\rangle + \langle \nabla f(x_k), (x_k + \beta^m \tilde{d}_k) - x_k(\beta^m)\rangle\}. \tag{4.6.1}$$

Also, a variation of the Goldstein step size rule [9] can be employed, in which $\sigma < 0.5$ and α is chosen such that

$$\begin{aligned}(1 - \sigma)\{\alpha\langle d_k, D_k d_k\rangle + \langle \nabla f(x_k), (x_k + \alpha\tilde{d}_k) - x_k(\alpha)\rangle\} \geq \\ f(x_k) - f(x_k(\alpha)) \geq \sigma\{\alpha\langle d_k, D_k d_k\rangle + \\ \langle \nabla f(x_k), (x_k + \alpha\tilde{d}_k) - x_k(\alpha)\rangle\}.\end{aligned} \tag{4.6.2}$$

The rule (4.6.2) is the counterpart of (4.6.1). The reader can easily construct the counterpart to (4.4.14).

4.6.3 Variations on the Projections

There is one central observation, namely, the projections of $D_k d_k$ and d_k^+ on any closed convex set for which d_k is a direction of recession, result in descent directions. By employing different sets with this property, variations on the algorithm result since different directions may be obtained and different arcs may be searched.

The first variation is to replace C_k in (4.4.9) by $(\Omega_k - x_k)$, i.e.

$$\tilde{d}_k = \arg\min\{\|z - D_k d_k\| \mid z \in \Omega_k - x_k, \langle z, d_k^+\rangle = 0\} \tag{4.6.3}$$

where

$$\Omega_k = \{z \mid \langle a_i, z\rangle \leq b_i, \ \forall\, i \in I_k\}.$$

Evidently

$$\Omega_k - x_k \ \ C_k$$

and as a result d_k is a direction of recession of $\Omega_k - x_k$, which implies that $\tilde{d}_k$ defined by (4.6.3) is a descent direction.

Interestingly, this variation gives rise to a variation in the stepsize search. Since the set $\{z \mid z \in C_k, \ \langle z, d_k^+\rangle = 0\}$ is a cone, it is easily verified that

$$\alpha d = \arg\min \{\|\alpha D_k d_k - z\| \mid z \in C_k, \ \langle z, d_k^+\rangle = 0\}.$$

Thus, (4.4.11) can be interpreted as

$$x_k(\alpha) = P[x_k + \alpha d_k^+ + q_k(\alpha)]$$

where

$$q_k(\alpha) = \arg\min \{\|\alpha D_k d_k - z\| \mid z \in C_k, \ \langle z, d_k^+\rangle = 0\}.$$

Since C_k can be replaced by $\Omega_k - x_k$, a new algorithm results by using the arc

$$x_k(\alpha) = P[x_k + \alpha d_k + \tilde{q}_k(\alpha)]$$

where

$$\tilde{q}_k(\alpha) = \arg\min \{\|\alpha D_k d_k - z\| \mid z \in \Omega_k - x_k, \ \langle z, d_k^+\rangle = 0\}.$$

Indeed, the particular algorithm suggested in [6] can be considered to be an implementation of the last variation for an orthant constraint.

4.7. Multicommodity Network Flow Problems

4.7.1 Application of the Algorithm to the Multicommodity Flow Problem

In this last section we apply algorithm (4.4.3)-(4.4.14) to a classical nonlinear multicommodity network flow problem and present some computational results.

We consider a network consisting of N nodes, 1, 2,..., N, and a set of directed links denoted by £. We assume that the network is connected in the sense that for any two nodes m, n, there is a directed path from m to n. We are given a set W of ordered node pairs referred to as origin-destination (or OD) pairs. For each OD pair $w \in W$, we are given a set of directed paths P_w that begin at the origin node and terminate at the destination node. For each $w \in W$ we are also given a positive scalar r_w referred to as the input of OD pair w. This input must be optimally divided among the paths in P_w so as to minimize a certain objective function.

For every path $p \in P_w$ corresponding to an OD pair $w \in W$ we denote by x^p the flow travelling on p. These flows must satisfy

$$\sum_{p \in P_w} x^p = r_w \qquad \forall\, w \in W \tag{4.7.1}$$

$$x^p \geq 0 \qquad \forall\, p \in P_w,\ w \in W \tag{4.7.2}$$

Equations (4.7.1), (4.7.2) define the constraint set of the optimization problem - a Cartesian product of simplices.

In Examples (4.B) and (4.C) we discussed the application of our method to the case of a simplex constraint. It is not difficult to see that if we take a "diagonal" metric on the space, the multicommodity flow problem decomposes in the sense explained below.

Let x denote the vector of variables x^p, $p\epsilon P_w$, $w\epsilon W$, and let x^W denote the vector of variables x^p, $p\epsilon P_w$. Let $C_x(x^W)$ and $\Gamma_x(x^W)$ denote the cone and subspace, respectively, in $R^{|W|}$, generated at x, when all variables aside from those in x^W are considered fixed and $\epsilon = \epsilon(x)$. Then

$$C_x = \prod_{w\epsilon W} C_x(x^W)$$

$$\nabla f(x) = (\ldots, \nabla_{x^W} f(x), \ldots)$$

and

$$\Gamma_x = \prod_{w\epsilon W} \Gamma_x(x^W).$$

Thus all projections decompose and therefore in many respects the multicommodity flow problem is not different from the problem with a single simplex constraint. The only points where the "interaction" among the simplices appears is in computing ϵ_k, and in computing $D_k d_k$.

To every set of path flows $\{x^p | p\epsilon P_w, w\epsilon W\}$ satisfying (4.7.1), (4.7.2) there corresponds a flow f^a for every link $a\epsilon\mathcal{L}$. It is defined by the relation

$$f^a = \sum_{w\epsilon W} \sum_{p\epsilon P_w} 1_p(a) x^p \qquad \forall\ a\epsilon\mathcal{L} \qquad (4.7.3)$$

where $1_p(a) = 1$ if the path p contains the link a and $1_p(a) = 0$ otherwise. If we denote by f the vector of link flows we can write relation (4.7.3)) as

$$f = Ex \qquad (4.7.4)$$

where E is the arc-chain matrix of the network.

For each link $a \in \mathcal{L}$ we are given a convex, twice continuously differentiable scalar function $D_a(f^a)$ with strict positive second derivative for all $f^a \geq 0$. The objective function is given by

For each link $a \in \mathcal{L}$ we are given a convex, twice continuously differentiable scalar function $D_a(f^a)$ with strict positive second derivative for all $f^a \geq 0$. The objective function is given by

$$D(f) = \sum_{a \in \mathcal{L}} D_a(f^a). \tag{4.7.5}$$

By using (4.7.4) we can write the problem in terms of the path flow variables x^p as

$$\text{minimize } J(x) = D(Ex)$$

subject to:

$$\sum_{p \in P_w} x^p = r_w \qquad \forall\ w \in W$$

$$x^p \geq 0 \qquad \forall\ p \in P_w,\ w \in W.$$

In communication network applications the function D may express, for example, average delay per message [10],[11] or a flow control objective [12], while in transportation networks it may arise via a user or system optimization principle formulation [13],[14],[15]. We concentrate on the separable form of D given by (4.7.5), although what follows admits an extension to the non-separable case.

A Newton-like method will be obtained if we chose $D_k d_k$ so that $x_k + D_k d_k$ is the minimum of the quadratic approximation to f on $x_k + \Gamma_k$. For this we must find $A\bar{v}$ where $\bar{v}$ solves

$$\underset{v}{\text{minimize}}\ \langle \nabla J(x_k), Av \rangle + \frac{1}{2} \langle Av, \nabla^2 J(x_k) Av \rangle \tag{4.7.6}$$

and where A is a matrix such that its columns are linearly independent and span Γ_k.

The particular structure of the objective function (4.7.5) gives rise to a Hessian matrix which makes the solution of (4.7.6) relatively easy to obtain. Indeed, using (4.7.5) we can rewrite (4.7.6) as

$$\underset{v}{\text{minimize}} \; \langle E'\nabla D(f_k), Av\rangle + 1/2\langle Av, E'\nabla^2 D(f_k)EAv\rangle, \tag{4.7.7}$$

where $f_k = Ex_k$. A key fact is that problem (4.7.7) can be solved by the Conjugate Gradient (C-G) method using graph type operations and the fact that $\nabla^2 D(f_k)$ is diagnonal without explicitly storing the matrix

$$A'E'\nabla^2 D(f_k)EA.$$

Note that a solution to (4.7.7) exists since $E'\nabla D(f_k)$ is in the range of the nonnegative definite matrix $E'\nabla^2 D(f_k)E$.

4.7.2 Implementation of the C-G Method

A scaled version of the C-G method solves the unconstrained minimization problem

$$\min_z \; b'z + \frac{1}{2} z'Hz \tag{4.7.8}$$

as follows:

A positive definite symmetric matrix S is chosen, and a sequence $\{z_m\}$ is generated according to the iteration

$$z_0 = 0, \; z_{m+1} = z_m + \gamma_m p_m, \qquad m = 0,1,\ldots,$$

where the conjugate direction sequence $\{p_m\}$ is given recursively by

$$p_0 = -Sr_0,\ p_m = -Sr_m + \beta_m p_{m-1}, \qquad m = 1,2,\ldots,$$

the residual sequence $\{r_m\}$ is defined by

$$r_m = Hz_m + b, \qquad m = 0,1,\ldots,$$

and the scalars γ_m and β_m are given by

$$\gamma_m = \frac{r_m' S r_m}{p_m' H p_m}, \qquad m = 0,1,\ldots,$$

$$\beta_m = \frac{r_m' S r_m}{r_{m-1}' S r_{m-1}}, \qquad m = 1,2,\ldots .$$

Applying the method to (4.7.7.) with the inner product that corresponds to the identity matrix and $S = A'\bar{S}A$ where $\bar{S}$ is a diagonal matrix, a sequence $\{v_m\}$ is generated according to

$$v_0 = 0 \quad v_{m+1} = v_m + \gamma_m p_m, \qquad m = 0,1,\ldots,$$

$$p_0 = -A'\bar{S}Ar_0,\ p_m = -A'\bar{S}Ar_m + \beta_m p_{m-1}, \qquad m = 1,2,\ldots,$$

$$r_m = A'E'\nabla^2 D(f_k)EAv_m + A'E'\nabla D(f_k), \qquad m = 0,1,\ldots,$$

$$\gamma_m = \frac{r_m'A'\bar{S}Ar_m}{p_m'A'E'\nabla^2 D(f_k)EAp_m} \qquad m = 0,1,\ldots,$$

$$\beta_m = \frac{r_m'A'\bar{S}Ar_m}{r_{m-1}'A'SAr_{m-1}} \qquad m = 1,2,\ldots .$$

Since we are interested in $A\bar{v}$ we can work directly with a sequence $\{Av_m\}$ which will be given by

$$Av_0 = 0 \quad , \quad (Av_{m+1}) = (Av_m) + \gamma_m p_m, \qquad m = 0,1,\ldots,$$

$$p_0 = -(AA')\bar{S}r_0 \qquad p_m = -(AA')Sr_m + \beta_m p_{m-1} \qquad m = 1,2,\ldots,$$

$$r_m = (AA')[E'\nabla^2(f_k)E(Av_m) + E'\nabla D(f_k)] \qquad m = 0,1,\ldots,$$

$$\gamma_m = \frac{r_m'\bar{S}r_m}{p_m'E'\nabla^2 D(f_k)Ep_m} \qquad m = 0,1,\ldots,$$

$$\beta_m = \frac{r_m'\bar{S}r_m}{r_{m-1}'Sr_{m-1}} \qquad m = 1,2,\ldots \quad .$$

As is well known ([44], [43]) this method will find the solution $A\bar{v}$ of (4.7.7) in at most n_k steps where n_k is the dimension of Γ_k, regardless of the choice of $\bar{S}$. Since n_k might be huge we are primarily interested in an approximate implementation whereby only a few C-G iterations of the method are performed. Under these circumstances the choice of $\bar{S}$ can have a substantial effect on the quality of the final solution. A suitable choice is to take $\bar{S}$ to be the inverse to the diagonal approximation to

$$AA'E'\nabla^2 D(f_k)EAA'.$$

The key to the distributed manner and the facility with which the C-G steps above can be carried out is the interpretation of the various matrix multiplications as graph type operations. Since AA' is block diagonal with respect to OD pairs and $\nabla^2 D(f_k)$ is diagonal the only "real" multiplication is by E and E'. Thus for example to compute

$$E'\nabla^2 D(f_k)E(Av_m)$$

(notice that $E'\nabla^2 D(f_k)Ep_m$ can be derived from it) we interpret (Av_m) as a vector of path flows and then EAv_m is the corresponding vector of link flows; likewise we interpret $\nabla^2 D(f_k)EAv_m$ as a vector of link delays and then $E'\nabla^2 D(f_k)EAv_m$ is the corresponding vector of path delays. Thus if nodes and links are viewed as processors communicating by exchanging messages. The multiplications above can be carried out by sending messages back and forth along paths. The other multiplication needed, that of two vector of the dimension of the paths, as well as coordinating the end and the beginning of a new iteration can be done by flooding or by employment of a "leader" node.

As the reader can see, the linearity of the link flows in the path flows is essential for the method. This precludes, for example, the use of fractions as routing variables. Yet, working in the space of path flows has its price. The number of paths can be exponentially large. We propose two remedies. Both are not attractive.

