PRIORITY ASSIGNMENT IN INTEGRATED SERVICES NETWORKS

by

JEAN MICHEL RÉGNIER

B.Sc.A. University of Montreal
(1981)

M.Sc.A. University of Montreal
(1982)

SUBMITTED TO THE DEPARTMENT OF

ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May, 1986

©Massachusetts Institute of Technology, 1986

Signature of Author_____
Department of Electrical Engineering and Computer Science
May 16, 1986

Certified by_____
Professor Pierre A. Humblet
Thesis Supervisor

Accepted by_____
Professor Arthur C. Smith
Chairman, Departmental Commitee on Graduate Students

Priority Assignment in Integrated Services Networks

by

Jean Michel Régnier

Submitted to the Department of
Electrical Engineering and Computer Science
on May 2, 1986
in Partial Fulfillment of the requirements
for the Degree of Doctor of Philosophy

Abstract

Our main results demonstrate that priority queuing is an effective means of controlling the delay of the users in an integrated services network, and that priority queuing can be coupled with a flow control mechanism to simultaneously control the rate and the delay of the users.

We begin by specializing the work of Coffman [Co80] on the characterization of the delays realizable in a single server queuing system to the case in which the service time of the users are drawn from a common exponential distribution. In this context we obtain a simple characterization of the set of realizable delays and we construct a simple time–invariant queuing strategy via which any realizable delay assignment can be realized. We then attach a delay cost function to each user and we investigate the problem of finding realizable delay assignments which minimize the costs. Two formulations are investigated. In the first formulation the objective is to minimize the sum of the user's costs. We give a set of necessary and sufficient optimality conditions for this problem and, based on these conditions, we construct a simple algorithm for solving the problem. In the second formulation the objective is to find a min–max fair delay assignment. We show that this problem is intimately related to the preceeding problem and propose a simple algorithm for solving it.

Equipped with the results obtained in the single server case we investigate how priority queuing can be used to control the delay of the users in an integrated services network. Specifically, quantifying the end–to–end delay preferences of the users on an individual basis through associated delay–cost functions, we consider the problem of selecting the queuing strategies on the links of the network so as to minimize the overall delay cost. We give a set of necessary and sufficient optimality conditions for this problem and we propose two distributed for algorithms solving it. The first algorithm is approximate in the sense that in general it does not converge to an optimal solution. However by selecting the parameters of the algorithm appropriately the solution produced can be brought as close to optimality as desired. Moreover the first algorithm terminates in finite time and works in a completely uncoordinated manner. The second algorithm always converge to an optimal solution but requires more coordination than the first algorithm.

Finally we propose a formulation of the flow control problem in which the interactions between the rates and the delays are fully considered. In the formulation the cost function of each user depends explicitly on the rate and the end–to–end delay of the user. The objective is to select the rate of the users and the queuing strategies on the links of the network so as to minimize the overall cost. We give a set of necessary and sufficient optimality conditions for this problem. Then, based on these conditions, we construct a distributed algorithm solving the problem.

Thesis Supervisor: Dr. Pierre A. Humblet

Title: Associate Professor of Electrical Engineering

à Denise

Acknowledgments.

I wish to express my sincere gratitude to my parents, Gabrielle Bélisle and Michel Régnier, for their unfailing support throughout the course of my studies, and to my wonderful wife, Denise, whose faith in me and understanding made this thesis possible.

I would like to thank Professor P. A. Humblet, my thesis supervisor, for all the assistance he has provided in the course of this research. His guidance, suggestions and support have been invaluable.

I would like to thank Professors R. G. Gallager and J. N. Tsitsiklis, my thesis readers, for numerous fruitful discussions and for their careful reading of the manuscript. I also thank Professor Gallager for having brought the problem to my attention in the first place.

I would like to thank Professor A. Drake, my advisor, for his counseling and encouragements.

I would like to thank my office–mates: Ellen Hahne, Patrick Hossein, Atul Khanna, Jay Kuo, Whay Lee, Utpal Mukherji, John Spinelli and Kevin Tsai, who have greatly contributed to the quality of my life at M.I.T. I have especially enjoyed many discussions with Jay, John, Kevin and Utpal.

I thank my friends from the Shotokan Karate Club for many very enjoyable hours.

I thank Dr. W. H. Cameron and Dr. R. Jackson from Bell–Northern Research for their friendship and encouragements.

I thank the National Sciences and Engineering Research Council of Canada and Bell–Northern Research for their financial support.

This work was supported in part by the National Sciences Foundation under Grant NSF–ECS–8310698. This funding was greatly appreciated.

Table of Contents.

List of Figures.

Notation.

General rules:

An arrow over a variable indicates that the variable is a vector. A subscript usually refers to a virtual circuit (v.c.) attribute while a superscript usually refers to a link attribute. A superscript enclosed in square brackets refers to an estimated value and a superscript enclosed in parenthesis denotes a variable indexed by an iteration number.

The variables listed here are only those appearing in the body of the text. Variables used only locally, for example inside a proof, are omitted.

Notation:

A_i: Threshold below which the delay of v.c. i is equal to the delay of v.c. $i-1$.

(A_o): Auxiliary problem in the system–oriented approach.

B: Busy period.

$B(i, \vec{w}, \vec{R})$: Shorthand notation for the right hand side of the i^{th} feasibility constraint.

$B^l(i, \vec{w}^l, \vec{R})$: Shorthand notation for the right hand side of the i^{th} feasibility constraint on link l.

$B^{l\prime}(i, \vec{w}^l, \vec{R})$: First derivative of the right hand side of the i^{th} feasibility constraint on link l.

c_i: Delay cost per unit rate of v.c. i in the linear cost case.

$C_i(D_i)$: Cost of assigning the end–to–end delay D_i to v.c. i.

$C_i^l(D_i^l)$: Cost of assigning the delay D_i^l to v.c. i on link l, assuming that the delay of v.c. i on the other links is fixed.

$[C_i]^{-1}(\cdot)$: Inverse of the function $C_i(\cdot)$.

$[C_i']^{-1}(\cdot)$: Inverse of the function $C_i'(\cdot)$.

(CPP): Convex programming problem.

$\vec{D}$: Vector containing all the delays; i.e, $\vec{D} = (D_1, \ldots, D_V)$ in the single link case and $\vec{D} = (D_i^l,\ l \in \mathcal{L}_i,\ i = 1, \ldots, V)$ in the multiple link case.

$\vec{D}^l$: Vector containing the delay on link l of the v.c.'s using link l.

$D(i)$: Destination of node i.

D_i: End–to–end delay of v.c. i.

$D_{i,n}$: Delay of the n^{th} packet generated by v.c. i.

d_i^l: Reference value of the delay ov v.c. i on link l.

$d_i^{[l]}$: Estimate of the end–to–end delay of v.c. i maintained by link l.

D_i^l: Delay of v.c. i on link l.

$D_i^l(k)$: Delay of v.c. i on link l at time k.

ds_i^l: Identity of the link downstream to l on i's path.

$ds_i^{[l]}$: Estimate maintained by link l of the overall delay of v.c. i on the links of i's path downstream to l.

Ds_i^l: Set of links on i's path downstream to l.

$du_i^{[l]}$: Estimate maintained by link l of the overall delay of v.c. i on the links of i's path upstream to l.

e_i: Priority group of v.c. i.

e_i^l: Priority group of v.c. i on link l.

$e-1$: Priority group immediately preceeding the priority group e.

E: Set containing all the priority groups.

E^l: Set containing all the priority groups on link l.

$E[N_g]$: Expected load due to the packets of the v.c.'s in g, as seen at a random instant in steady–state.

f_i^l: Delay group of v.c. i on link l.

$f-1$: Delay group immediately preceeding the delay group f on the link on which f is defined.

F^l: Set containing all the delay groups on link l.

(FC_s): The flow control problem in the system–oriented approach.

g: Subset of the set of indices $\{1,\ldots,V\}$.

$g(t)$: Virtual load due to the packets of the v.c.'s in g at time t.

G: Number of iterations required to determine the zero of $G_s(\cdot)$ in the system–oriented approach or to determine the zero of $G_u(\cdot)$ in the user–oriented approach.

$G_i(R_i, N_i)$: Cost of assigning the rate R_i and the average number of packets N_i to v.c. i.

$G_s(\cdot)$: Compact representation of the minimum slackness in the equations in step (3) of Alg_P_s.

$G_u(\cdot)$: Compact representation of the minimum slackness in the equations in step (3) of Alg_P_u.

H: Set of weakly feasible delay assignments satisfying the ordering $D_1 \leq \cdots \leq D_V$.

$H(i)$: Home of node i.

$\bar{i}(e)$: V.c. in the priority group e having the highest position in the ordering.

$\underline{i}(e)$: V.c. in the priority group e having the lowest position in the ordering.

$\bar{i}(f)$: V.c. in the delay group f having the highest position in the ordering on the link on which f is defined.

$\underline{i}(f)$: V.c. in the priority group f having the lowest position in the ordering on the link on which f is defined.

$J_i(D_i)$: Optimal value of the minimization with respect to the delay of the v.c.'s $i+1, \ldots, V$ when v.c. i is assigned the delay D_i in the right hand side of equation (3.11).

k^-: Time immediately preceeding time k.

K: Constant.

L: Number of links in the network.

L_i: Number of links in the path of v.c. i.

$\mathcal{L}_i$: Set of links in the path of v.c. i.

N: Number of nodes in the network.

$\vec{N}$: Vector containing all the average number of packets $\vec{N} = (N_i^l,\ l \in \mathcal{L}_i,\ i = 1, \ldots, V)$.

N_i: Average number of packets of v.c. i.

$\vec{N}^l$: Vector containing the average number of packets on link l of the v.c.'s using link l.

N_i^l: Average number of packets of v.c. i on link l.

(NP_s): System–oriented problem in the multiple link case.

p_i: Probability that a packet is assigned the i^{th} priority in the aggregate stream consisting of the packets of v.c. i and of the packets of v.c.'s $1,\ldots,i-1$ which have not been assigned a priority greater than i.

(P_s): System–oriented problem in the single link case.

(P_u): User–oriented problem in the single link case.

$(P_{u,k})$: k^{th} problem in the hierarchy.

$q(\cdot)$: Dual functional.

$Q(\cdot)$: Queuing strategy.

$\vec{R}$: Vector containing the rate of the v.c.'s.

R_i: Rate of v.c. i.

S^*: Optimal value of $S(\cdot)$.

$S(\vec{D})$: Total delay cost corresponding to the assignment $\vec{D}$.

$S(\vec{R},\vec{N})$: Total cost corresponding to the rate assignment $\vec{R}$ and to the average number of packets assignment $\vec{N}$.

$\mathcal{S}_\gamma$: System–functional priority group whose characteristic number is γ.