The standard way of working in the space of path flows is to maintain a list of subset of the paths, those that currently carry flow, appending to it at the end of each iteration the path of the smallest marginal delay. If one appends a new path only if the improvement in the marginal delay, compared to the current minimal one in the subset, is above some threshold, then it looks like the number of paths used will be reduced. Alternatively, one can work with a restricted a priori fixed set of paths for each OD pair.

Overcoming the exponential growth of the number of paths which carry flow (which does not necessarily happen for a centralized algorithm) is a topic for further research.

4.7.3 Computational Results:

A version of the algorithm was run on three examples of the

multicommodity flow problem. The networks are shown in Figures 4.5 - 4.7. Each OD pair was restricted to use only two prespecified paths. This reduced the programming load significantly, yet captured the essence of the algorithm. It is conjectured that the results we obtained are representative of the behavior of the algorithm when applied to more complex multicommodity flow problems.

Since there was no difficulty in deciding whether the particular run was convergent due to the fact that superlinear convergence can be detected by observing the reduction of the cost function, we decided to implement a heuristic version of the algorithm and thereby save computational overhead. By and large, we have tried to avoid the line search. Without it, aside from minimal coordination in doing vector multiplication and the graph type operations mentioned above, the algorithm decomposes according to the OD pairs. We tried various simple stopping rules for the C-G iteration. We have settled on the one which will be described shortly.

The algorithm was operated in three modes distinguished by the rules according to which the C-G method was stopped. In the first mode (denoted by Newton) the C-G iteration was run to the exact solution of problem (4.7.7). In the second mode, (denoted by Approximate Newton) the C-G iteration was run until its residual was reduced by a factor of 1/8 over the starting residual. Finally, in the third mode the C-G method was allowed to perform only one step (to be denoted by 1-step). We used different values ε_k for different OD pairs, according to a variation of (4.4.1) (with $\varepsilon = 0.2$).

The projection metric we used was of the pseudometric type as suggested in Example (4.C). In all three modes, in addition to their particular stopping rule, the C-G method was stopped whenever for any OD pair w the

flow on the path with the smaller entry in the gradient became negative. The last point in the sequence of points generated by the stopped C-G method subiteration was connected by a line to the point preceeding it. The point on the line at which the particular path flow became zero was taken as the result of the C-G iteration.

We used two types of objective functions. The first is

$$D_a(f^a) = \frac{f^a}{C_a - f^a} \qquad \forall\ a \in \mathcal{L}$$

where C_a is a given positive scalar expressing the "capacity" of link a. This function is typically used to express queuing delay in communication networks following Kleinrock's Independence Assumption [19]. The second type was taken to be quadratic. We used two sets of inputs, one to simulate heavy loading and one to simulate light loading. For each combination of cost function, input and network we present the results corresponding to the three versions in Tables 4.3, 4.6 and 4.9.

Our main observation from the results of Table 4.3, 4.6 and 4.9 as well as additional experimentation with multicommodity flow problems is that in the early iterations the 1-step method makes almost as much progress as the other two more sophisticated methods but tends to slow down considerably after reaching the vicinity of the optimum. Also the approximate Newton method does almost as well as Newton's method in terms of number of iterations. However the computational overhead per iteration for Newton's method is considerably larger. This is reflected in the results which show in ten cases out of twelve a larger number of conjugate gradient subiterations for Newton's method. Throughout our computational experiments the approximate Newton method based on conjugate gradient subiterations has performed very well and together with its variations is in our

view the most powerful class of methods available at present for nonlinear multicommodity network flow problems.

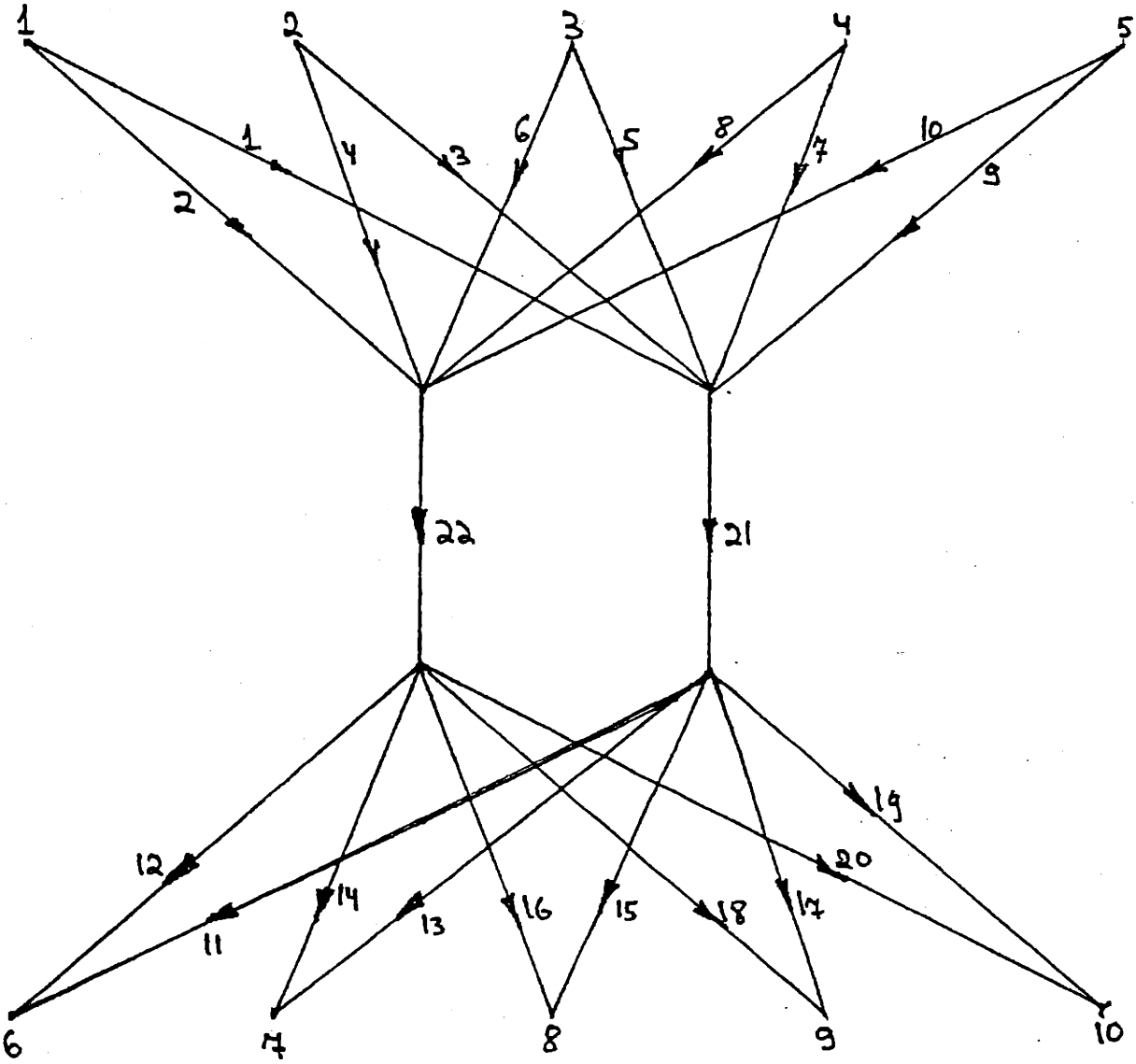


Figure 4.5: Net 1

initially all flows traverse link 21.

$$M = \begin{array}{|c|c|c|c|c|} \hline 10.5 & 20 & 7.5 & 10 & 10 \\ \hline 15 & 5 & 9 & 7.5 & 7.5 \\ \hline 3 & 15 & 6 & 8 & 3 \\ \hline 10 & 6 & 10 & 10 & 14 \\ \hline 50 & 35 & x & x & x \\ \hline \end{array}$$

$$C_1 = m_{ij},\ i = \left\lfloor \frac{1}{5} \right\rfloor + 1,\ j = 1 - 5(i - 1)$$

Table 4.1: Net 1, Capacities

origin \ destination	6	7	8	9	10
1	0.1	1	1.5	2	2.5
2	1	1	1	1	1
3	0.5	0.5	1.5	1.5	3.5
4	0.25	0.25	2	0.25	0.25
5	0.75	0.75	0.75	0	0

Table 4.2: Net 1, Low Input

High Input = Low Input x 1.75

	Initial Value	Final Value	# of Itera-tions	Total # of C-G Subit-erations
Low Load				
Nonquadratic Objective	$1.600616 \cdot 10^{6}$			
Newton		8.743550	16	29
Approximate Newton		8.758665	16	16
1-Step		8.758665	16	16
Quadratic Objective	$1.866326 \cdot 10^{1}$			
Newton		7.255231	5	17
Approximate Newton		7.255231	7	13
1-Step		7.255231	12	12
High Load				
Nonquadratic Objective	$9.759996 \cdot 10^{6}$			
Newton		$3.737092 \cdot 10^{1}$	14	117
Approximate Newton		$3.737745 \cdot 10^{1}$	15	30
1-Step		$3.747400 \cdot 10^{1}$	15	15
Quadratic Objective	$9.759996 \cdot 10^{6}$			
Newton		$1.521299 \cdot 10^{1}$	5	24
Approximate Newton		$1.521299 \cdot 10^{1}$	13	27
1-Step		$1.521301 \cdot 10^{1}$	16	16

Table 4.3: Net 1, Computational Results.

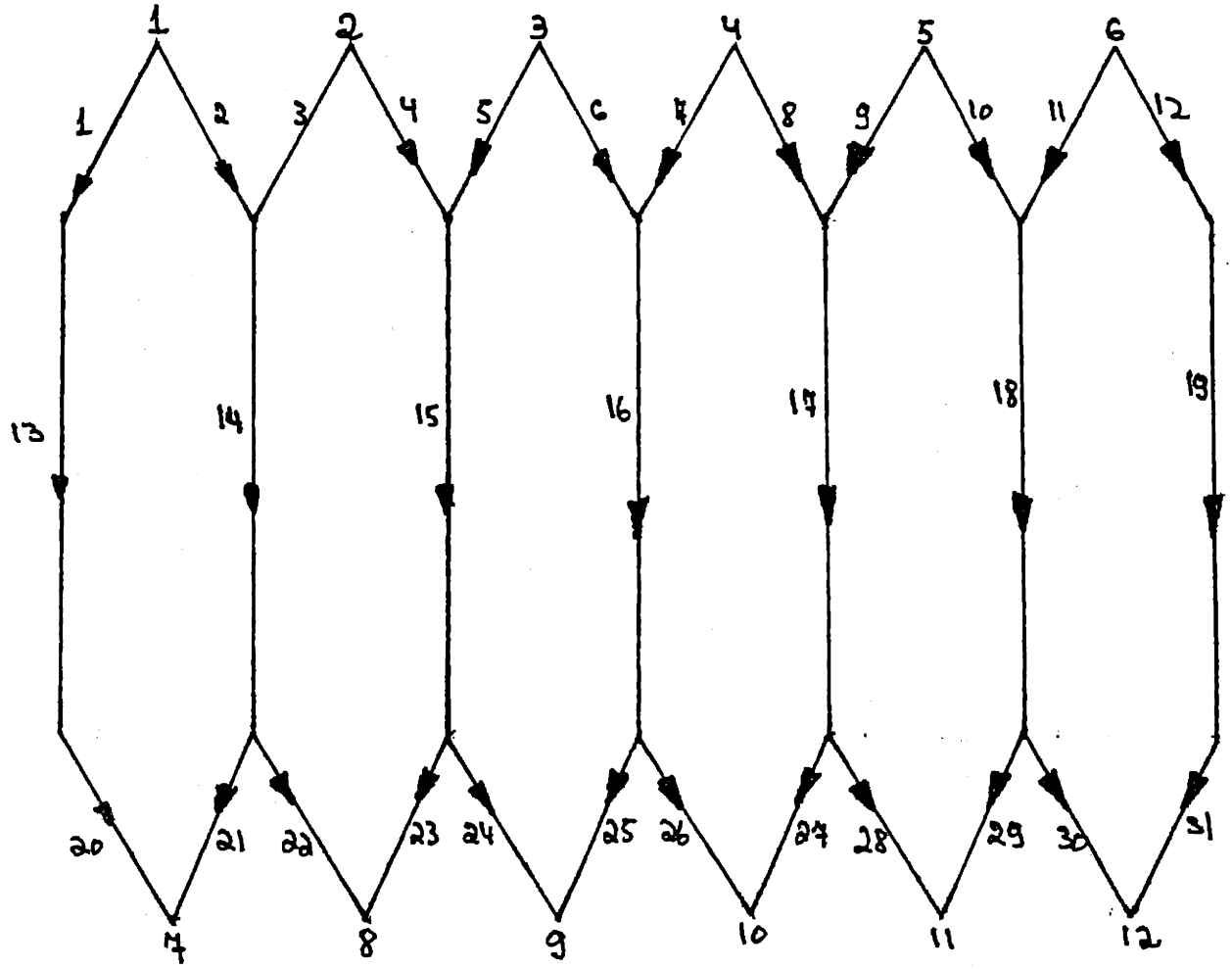


Figure 4.6: Net 2

Initially all flows traverse the paths starting with an even link number.