T: Index used in Alg_P_s and Alg_P_u to keep track of the v.c.'s whose delay has already been assigned.

us_i^l: Identity of the link upstream to l on i's path.

Us_i^l: Set of links on i's path upstream to l.

$\mathcal{U}_\gamma$: User–functional priority group whose characteristic number is γ.

V: Total number of v.c.'s.

V^l: Number of v.c.'s sharing link l.

$\mathcal{V}^l$: Set of v.c.'s sharing link l.

$\vec{w}$: Ordering of the v.c.'s.

w_i: Position of v.c. i in the ordering $\vec{w}$.

$\vec{w}^l$: Ordering of the v.c.'s on link l.

w_i^l: Position of v.c. i in the ordering $\vec{w}^l$.

z_i: Marginal average number of packets cost associated with a variation of the rate of v.c. i.

$\overline{z}_i$: Approximation to z_i used to determine if the rate of v.c. i should be increased.

$\underline{z}_i$: Approximation to z_i used to determine if the rate of v.c. i should be reduced.

z_i^l: Marginal average number of packets cost on link l associated with a variation of the rate of v.c. i.

$\overline{z}_i^l$: Approximation to z_i^l used to determine if the rate of v.c. i should be increased.

$\underline{z}_i^l$: Approximation to z_i^l used to determine if the rate of v.c. i should be reduced.

Δ_n: Variation of the average number of packets in an iteration of Alg_NP_s_a.

$\overline{\Delta}_n$: Parameter of Alg_NP_s_a.

ΔC_{ij}: Difference in the marginal delay cost per unit rate of v.c.'s i and j.

Δd_i^l: Variation of the delay of v.c. i on link l from its reference value.

$\Delta_n G_{ij}$: Difference in the marginal average number of packets cost of v.c.'s i and j.

$\Delta_r G_i$: Net marginal cost associated with a variation of the rate of v.c. i.

ε_d: Parameter of Alg_NP_s_a.

ϕ_i: Overall arrival rate of the packets of priority greater than i.

γ_d: Parameter of Alg_NP_s_a.

γ_i: Arrival rate of the aggregate stream consisting of the packets of v.c. i and of the packets of v.c.'s $1, \ldots, i-1$ which have not been assigned a priority greater than i.

γ_k^*: k^{th} worst delay cost in the optimal solution of the problem (P_u).

λ_i: Dual variable associated with the i^{th} feasibility constraint in a problem (A_s).

λ_i^*: Lagrange multiplier associated with the i^{th} feasibility constraint in a problem (A_s) defined based on an ordering valid for the optimal solution of the problem (P_s).

$\lambda_i^{*\,l}$: Lagrange multiplier associated with the i^{th} feasibility constraint on link l in a problem (A_s) defined on link l based on an ordering on link l valid for the optimal solution of the problem (P_s^l).

μ: Capacity of the link in the single link case.

μ^l: Capacity of link l.

ν: Parameter used to define the delay groups.

$\vec{\psi}$: Dual variable in the dual problem corresponding to the problem $(P_{u,1})$.

$\vec{\psi}^*$: Lagrange multiplier vector in the dual problem corresponding to the problem $(P_{u,1})$.

ρ: Load on the link in the single link case.

ρ^l: Load on link l.

σ: Parameter of Alg_FC_s.

$\overline{\tau}$: Parameter of Alg_NP_s_e and Alg_FC_s.

τ_{ij}^l: Parameter in an update between v.c.'s i and j on link l in Alg_NP_s_e and Alg_FC_s.

$\Im(\vec{x})$: Lexicographic ordering of the vector $\vec{x}$.

1 Introduction

1.1 Introduction

The recent breakthrough in very large scale integration technology has resulted in a substantial decrease in the cost per unit of computation, especially in small to medium–size computers. This has led to the gradual replacement of the traditional mainframe by less powerful but more numerous and versatile machines. Also, the reduction in data processing costs has led to the introduction of a wide variety of new communication services. Typical examples are video, electronic mail and packetized voice. This evolution relies heavily on the ability to transmit information. As a consequence considerable developments have been made in the field of integrated services networks (i.s.n.).

The users of a network do not care about the internal operation of the network. Their prime interest is the quality of the received service, which, to a great extent, depends on two parameters: rate and delay. The rate of a user is a measure of the quantity of information that the user can input into the network per time unit while the delay is a measure of the time taken by the network to deliver this information to its destination. Obviously users prefer high rates and low delays but, equally obviously, these objectives are contradictory. The rates and the delays are tightly coupled through a set of feasibility conditions which limits considerably the possible choices.

A fundamental task of a network is to insure that users have acceptable rates and delays. The most popular approach to this problem consists of determining first an acceptable rate assignment, but without considering explicitly the delays. This problem is often called the flow control problem in the literature. In a second step, when the rates are known, a corresponding set of delays can be determined.

In most networks the constraints linking the rates and the delays are so strong that the delays are well–defined functions of the rates. In other words in these networks the knowledge of the rates is sufficient to completely specify the delays. The drawback in this situation is that, given a rate assignment, there is no possibility of adjusting the delay assignment. This does not present a problem if delay is of secondary importance. In recent years however, due to the introduction of several new types of users, delay has become

a much more sensitive issue. This has motivated several authors to suggest means for providing more flexibility regarding the choice of the delay assignment. In particular one of the most promising means for achieving this objective is priority queuing. Priority queuing allows preferences to be established among users, which, to a certain extent, uncouples the delay assignment from the rate assignment. In fact the additional flexibility provided by priority queuing is viewed as increasingly essential in i.s.n.'s as the requirements of the users become increasingly diversified.

Fully exploiting the flexibility provided by the use of priority queuing to select an acceptable delay assignment is a problem that has received very little attention in the literature. Our first major objective in this research is to propose a solution to this problem, which we will hereafter call the delay assignment problem.

The classical approach to flow control consists of defining a cost function whose role is to establish an order of preference among the possible rate assignments. Then the problem is solved by finding a rate assignment that minimizes this function. A major drawback of this approach is that the impact of the rate on the delay is not considered in the selection of the rate assignment, so that very poor delay assignments may result from this procedure. Indeed, this problem has been widely recognized. It is usually overcome by adding extra constraints in the flow control problem eliminating all the rate assignments leading to delay assignments not satisfying some a priori specified criterion. Of course although this approach guarantees a minimum delay performance, it is nowhere near considering the tradeoff existing between rate and delay.

The second objective of this research is to integrate the flow control and the delay assignment problems into a formulation that accounts fully for the interaction between rate and delay. In this formulation the cost function depends explicitly on both the rate and the delay of each user. The objective is to find simultaneously the rate and the delay assignment that minimizes this function.

1.2 Summary of related work

Traditionally one of the most severe constraints in the design of flow control schemes was the limited nodal storage capacity. The main design challenge was to construct a scheme robust enough to avoid the possibility of buffer overflow or deadlock in overload conditions [Ra76,La77,Ir78]. These factors were important because they usually resulted in an important drop of throughput.

Memory is no longer a scarce resource *, so that its management should not be of prime concern. Of course buffer overflows and deadlocks should still be avoided. However, it is now neither difficult nor expensive to equip the nodes with sufficient storage to keep the probability of occurence of these events very small. In this context buffer overflows and deadlocks should be handled by a dedicated prevention and/or recovery procedure independent of the flow control mechanism. In fact buffer overflows and deadlocks should now not impact the flow control mechanism more than other rare events, such as link or node failure, maintenance or network start–up.

As a result of the decrease in memory cost the emphasis in flow control has gradually shifted from the local management of memory to a more global perspective. In particular end–to–end flow control schemes are becoming increasingly popular. These schemes are characterized by the fact that their objective is defined directly in terms of the needs of the users. For example maintaining a certain fairness in the rate of the users [Ro78] or insuring a minimum throughput and delay performance to each user [Mu86] are typical end–to–end objectives.

As the performance of a network is always ultimately judged with respect to the quality of the service given to the users, it is our belief that schemes based on end–to–end criteria inherently lead to the best performance. In the remainder of this section we concentrate on a survey of the relevant end–to–end schemes presented in the literature in the context of this thesis. For a more extensive discussion on the flow control and delay issues the reader is refered to [Ge80,Gr83,Ta81].

* For example a 256K RAM that did not exist in 1981 was worth $40 in 1983. It is now sold for $5 [Ti85].

In the context of integrated voice and data networks Ibe [Ib81] and Gafni [Gaf82] have proposed flow control schemes where voice traffic is always given priority over data traffic. The rationale is to provide low end-to-end delay to voice traffic while maintaining high throughput for data traffic. A fundamental assumption behind these schemes is that a delay increase is much more damaging to a voice conversation than to a data exchange session. In practice this assumption is widely accepted. It is motivated by the belief that once a threshold is reached speech intelligibility degrades rapidly as delay increases (see for example [We79,Gr81]).

Systematically giving priority to voice traffic may sometimes be questionable. A trivial example is the case of a voice conversation sharing a link with an interactive data exchange session. If the delay of a voice conversation is well below threshold, it may be increased without impairing the quality of the conversation*. However, this may substantially improve the situation of the data session, thus yielding a better overall performance.

A more serious drawback of the above schemes is the implicit requirement that users can be ranked according to some label, and that the ranking is independent of the network state. Indeed, in the Ibe and Gafni schemes, users are labelled as being either "voice" or "data". The ranking consists of prescribing that the users labelled "voice" be given higher priority than the users labelled "data".

This formulation assumes that the users labelled "data" have relatively similar requirements, so that they can meaningfully be considered as an homogeneous group. However, with the introduction of a wide range of new services, this assumption may be questioned. For example there are enormous differences in the rate and delay requirements of a data session depending if it is video signal, electronic mail, file transfer or interactive session.

Of course this problem can be overcome by generalizing the approach. Namely, the label "data" can be subdivided into several more specific labels, such as "video signal", "electronic mail", etc. This, however, brings two important problems. First, a more detailed ranking is required. With only two labels; i.e., "voice" and "data", the ranking "voice in front of

* As long as it is kept below the threshold at which it becomes perceptible the delay of a voice conversation is unimportant.

data" is not too restrictive. With many kinds of users the ranking may become much more restrictive. Second, the ranking must be independent of the network state. With only the "voice" and "data" labels this requirement is not very restrictive because the ranking "voice in front of data" should prevail in most circumstances. Otherwise stated, in a voice and data network, giving priority to data may only be justified in a few exceptional situations. With many different kinds of users the insensitivity of the ranking to variations of the network state becomes a problem. This is because the new, more detailed, ranking cannot be as robust as the old "voice in front of data" ranking. For example a video signal should in general be given higher priority than an interactive session. However, if an interactive session is heavily penalized on one link, it may be desirable to give it a higher priority than the video signals on the other links, so as to compensate the impact of the heavily loaded link.