$c_{13} = 12$	$c_{14} = 10$	$c_{15} = 8$	$c_{16} = 6$
$c_{17} = 4$	$c_{18} = 2$	$c_{19} = 1$	

Table 4.4: Net 2. Capacities

All other link capacities are 99

inp(1,7) = 9.9	inp(2,8) = 7.7	inp(3.8) = 5.4
inp(4.10) = 3.2	inp(5.11) = 1	inp(6,12) = 0.4

Table 4.5: Net 2, Low input

High input = Low input x 1.4

	Initial Value	Final Value	# of Itera-tions	Total # of C-G Subit-erations
Low Load				
Nonquadratic Objective	$1.399349 \cdot 10^{2}$			
Newton		$1.087936 \cdot 10^{1}$	13	44
Approximate Newton		$1.087936 \cdot 10^{1}$	14	34
1-Step		$1.120913 \cdot 10^{1}$	16	16
Quadratic Objective	$1.461559 \cdot 10^{1}$			
Newton		9.561953	3	4
Approximate Newton		9.561953	3	4
1-Step		9.561953	6	6
High Load				
Nonquadratic Objective	$3.863577 \cdot 10^{5}$			
Newton		$5.767141 \cdot 10^{1}$	12	31
Approximate Newton		$4.690345 \cdot 10^{2}$	16	16
1-Step		$4.690345 \cdot 10^{2}$	16	16
Quadratic Objective	$8.916975 \cdot 10^{0}$			
Newton		5.967379	3	7
Approximate Newton		5.967379	3	6
1-Step		5.967379	5	5

Table 4.6: Net 2. Computational Results.

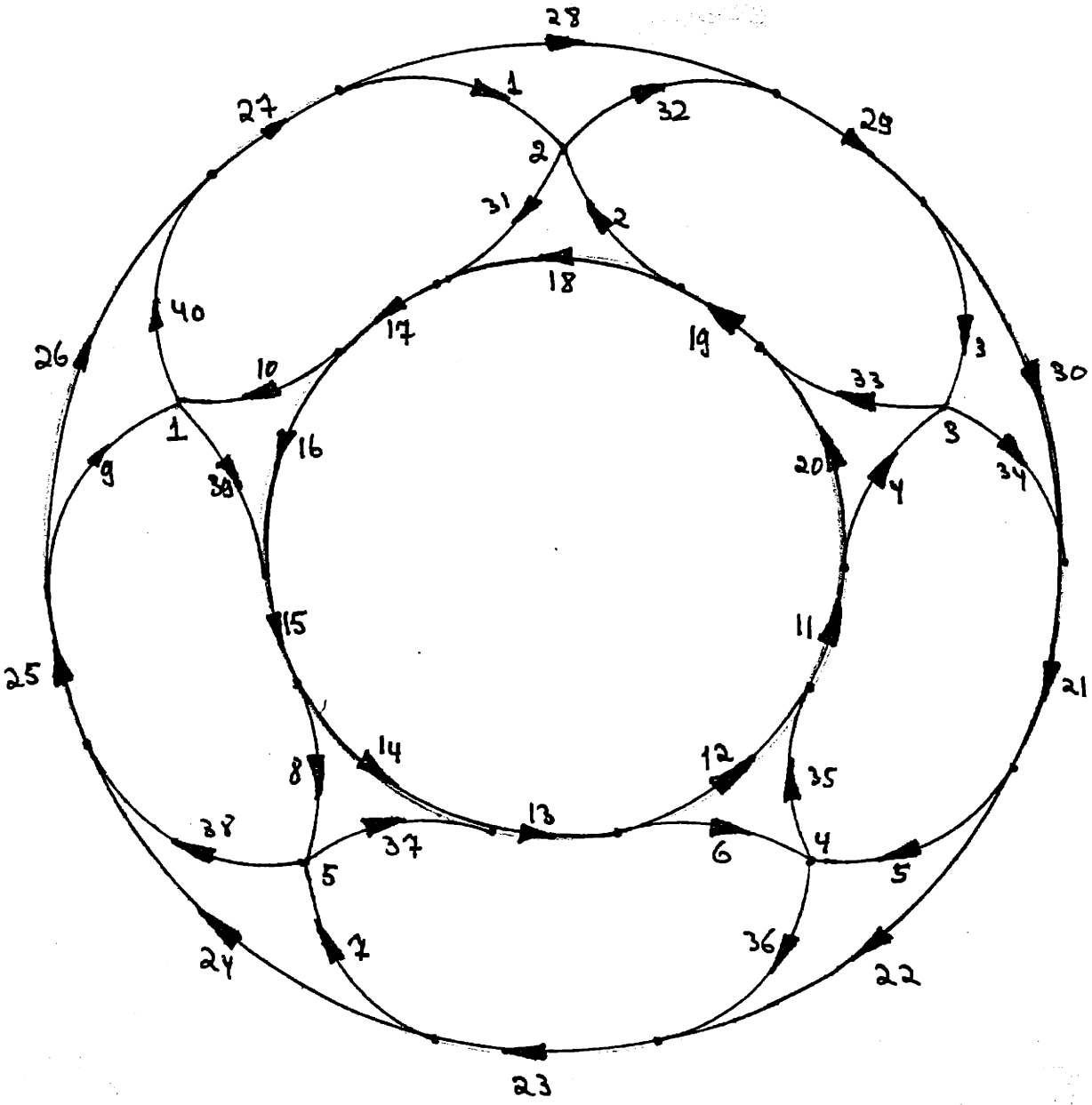


Figure 4.7: Net 3

Permisable paths are those that do not traverse the "circles" only in the initial or final leg. Initially all flows are directed on the longer path among two possible for each OD pair.

$$M = \begin{array}{|c|c|c|c|c|c|c|} \hline 7 & 9 & 7 & 9 & 4 & 5 & 9 \\ \hline 6 & 11 & 8 & 35 & 36 & 37 & 38 \\ \hline 39 & 35 & 36 & 37 & 38 & 39 & 39 \\ \hline 38 & 37 & 36 & 35 & 39 & 38 & 37 \\ \hline 36 & 35 & 12 & 14 & 14 & 12 & 13 \\ \hline 13 & 11 & 25 & 15 & 11 & x & x \\ \hline x & x & x & x & x & x & x \\ \hline \end{array}$$

$$C_1 = m_{ij} \qquad i = \left\lfloor \frac{1}{7} \right\rfloor + 1, \qquad j = 1 - 7(i-1)$$

Table 4.7: Net 3. Capacities

origin \ destination	1	2	3	4	5
1	0	1	2	3	4
2	4	0	3	2	1
3	2	3	0	4	1
4	3	2	1	0	4
5	3	4	1	2	0

Table 4.8: High inputs.

Low input = High inputs x 0.8

	Initial Value	Final Value	# of Itera-tions	Total # of C-G Subit-erations
Low Load				
Nonquadratic Objective	$2.301192 \cdot 10^{1}$			
Newton		$1.566526 \cdot 10^{1}$	8	30
Approximate Newton		$1.566526 \cdot 10^{1}$	10	24
1-Step		$1.568205 \cdot 10^{1}$	16	16
Quadratic Objective	$1.862375 \cdot 10^{1}$			
Newton		$1.319755 \cdot 10^{1}$	7	13
Approximate Newton		$1.319755 \cdot 10^{1}$	6	6
1-Step		$1.319755 \cdot 10^{1}$	6	6
High Load				
Nonquadratic Objective	$1.166363 \cdot 10^{5}$			
Newton		$1.113463 \cdot 10^{5}$	10	48
Approximate Newton		$1.113463 \cdot 10^{5}$	16	54
1-Step		$1.113478 \cdot 10^{5}$	16	16
Quadratic Objective	$3.077163 \cdot 10^{1}$			
Newton		$2.135313 \cdot 10^{1}$	8	10
Approximate Newton		$2.135313 \cdot 10^{1}$	9	9
1-Step		$2.135313 \cdot 10^{1}$	9	9

Table 4.9: Net 3, Computational Results.

5. Data Flow-Control

5.1 Introduction

In [12], Gallager and Golestaani introduced a new approach to data flow control. They proposed to divide the task of flow-control, in the sense of deciding whether to admit a new packet into the network or not, into two subtasks. The two subtasks have complementary roles, one is concerned with the relatively long-term effect of admitted packets, while the other is concerned with the relatively short-term effect.

The first subtask is quasi-static adjustment of the average rate of session flows admitted into the network to the changing environment around the network. This environment is changing due to old sessions termination, new sessions activation, and changing demands imposed on the network by sessions in progress. These changes occur on a rather large time scale, in which we can ignore the individual packet and treat the streams of packets as continuous flows which are not accumulated at intermediate nodes. In this context we adjust the average of the admitted flows so as to strike a balance between the desirability of increasing session rates, and the need to avoid congestion which may hamper the throughput.

The second subtask is instantaneous adjustment of packet admittance to the instantaneous, changing, state of the network. This state is changing due to possible instantaneous fluctuations of the flow admitted to the network and due to the non-deterministic process which governs the delivery of a packet from its origin to its destination. These changes occur on a short time scale in which we deal with the individual packet. In this context we adjust the instantaneous session rates. At times, when the network is momentarily relatively far from congestion, we allow higher rates, while at times, when the network approaches congestion, we cut the rates down.

Certainly, ultimately, it is the second subtask which will determine the average rate of admitted flows, since what is done on the short time scale determines what happens on the longer one. In Section 5.2, the main thrust of this chapter, we address the problem of tuning the mechanism by which the second subtask is performed, so as to achieve the averages determined in the first one.

The mechanism we will consider is that of "window" flow-control. Defined in a broad way, window flow-control is any mechanism which restricts the instantaneous number of packets in the network, which are associated with some generic, network related entities (e.g. links, sessions, network), from exceeding some prespecified values. Thus, for example, the isarithmic flow-control scheme [21] can be considered as a particular implementation of a window associated with the whole network. Similarly, flow-control schemes which are based on buffer thresholds, can be considered as a window associated with a link. Certainly, schemes which are based on the "intersection" of several windows are also possible.

In its narrower and commonly used meaning [21], window flow-control refers to the scheme by which the number of unacknowledged packets for each session is restricted from exceeding a prespecified value. In [36], Gallager suggests that exercising window flow-control by windows associated with entities which are, in some sense, more detailed than sessions,can provide means of achieving dynamic routing. In particular when routing is done, as in this Thesis, by the use of path flow variables, a window can be associated with every session-path pair. To see why such a window can provide dynamic routing, we draw an analogy from the isarithmic flow-control scheme.

A serious objection to the use of isarithmic flow-control as an implementation of a global, network wide window, is that when a local congestion arises, the scheme reduces also the input rate of those sessions which do not affect the congested area. This objection applies to session windows also. A session that uses several paths should not be cut back if only one of these paths is congested. Rather, the uncongested paths should be utilized more while reducing the utilization of the congested path. This can be more closely achieved by using session-path windows.

In Section 5.3 we will take a step toward demonstrating that session-path windows provide, to some degree, dynamic routing. In the next section we propose an iterative algorithm for calculating the sizes of either session windows or session-path windows which give rise to desired session rates or path rates, respectively.

5.2 An Algorithm for Adjusting Windows to Rates

In this section we propose an algorithm by which we can find a window for each session that results in a set of prespecified average input rates. The result, evidently, applies to the session-path window since each session-path pair can be considered as a session.

Unlike data routing in which we assume approximate delay functions and use their first and second derivatives, windows and input rates are bound together through the real delay functions. One should feel uneasy in assuming these functions are known exactly, let alone take their first or second derivatives. Below we propose an algorithm which will adjust windows to achieve prespecified rates without the exact knowledge of the delay functions.

Let S be the set of sessions and assume a certain fixed routing is given. Evidently, under fixed routing, the average flow on each link a, out of the set of links $\mathcal{L}$, is a function of the vector of average session inputs $r=(r^1,r^2,\ldots)'$. Let $D_s(r)$ denote the average time that elapses from the moment session s launches a packet into the network until an acknowledgement of its receipt arrives back. Let $v = (\ldots, vs,\ldots)'$ denote the vector of windows where, v^s is the window allocated to session s. Assuming that a packet is always available to be sent, the relation between rs and vs using Little's formula [39], satisfies

$$v^s = r^s \cdot D_s(r) \qquad \forall\, s\in S\ . \qquad (5.2.1)$$

Let Ω be defined by

$$\Omega = \{r \mid r^s \geq 0,\ D_s(r) < \infty\ \forall\ s\in S\}$$

and denote the mapping defined by (5.2.1) by $h:R^{|S|} \to R^{|S|}$. It turns out [35], [12] that under various assumptions including the assumption that

$$D_s(r) > \beta > 0 \qquad \forall\, r\in\Omega,\ \forall\, s\in S \qquad (5.2.2a)$$

for some $\beta > 0$,

$$\frac{\partial D_s(r)}{\partial r^s} \geq \frac{\partial D_{s'}(r)}{\partial r^{s'}} \qquad \forall s, s'\in S\ , \qquad (5.2.2b)$$

the Jacobian of h, denoted by P(r) satisfies

$$P(r) = T(r) + R(r)B'M(r)B \qquad (5.2.3)$$

where T(r) denotes a diagonal matrix whose (m,m) entry is $D_m(r)$, R(r) denotes a diagonal matrix whose (m,m) entry is r^m, M(r) is a diagonal matrix and B is a matrix which is function of the routing only. It is shown there [35] that P(r) is nonsingular on Ω and that the mapping $h:\Omega\to h(\Omega)$ is

one to one and therefore $h^{-1}: h(\Omega) \to \Omega$ exists and by the Inverse Function Theorem [39], is continuously differentiable. We add here the following proposition which will proved in Appendix (5.A).

Proposition (5.A): Given that $h: \Omega \to h(\Omega)$ is such that $P(r)$ is non-singular on Ω, $D_s(o) < \infty$ and $D_s(\cdot)$ is convex continuous and differentiable on Ω for all $s \in S$, then under assumption (5.2.2) $h(\Omega) = (R^{|S|})^+$, where

$$(R^{|S|})^+ = \{r \mid r^s > 0 \quad \forall\, s \in S\}.$$

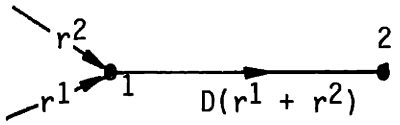
With Propositon (5.A) and the existence of h^{-1} which is continuously differentiable, we can try to minimize a function $J^*(h^{-1}(v))$ on the nonnegative orthant.

Since h, as mentioned above, is not available, one might hope that

$$\Delta v^s = -\eta \frac{\partial J^*}{\partial r^s} \qquad \forall\, s \in S$$

is a descent direction. This is false in some cases as the following example shows. (cf. [38] page 67).

Consider a network of one link (1,2) and two sessions originating at node 1 and terminating at node 2, as in Figure (A.1)



Assume $D_1(r^1+r^2) = D_2(r^1+r^2) = D(r^1+r^2)$ where D is of the Kleinrock type,

$$D(r^1+r^2) = \frac{1}{C-r^1-r^2}$$

and C is the capacity of link (1,2). If r^1 is very close to the capacity and r^2 is zero then increasing both windows v^1 and v^2 will virtually not increase the total flow r^1+r^2. It will just change the components of the

flow to satisfy

$$\frac{r^1}{r^2} = \frac{v^1}{v^2} \quad .$$

Thus r^2 will increase at the expense of an almost identical decrease in r^1. As a result if

$$\frac{\partial J^*}{\partial r^1} < \frac{\partial J^*}{\partial r^2} < 0$$

we get that

$$-\Delta r_1 \tilde{=} \Delta r_2 = \Delta r > 0$$

implying

$$\Delta J^* = \Delta r\left(\frac{\partial J^*}{\partial r^2} - \frac{\partial J^*}{\partial r^1}\right) > 0 \quad .$$

Gallager [12] showed that a diagonal scaling of $\nabla_r J^*$ by $R(r)$ results in a descent direction in the window space. To see this let

$$\Delta v = -\eta R(r) \nabla_r J^* . \tag{5.2.4}$$

It is known that $\nabla_r J^*$ and $\nabla_v J^*(h^{-1}(\cdot))$ can be connected through

$$\nabla_r J^*(r) = P'(h^{-1}(v)) \cdot \nabla_v J^*(h^{-1}(v))|_{v = h(r)} .$$

Using (5.2.3) and the equations above we get

$$\nabla_v J^*(h^{-1}(v))' \cdot \Delta v = -\eta \nabla_v J^*(h^{-1}(v))' R(r) \nabla_r J^*(r) = \tag{5.2.5}$$

$$= -\eta \nabla_v J^*(h^{-1}(v))' \cdot R(r) P[h^{-1}(v)]' \nabla_v J^*(h^{-1}(v)) =$$

$$= -\eta \nabla_v J^*(h^{-1}(v))' [R(r)T(r) + R(r)B'M(r)BR(r)] \nabla_v J^*(h^{-1}(v))$$

$$\geq 0$$

Similarly, when $\nabla_v J^*(h^{-1}(v))$ is available taking

$$\Delta r = -\eta R(r)\nabla_v J^*(h^{-1}(v))$$

results in

$$\nabla_r J^{*\prime}\cdot\Delta r = -\eta\nabla_r J^{*\prime}\cdot R(r)\cdot\nabla_v J^*(h^{-1}(v)) \qquad (5.2.6)$$

$$= -\eta(P(r)'\nabla_v(J^*\cdot h^{-1}))'R(r)\cdot\nabla_v(J^*\cdot h^{-1})$$

$$= -\eta\nabla_v(J^*\cdot h^{-1})'P(r)R(r)\cdot\nabla_v(J^*\cdot h^{-1})$$

$$= -\eta\nabla_v(J^*\cdot h^{-1})'[T(r)R(r) + R(r)B'M(r)BR(r)]\nabla_v(J^*\cdot h^{-1})$$

$$\geq 0.$$

The last inequalities in (5.2.5) and (5.2.6) hold because $P(r)R(r)$ is nonnegative definite on $\Omega\cap(R^{|S|})^+$. Moreover, the inequalities are strict on $\Omega\cap[\text{int}(R^{|S|})^+]$, since there $P(r)R(r)$ is positive definite.

Let $\bar{r}>0$ be a desirable vector of average input rates. Take J^* to be

$$J^*(r) = \|r - \bar{r}\|^2.$$

If r_0 is such that the ball of radius $\|\bar{r}-r_0\|$ does not intersect the axes, then within this ball the eigenvalues of $P(r)R(r)$ are bounded away from zero, and since we have a descent direction, then by taking fixed small enough η or by using an Armigo stepsize rule (cf. Chapter 4) in iteration (5.2.4) we are guaranteed to converge to a $\bar{v}$ such that $\bar{r}=h^{-1}(\bar{v})$.

If an entry of $\bar{r}$ equals zero, we take the corresponding $\bar{v}$ to be zero. It is not difficult to realize that we are now back at the previous case but in a space of smaller dimension. Similar arguments hold to achieve $\bar{r}$ such that $\bar{v}=h(r)$.

In practice to achieve a desired $\bar{r}$ we use an approximation to $D_s(r)$ to

find a point v_0 close to v, and then we iterate, using small η. Notice that such an iteration requires only the monitoring of the average rate of the input traffic.

When we want to have a window per session-path then to achieve a desired path flow, we treat each session-path as a separate session. If a session utilizes several paths and a packet arrives when some of the paths are ready to accept it, we put the packet on a particular path with probability which is proportional to its desired path flow.

5.3 Dynamic Routing by Session-Path Window

In this section we are going to look at flow control from the strict point of view of whether it can provide means of exercising dynamic routing. To this end we will assume a network with infinite buffers. The question we address is whether in such a network a window flow control mechanism has any possibility of reducing delay.

Traditionally, packet delay is measured as the time that elapses from the moment a packet is launched into the network to the moment that it arrives at the destination. With flow control, packets may have to wait at the host node until they can be launched into the network. Evidently this will not happen without flow control. To compare the two cases, we assume a given packet arrival rate to a host. The host has an infinite buffer where it stores the arriving packets until they can be transmitted. We will measure delay as the time that elapses from the moment a packet arrives at the host until it is delivered to the destination.

Consider the network in Figure (5.2),

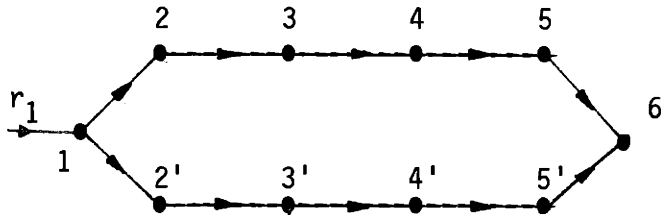


Fig. (5.2)

We model this network as a buffer feeding two networks of queues, each network consisting of four tandem queues. The queues capture the interfering traffic which is omitted in Fig (5.2). We assume all queues are identical.

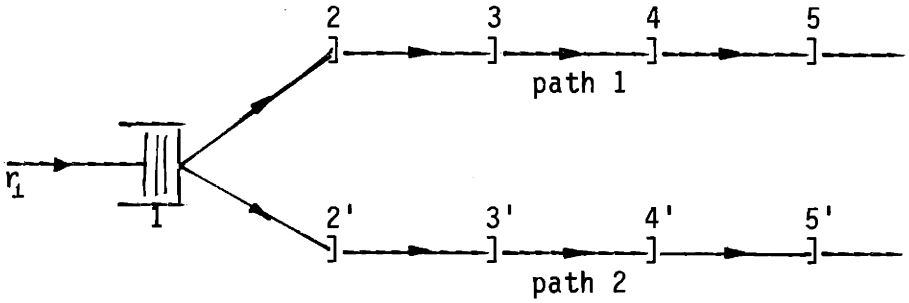
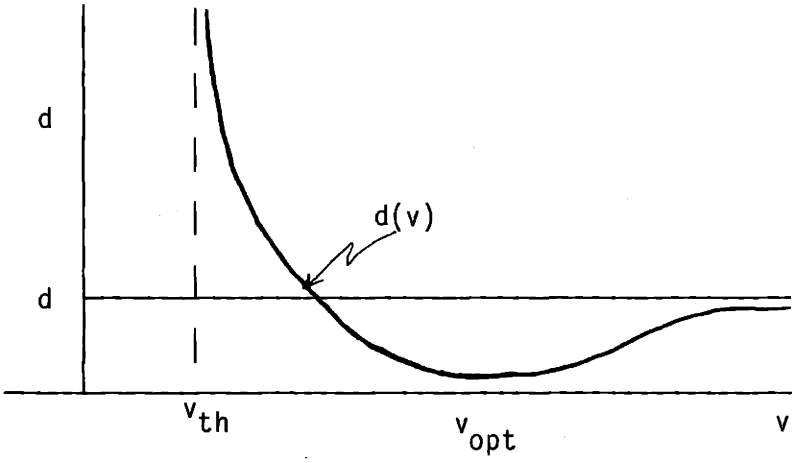


Fig. (5.3)

Modeling node 1, just as a buffer, amounts to the assumption that the transmission delay on links (1,2) and (1,2') is negligible when compared to each path delay.

With no flow-control, packets which arrive at buffer 1 are immediately sent to queue 2 or 2'. We assume this is done in a Round Robin fashion. We compare this with the flow controlled case, in which the total number of packet at queues 2 to 5, corresponding to path 1, as well as the number of packets at queues 2' to 5', corresponding to path 2, are restricted from exceeding some value $\hat{v}$. A path will be said to be blocked when the number of packets on it equals $\hat{v}$. We assume that when no path is blocked, packets are sent to the two paths in a Round Robin fashion. When the two paths are

blocked, packets are queued at buffer 1, and when only one path is blocked, packets are sent to the path which is not blocked. Let d be the average packet delay encountered in the uncontrolled case. We conjecture that the behaviour of the average delay per packet d as v changes, is as shown in Figure (5.4).



Fig(5.4)

This behaviour can be explained as follows. For small v there is a high probability that when the two paths are blocked, queues 2 and 2' are empty. Thus by not sending a packet, waiting at buffer 1, we do not perform work on a "customer" when otherwise we could, and the system might be unstable. Once we take v above some threshold v_{th} the system becomes stable. Then, when only one path is blocked we gain the information that the path is "congested" and flow is diverted to the other path until the "congestion" subsides. The ratio of the probability that exactly one path is blocked to the probability that the two paths are blocked simultaneously and queues 2 and 2' are empty, grows to infinity as v increases and therefore the relative frequency of the "good effect" by far exceeds that of the "bad effect". This justifies the reduced delay compared to the uncontrolled case.

This model also gives an indication about how to distribute the windows when exercising session-link windows. Certainly a path with all packets concentrated at the first queue is more congested than a path with the same number of packets all concentrated at the last queu. Intuitively, since we try to get information so as to equalize the average amount of work remaining in each queue, we should have the windows at queues 2 and 2' much smaller than those at queues 5 and 5'. In the context of many OD pairs this will have the effect of giving "priority" to flow which has already undergone large delay compared to a flow which has just entered the network.

To summarize, we have shown in this section that the claim in [36] may be plausible. Analyzing this claim in more detail is a topic for further investigation.

6. Integrated Network

6.1 Introduction

We consider three types of networks. The first handles data only, the second handles voice only, and the third handles voice and data together. Correspondingly we have three sections. In the first we propose the implementation of the data routing and the data flow control algorithms of the previous chapters so as to achieve a reasonable objective. In the second we do the same for voice, and in the third we propose a way of operating the four algorithms in a single network.

For a data only network, the joint routing and flow-control algorithm we propose, follows the suggestion made by Golestaani in [35]. The only difference is our use of the particular routing algorithm introduced in Chapter 4, and the adjustment of windows by the iterative algorithm, suggested in Chapter 5.

For a voice only network, the joint routing and flow-control algorithm is designed to result in a fair allocation in the sense defined in Chapter 3, of rates to sessions. Here like in Chapter 2, we assume each session is routed on a virtual circuit which lasts for the whole duration of the session, and the routing decision is the allocation of a path to an incoming session requesting one. We use an objective function whose minimization, similar to a penalty function, results in a fair allocation in an asymptotic sense.

Finally, for a network which handles voice and data simultaneously, we propose the concurrent use of the previous routing and flow control algorithms. To get them to interact in a meaningful way we let the voice take priority over the data within the queues and we let the data make use of

the capacity left over by the voice. The rates allocated to voice sessions are determined according to artificially calculated link capacities. These artifical link capacities are calculated so as to strike a balance between the maximal rate assigned to a voice session on a particular link and the delay the data experiences on the same link. We conclude with a discussion of various features of this algorithm.