From a conceptual standpoint introducing labels and ranking them is a mean of artificially constraining the problem. As new labels are introduced the ranking must become increasingly detailed, and thus increasingly restrictive.

In [Th84] Thabit proposes a link scheduling algorithm circumventing the above difficulties. A convex non–decreasing cost function is associated to each user, the precise form of the function depending on the nature of the user. It translates, on a packet basis, to the dissatisfaction corresponding to a given end–to–end delay. In the proposed algorithm each link, knowing each packet delay on the upward links and estimating it on the downward links, schedules for transmission the packet most likely to incur the highest delay cost. Two important features of the scheme are its ability of accounting without added complexity for heterogeneous user requirements and of automatically maintaining packet sequencing.

Transmission scheduling requires that each link estimates the delay of each packet on the downward links. This is a central difficulty of the scheme. It is extremely difficult to construct estimators capable of reliably tracking the current state of the network. A second difficulty is that, even with perfect knowledge of all the parameters, scheduling the transmissions is a notoriously hard problem, especially in a distributed context (see for

example [So74,St74]). Typically, the processors in a network are incapable of solving such a problem in near–real time.

In [Wo82] Wong et al. suggested a measure of fairness based on the minimization of the deviations of the individual users' average delay from the overall average delay. Relatively to this performance measure two priority schemes are investigated: one where the non–preemptive priority of a user is set equal to the number of hops on its path (here, contrary to ordinary usage but in a more logical manner, a higher number corresponds to a higher priority) and the other being Kleinrock's linear time dependent priority scheme with suitably optimized parameters ([Kl76] chap. 3).

We disagree with this definition of fairness because it involves the users' end–to–end delay only through their deviations from the overall average delay. By increasing everybody's delay it is possible to reduce the deviations, thereby obtaining a better fairness. The delays can be increased by using a non work–conserving queuing strategy; for example by delaying the departure of the packets from their exit node. In [Wo82] the choice of priority schemes artificially prevented this situation from happening. However, Wong's formulation has two very appealing features. First, the objective of the scheme is defined in terms of the users' end–to–end average delay. Also, the use of priority queuing provides a considerable freedom in the choice of the average delay assignment. These two features will be central to our formulation.

A question that may be raised concerning Wong's scheme is the extent to which the choice of priority scheme limits the performance of the system. Otherwise stated, are there priority schemes that can result in a better performance than the two particular schemes considered by Wong et al? Another question concerns the coordination with a flow control algorithm. This will be required because Wong's scheme does not update rates and because of the tight coupling between rate and delay.

Initially proposed by Kleinrock [Kl78], the "power" of a network is a concept that has recently received considerable attention. In its most basic form the power is defined as the ratio of the average rate to the average delay.

Intuitively, maximizing the power is a very appealing objective for a flow control scheme [Ba81,Ja80]. Indeed, increasing the power of a network means increasing the rates or decreasing the delays. Clearly both alternatives are obviously desirable. An important and novel feature in this approach to flow control is that the selection of the rates depends explicitly on the resulting delay assignment. In fact in this approach the interaction between rate and delay is critical as both parameters have a comparable impact on the power.

Although it has some interesting properties and is intuitively appealing, the power has also several drawbacks. Jaffe has shown in [Ja81] that the power is a non–convex, non–decentralizable function. This means that it is in general difficult to produce an assignment minimizing the power, especially if the computations are to be distributed among the processors. The power is also an inconsistent and incomplete measure. Given two independent networks whose assignments are locally optimal (i.e., minimize the power of their respective network), when these networks are considered as subsets of a bigger network the assignments become non optimal. This is a gross inconsistency: as the networks have no interactions, considering them independently or as subsets of a bigger network should be irrelevant. Also, minimizing the power only imposes a set of constraints on the rate assignment. This set is in general not alone sufficient to determine the rate assignment.

1.3 Overview.

The main result presented in chapter 2 is a simple characterization of the set of average delays realizable in a single server queue shared by several competing users. This result is a specialization of the work of Coffman [Co80] to the case in which the exponential distribution of the service time of the jobs of each user is the same for all users. A significant consequence of this result is that the complexity of the problem of determining if a given delay assignment is realizable is only $O(V \log V)$, V being the number of users. In the more general case considered in [Co80] the complexity increases exponentially with V.

The proof of this result is based on a "universal" queuing scheme which is in itself a useful by–product. The scheme is universal because any realizable average delay assignment can be realized using the scheme. In other words if an average delay assignment is realizable in some way, it can also be realized by using the scheme. Another important property of the scheme is its simplicity. Given a realizable average delay assignment it is easy to construct the particular instance of the scheme that realizes the assignment.

In chapter 3 we exploit the results obtained in chapter 2 to obtain means of achieving delay assignments that are in some sense desirable. For that purpose each user is assigned a cost function quantifying its delay preferences. Two formulations are investigated. In the first formulation the objective is to minimize the sum of the user's costs. Obviously an average delay assignment achieving the minimum is desirable for it leads to the greatest overall users' satisfaction. We give a set of conditions necessary and sufficient for guaranteeing the optimality of a delay assignment for this problem. Then, based on these conditions, we present a simple algorithm that solves the problem.

In the second formulation the objective is to find a min–max fair delay assignment. The philosophy behind the min–max approach is to insure that, as much as possible, the users are served equally well by the network, independently of the particular constraints each user may impose on the network. We show that this problem is intimately related to the preceeding problem and propose a simple algorithm solving it.

In chapter 4 the attention is focussed on virtual circuit (v.c.) i.s.n.'s. These are networks in which each user communicates through a "virtual circuit"; namely a fixed

path, established during the set–up of the communication, and subsequently used for the whole duration of the communication. As there is a one–to–one corespondence between the v.c.'s and the users, we will often, in the remainder of this thesis, use the two terms interchangeably.

The objective in chapter 4 is to develop a means of using priority queuing in an i.s.n. so as to better control the end–to–end delay of the users. The approach taken is similar to that of chapter 3. Namely, the users' end–to–end delay preferences are quantified on an individual basis through associated end–to–end delay cost functions. The problem is to select the delay of the users on the links of the network so as to minimize the overall delay cost. We give a set of necessary and sufficient optimality conditions for this problem and then present two distributed algorithms solving it. The first algorithm is approximate in the sense that it does not, in general, converge to an optimal solution. However, via a suitable choice of the parameters of the algorithm, the assignment produced can be brought as close to optimality as desired. Moreover, the first algorithm converges in finite time and works in a completely uncoordinated manner. The second algorithm always converges to an optimal assignment. It requires however a greater coordination than the first algorithm.

In chapter 5 we propose a formulation of the flow control problem in which the interactions between the rates and the delays are fully considered. In the formulation the cost functions depend explicitly on the rate and the end–to–end delay of the users. The objective is to select the rate of each v.c. and the queuing strategy at each link so as to minimize the overall rate and delay cost. We give a set of necessary and sufficient optimality conditions for this problem. Then based on these conditions we construct a distributed algorithm solving the problem.

Finally we conclude this work with a discussion focussing on the possibilities for future research. We also comment qualitatively on some of the design issues arising in our formulation, for example regarding the choice of the cost functions.

2 Steering Average Delay Through Priority Queuing.

Our objective in this chapter is to investigate how priority queuing can be used to control the average delay of the users in a single–server queuing system.

The chapter is divided in 3 sections. In the first section we define the system and introduce the assumptions under which it will be analyzed. In the second section we summarize some results already available in the literature and derive some new results concerning the characterization of the average delays realizable in a single–server queuing system satisfying the framework introduced in section 1. In the last section we comment on the results presented in the chapter and mention some possible generalizations.

2.1 Modelization of the system.

We consider a system in which several v.c.'s are competing for the transmission capacity of a link. The link is modelled as a single server whose service rate is μ bits/sec. There are V competing v.c.'s, labelled $1, \ldots, V$. These v.c.'s generate packets which, upon arrival at the link, are queued for transmission. We associate with each v.c. i a rate R_i. R_i represents the rate (in packets per second) at which packets of v.c. i are generated. In the first chapters of this thesis the rates are assumed to be constant. This assumption will be removed in chapter 5.

We assume that:

(A.2.1.1) All arrival processes are Poisson processes independent of each other and of the state of the system.

(A.2.1.2) All packet lengths are drawn independently from a common exponential distribution.

(A.2.1.3) $\sum_{i=1}^{V} R_i < \mu$.

Also, we assume without loss of generality (w.l.o.g.) that the mean packet length is unity. These assumptions are very common in the literature.

We define $\rho = \frac{1}{\mu}\sum_{i=1}^{V} R_i$ as the load on the system. Note that assumption (A.2.1.3) implies that $\rho < 1$. This is a well–known stability condition for queuing systems. It merely says that the capacity of the system is sufficient for handling the demand.

We define a queuing strategy as a function $Q(\cdot, t)$ indicating the identity of the packet being transmitted at time t, or 0 if no packet is being transmitted. Equivalently, we may regard a queuing strategy as a set of rules which completely determine the operation of the link regarding the scheduling of the packet transmissions. Typical example of queuing strategies are first in first out (f.i.f.o.), last in first out (l.i.f.o.), and round–robin. We assume that a queuing strategy $Q(\cdot, t)$ satisfies:

(A.2.2.1) $Q(\cdot, t) = 0$ if and only if the system is idle.

(A.2.2.2) If t falls within busy period B then the value of $Q(\cdot, t)$ depends only on those arrival times, departure times, execution intervals and class indices which apply to packets waiting, being served or having received service during B up to time t.

(A.2.2.3) $Q(\cdot, t)$ may be deterministic or may be governed by some probability law whose parameters are restricted to those of assumption (A.2.2.2).

These assumptions were also made by Coffman in a context similar to ours [Co80]. In fact we will require some of Coffman's results in the sequel. In these assumptions some new notation has been used. This notation is defined as follows. A busy period is a time interval in which there is always at least one packet present in the system. An idle period is a time interval in which the system is empty. An execution interval is a time interval in which the server is devoted to the service of a particular packet. In non–preemptive queuing each packet has exactly one execution interval; it corresponds to its total service time. In preemptive queuing a packet may require several execution intervals; this results from the possibility that the packet be ejected from the server by an higher priority packet.

Assumption (A.2.2.1) imposes that the server cannot stand idle in front of waiting packets, which is obviously reasonable. The major limitation imposed by assumption (A.2.2.2) is that the scheduling decisions must be independent of the service time or of what remains of the service time of preempted packets. Because the length of a packet is usually known before its transmission this assumption is more restrictive. It may be justified by practical considerations. First, the requirement that packets of a given v.c. be delivered in order puts a severe limitation on schemes allowing service time dependent scheduling. Second, the packet service times are in practice more clustered than what is predicted by the exponential distribution assumption. This tends to weaken the impact of assumption (A.2.2.2), especially if, as is typically the case, interrupted transmissions are lost or if the amount of preemption is small.

A well–known model via which a common exponential distribution for the packet lengths can be generated while insuring that assumption (A.2.2.2) holds consists of pretending that in a very small time interval Δt the probability that a packet departs is $\mu\Delta t$ if the server works on the packet during the interval and 0 otherwise. Of course the applicability of this model in the context of a data–communication network is questionable as the packet length is usually known beforehand. It is, however, consistent with the extremely popular Kleinrock's independence assumption [Kl76].

Finally because preemption is allowed we must specify how the preempted packets resume their service in order to completely specify the behavior of the system. For this purpose we assume that:

(A.2.3) A preempted packet resumes its service from the point where it was left upon the last preemption.

Queuing systems in which assumption (A.2.3) holds are called preemptive–resume systems. The advantage of such systems is that the work already done on a packet is never lost.

Although preemptive–resume systems are not currently widely used in communication networks, we believe that in general their implementation does not present any serious conceptual problem. In fact we will later present a very simple scheme via which most preemptive–resume systems can easily be implemented.

It may be noted that assumption (A.2.3) can be replaced by the assumption that when a preempted packet resumes its service, it draws a new length from the exponential distribution independently of its previous length and of the state of the system. Systems in which this latter assumption is satisfied are called preemptive–repeat with resampling systems. The equivalence of the two assumptions in our context follows from the memoryless property of the exponential distribution.

Although we could have generalized assumption (A.2.3) to include preemptive–repeat with resampling systems, we have preferred not to do so. This is because in practice the packet lengths are not exponential, so that we have really no firm ground to justify the generalization.

2.2 Characterization of the set of realizable delays.

We first introduce some notation and terminology and state some fundamental results of queuing theory. Following this we define a queuing strategy which will play an important role in the proof of some of the results to be presented in the section. Next we present some new results concerning the characterization of the delays realizable in queuing systems fitting the framework of section 2.1. Finally we conclude the section by introducing some new notation related to the results which we will subsequently heavily use throughout the thesis.

2.2.1 Some fundamental results of queuing theory.

Consider a queuing system satisfying assumptions (A.2.1)–(A.2.3). Let $D_{i,n}$ be the total system delay of the n^{th} packet of v.c. i for a particular sample function of the queuing process in this system. Let:

$$D_i = \lim_{k\to\infty} \frac{1}{k} \sum_{n=1}^{k} D_{i,n} \tag{2.1}$$

D_i is the average system delay of v.c. i. A fundamental fact of queuing theory is that in systems satisfying assumptions (A.2.1)–(A.2.3) D_i is finite and independent of the particular sample function of the queuing process [Kl76]. Indeed, this is why our notation for D_i does not depend on the particular sample function. In the remaining of this thesis we call D_i simply the delay of v.c. i.

Let g be any non–empty subset of the set of indices $\{1,\ldots,V\}$. We define $E[N_g]$ as the virtual load due to the packets of the v.c.'s in g, as seen at a random time in the steady–state. $E[N_g]$ represents the expected time that it would take for a server of unit capacity to serve all the packets of the v.c.'s in g present in the system at a random instant if, starting from this instant, the server would work uniquely on these packets. We also define $g(t)$ as the virtual load due to packets of v.c.'s in g at time t for a particular sample function of the queuing process. $g(t)$ is a function which, at the arrival instant of a packet of one of the v.c.'s in g, jumps by an amount equal to the service time of the packet. In between jumps $g(t)$ decreases at a slope of μ in the intervals in which the server works on

packets of v.c.'s in g and is constant in the intervals in which the server does not work on packets of v.c.'s in g.

It can be shown that in a queuing system satisfying assumptions (A.2.1)–(A.2.3) any sample function of the queuing process satisfies [Co80]:

$$E[N_g] = \lim_{T\to\infty} \frac{1}{T} \int_0^T g(t)\, dt \tag{2.2}$$

Another well–known result of queuing theory is that all the queuing strategies that satisfy assumptions (A.2.1)–(A.2.3) also satisfy a conservation equation. This equation is usually called Kleinrock's conservation equation. We state it without proof in the following lemma. The proof can be found in several references, for example in [Co80].

Lemma 2.1 (Kleinrock's conservation equation): Let g be any non–empty subset of the set of indices $\{1,\ldots,V\}$. For any queuing system satisfying assumptions (A.2.1)–(A.2.3) the following relation holds:

$$\sum_{i\in g} R_i D_i = E[N_g]$$

Moreover, if $g = \{1,\ldots,V\}$:

$$E[N_g] = \frac{R_1 + \cdots + R_V}{\mu - R_1 - \cdots - R_V}$$

Let g_f and g_l be a non–trivial partition of the set of indices $\{1,\ldots,V\}$. In a queuing system satisfying assumptions (A.2.1)–(A.2.3) we say that the v.c.'s in g_f have full priority over the v.c.'s in g_l whenever the queuing strategy does not allow the server to work on a packet of a v.c. in g_l when a packet of a v.c. in g_f is present in the system. In this situation we also equivalently say that the v.c.'s in g_l are of lowest priority as compared to the v.c.'s in g_f.

An immediate consequence of this definition of full priority is that whenever there exist g_f and g_l such that the v.c.'s in g_f have full priority over the v.c.'s in g_l the delay of the v.c.'s in g_f satisfy:

$$\sum_{i\in g_f} R_i D_i = \frac{\sum_{i\in g_f} R_i}{\mu - \sum_{i\in g_f} R_i} \tag{2.3}$$

Indeed, this follows immediately from the fact that lemma 2.1 then applies to the v.c.'s in g_f considered in isolation.

Consider a pair of v.c.'s i and j. We say that v.c. i has full priority over v.c. j or equivalently that v.c. j is of lowest priority as compared to v.c. i whenever $i \in g_f$ and $j \in g_l$ for some g_f and g_l as defined above. Similarly, we say that v.c.'s i and j are of comparable priority if neither v.c. has full priority over the other.

We list without proof some well–known results of priority queuing in the next lemma. The proofs can be found in several references, for example in [Ja68].

Lemma 2.2: In any queuing system satisfying assumptions (A.2.1)–(A.2.3):

1) If v.c. i has full priority over v.c. j and v.c. j has full priority over v.c. k, then v.c. i has full priority over v.c. k.
2) If g_f and g_l are such that the v.c.'s in g_f have full priority over the v.c.'s in g_l, then for any v.c.'s $i \in g_f$ and $j \in g_l$:

$$D_i \leq \frac{1}{R_i}\left[\frac{\sum_{p\in g_f} R_p}{\mu - \sum_{p\in g_f} R_p} - \frac{\sum_{p\in g_f} R_p - R_i}{\mu - \sum_{p\in g_f} R_p + R_i}\right]$$

$$D_j \geq \frac{1}{R_j}\left[\frac{\sum_{p\in g_f} R_p + R_j}{\mu - \sum_{p\in g_f} R_p - R_j} - \frac{\sum_{p\in g_f} R_p}{\mu - \sum_{p\in g_f} R_p}\right]$$

Note that it follows immediately from the proposition 2 of the lemma that if v.c. i has full priority over v.c. j, then $D_i < D_j$.

2.2.2 The cascade scheme.

We now describe a priority scheme which will be used in the sequel. The scheme, called the cascade scheme, is depicted in Figure 2.1. The input of the cascade scheme are the V v.c.'s. The output consists of V streams, called the priority streams, and labelled $1,\ldots,V$. By construction each priority stream i, $i = 1,\ldots,V$, has the properties of having full priority over the priority streams $i+1,\ldots,V$ and of being of lowest priority as compared to the priority streams $1,\ldots,i-1$.

The priority streams are constructed as follows. Upon arrival in the system a packet of v.c. 1 is given with probability p_1, and independently of the state of the system the

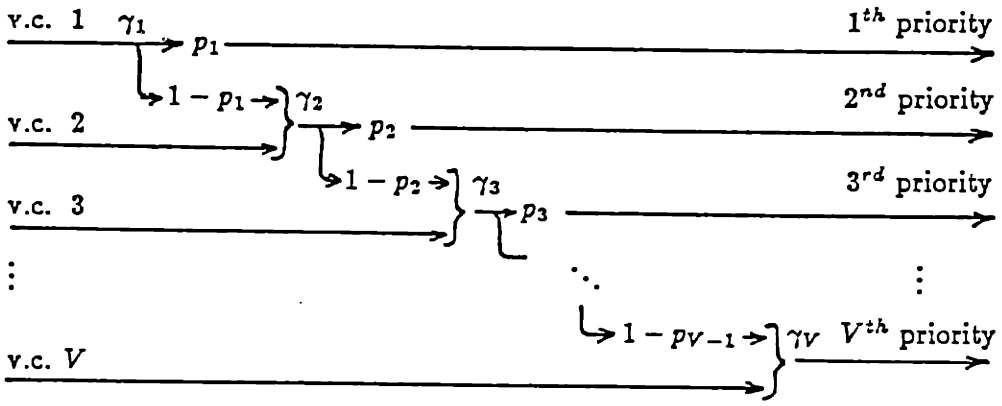


Figure 2.1: The cascade scheme.

highest "1" priority*. The packets of v.c. 1 which are given the highest priority constitute the priority stream 1. The packets of v.c. 1 which are not given the highest priority cascade down and are lumped into the stream of packets of v.c. 2. In the aggregate stream 2 (consisting of the packets of v.c. 2 and of the packets of v.c. 1 rejected from the top priority) a packet is given with probability p_2, and independently of the state of the system the second highest priority. The packets which are given the second highest priority constitute the priority stream 2. The packets of the aggregate stream 2 which are not given the second priority cascade down and are lumped into the stream of packets of v.c. 3. This cascading continues until v.c. V is reached, at which point packets are all given the lowest priority. All priorities are preemptive and, within each priority stream, packets are served on a f.i.f.o. basis.

* Throughout this work, as in most of the literature, priorities are numbered in their reverse order of precedence.

The scheme, as described above, does not necessarily preserve packet sequencing. For example a packet of v.c. 1 may cascade all the way down to the lowest priority stream where it is likely to experience a long delay. Another packet of v.c. 1 arriving subsequently will "pass" the first packet if it is routed to an higher priority stream and if the first packet is still in the system when the second packet arrives. For this reason we add the following rule to the cascade scheme: "when a packet of a given v.c. is transmitted it must be the oldest packet of this v.c. in the system but this packet exchanges its priority and position in queue with the packet (of the same v.c.) that would otherwise have been transmitted". Clearly, this rule does not affect the average delays while it ensures conservation of packet sequencing.