6.2 Combined Data Routing and Flow-Control

In [35], Golestaani proposed adjusting routing and inputs so as to strike a balance between a penalty for reducing inputs below the values demanded by users, and a penalty for increasing the network congestion. Specifically, letting r_w^* represent the value of the average input demanded between origin and destination (OD) pair $w \epsilon W$, Golestaani suggested replacing problem (4.7.5) by the problem

$$\text{Min} \quad D(Ex) + \sum_{w \epsilon W} e_w(r_w) \tag{6.2.1}$$

subject to

$$\sum_{p \epsilon P_w} x^p = r_w \qquad \forall\, w \epsilon W, \tag{6.2.2}$$

$$x^p \geq 0 \qquad \forall\, p \epsilon P_w,\ w \epsilon W, \tag{6.2.3}$$

$$0 \leq r_w \leq r_w^* \qquad \forall\, w \epsilon W, \tag{6.2.4}$$

where all notations carry over from problem (4.7.5), and $e_w(\cdot)$ is a nonnegative nonincreasing convex function on the interval $[0, r_w^*]$ for all $w \epsilon W$.

He then proposed to rewrite (6.2.2) - (6.2.4) as

$$\sum_{p \epsilon P_w} \tilde{x}^p + y_w = r_w^* \qquad \forall\, w \epsilon W \tag{6.2.5}$$

$$\tilde{x}^p \geq 0 \quad y_w \geq 0 \qquad \forall\, p \in P_w,\ w \in W \qquad (6.2.6)$$

and correspondingly replace $e_w(r_w)$ by $g_w(y_w)$ for all $w \in W$ where

$$g_w(y_w) = e_w(\overset{*}{r}_w - y_w). \qquad (6.2.7)$$

It can then be easily seen that the problem

$$\text{Min} \quad D(E\tilde{x}) + \sum_{w \in W} g_w(y_w) \qquad (6.2.8)$$

subject to (6.2.5) - (6.2.6) is equivalent to problem (6.2.1) - (6.2.4) through the transformation

$$x^p = \tilde{x}^p \qquad \forall\, p \in P_w,\ w \in W,$$

$$y_w = \overset{*}{r}_w - r_w \qquad \forall\, w \in W.$$

Problem (6.2.8) can now be viewed as a routing problem with separable objective function to which the algorithm of Chapter 4 applies. Morever in problem (6.2.8) we have the same OD pair set as is problem (4.7.5); therefore it is easily seen that the process of solving (6.2.8) can be simulated on the network which corresponds to problem (4.7.5). This simulation process, when done using the algorithm of Chapter 4 can be seen to preserve all the decomposition and distributed computation properties mentioned there. Thus the algorithm will result in a sequence of inputs and path flows converging superlinearly to the solution of problem (6.2.1).

The fact of the matter is that our algorithm is not intended to solve problem (6.2.8) in the abstract but rather to serve as a quasi-static algorithm in a network in which we deal with flows and inputs consisting of actual packets sent by users. In this context it is the flow-control we exert on inputs that will determine the average input rates. Thus we have

to adjust our flow-control parameters from iteration to iteration so as to achieve the average input rates dictated when solving (6.2.8).

When exerting the flow-control on the inputs by windows, either for each session or for each session-path pair, we can adjust the windows to achieve a desired input rates using the algorithm suggested in Chapter 5. Thus when a window per session is employed, we first adjust the routing fractions of the outgoing paths and then iterate on the session windows to achieve the desired average rate; when a window per each session-path pair is employed, we determine the new desired path flows and then iterate on the session path windows to achieve these path flows.

In order that the window iteration will not follow the congestion sample path and by that exacerbate a dangerous situation, the time between two window iterations should not be too short. On the other hand, this time should not be too long in order that we execute more window iterations in between two routing updates. A reasonable time in between two window iterations is that which is comparable to the time it takes a temporary congestion to subside. Correspondingly, a reasonable time in between two routing updates is one which allows for a number of window iterations which results in inputs which are acceptable approximations to the desired ones. Notice that the time between routing updates should not be too short anyway, since in the quasi-static approach we should measure flow on a time scale on which the accumulation of flows at intermediate nodes can be neglected.

6.3 Combined Voice Routing and Flow-Control

In chapter 3 we have introduced a voice flow-control algorithm which allocates rates to sessions so as to yield a fair allocation, in a sense

explained there, over the set defined by (3.3.6) - (3.3.8). When, in addition, we have the freedom to determine routing also, some of the parameters in (3.3.6) - (3.3.8) are not fixed anymore, but are to be determined by the routing. Thus the set defined now by (3.3.6) - (3.3.8) is larger than before. A natural objective for a joint routing and flow control algorithm will be to find a fair allocation of rates over this larger set.

Specifically, retaining the notations used in Chapter 3, every session $s\varepsilon S$ is associated with an origin-destination (OD) pair $w\varepsilon W$. Abusing notation, we let w denote also the set of all $s\varepsilon S$ which are asssociated with $w\varepsilon W$; then the objective of the joint voice flow control and routing is to find a fair allocation $(\ldots,f^{-1}(\overline{\gamma}_s),\ldots)'$ over the set of vectors $(\ldots,f^{-1}(\gamma_s),\ldots)'$ such that

$$z_{p_s} \cdot f_s^{-1}(\gamma^s) \leq g_a\,(c_a - F^a)\cdot 1_{p_s}(a)\cdot z_{p_s} \qquad \forall p_s \varepsilon P_w:\ s\varepsilon w \qquad (6.3.1)$$

$$\forall\ s\varepsilon S,\ \forall\ a\varepsilon \mathcal{L}$$

$$\sum_{P_s\varepsilon P_w:\ s\varepsilon w} z_{p_s} = 1 \quad z_{p_s} \varepsilon\{0,1\} \qquad \forall p_s \varepsilon P_w:\ s\varepsilon w,\ \forall s\varepsilon S,\ \forall\ w\varepsilon W \qquad (6.3.2)$$

$$F^a \leq c_a,\ \gamma^s \geq 0 \quad F^a = \sum_{s\varepsilon S} \gamma^s \cdot 1_{p_s}(a)\cdot z_{p_s} \qquad \forall p_s \varepsilon P_w:\ s\varepsilon w,\ \forall s\varepsilon S,\ \forall w\varepsilon W \qquad (6.3.3)$$

From relation (6.3.1) it is not difficult to see that there exists $\overline{m}$ such that for all $m \geq \overline{m}$ minimizing

$$\sum_{a\varepsilon \mathcal{L}} [g_a(c_a - F^a)]^{-m} \qquad (6.3.4)$$

subject to (6.3.2), (6.3.3) and to

$$z_{p_s}\cdot 1_{p_s}(a)\cdot f_s^{-1}(\gamma^s) = \min_{a:1_{p_s}(a)=1} g_a(c_a - F^a)\cdot 1_{p_s}(a) z_{p_s} \qquad \forall\ s\varepsilon S,\ a\varepsilon \mathcal{L}, \qquad (6.3.5)$$

will solve the new fair allocation problem. A similar observation was made by Gallager in a different context [26].

We have already seen in Chapter 2 that by routing new incoming sessions on the minimal marginal delay path, we obtain under various assumptions detailed there, almost the same effect as if we had the possibility of rerouting virtual circuits. Thus our rule for routing will be to measure $c_a - F^a$ periodically, update the shortest marginal delay path with respect to the delay function (6.3.4) using some fixed large m, and then, in between updates, direct all incoming new sessions to the recently chosen paths.

This routing algorithm will work provided that the $c_a - F^a$ we measure corresponds to a fair allocation subject to the current routes so that (6.3.5) holds. This will be approximately true if we employ the flow-control algorithm of Chapter 3 and we assume that it maintains the rates close to the fair allocation one at all times, as a good quasi-static algorithm should do.

6.4 Combined Voice and Data

We consider a network serving both voice and data together. Because of the hard constraints on voice delay, we asssume that the voice packets are given priority in the queues over the data packets. Thus as far as the voice is concerned, the data does not exist. This may cause the data to be squeezed out of the network by the voice. In an extreme case the voice might obtain an encoding rate above ear's resolution at the expense of the data. To prevent this from happening we impose an additional constraint to (6.3.1) - (6.3.3) and assign the voice rates so as to achieve a fair allocation over the resulting smaller set.

Let $\tilde{D}_a(f^a, F^a)$ denote an approximation to the delay experienced by the data on link $a \in L$ when the average voice traffic on link $a \in \mathcal{L}$ is F^a and the average data traffic on that link is f^a. A reasonable compromise between data delay and voice rate can be achieved by imposing the condition

$$e(f_s^{-1}(\gamma^s)) \cdot 1_{p_s}(a) \geq \tilde{D}_a(f^a, F^a) \cdot 1_{p_s}(a) \qquad \forall\, a \in \mathcal{L},\ \forall\, s \in S \tag{6.4.1}$$

where $e(\cdot)$ is a nonnegative monotonically decreasing function, approaching infinity as its argument reduces to zero. Since f_s^{-1} is monotonically non-decreasing, (6.4.1) bounds γ^s on link $a \in L$ as a function of the delay experienced by the data on that link. We now propose a joint flow-control and routing algorithm for voice and data which will satisfy (6.4.1). For voice, the objective of the joint routing and flow-control algorithm will be to achieve a fair allocation of rates over the set defined by (6.3.1) - (6.3.3) together with (6.4.1). For data, the objective will be to minimize (6.2.8) where $\tilde{D}_a(f^a, F^a)$ replaces $D_a(f^a)$ in (4.7.5).

To achieve these two interacting objectives, we propose to employ the joint routing and flow-control algorithms of the previous data and voice sections, for data and voice, respectively. The voice rate will be influenced by the data delay by viewing c_a in condition (6.3.1), (6.3.3) as a parameter $\tilde{c}_a$. From (6.3.1) and (6.4.1), at equilibrium we have

$$\tilde{D}_a(f^a, F^a) \leq e(g_a[c_a - F_a]) \qquad \forall\, a \in \mathcal{L}. \tag{6.4.2}$$

Using Assumption (3.A) on g_a we observe that the composition of g_a with e can, without loss of generality, be denoted by e_a again since it satisfies the properties attributed to e, above. We propose substituting different $\tilde{c}_a$ from iteration to iteration according to the algorithm

$$(\tilde{c}_a)_{k+1} = \max\{0,\min\{c_a,(\tilde{c}_a)_k + \tilde{\alpha}_k^a[e_a((\tilde{c}_a)_k - F_k^a) - D_a(f_k^a,F_k^a)]\}\} \qquad (6.4.3)$$

$$\forall\ a\varepsilon\mathcal{L},\ k=0,1,\ldots,$$

for some $0 < \tilde{\alpha}_k^a < 1$. The determination of $\tilde{\alpha}_k^a$, as well as the exact model and assumptions under which the resulting combined voice and data algorithm will converge, is a subject for further research.

On the intuitive level, the working of the algorithm should be apparent. Assume we are currently at equilibirum. If the penalty $e_w(r_w)$ for an OD data pair $w\varepsilon W$ is increased by a small increment, the flow on the paths utilized by w will increase accordingly. By the data routing algorithm this increase will be distributed among the paths as to equalize the marginal data path delays. By iteration (6.4.3) the increase in the data delay will cause a reduction of $\tilde{c}_a$. This by the fair allocation algorithm for voice, will result in a decrease in F^a. Morever the total change in $\tilde{c}_a - F^a$ will be negative by Assumptions (3.A), (3.B) and the fair allocation property. Thus we obtain a new equilibirum in which the data delay was increased by an increment and the voice rate was decreased by an increment. Moreover, when the number of voice sessions on a particular link increases, it causes an increase in F^a and a new equilibirum is obtained at a higher F^a, $\tilde{c}_a$ and higher data delay. The higher data delay on the other hand causes a data input reduction.

It can be observed that the algorithm maintains the distributed nature similar to the the other algorithms in this Thesis since iteration (6.4.3) can be implemented by each link using only variables local to the link. Moreover the algorithm, by dynamically allocating the capacity, trades off between voice and data where this tradeoff takes into consideration the

priority functions of the various data and voice sessions. The reader can easily convince himself of this by examining the process described in the previous paragraph using different priority functions.

References

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

[2] E.S. Levitin and B.T. Poljak, "Constrained Minimization Problems", U.S.S.R. Comp. Math. Phys., Vol. 6, 1966, pp. 1-50.

[3] U.M. Garcia-Palomares, "Superlinearly Convergent Algorithims for Linearly Constrained Optimization", in Nonlinear Programming 2, O.L. Mangasarian, R.R. Meyer and S.M. Robinson (eds.), Academic Press, NY, 1975, pp. 101-121.

[4] P.E. Gill, W. Murray, and M.H. Wright, Practical Optimization, Academic Press, NY, 1981.

[5] J.C. Dunn, "Global and Asymptotic Rate of Convergence Estimates for a Class of Projected Gradient Processes", SIAM J. on Control and Optimization, Vol. 18, 1981, pp. 659-674.

[6] D.P. Bertsekas, "Projected Newton Methods for Optimization Problems with Simple Constraints", SIAM J. on Control and Optimization, Vol. 20, 1982, pp. 221-246.

[7] D.P. Bertsekas, "On the Goldstein-Levitin-Poljak Gradient Projection Method", Proc. 1974 IEEE Conf. on Decision and Control, Pheonix, AZ, pp. 47-52. Also IEEE Trans. Automat. Control, Vol. 20, 1976, pp. 174-184.

[8] J.-J. Moreau, "Convexity and Duality", in Functional Analysis and Optimization, E.R. Caianiello (ed.), Academic Press, NY, 1966.

[9] E. Polak, "Computational Methods in Optimization: A Unified Approach", Academic Press, 1971.