Let $\gamma_1 = R_1$, $\phi_1 = 0$ and define recursively for $i = 2, \ldots, V$.

$$\gamma_i = R_i + (1 - p_{i-1})\gamma_{i-1} \tag{2.4}$$

$$\phi_i = \phi_{i-1} + p_{i-1}\gamma_{i-1} \tag{2.5}$$

γ_i is the arrival rate of the aggregate steam i; namely the arrival rate of the packets of v.c. i and of the packets of the v.c.'s $1, \ldots, i-1$ which have not been assigned one of the first $i-1$ priorities (see Figure 2.1). ϕ_i is the arrival rate of the packets whose priority is greater than i. Note that it follows from the definition of γ_i that the arrival rate of priority stream i is respectively $\gamma_i p_i$ or γ_i depending if $i < V$ or $i = V$.

It is not difficult to see that the arrival processes of the priority steams are Poisson processes independent of each other and independent of the state of the system. Also, it is readily verified that the cascade scheme satisfies assumptions (A.2.2).

2.2.3 Characterization of the realizable delays.

Let $\vec{D} = (D_1, \ldots, D_V)^*$ be a given delay assignment. In a queuing system satisfying assumptions (A.2.1) and (A.2.3) we say that $\vec{D}$ is feasible (or realizable) if there exists a queuing strategy satisfying assumptions (A.2.2) which, if used to schedule the transmissions, produces $\vec{D}$.

* An arrow over a variable indicates that the variable is a vector.

Our first result gives a set of necessary and sufficient conditions for guaranteeing the feasibility of a given delay assignment. The result also shows that the cascade scheme is universal in the sense that whenever a delay assignment is realizable, it can be realized using the cascade scheme.

In the result it is assumed that the delay assignment satisfies $D_1 \leq \ldots \leq D_V$. Clearly this assumption is not restrictive since it can always be enforced via an appropriate re–labelling of the v.c.'s.

Theorem 2.1: Let $\vec{D}$ be a delay assignment satisfying $D_1 \leq \cdots \leq D_V$. Then the following propositions are equivalent:

a) $\vec{D}$ is realizable.

b) $\vec{D}$ is realizable using the cascade scheme and the choice of the p_i, $i = 1, \ldots, V-1$, is unique.

c) The following equations are satisfied:

$$\begin{aligned} R_1 D_1 &\geq \frac{R_1}{\mu - R_1} \\ \vdots \quad &\geq \quad \vdots \\ R_1 D_1 + \cdots + R_{V-1} D_{V-1} &\geq \frac{R_1 + \cdots + R_{V-1}}{\mu - R_1 - \cdots - R_{V-1}} \\ R_1 D_1 + \cdots + R_{V-1} D_{V-1} + R_V D_V &= \frac{R_1 + \cdots + R_V}{\mu - R_1 - \cdots - R_V} \end{aligned}$$

Proof:

We show that $a \Rightarrow c$, $c \Rightarrow b$, $b \Rightarrow a$.

1) $a \Rightarrow c$

This result has been shown in [Co80]. We repeat the proof here for completeness. The last equation in the set of equations in proposition (c) is true because it is Kleinrock's conservation equation. Consider any other equation in the set, say the i^{th} equation:

$$R_1 D_1 + \cdots + R_i D_i \geq \frac{R_1 + \cdots + R_i}{\mu - R_1 - \cdots - R_i} \tag{2.6}$$

Let g_i be the set containing the indices $\{1, \ldots, i\}$. From lemma 2.1we have

$$R_1 D_1 + \cdots + R_i D_i = E[N_{g_i}] \tag{2.7}$$

We also know from equation (2.2) that for any sample function of the queuing process:

$$E[N_{g_i}] = \lim_{T\to\infty} \frac{1}{T} \int_0^T g_i(t)\, dt \tag{2.8}$$

where $g_i(t)$ is the virtual load due to packets of v.c.'s in g_i at time t.

Consider the virtual load function, say $\tilde{g}_i(t)$, resulting from the same realization of arrival instants and service times as in $g_i(t)$ but with a queuing strategy that gives full preemptive priority to the packets of the v.c.'s in g_i. Because $g_i(t)$ and $\tilde{g}_i(t)$ jump by the same amounts at the same times and because $g_i(t)$ cannot decrease faster than $\tilde{g}_i(t)$, we have $g_i(t) \geq \tilde{g}_i(t)$ for all $t \geq 0$. This means:

$$\lim_{T\to\infty} \frac{1}{T} \int_0^T g_i(t)\, dt \geq \lim_{T\to\infty} \frac{1}{T} \int_0^T \tilde{g}_i(t)\, dt = E[\tilde{N}_{g_i}] \tag{2.9}$$

where $E[\tilde{N}_{g_i}]$ is the expected virtual load due to the packets of the v.c.'s in g_i under the preemptive queuing strategy. However under the preemptive strategy lemma 2.1 applies to the v.c.'s in g_i in isolation. That is:

$$E[\tilde{N}_{g_i}] = \frac{R_1 + \cdots + R_i}{\mu - R_1 - \cdots - R_i} \tag{2.10}$$

Combining equations (2.7), (2.8), (2.9) and (2.10) we get equation (2.6).

2) $c \Rightarrow b$

Let D_{ps_i} be the delay of the i^{th} priority stream in the cascade scheme. Since the i^{th} priority stream is of lowest priority as compared to the priority streams $1, \ldots, i-1$ and has full priority over the priority streams $i+1, \ldots, V$, it follows from lemma 2.1 that:

$$D_{ps_i} = \frac{1}{p_i\gamma_i}\left[\frac{p_i\gamma_i + \phi_i}{\mu - p_i\gamma_i - \phi_i} - \frac{\phi_i}{\mu - \phi_i}\right] \tag{2.11}$$

Consider a particular v.c. $i < V$. The delay of v.c. i in the cascade scheme is a weighted average of two contributions. A proportion p_i of the packets of v.c. i is routed to the priority stream i. As these packets constitute a Poisson process independent of the state of the system, it follows that these packets must experience the average delay D_{ps_i}. On the other hand the packets of v.c. i which are not given the i^{th} priority are treated exactly as the packets of v.c. $i+1$. As these packets constitute a Poisson process independent of the

arrival process of the packets of v.c. $i+1$, and as the arrival process of the packets of v.c. $i+1$ is also Poisson, it follows that the packets of v.c. i which are not given priority i must experience the average delay D_{i+1}. Similarly, it is not difficult to see that the average delay of the packets of v.c. V is identical to that of the packets of the V^{th} priority stream. Using these facts we can write:

$$D_i = \begin{cases} p_i D_{ps_i} + (1-p_i) D_{i+1}, & i < V \\ D_{ps_V}, & i = V \end{cases} \tag{2.12}$$

To prove the proposition we must show that when the assignment $\vec{D}$ satisfies the set of equations in proposition (c) there exists $p_i \in [0,1]$, $i = 1, \ldots, V-1$, such that the above equations are satisfied. Moreover, we must show that the choice of the p_i, $i = 1, \ldots, V-1$, is unique. We use an induction argument. We first show as the initial step of the induction that there exists a unique $p_1 \in [0,1]$ for which equation (2.12) is satisfied.

Using equation (2.11), equation (2.12) becomes:

$$D_1 = \frac{p_1}{\mu - R_1 p_1} + (1-p_1) D_2 \tag{2.13}$$

Define:

$$f_1(p) = \frac{p}{\mu - R_1 p} \tag{2.14}$$

which for $p \in [0,1]$ is strictly convex and non–decreasing. Also $f_1(0) = 0$ and, using the first equation in the set of equations in proposition (c):

$$f_1(1) = \frac{1}{\mu - R_1} \leq D_1 \tag{2.15}$$

Consider the equation:

$$D_1 + (p-1) D_2 = f_1(p) \tag{2.16}$$

As shown if Figure 2.2 the fact that $D_1 \leq D_2$ and the properties of the function $f_1(\cdot)$ guarantee that this equation has at least one root in the interval $[0,1]$. This proves that at least one $p_1 \in [0,1]$ satisfying equation (2.12) exists. To complete the initial step of the induction we must show that p_1 is unique.

If strict inequality is achieved in equation (2.15) it is easy to see that p_1 must be unique (this is the case depicted in Figure 2.2). If equality holds in equation (2.15) two cases are

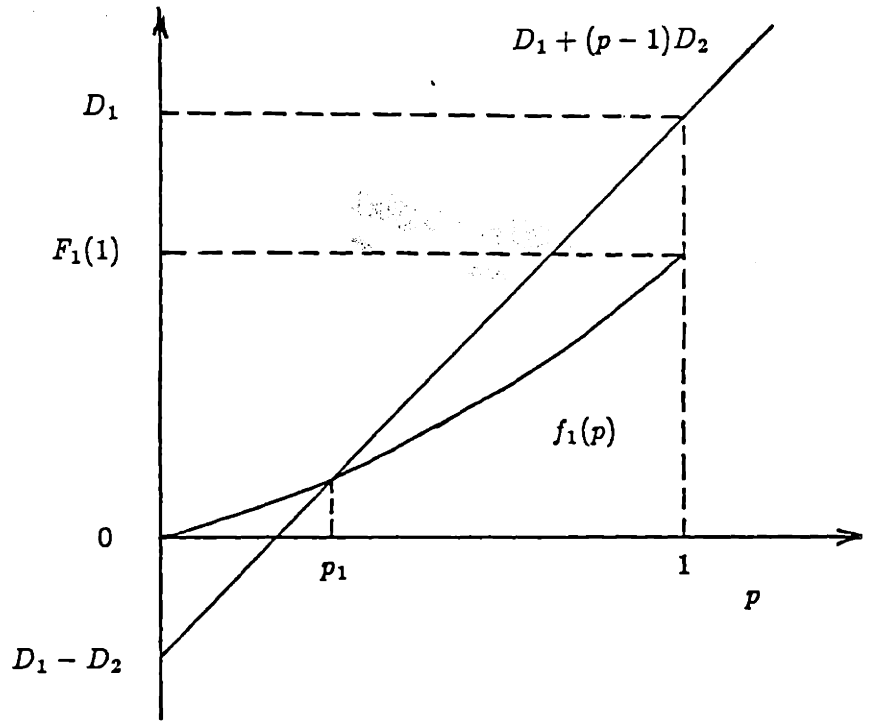


Figure 2.2.

possible, as shown in Figures 2.3–a and 2.3–b. Case (a) corresponds to the situation in which equation (2.15) has more than one root in the interval $[0, 1]$ while case (b) corresponds to the situation in which the root is unique. Note that $p = 1$ must be a root in both cases.