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

[11] D.P. Bertsekas, E.M. Gafni and R.G. Gallager, "Second Derivative Algorithms for Minimum Delay Distributed Routing in Networks", Report LIDS-R-1082, Laboratory for Information and Decision Systems, Mass. Inst. of Tech., Camb., MA, May 1979.

[12] Algorithms for Data Networks", Proceedings of Fifth International Conference on Computer Communication (ICCC-80), Atlanta, GA, Nov. 1980, pp. 779-784.

[13] D.P. Bertsekas and E.M. Gafni, "Projection Methods for Variational Inequalities with Application to the Traffic Assignment Problem", in Math. Prog. Study, D.C. Sorenson and R.J.-B. Wets (Eds.), North-Holland Pub. Co., Amsterdam, 1982, pp. 139-159.

[14] S. Dafermos, "Traffic Equilibrium and Variational Inequalities", Transportation Science 14, 1980, pp. 42-54.

[15] H.Z. Aashtiani and T.L. Magnanti, "Equilibria on a Congested Transportation Network", SIAM J. of Algebraic and Discrete Math., Vol. 2, 1981, pp. 213-226.

[16] D.P. Bertsekas and E.M. Gafni, "Projected Newton Methods and Optimization of Multicommodity Flows", LIDS Rep. P-1140, M.I.T., Cambridge, MA, 1981.

[17] F.H. Moss, "The Application of Optimal Control Theory to Dynamic Routing in Data Communication Networks", Ph.D. Disssertation, Massachusetts Inst. of Tech., Camb., 1976.

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

[19] L. Kleinrock, Communication Nets: Stochastic Message Flow and Delay, New York: McGraw Hill, 1964.

[20] A.A. Assad, "Multicommodity Network Flows - A Survey", Networks, Vol. 8, 1978, pp. 37-91.

[21] M. Gerla and L. Kleinrock, "Flow Control: A Comparative Study", IEEE Trans. Commun., Vol. COM-28, Apr. 1980, pp. 553-574.

[22] L. Kleinrock and C.W. Tseng, "Flow Control Based on Limiting Permit Generation Rates", Proceedings of Fifth International Conference on Computer Communications (ICCC-80), Atlanta, GA, Nov. 1980, pp. 785-790.

[23] A. O'Leary, "Distributed Routing", Report LIDS-TH-1064, Laboratory for Information and Decision Systems, Mass. Inst. of Tech., Camb., MA, Jan. 1981.

[24] H.Z. Aashtiani, "The Multi-Modal Traffic Assignment Problem", Ph.D. Thesis, Sloan School of Management, Mass. Inst. of Tech., Camb., MA., May 1979.

[25] D.P. Bertsekas, "A Class of Optimal Routing Algorithms for Communication Networks", Proc. Fifth Int. Conf. Comp. Comm., Atlanta, GA, Oct. 1980, pp. 71-75.

[26] F. Ros-Peran, "Routing to Minimize the Maximum Congestion in a Communication Network", Ph.D. Dissertation, Department of Elect. Eng. and Comp. Science, Mass. Inst. of Tech., Camb., MA, 1979 (Also LIDS-TH-885)

[27] J. Defenderfer, "Comparative Analysis of Routing Algorithms for Computer Networks", Ph.D. Dissertation, Department of Elect. Eng. and Comp. Science, Mass. Inst. of Tech., Camb., MA, 1977. (Also LIDS-TH-756)

[28] S. Even, A. Itai, and A. Shamir, "On the Complexity of Timetable and Multicommodity Flow Problems", SIAM J. Comp., Vol. 5, No. 5, Dec 1976, pp. 691-703.

[29] K.S. Vastola, "A Numerical Study of Two Measures of Delay for Network Routing", M.S. Thesis, Dept. of Elect. Eng., Univ. of IL, Urbana, IL, Sept. 1979.

[30] A. Segall, "Optimal Distributed Routing for Virtual Line-Switched Data Networks", IEEE Trans. on Comm., Vol. COM-27, No. 1, Jan 1979, pp. 201-209

[31] T. Beally, B. Gold, and S. Seneff, "A Technique for Adaptive Voice Flow Control in Integrated Packet Networks", IEEE Trans. Comm., Vol. COM-28, March 1980.

[32] J.M. Jaffee, "A Decentralized 'Optimal', Multiple-User, Flow and Control Algorithm", Proc. of Fifth Int. Conf. on Comp. Comm. (ICCC-80), Atlanta, GA, Nov. 1980, pp. 839-844.

[33] H. Hayden, "Voice Flow Control in Integrated Packet Network", M.S. Thesis, Dept. of Elect. Eng. and Comp. Science, Mass. Inst. of Tech., Camb., MA, 1981.

[34] M. Gerla and Per-Olov Nilsson, "Routing and Flow Control Interaction in Computer Networks", Department of Computer Science, Univ. of CA at Los Angeles, CA, 1980.

[35] S.J. Golestaani, "A Unified Theory of Flow Control and Routing in Data Communication Networks", Ph.D. Dissertation, Dept. of Elect. Eng. and Comp. Science, Mass. Inst. of Tech., Camb. MA, Jan. 1980.

[36] R. G. Gallage, D.P. Bertsekas and P.A. Humblet, "Data Network Reliability", Jan. 1981, Proposal to ARPA.

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

[38] O.C. Ibe, "Flow Control and Routing in an Integrated Voice and Data Communication Network", Ph.D. Dissertation, Dept. of Elec. Eng. and Comp. Science, Mass. Inst. of Tech., Camb. MA, 1981.

[39] W. Rudin, Principles of Mathematical Analysis, McGraw Hill, Inc., New York, 1976.

[40] J.D.C. Little, "A Proof for the Qucueing Formula: L= W", Operation Research, Vol. 9, 1961, pp. 386-387.

[41] G. Ciovello and P. Vena, "Integration of Circuit/Packet Switching in SENET (Slotted Envelope Network) Concept", NTC Conference Records, Dec. 1975, pp. 42-12 to 42-17.

[42] A.V. Aho, J.E. Hopcroft and J.D. Ullman, The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading, MA, 1974.

[43] M. Avriel, Nonlinear Programming: Analysis and Methods, Prentice Hall, Inc., NJ, 1976.

[44] M.R. Hestenes, Conjugate Direction Methods in Optimization, Springer-Verlay, NY, 1980.

[45] P. Baran, "On Distributed Communication Networks", IEEE Trans. Communication Systems, March 1964, pp. 1-9.

[46] J.G. Foschini and J. Salz, "A Basic Dynamic Routing Problem and Diffusion", IEEE Trans. Communication, Vol. COM-26, No. 3, March 1978, pp. 320-327.

[47] G. Karog, L. Fransen and E. Kline, "Multirate Processor (MRP)", Naval Research Laboratoy Report, Sept. 1980.

Appendix (2.A)

Proof of Lemma (2.A):

For all $w\epsilon W$ let $\overline{p}\epsilon P_w$ denote the path such that $\overline{u^p}(k\Delta t)=1$, then if $\tilde{x}$ is such that for all $w\epsilon W$, $\tilde{x}^p$ $p\epsilon P_w$ satisfies

$$\tilde{x}^p = \begin{cases} \lambda_w \mu^{-1} & p=\overline{p} \\ 0 & p\epsilon P_w,\ p\neq\overline{p} \end{cases},$$

it solves the problem in the lemma since for all $w\epsilon W$ there exist constants q^w such that for all $p\epsilon P_w$

$$\partial \frac{D(\overline{E}x(k\Delta t))}{\partial x^p} = \partial \frac{V(x(k\Delta t))}{\partial x^p} + q^w ,$$

i.e. $\overline{p}\epsilon P_w$ satisfies also

$$\overline{p} = \arg\min_{p\epsilon P_w} \frac{\partial V(x(k\Delta t))}{\partial x^p} .$$

Now, it can be seen that every stochastic process in (2.2.3) corresponds to an exponential random variable, and thus by the memoryless property of the exponential distribution and the independence of these exponential random variables, we get for $p\epsilon P_w$, $p\neq\overline{p}$

$$E\left(x^p([k+1]\Delta t) \Big/ x(k\Delta t)\right) = x^p(k\Delta t)\cdot e^{-\mu\Delta t} \qquad (2.A.1)$$

To find the expression for $p=\overline{p}$ we first find the average of the number of arrivals at w between $k\Delta t$ and $(k+1)\Delta t$ which do not terminate before $(k+1)\Delta t$. Denote this number by Z_w, and the number of arrivals by $\overline{N}$, then

$$E(Z_w) = E_{\overline{N}}\left[E(Z_w \Big/ A_w(k+1)\Delta t - A_w(k\Delta t) = \overline{N})\right] = \qquad (2.A.2)$$

$$= E_{N}\left(N \cdot \frac{1}{\Delta t}\int_{0}^{\Delta t} e^{-\mu t}dt\right) = E_{N}\left(N \cdot \frac{1}{\mu\Delta t} \cdot (1 - e^{-\mu\Delta t})\right)$$

$$= \lambda_w \cdot \Delta t \frac{1}{\mu\Delta t}(1 - e^{-\mu\Delta t}) = \lambda_w \cdot \mu^{-1}(1 - e^{-\mu\Delta t})$$

where the third equality follows because, given the number of Poisson arrivals in an interval, then the arrival times are uniformally distributed in the interval. Thus for $p = \overline{p}$ we have

$$E\left[x^{\overline{p}}([k+1]\Delta t)\Big/x(k\Delta t)\right] = x^{\overline{p}}(k\Delta t)\cdot e^{-\mu\Delta t} + \lambda_w \mu^{-1}(1- e^{-\mu\Delta t}). \qquad (2.A.3)$$

From (2.A.1) - (2.A.3) we get for $p\neq\overline{p}$

$$E(x^p[(k+1)\Delta t] - x^p(k\Delta t)\Big/x(k\Delta t))$$

$$= (e^{-\mu\Delta t} - 1)x^p(k\Delta t) = (1 - e^{-\mu\Delta t})(\tilde{x}^p - x^p(k\Delta t))$$

while for $p=\overline{p}$

$$E\left[x^{\overline{p}}[(k+1)\Delta t] - x^{\overline{p}}(k\Delta t)\Big/x(k\Delta t)\right] =$$

$$(\lambda_w\mu^{-1} - x^{\overline{p}}(k\Delta t))(1 - e^{-\mu\Delta t}) = (1 - e^{-\mu\Delta t})(\tilde{x}^{\overline{p}} - x^{\overline{p}}(k\Delta t))$$

Q.E.D.

Proof of Lemma (2.B):

For $p\epsilon P_w$ $p\neq\overline{p}$, using the fact that the variance of a Bernoulli random variable with probability of success p, is pq we get

$$E\left[(x^p[(k+1)\Delta t] - x(k\Delta t))^2\Big/x(k\Delta t)\right] = \qquad (2.A.4)$$

$$E^2\left[x^p[(k+1)\Delta t] - x(k\Delta t)\Big/x(k\Delta t)\right] +$$

$$\sigma^2(x^p([k+1]\Delta t) - x(k\Delta t) \Big/ x(k\Delta t))$$

$$= \quad x^p(k\Delta t)(1 - e^{-\mu\Delta t})^2 + x^p(k\Delta t)e^{-\mu\Delta t}(1 - e^{-\mu\Delta t})$$

while for $p \varepsilon P_w$, $p=\overline{p}$ we get with Z_w and $\overline{N}$ as defined in the proof of Lemma (2.A),

$$E\left[(x^{\overline{p}}([k+1]\Delta t) - x^{\overline{p}}(k\Delta t))^2 \Big/ x^{\overline{p}}(k\Delta t)\right] \leq \qquad (2.A.5)$$

$$E(Z_w^2) + E\left[(x^{\overline{p}}([k+1]\Delta t - x^{\overline{p}}(k\Delta t))^2 \Big/ x(k\Delta t),\ Z_w=0\right]$$

For $E(Z_w^2)$ we have

$$E(Z_w^2) = E_{\overline{N}}\, E(Z^2/\overline{N}) = E_{\overline{N}}\left[(\overline{N}\frac{1}{\mu\Delta t}(1 - e^{-\mu\Delta t}))^2\right.$$

$$\left. + \overline{N}\frac{1}{\mu\Delta t}(1 - e^{-\mu\Delta t})\left(1 - \frac{1 - e^{-\mu\Delta t}}{\mu\Delta t}\right)\right] =$$

$$= \left[\lambda_w\Delta t + (\lambda_w\Delta t)^2\right]\left(\frac{1}{\mu\Delta t}(1 - e^{-\mu\Delta t})\right)^2 +$$

$$+ \lambda_w\Delta t\left(\frac{1 - e^{-\mu\Delta t}}{\mu\Delta t}\right)\left(1 - \frac{1 - e^{-\mu\Delta t}}{\mu\Delta t}\right).$$

For the second term in the right hand side of (2.A.5) we have the same expression as (2.A.4) and Lemma (2.B) follows.

Q.E.D.

Appendix (3.A)

Proof of Lemma (3.A)

We prove (3.4.5) inductively. Assume γ_k satifies (3.4.5). Then it can be easily verified from (3.4.3) and (3.4.4) that

$$0 < \alpha_k^a < 1. \tag{3.A.1}$$

Also by (3.4.2)

$$\gamma_{k+1}^s = \min_{a \text{ s.t. } 1_{p_s}(a)=1} [(1 - \alpha_k^a)\gamma_k^s + \alpha_k^a f_s g_a (c_a - \sum_{t\in S} \gamma_k^t 1_{p_t}(a))]. \tag{3.A.2}$$

Since by assumptions (3.A), the induction hypothesis and (3.A.1) the two terms in the R.H.S. of (3.A.2) are nonnegative we get

$$\gamma_{k+1}^s \geq 0.$$

Let $\sum_{t\in S} \gamma_k^t 1_{p_t}(a)$ be denoted by F_k^a and let $\sum_{t\in S} f_t g_a [c_a - (\cdot)] 1_{p_t}(a)$ be denoted by $G_a(\ )$. Then $G_a(\ )$ is convex decreasing and (3.4.3) and (3.4.4) can be written as

$$\alpha_a^k = \frac{1}{1 + \dfrac{(G_a(0) - G_a(F_k^a))}{F_k^a}} \quad \text{or} \quad \alpha_a^k = \frac{1}{(1 - G_a'(F_k^a))}, \tag{3.A.3}$$

respectively.

From (3.4.2) we have

$$F_{k+1}^a \leq F_k^a + \alpha_a^k [G_a(F_k^a) - F_k^a]. \tag{3.A.4}$$

We will now distinguish between two cases:

case a: $(G_a(F_k^a) < F_k^a)$

In this case it follows from (3.A.4) and (3.A.1) that

$$F^a_{k+1} \leq F^a_k$$

and therefore by the induction hypothesis

$$F^a_{k+1} < c_a$$

<u>case b</u>: $(G_a(F^a_k) \geq F^a_k)$

By convexity of $G_a(\cdot)$ we have for all $z<F^a_k$

$$G'_a(F^a_k) \geq \frac{G_a(z) - G_a(F^a_k)}{z - F^a_k}.$$

Taking z=0 we obtain

$$-G'_a(F^a_k) \leq \frac{G_a(0) - G_a(F^a_k)}{F^a_k}$$

and therefore by the nonnegativity of $G_a(F^a_k) - F^a_k$ we get using (3.A.4) and the fact that $G_a(\ )$ is decreasing

$$F^a_{k+1} \leq F^a_k + \frac{1}{1-G'_a(F_k)} (G_a(F^a_k) - F^a_k)$$

or by the nonnegativity of $1- G'_a(F^a_k)$

$$G_a(F^a_k) - F^a_k + (G'_a(F^a_k) - 1)(F^a_{k+1} - F^a_k) \geq 0. \qquad (3.A.5)$$

Thus (3.A.5) holds for α^a_k as defined by (3.4.3) and (3.4.4). Define

$$H_a(F^a_k) = G_a(F^a_k) - F^a_k.$$

Obviously $H(\cdot)$ is convex and therefore by the subdifferential relation and (3.A.5) we obtain

$$G_a(F^a_{k+1}) - F^a_{k+1} = H_a(F^a_{k+1}) \geq 0$$

and as a result

$$G_a(F^a_{k+1}) \geq F^a_{k+1}. \tag{3.A.6}$$

W.l.o.g. define $G_a(z) = 0$ for $z \geq c_a$ and get from (3.A.6)

$$F^a_{k+1} < c_a,$$

and (3.4.5) is proved.

Once we have proved (3.4.5) we can strengthen (3.A.1) by observing that now (3.4.3) and (3.4.4) imply

$$\liminf_{k\to\infty} \alpha^a_k > 0. \tag{3.A.7}$$

Notice that once F^a_k belongs to case b) for some $\bar{k}$, it will belong to this case for all $k>\bar{k}$; while if F^a_k belongs to case a) it is reduced toward the region of case b) as implied by (3.A.7) and (3.A.4) at least as fast as a geometric progression so that at most in the limit we are in case b) and thus

$$\limsup_{k\to\infty} F^a_k$$

satisfies case b) which proves the lemma.

Q.E.D.

Appendix (4.A)

Proof of Lemma (4.A): a) Fix $x\varepsilon X$, $z\varepsilon H$ and $\gamma>1$. Denote

$$a = x + z, \quad b = x + \gamma z \tag{4.A.1}$$

Let $\overline{a}$ and $\overline{b}$ be the projections on X of a and b respectively. It will suffice to show that

$$\|\overline{b} - x\| \leq \gamma\|\overline{a} - x\|. \tag{4.A.2}$$

If $\overline{a} = x$ then clearly $\overline{b} = x$ so (4.A.2) holds. Also if $a\varepsilon X$ then $\overline{a} = a = x + z$ so (4.A.2) becomes $\|\overline{b} - x\| \leq \gamma\|z\| = \|b - x\|$ which again holds since P is nonexpansive. Finally if $\overline{a} = \overline{b}$ then (4.A.2) also holds. Therefore it will suffice to show (4.A.2) in the case where $a \neq b$, $a \neq x$, $b \neq x$, $a\varepsilon X$, $b\varepsilon X$ shown in Figure (4.A.1).

Let H_a and H_b be the two hyperplanes that are orthogonal to $(\overline{b} - \overline{a})$ and pass through $\overline{a}$ and $\overline{b}$ respectively. Since $\langle\overline{b} - \overline{a}, b - \overline{b}\rangle \geq 0$ and $\langle\overline{b} - \overline{a}, a - \overline{a}\rangle \leq 0$ we have that neither a nor b lie strictly between the two hyperplanes H_a and H_b. Furthermore x lies on the same side of H_a as a, and $x\notin H_a$. Denote the intersections of the line $\{x + \alpha(b - x) | \alpha\varepsilon R\}$ with H_a and H_b by s_a and s_b respectively. Denote the intersection of the line $\{x + \alpha(\overline{a} - x) | \alpha\varepsilon R\}$ with H_b by w. We have

$$\gamma = \frac{\|b - x\|}{\|a - x\|} \geq \frac{\|s_b - x\|}{\|s_a - x\|} = \frac{\|w - x\|}{\|\overline{a} - x\|} = \frac{\|w - \overline{a}\| + \|\overline{a} - x\|}{\|\overline{a} - x\|}$$

$$\geq \frac{\|\overline{b} - \overline{a}\| + \|\overline{a} - x\|}{\|\overline{a} - x\|} \geq \frac{\|\overline{b} - x\|}{\|\overline{a} - x\|} \tag{4.A.3}$$

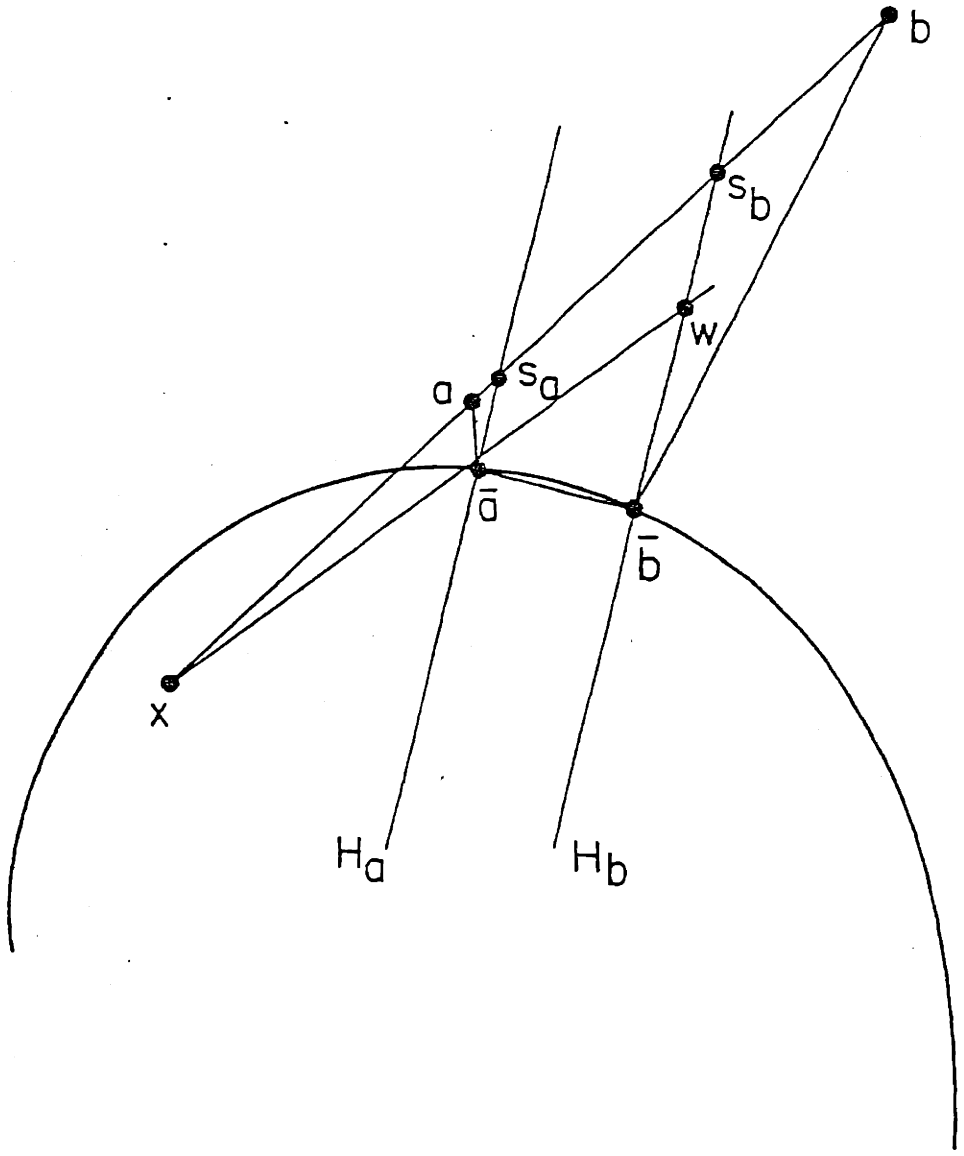


Figure 4.A.1

where the third equality is by similarity of triangles, the next to last inequality follows from the orthogonality relation $\langle w - \overline{b}, \overline{b} - \overline{a} \rangle = 0$, and the last inequality is obtained from the triangle inequality. From (4.A.3) we obtain (4.A.2) which was to be proved.

b) Since y is a direction of recession of Ω, we have

$$P_{\Omega}(x + z) + y \in \Omega \ . \tag{4.A.4}$$

Thus by definition of projection on a closed convex set

$$\langle (x + z) - P_{\Omega}(x + z), (P_{\Omega}(x + z) + y) - P_{\Omega}(x + z) \rangle \leq 0 \tag{4.A.5}$$

or equivalently

$$\langle (x + z) - P_{\Omega}(x + z), y \rangle \leq 0,$$

and (4.3.25) follows.

Q.E.D.

Appendix (4.B)

We develop the main arguments for the proof of Proposition (4.C) through a sequence of Lemmas. In what follows we use the word "eventually" to mean "there exists $\bar{k}$ such that for all $k \geq \bar{k}$", where $\bar{k}$ may be different for each case.

Lemma (4.B.1): Under the conditions of Proposition (4.C), $\lim_{k\to\infty} x_k = \bar{x}$ and eventually

$$I_k = A_{\bar{x}}. \tag{4.B.1}$$

Proof: By relation (4.4.14), since $\bar{x}$ is a limit point and the algorithm decreases the value of the objective function at each iteration, we have

$$\lim_{k\to\infty} \|x_{k+1} - x_k\| = 0,$$

which implies, again by the descent property and the fact that $\bar{x}$ is a strict local mimimum

$$\lim_{k\to\infty} x_k = \bar{x}. \tag{4.B.2}$$

Therefore from (4.5.5)

$$\lim_{k\to\infty} \varepsilon(x_k) = \varepsilon(\bar{x}) = 0. \tag{4.B.3}$$

Since the set I is finite it follows from (4.5.2), (4.5.6) and (4.B.3) that eventually

$$I_k \subset A_{\bar{x}}. \tag{4.B.4}$$

To show the reverse inclusion we must show that eventually

$$\langle a_i, x_k \rangle \geq b_i - \varepsilon(x_k)\|a_i\|, \qquad \forall\, i \in A_{\bar{x}}. \tag{4.B.5}$$

By the Cauchy-Schwartz inequality (B.3) and (4.5.5) we have eventually

$$\varepsilon(x_k)\|a_i\| = \|x_k - P[x_k - \nabla f(x_k)]\| \cdot \|a_i\| \geq \langle P[x_k - \nabla f(x_k)] - x_k, a_i \rangle.$$

Therefore in order to show (4.B.5) it suffices to show that eventually

$$\langle a_i, P[x_k - \nabla f(x_k)] \rangle = b_i, \qquad \forall\ i \in A_{\bar{x}}$$

or equivalently

$$P[x_k - \nabla f(x_k)] \in \bar{x} + N_{\bar{x}}.$$

Since $x_k \to \bar{x}$ this follows from Assumption (4.C).

Q.E.D.