If case (a) prevails we must have:

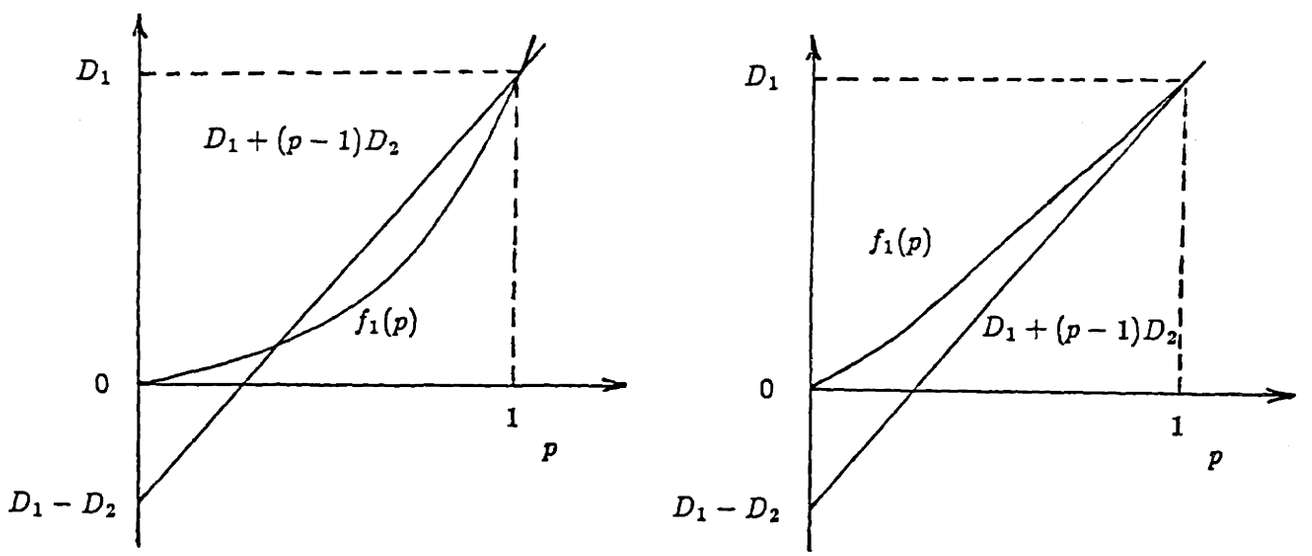
$$\left[\frac{df_1(p)}{dp}\right]\Big|_{p=1} > \frac{d}{dp}\left[D_1 + (p-1)D_2\right] \tag{2.17}$$

Evaluating the derivatives we obtain:

$$\frac{\mu}{(\mu - R_1)^2} > D_2 \tag{2.18}$$

On the other hand, it is known that the second equation in the set of equations in proposition (c) holds; i.e.,

$$R_1 D_1 + R_2 D_2 \geq \frac{R_1 + R_2}{\mu - R_1 - R_2} \tag{2.19}$$



a) p not unique

b) p unique

Figure 2.3.

If $p = 1$ is a root of equation (2.16) it follows that $D_1 = f_1(1) = 1/(\mu - R_1)$. Replacing in the last equation we obtain:

$$\begin{aligned} D_2 &\geq \frac{1}{R_2}\left[\frac{R_1 + R_2}{\mu - R_1 - R_2} - \frac{R_1}{\mu - R_1}\right] \\ &\geq \frac{\mu}{(\mu - R_1 - R_2)(\mu - R_1)} \end{aligned} \tag{2.20}$$

Equation (2.20) contradicts equation (2.18). It follows that only case (b) is possible, which proves that p_1 is unique. This concludes the proof of the initial step of the induction.

The induction step can be proven using essentially the same argument as the initial step. However, as the proof of the induction step does not bring much additional insight, we have relegated it to section 1 of Appendix A.

3) $b \Rightarrow a$

This follows immediately from the fact that $\vec{D}$ can be realized using the cascade scheme.

Q.E.D.

The following corollary provides a quick way of determining the set of p_i via which a given realizable delay assignment can be realized in the cascade scheme.

Corollary 2.1: Assume that $\vec{D}$ is feasible. Also, assume (w.l.o.g.) that $D_1 \leq \cdots \leq D_V$. Then the probabilities $p_1, \ldots, p_{V-1}$ which, in the cascade scheme, realize $\vec{D}$ are given recursively for $k = 1, \ldots, V-1$, by:

$$p_k = \frac{K_k - \sqrt{K_k^2 - 4D_{k+1}\gamma_k(D_{k+1} - D_k)(\mu - \phi_k)}}{2D_{k+1}\gamma_k}$$

where:

$$K_k = (\mu - \phi_k)D_{k+1} + \gamma_k(D_{k+1} - D_k) - \frac{\mu}{\mu - \phi_k}$$

and where γ_k and ϕ_k are as in equations (2.4) and (2.5).

This corollary is proven in the section 2 of Appendix A.

Another consequence of theorem 2.1 is the following, intuitively obvious, corollary.

Corollary 2.2: The set of feasible delays is convex.

This corollary is proven in the section 3 of Appendix A.

The proof of this corollary relies on the argument that if two delay assignments are feasible, any convex combination of these assignments can be realized simply by using the strategy realizing each assignment during an appropriate proportion of the time. This, however, does not imply that in order to realize a convex combination of feasible delays one has to resort to a time–dependent queuing strategy. Indeed since the convex combinations are feasible it follows that they can be realized using the cascade scheme, which is time–independent.

It is essential that the rates be constant for the set of realizable delays to be convex: jointly the set of feasible rates and delays is not a convex set. This may be seen by considering the set of equations of theorem 2.1–c when both $\vec{R}$ and $\vec{D}$ are allowed to vary. Intuitively, this can be explained from the fact that the rate enters as a weighting factor in the computation of the average delay. For example suppose that for a rate of 1 the average delay of a v.c. is 1 but that when the rate is increased to 2 the average delay increases to

2. If the v.c. is given a rate of 1 and of 2 during equal amounts of time, the overall average delay is not 1.5 but 1.66 because twice as many packets experience a delay of 2 than a delay of 1.

The cascade scheme possesses some nice properties; e.g., ease of implementation, time invariance, conservation of packet sequencing. However, no claims of any sort of optimality are being made on this scheme. Indeed, it is our belief that many other schemes share some of these properties.

The result presented next is the specialization of lemma 1, lemma 2 and theorem 1 of [Co80] to the case in which common service time prevails. We omit the proof of this result here. The proof may be found in [Co80].

Theorem 2.2: Consider a queuing system satisfying assumptions (A.2.1)-(A.2.3). The following propositions are equivalent.

a) $\vec{D}$ is realizable.

b) $\vec{D}$ belongs to the convex hull of the $V!$ distinct delay assignments corresponding to the possible permutations of V distinct preemptive priorities among the V v.c.'s.

c) For all $g \subset \{1,\ldots,V\}$:

$$\sum_{i\in g} R_i D_i \geq \frac{\sum_{i\in g} R_i}{\mu - \sum_{i\in g} R_i}$$

and if $g = \{1,\ldots,V\}$ equality is achieved in the above equation.

2.2.4 Some notation.

In this section we introduce some notation related to the results presented in the last section. We also state some simple results which follow from the results of the last section and which will be used repeatedly throughout the thesis.

Let $\vec{w} = (w_1,\ldots,w_V)$ be a permutation of the vector $(1,\ldots,V)$. We call $\vec{w}$ an ordering of the v.c.'s on the link and we call w_i, $i = 1,\ldots,V$, the position of v.c. i in the ordering. We say that v.c. i has an higher (resp. lower) position than v.c. j in the ordering if $w_i < w_j$ (resp. $w_i > w_j$). An ordering $\vec{w}$ is valid for an assignment $\vec{D}$ if for all i, j, $1 \leq i,\ j \leq V$:

$$w_i < w_j \ \Rightarrow\ D_i \leq D_j \tag{2.21}$$

The equations in theorem 2.1–c are called the feasibility constraints. The feasibility constraint whose left hand side contains exactly k terms is called the k^{th} feasibility constraint. We say that v.c. j appears in the k^{th} feasibility constraint if the left hand side of the feasibility constraints $1,\ldots,k-1$ does not contain the term R_jD_j but if the left hand side of the feasibility constraints $k,\ldots,V$ contains it.

It may be noted that the feasibility constraints are not unique when several v.c.'s have the same delay. This is due to the fact that the first $V-1$ feasibility constraints depend on the ordering in which the delays of the v.c.'s are ranked, and that this ordering is not unique when the delays of several v.c.'s are equal. However it is clear that given a valid ordering these feasibility constraints are unique. Indeed given a valid ordering, say $\vec{w}$, the k^{th} feasibility constraint, $k=1,\ldots,V-1$ then becomes:

$$\sum_{p|w_p\leq k} R_pD_p \geq \frac{\sum_{p|w_p\leq k} R_p}{\mu - \sum_{p|w_p\leq k} R_p} \tag{2.22}$$

By definition when the feasibility constraints are constructed using the ordering $\vec{w}$ the v.c. appearing in the k^{th} feasibility constraint, $k=1,\ldots,V$, is always the v.c., say i, satisfying $w_i=k$. In the remainder of this work we will always specify the ordering on which the feasibility constraints are based, so that the feasibility constraints will always be known unambiguously.

Consider a given delay assignment. Suppose that this assignment is such that for a given ordering valid for it the k^{th} feasibility constraint is satisfied with equality. Then it follows from lemma 2.2 that the delay of the v.c.'s involved in this constraint are strictly smaller than the delay of the other v.c.'s. This implies that in any other ordering valid for the assignment the positions of the v.c.'s involved in this constraint must be smaller than the positions of the other v.c.'s, so that this constraint must in fact be the k^{th} feasibility constraint in all the ordering valid for the assignment. This shows that the feasibility constraints satisfied with equality are common to all the orderings valid for the assignment. Similarly, it is not difficult to see that if for an ordering valid for the assignment the k^{th} feasibility constraint is satisfied with strict inequality, then the k^{th} feasibility constraint is

satisfied with strict inequality for all the orderings valid for the assignment. Note however that in this case the constraint may change depending on the ordering.

Let $\vec{D}$ be a feasible delay assignment and let $\vec{w}$ be an ordering valid for this assignment. Consider a pair of v.c.'s i and j and assume w.l.o.g. that $w_i < w_j$. It is not difficult to see that v.c. i has full priority over v.c. j if and only if at least one of the feasibility constraints $w_i, \ldots, w_j - 1$ is satisfied with equality. Similarly v.c.'s i and j are of comparable priority if and only if the feasibility constraints $w_i, \ldots, w_j - 1$ are all satisfied with strict inequality. These facts motivate the following definitions.

Given a feasible delay assignment $\vec{D}$ we define the priority group of v.c. i, denoted e_i, as follows:

$$e_i = \{j \mid j \text{ is of comparable priority to } i\} \tag{2.23}$$

We also define:

$$E = \{e_i,\ i = 1, \ldots, V\} \tag{2.24}$$

It is easy to see that if e and $\tilde{e}$ are two distinct priority groups, then:

$$e \cap \tilde{e} = \phi \tag{2.25}$$

and either:

$$i \text{ has full priority over } j \text{ for all } i \in e,\ j \in \tilde{e} \tag{2.26}$$

or:

$$j \text{ has full priority over } i \text{ for all } i \in e,\ j \in \tilde{e} \tag{2.27}$$

Equations (2.26) and (2.27) can be used to order the priority groups. Indeed, we will say that $e < \tilde{e}$ whenever e and $\tilde{e}$ satisfy equation (2.26) and conversely we will say that $e > \tilde{e}$ whenever e and $\tilde{e}$ satisfy equation (2.27). It is not difficult to see that this is a well–defined order relationship.

Let e be a priority group such that $e > \tilde{e}$ for some $\tilde{e} \in E$. We define $e - 1$ to be the priority group immediately preceeding e. Namely $e - 1$ is the priority group satisfying $e - 1 < e$ and $e - 1 \geq \hat{e}$ for all $\hat{e} \in E$, $\hat{e} < e$.

Let $\vec{w}$ be an ordering valid for the assignment $\vec{D}$ and let e be a priority group. Given the ordering $\vec{w}$ we denote by $\bar{i}(e)$ and $\underline{i}(e)$ the v.c.'s in e having respectively the highest and the lowest position in the ordering. Namely $\bar{i}(e)$ and $\underline{i}(e)$ satisfy:

$$\begin{aligned} w_{\bar{i}(e)} &\leq w_j \quad \text{for all } j \in e \\ w_{\underline{i}(e)} &\geq w_j \quad \text{for all } j \in e \end{aligned} \tag{2.28}$$

Finally given an assignment $\vec{D}$ and an ordering $\vec{w}$ valid for this assignment we define for convenience:

$$B(i, \vec{w}, \vec{R}) = \frac{\sum_{p|w_p \leq i} R_p}{\mu - \sum_{p|w_p \leq i} R_p} \quad 1 \leq i \leq V \tag{2.29}$$

2.3 Comments.

An immediate consequence of theorem 2.1 is that it allows us to determine in $O(V \log V)$ computations if a given delay assignment is feasible. The major task is the ordering of the delays which takes $O(V \log V)$ comparisons. On the other hand if theorem 2.2 is used the number of computations grows exponentially. This is because the number of distinct subsets in theorem 2.2–c grows exponentially. It is interesting to note how the common service time assumption, which seems to be a modest assumption, reduces the complexity of the problem.

In this chapter we obviously consider preemptive queuing as acceptable. This is relatively unusual. Indeed almost all queuing strategies mentioned in the data–communication literature are non–preemptive. Our main reason for allowing preemption is that it leads to a larger feasible delay set. In other words, it gives more flexibility in the choice of a delay assignment. However a second important reason for allowing preemption is that it is not as costly as it may look. In fact we believe that the overhead incurred for allowing preemption in the context of a v.c. communication network is very small.

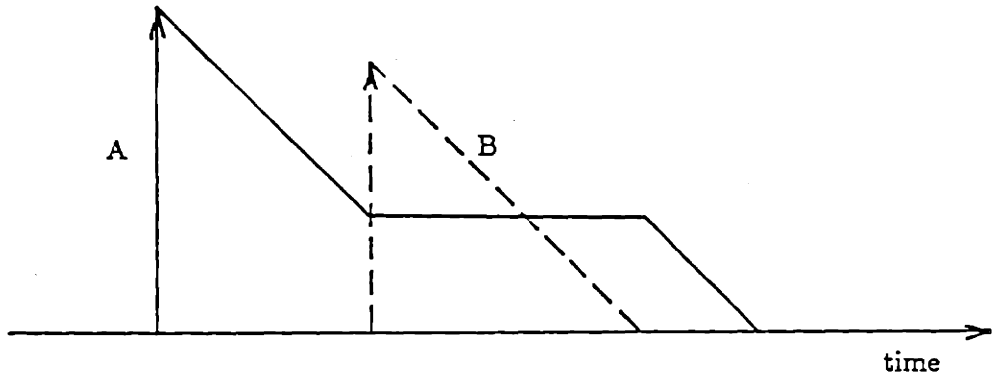
A first thing to notice is that even if preemption is allowed one is not forced to resort to a preemptive queuing strategy to realize a delay assignment. Indeed a large class of feasible delay assignments can be realized by non–preemptive queuing strategies. A useful result in this context is that any delay assignment realizable by a non–preemptive queuing strategy can also be realized by the non–preemptive version of the cascade scheme. This version of the cascade scheme is as described in section 2.2.2 except that all priorities are non–preemptive. More information concerning the non–preemptive version of the cascade scheme and concerning the characterization of the delay assignments realizable by non–preemptive queuing strategies is provided in the section 4 of Appendix A.

Another argument in favor of the use of preemptive queuing is that in v.c. communication networks preemptive queuing can be implemented very easily and efficiently. Indeed it is possible to imagine simple schemes in which the overhead incurred in a preemption is much smaller than the pure loss of the transmitted portion of the preempted packet*. For

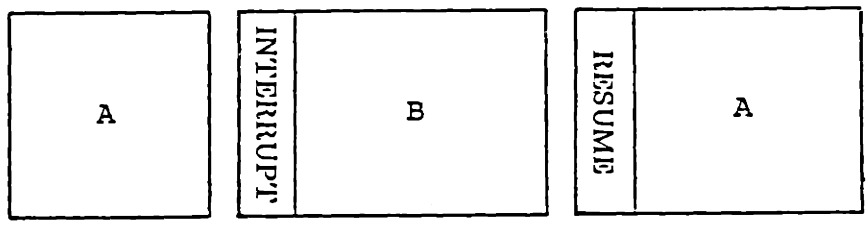
* This idea has been suggested by Prof. Gallager.

example the cascade scheme can easily be implemented by serving the preempted packets in a l.i.f.o. manner. To illustrate this point consider the following example, as depicted in Figure 2.4. Initially the link is transmitting packet A which is the only packet present in the system. Before A's transmission terminates a higher priority packet B arrives, causing an interruption in A's transmission. However, the transmitted portion of packet A is not lost. A flag, called INTERRUPT, is placed in front of message B. This flag informs the destination node that A's transmission has been interrupted and that the message that caused the interruption is incoming. When A's tranmission resumes, after completion of B's transmission, another flag, called RESUME, is inserted. Again the role of this flag is to notify the destination node that A's transmission is ready to resume. One may see that the scheme provides enough information to allow the destination node to uniquely decode the sequence of interleaved packets. It may also be noted that this scheme requires only two flags per preemption, which is a very small transmission overhead. Moreover, as the packets of each v.c. are transmitted in sequence, only one buffer per v.c. is required at the receiving node for storing the packets whose transmission is not yet completed. This is a very small memory overhead.

Finally, although we believe that the use of preemption may be advantageous, it is worth noting that allowing preemption is not at all essential. Indeed all the results presented in this chapter can be re–established in the context of non–preemptive queuing only. Essentialy only slight modifications are required to account for the new form of the feasible delay set. Details are omitted here but these modified results are presented in the section 4 of Appendix A.



a) Activity of the server.



b) Transmitted sequence.

Figure 2.4

3 Minimization of delay cost – The single link case.

In this chapter we consider the problem of assigning average delays to v.c.'s so as to achieve a desirable point of operation in queuing systems satisfying the framework developed in chapter 2. We consider two approaches to this problem. These two approaches, which have been called the system–oriented approach and the user–oriented approach, differ essentially in the choice of the objective to be achieved. In the system–oriented approach the main concern is to maximize the overall wealth while in the user–oriented approach the emphasis is rather to achieve a certain fairness among the users.

The chapter is divided in five sections. In the first section we formulate the problems in the system–oriented approach and in the user–oriented approach. In the second section we state some well–known results of convex programming that will be heavily used in the sequel. In the third and fourth sections we respectively study the problems in the system–oriented approach and in the user–oriented approach. Finally in the last section we comment on the results developped in the chapter and establish a fundamental relationship between the problems in the system–oriented approach and in the user–oriented approach.

3.1 Introduction.

We associate a delay cost function $C_i(\cdot)$ to each v.c. i. $C_i(D_i)$ quantifies the dissatisfaction of v.c. i when its average delay is D_i.

In the system–oriented approach the objective is to maximize the overall wealth or, equivalently, to minimize the total dissatisfaction. We call this approach system–oriented because the concerns of the users are not treated on an individual basis but are instead only considered through an aggregate system–wide measure. A queuing strategy is optimal in the system–oriented approach if the average delay assignment that it produces is a solution

of the following problem, hereafter referred to as (P_s) *:

$$(P_s) \qquad \min_{\vec{D} \text{ feasible}} S(\vec{D})$$

where:

$$S(\vec{D}) = \sum_{i=1}^{V} C_i(D_i) \tag{3.1}$$

The additive form of the objective function justifies the attribute "system" to the approach. V.c.'s are only considered through their impact on the global performance measure $S(\cdot)$.

In the user–oriented approach the aim is to achieve a fair delay assignment. Unlike the system–oriented approach the v.c.'s in the user–oriented approach are explicitly considered on an individual basis.

The user–oriented approach involves the notion of fairness. We will consider fairness in the min–max sense. The philosophy behind the min–max fairness is to make the system as transparent to the user as possible. In particular an important objective of the min–max fairness is to uncouple as much as possible the quality of the service given to the users from the individual constraints imposed by each user on the network.

In order to define precisely the problem in the user–oriented approach some new notation is needed. This notation is now introduced. Given an arbitrary vector $\vec{x}$ we define the lexicographic ordering of $\vec{x}$, denoted $\Im(\vec{x})$, as the non–increasing permutation of $\vec{x}$. For example the lexicographic ordering of the vector $(4,1,8,2,7)$ is the vector $(8,7,4,2,1)$. Also, given two vectors $\vec{x} = (x_1, \ldots, x_n)$ and $\vec{y} = (y_1, \ldots, y_n)$, we say that $\vec{x}$ is lexicographically smaller than $\vec{y}$ if $x_i \geq y_i$ for some i implies that $x_j < y_j$ for some $j < i$. For example the vector $(7,7,7)$ is lexicographically smaller than the vector $(8,0,0)$.

Now the problem in the user–oriented approach, hereafter referred to as (P_u), can be defined as follows:

$$(P_u) \qquad \min_{\vec{D} \text{ feasible}} \Im\big(C_1(D_1), \ldots, C_V(D_V)\big)$$

* By writing "min" in the following problem the existence of a minimum is implicitly assumed. This is justified because the delay of a v.c. cannot be greater than what it would be if the v.c. was given alone the lowest priority. This means that the feasible set can be made compact, insuring the existence of a minimum.

where the minimization in this problem should be interpreted in the lexicographic sense; i.e., the objective is to find the delay assignment which leads to the smallest lexicographic ordering of the cost vector.