Lemma (4.B.2): Under the conditions of Proposition (4.C) for each $\bar{\alpha} \in (0,1]$, eventually we have

$$x_k(\alpha) \in \bar{x} + N_{\bar{x}}, \qquad \forall\ \alpha \in [\bar{\alpha}, 1]. \tag{4.B.6}$$

Proof: From Lemma (4.B.1) we have $x_k \to \bar{x}$ and eventually $C_k = \bar{C}$ where

$$\bar{C} = \{z \mid \langle z, a_i \rangle \leq 0, \ \forall\ i \in A_{\bar{x}}\}. \tag{4.B.7}$$

Since the projection of $-\nabla f(\bar{x})$ on $\bar{C}$ is the zero vector and d_k is eventually the projection of $-\nabla f(x_k)$ on $\bar{C}$ it follows that

$$\lim_{k \to \infty} d_k = 0, \tag{4.B.8}$$

Since $\tilde{d}_k$ is the projection of $D_k d_k$ on a subset of C_k, and $\{\|D_k\|\}$ is bounded above [cf. (4.4.8), (4.4.9)], it follows that

$$\lim_{k \to \infty} \tilde{d}_k = 0. \tag{4.B.9}$$

Since $-\nabla f(x_k) = d_k^+ + d_k$ and $g_k = d_k^+ + \tilde{d}_k$ we obtain

$$\lim_{k\to\infty} g_k = \nabla f(\bar{x}). \tag{4.B.10}$$

A simple argument shows that Assumption (4.B) implies that for all $\alpha\in[0,1]$

$$P(y)\in\bar{X} + N_{\bar{x}} \qquad \forall\, y \text{ such that } \|\bar{x} - \alpha\nabla f(\bar{x}) - y\| \leq \alpha\delta \tag{4.B.11}$$

For any $\bar{\alpha}\in(0,1]$, equation (4.B.10) shows that we have eventually

$$\|\bar{x} - \alpha\nabla f(\bar{x}) - (x_k - \alpha g_k)\| \leq \alpha\delta, \qquad \forall\, \alpha\in[\bar{\alpha},1].$$

Therefore from (4.B.11) we have eventually

$$x_k(\alpha) = P(x_k - \alpha g_k)\in\bar{X} + N_{\bar{x}}, \qquad \forall\, \alpha\in[\bar{\alpha},1].$$

Q.E.D

Lemma (4.B.3): Under the conditions of Proposition (4.C)

$$\liminf_{k\to\infty} \alpha_k > 0.$$

Proof: From Lemma (4.B.1) we have $I_k = A_{\bar{x}}$ and $x_k \to \bar{x}$, while from (4.B.8) we have $\|g_k\| \to \|\nabla f(\bar{x})\|$. Therefore from Proposition (4.A) part b) [cf. (4.3.27)] it follows that there exists $\alpha > 0$ such that eventually

$$\langle\nabla f(x_k), x_k - x_k(\alpha)\rangle \geq \alpha\langle d_k, D_k d_k\rangle + \frac{1}{\alpha}\|x_k(\alpha) - (x_k + \alpha\tilde{d}_k)\|^2, \qquad \forall\, \alpha\in(0,\alpha].$$

Using this relation we get that eventually

$$f(x_k) - f[x_k(\alpha)] \geq$$

$$\geq \langle \nabla f(x_k), x - x(\alpha) \rangle - \frac{L}{2} \| x_k(\alpha) - x_k \|^2$$

$$\geq \alpha \langle d_k, D_k d_k \rangle + \frac{1}{\alpha} \| x_k(\alpha) - (x_k + \alpha \tilde{d}_k) \|^2 - \frac{L}{2} \| x_k(\alpha) - x_k \|^2$$

$$\geq \alpha \langle d_k, D_k d_k \rangle + \frac{1}{\alpha} \| x_k(\alpha) - (x_k + \alpha \tilde{d}_k) \|^2$$

$$-L \| \alpha \tilde{d}_k \|^2 - L \| x_k(\alpha) - (x_k + \alpha \tilde{d}_k) \|^2$$

$$\geq \alpha(1 - \alpha L \lambda_2) \langle d_k, D_k d_k \rangle +$$

$$+ \left(\frac{1}{\alpha} - L\right) \| x_k(\alpha) - (x_k + \alpha \tilde{d}_k) \|^2$$

where the second inequality follows from

$$\| x + y \|^2 \leq 2 \| x \|^2 + 2 \| y \|^2$$

the last inequality follows from (4.5.8) and L corresponds to an arbitrary nonempty bounded neighborhood of $\bar{x}$. Taking any $\alpha > 0$ satisfying

$$\bar{\alpha} \leq \hat{\alpha}, \quad 1 - \bar{\alpha} L \lambda_2 > \sigma, \; \bar{\alpha}\left(\frac{1}{\bar{\alpha}} - L\right) > \sigma$$

we obtain, using (4.5.14) that

$$\liminf_{k \to \infty} \alpha_k > \bar{\alpha}$$

and the lemma is proved. Q.E.D.

Proof of Proposition (4.C): The fact $\lim_{k \to \infty} \bar{x}_k = x$ is part of Lemma (4.B.1), while (4.5.7) follows from Lemmas (4.B.2) and (4.B.3).

In order to show (4.5.8) we note that from Lemma (4.B.1) and (4.B.8) we have eventually

$$C_k = \bar{C}, \quad C_k^+ = \bar{C}^+ \tag{4.B.12}$$

and

$$\lim_{k\to\infty} d_k^+ = -\nabla f(\bar{x}). \tag{4.B.13}$$

Equation (4.B.13) implies that eventually Assumption (4.B) holds with d_k^+ replacing $-\nabla f(\bar{x})$ and $\delta/2$ replacing δ. Therefore for all $i\epsilon A_{\bar{x}}$ and $\rho_i>0$ such that $\|\rho_i a_i\| < \frac{\delta}{2}$ we have

$$P(\bar{x} + d_k^+ \pm \rho_i a_i)\epsilon\bar{x} + N_{\bar{x}} \tag{4.B.14}$$

$$P(\bar{x} + d_k^+)\epsilon\bar{x} + N_{\bar{x}}. \tag{4.B.15}$$

For any $z_1, z_2 \epsilon H$ we have from a general property of projection on X

$$\langle z_1 - P(z_1), P(z_2) - P(z_1)\rangle \leq 0$$

$$\langle z_2 - P(z_2), P(z_1) - p(z_2)\rangle \leq 0.$$

By adding these two inequalities we obtain

$$\|P(z_1) - P(z_2)\|^2 \leq \langle z_1 - z_2, P(z_1) - P(z_2)\rangle, \quad \forall\, z_1, z_2 \epsilon H \tag{4.B.16}$$

By applying (4.B.16) we obtain

$$\|P(\bar{x} + d_k^+ \pm \rho_i a_i) - P(\bar{x} + d_k^+)\|^2 \tag{4.B.17}$$

$$\leq \langle \pm\rho_i a_i, P(\bar{x} + d_k^+ \pm \rho_i a_i) - P(\bar{x} + d_k^+)\rangle.$$

Since $\langle a_i, z\rangle = 0$ for all $z\epsilon N_{\bar{x}}$, $i\epsilon A_{\bar{x}}$ it follows from (4.B.14), (4.B.15)

that the right side of (4.B.17) is zero and therefore eventually

$$P(\bar{x} + d_k^+ \pm \rho_i a_i) = P(\bar{x} + d_k^+), \quad \forall\, i \in A_{\bar{x}}.$$

Since from (4.B.12) we have eventually $d_k^+ \in \bar{C}^+$, it follows that $P(\bar{x} + d_k^+) = \bar{x}$ and therefore also

$$P(\bar{x} + d_k^+ \pm \rho_i a_i) = \bar{x}, \quad \forall\, i \in A_{\bar{x}}.$$

Hence eventually

$$d_k^+ \pm \rho_i a_i \in \bar{C}^+, \quad \forall\, i \in A_{\bar{x}}$$

which implies that

$$\langle d_k^+ \pm \rho_i a_i, y \rangle \leq 0, \quad \forall\, y \in \bar{C},\ i \in A_{\bar{x}}. \tag{4.B.18}$$

Let

$$y \in \{z \mid z \in C_k,\ \langle z, d_k^+ \rangle = 0\}.$$

From (4.B.12) and (4.B.18) we have eventually

$$\langle a_i, y \rangle = 0 \qquad \forall\, i \in A_{\bar{x}},$$

or equivalently $y \in N_{\bar{x}}$. Hence eventually

$$N_{\bar{x}} \supset \{z \mid z \in C_k,\ \langle z, d_k^+ \rangle = 0\}$$

and it follows that

$$\text{span } N_{\bar{x}} = N_{\bar{x}} \supset \text{span}\{C_k \cap \{z \mid \langle z, d_k^+ \rangle = 0\}\} = \Gamma_k .$$

To show that the reverse inclusion note that if $y \in N_{\bar{x}}$ then by

Assumption (4.B) and (4.B.12) we have

$$\langle y, d_k^+ \rangle = 0.$$

Since $N_{\bar{x}} \subset \bar{C}$ and eventually $C_k = \bar{C}$ it follows that eventually $y \in C_k \cap \{z \mid \langle z, d_k^+ \rangle = 0\}$ and a fortiori $y \in \text{span}\{C_k \quad \{z \mid \langle z, d_k^+ \rangle = 0\}\} = \Gamma_k$ Therefore eventually

$$N_{\bar{x}} \subset \Gamma_k$$

and the proof of (4.5.8) is complete.

Since d_k is the projection of $-\nabla f(x_k)$ on $C_k \quad \{z \mid \langle z, d_k^+ \rangle = 0\}$ equation (4.5.9) follows easily from (4.5.8).

Also from (4.5.7) and (4.5.8) we have eventually $x_k \in \bar{x} + N_{\bar{x}}$, $\tilde{d}_k \in N_{\bar{x}}$, $d_k \in N_{\bar{x}}$ and d_k^+ is orthogonal to $N_{\bar{x}}$, while by Lemma (4.B.2) the vector x_{k+1} is the projection of $x_k + \alpha_k(\tilde{d}_k + d_k^+)$ on $\bar{x} + N_{\bar{x}}$. Therefore (4.5.10) and (4.5.11) follow.

Q.E.D

Appendix (5.A)

Proof of Proposition (5.A):

We prove the proposition inductively. It is easy to see that the Proposition is true for the one dimensional case. Assume the Proposition holds for the ($|S|$ -1)-dimensional case. Consider the $|S|$ dimensional one. By taking one entry equal zero and using assumption (5.2.2) to ignore the corresponding equation, it can be easily seen that we are back at an ($|S|$-1) -dimensional case with all assumptions holding and thus by the induction hypothesis we get that

$$h(\Omega \bigcap \partial(R^{|S|})^{+}) = \partial(R^{|S|})^{+} \qquad (5.A.1)$$

where ∂ denotes boundary and therefore $\partial(R^{|S|})^{+}$ is the boundary of the first orthant.

By (5.2.2) and (5.2.1) we get

$$h(r)\varepsilon(R^{|S|})^{+} \qquad \forall\ r\varepsilon\Omega \ . \qquad (5.A.2)$$

For all $0 < \delta_1 < \infty$ let $\Omega(\delta_1)$ be defined by

$$\Omega\ (\delta_1) = \{r \mid \| h(r) \| \leq \delta_1\} \qquad (5.A.3)$$

then by the continuity of h and $\| \ \|$ and by assumption (5.2.2) and since the Jacobian is nonsingular we get that

$$\text{int}\ \Omega\ (\delta_1) \neq \phi \qquad \forall\ \delta_1 > 0 \qquad (5.A.4)$$

$$\Omega\ (\delta_1) \subset \Omega\ (\delta_2) \subset \Omega \qquad \forall\ \delta_2 > \delta_1 > 0$$

and $\Omega\ (\delta_1)$ is a compact set for all $\delta_1 > 0$.

Let $\overline{B}(0,\delta_1)$ denote the closed ball of radius δ_1 around 0, then if

$$h(B\Omega(\delta_1)) \cap \overline{B}(0,\delta_1) = (R^{|S|})^{+} \cap \overline{B}(0,\delta_1) \tag{5.A.5}$$

for all $\delta_1 \geq 0$ we are done. Otherwise by (5.A.2) there exists $\delta_1 > 0$ such that

$$h(\Omega(\delta_1)) \cap \overline{B}(0,\delta_1) \subset (R^{|S|})^{+} \cap \overline{B}(0,\delta_1). \tag{5.A.6}$$

Let $\tilde{v}$ belong to the R.H.S. and not to the L.H.S. of (5.A.6). By the induction hypothesis

$$\tilde{v} \notin \partial(R^{|S|})^{+}$$

and by (5.A.3)

$$\tilde{v} \neq h(r) \qquad \forall\, r \in \Omega - \overline{\Omega}(\delta_1)$$

thus w.l.o.g. by increasing δ_1, we have that

$$\tilde{v} \in \text{int}\{(R^{|S|})^{+} \cap \overline{B}(0,\delta_1)\}. \tag{5.A.7}$$

Now, $\Omega(\delta_1)$ is compact, using the fact that a continuous function maps a compact set to a compact set we get that the set in the L.H.S of (5.A.6) is closed. Moreover from (5.A.4) we have that the intersection of the interior of the L.H.S of (5.A.6) and the R.H.S of (5.A.6) is non-empty. Therefore it follows from (5.A.7) that there exists a point v* such that

$$\overset{*}{v} \in \partial h(\Omega), \tag{5.A.8}$$

$$\overset{*}{v} \in \text{int}\,(R^{|S|})^{+} \tag{5.A.9}$$

and since the L.H.S. of (5.A.6) is closed there exists a point $r^{*} \in \Omega$ such that

$$v^* = h(r^*). \tag{5.A.10}$$

Moreover, (5.A.9),(5.A.10) combined with (5.2.1) imply that

$$r^* \in \text{int } \Omega \tag{5.A.11}$$

Using (5.A.11), (5.A.10), and the nonsingularity of the Jacobian $P(r^*)$ the Inverse Function Theorem [39] precludes (5.A.8).

Q.E.D.