We call a delay assignment achieving the minimum in (P_u) a min–max fair delay assignment. Compared to arbitrary feasible delay assignments, min–max fair delay assignments possess a property that justifies why they are considered to be fair [Gaf82, Gal84]. Specifically if $\vec{D}$ is a min–max fair delay assignment and $\vec{\tilde{D}}$ is an arbitrary feasible assignment, if $C_i(D_i) > C_i(\tilde{D}_i)$ for some i then it must be that for some $j \neq i$ $C_i(D_i) \leq C_j(D_j)$ and $C_j(D_j) < C_j(\tilde{D}_j)$. In words this means that if the assignment $\vec{\tilde{D}}$ provides a better service to some v.c. i than the min–max assignment $\vec{D}$, it must be that the assignment $\vec{D}$ instead provides a better service than the assignment $\vec{\tilde{D}}$ to some v.c. j not served as well as v.c. i. Clearly a min–max fair delay assignment is fair in the sense that it provides a service as equitable as possible to the users.

It is worth noting that a min–max delay assignment can be obtained by solving a hierarchy of nested problems. The first problem minimizes the maximum delay cost. This insures that the most penalized users get as much service from the network as they can possibly get. Solving this problem tells us what the worst delay cost is and who are the users experiencing it. However solving this problem is in general not sufficient to determine the delay of every user. This is because although the delay of the v.c.'s having the worst delay cost is known, the delay of the v.c.'s not having the worst delay cost is not yet known. To determine the delay of some of the v.c.'s not having the worst delay cost a second nested problem is defined. The objective of this second problem consists of minimizing the second largest delay cost subject to not increasing the delay cost of the v.c.'s having the largest delay cost. Solving this problem will tell us what the second delay cost is and who are the v.c.'s experiencing it. Clearly this method can be generalized to determine the delay of every v.c. This method is described in details in [Ro78] in the context of the routing problem, and in [Hay81] in the context of the flow control problem.

3.2 Some well–known results of convex programming.

In this section we state without proof some well–known results of convex programming. The reader is referred to [Be83] or [Av76] for an extensive treatment of this material.

Define:

$$(CPP) \qquad \begin{aligned} &\min \quad f(\vec{x}) \\ &\vec{x} \in X,\ g_j(\vec{x}) \leq 0,\ j = 1, \ldots, m \end{aligned}$$

where it is assumed that:

a) The set X is a convex subset of R^n.

b) The functions $f : R^n \rightarrow R$ and $g_j : R^n \rightarrow R$, $j = 1, \ldots, m$, are convex over X.

c) There exists at least one feasible solution to (CPP).

d) The optimal value of (CPP); f^*, is finite; i.e.,

$$f^* = \inf\left\{ f(\vec{x}) \mid \vec{x} \in X,\ g_j(\vec{x}) \leq 0,\ j = 1, \ldots, m \right\} < \infty \tag{3.2}$$

(CPP) is the archetype of a convex programming problem.

Let $\vec{\psi} \in R^m$. Define the Lagrangian as:

$$L(\vec{x}, \vec{\psi}) = f(\vec{x}) + \sum_{j=1}^{m} \psi_j g_j(\vec{x}) \tag{3.3}$$

and define a Lagrange multiplier vector as a vector $\vec{\psi}^*$ such that:

$$f^* = \inf_{\vec{x} \in X} L(\vec{x}, \vec{\psi}^*) \tag{3.4}$$

Further, define:

$$q(\vec{\psi}) = \inf_{\vec{x} \in X} L(\vec{x}, \vec{\psi}) \tag{3.5}$$

$q(\vec{\psi})$ is called the dual functional. The domain of $q(\cdot)$, denoted *dom* q, is the set of all vectors $\vec{\psi}$ such that $q(\vec{\psi}) > -\infty$. That is:

$$dom\ q = \left\{ \vec{\psi} \in R^m \mid \inf_{\vec{x} \in X} L(\vec{x}, \vec{\psi}) > -\infty \right\} \tag{3.6}$$

Assuming that *dom* q is non–empty the dual problem is defined as:

$$\begin{aligned} &\sup \quad q(\vec{\psi}) \\ &\vec{\psi} \geq 0 \\ &\vec{\psi} \in dom\ q \end{aligned} \tag{3.7}$$

We have the following results.

Theorem 3.1 (proposition 5.6 of [Be83]): Assume in (CPP) that there exists a vector $\vec{\bar{x}} \in X$ such that $g_j(\vec{\bar{x}}) < 0, j = 1, \ldots, m$ (i.e., assume that the set defined by the constraints $g_j(\vec{x}) \leq 0$ contains a strict interior point which also belongs to X). Then there exists a Lagrange multiplier vector and the optimal value f^* of (CPP) is equal to the optimal value of the dual problem.

Theorem 3.2 (proposition 5.2 of [Be83]) : The vectors $\vec{x}^*$ and $\vec{\psi}^*$ form an optimal solution–Lagrange multiplier pair for (CPP) if and only if the following Kuhn–Tucker conditions hold:

$$\vec{x}^* \in X,\ \vec{\psi}^* \geq 0,\ g_j(\vec{x}^*) \leq 0,\ \psi_j^* g_j(\vec{x}^*) = 0,\ j = 1, \ldots, m$$

$$L(\vec{x}^*, \vec{\psi}^*) = \inf_{\vec{x} \in X} L(\vec{x}, \vec{\psi}^*)$$

The condition $\psi_j^* g_j(\vec{x}^*) = 0,\ j = 1, \ldots, m$, is often referred to as the complementary slackness condition.

3.3 The system–oriented approach.

3.3.1 Characterization of the optimal solution.

In this section we of course assume that the assumptions (A.2.1)–(A.2.3) hold. In addition we also make the following assumption on the cost functions.

(A.3.1) For all i, $i = 1, \ldots, V$, $C_i(\cdot)$ is strictly convex, non–decreasing and twice continuously differentiable over the interval $[0, \infty[$.

Although in a slightly different context, the convexity assumption has been well justified in [Th84]. Also, as the users obviously prefer low delays, it is clear that assuming that the cost functions are non–decreasing is not restrictive. The "strictly" and "twice continuously differentiable" assumptions are not essential. However relaxing them does not provide much additional insight while it complicates the analysis and the notation. Finally we assume that the properties hold over the interval $[0, \infty[$ to insure the existence of a solution to the problem (P_s) with a finite objective function value. This assumption can be relaxed whenever it is not necessary to guarantee the preceeding condition.

Suppose that we add in the problem (P_s) the constraint that the delay assignment satisfies $D_1 \leq \cdots \leq D_V$. Using theorem 2.1–c the resulting auxiliary problem, which will be called (A_s), can be written as:

$$(A_s) \qquad \begin{array}{c} \min \quad S(\vec{D}) \\ \vec{D} \in H \\ D_1 \leq \cdots \leq D_V \end{array}$$

where:

$$H = \left\{ \vec{D} \;\middle|\; \frac{R_1 + \cdots + R_i}{\mu - R_1 - \cdots - R_i} - R_1 D_1 - \cdots - R_i D_i \leq 0 \quad i = 1, \ldots, V \right\} \tag{3.8}$$

In (A_s) the equality in Kleinrock's conservation equation has been relaxed. This can be done w.l.o.g. because the form of H and the assumptions that the cost functions are strictly convex and non–decreasing guarantee that any optimal solution to (A_s) must in fact achieve equality in Kleinrock's conservation equation. Indeed this can be seen as follows. Suppose

that (A_s) has an optimal solution $\vec{\tilde{D}}$ in which strict inequality holds in the last feasibility constraint. As $\tilde{D}_V$ appears only in the last feasibility constraint we must have $\tilde{D}_V = \tilde{D}_{V-1}$ for otherwise it would be possible to reduce $\tilde{D}_V$ without violating the feasibility constraints. This would result in a better solution since $C_V(\cdot)$ is strictly convex and non–decreasing. If $\tilde{D}_V = \tilde{D}_{V-1}$ it follows that the $V - 1^{th}$ feasibility constraint must be satisfied with strict inequality (indeed if the $V - 1^{th}$ constraint is satisfied with equality v.c. $V - 1$ must have full preemptive priority over v.c. V, in which case we cannot have $\tilde{D}_V = \tilde{D}_{V-1}$). Now if both the feasibility constraints $V - 1$ and V are satisfied with strict inequality, and since the v.c.'s $V - 1$ and V appear only in these feasibility constraints it follows that we can decrease $\tilde{D}_{V-1}$ and $\tilde{D}_V$, thus achieving a better solution, unless $\tilde{D}_{V-2} = \tilde{D}_{V-1} = \tilde{D}_V$. Continuing this way we eventually obtain that $\tilde{D}_1 = \cdots = \tilde{D}_V$ and that all the feasibility constraints are satisfied with strict inequality. This is clearly a contradiction since then everybody's delay can be reduced, achieving a better solution. From a practical point of view relaxing the equality in Kleinrock's conservation equation corresponds to relaxing assumption (A.2.2.1), namely allowing the server to stand idle in front of waiting packets. Hereafter a delay assignment satisfying the feasibility constraints when the left hand side in Kleinrock's conservation equation is allowed to be larger than the right hand side will be called a weakly feasible delay assignment. Also, to underline the fact that feasible delay assignments achieve equality in Kleinrock's conservation equation, we will now say that these assignments are strictly feasible.

It is not difficult to see that the constraints defining H are convex. Also it is clear in view of assumption (A.2.1.3) that H contains a strict interior point satisfying the ordering $D_1 \leq \cdots \leq D_V$ (for example it is readily verified that the point $D_1 = \cdots = D_V = 2/(\mu - R_1 - \cdots - R_V)$ is such a point). Hence identifying the constraints defining H with the constraints g_j in the formulation of (CPP) and the set defined by the constraint $D_1 \leq \cdots \leq D_V$ with the set X in (CPP), it follows that (A_s) fits the framework of (CPP), and that there exists at least one Lagrange multiplier vector for this problem. In addition it follows from theorem 3.1 that:

$$\min_{\vec{D}\in H,\ D_1\leq\cdots\leq D_V} S(\vec{D}) \quad = \quad \max_{\vec{\lambda}\geq 0} \quad q(\vec{\lambda}) \tag{3.9}$$

where $q(\vec{\lambda})$, the dual functional, is:

$$q(\vec{\lambda}) = \min_{D_1 \leq \cdots \leq D_V} S(\vec{D}) + \sum_{i=1}^{V} \lambda_i \left[\frac{R_1 + \cdots + R_i}{\mu - R_1 - \cdots - R_i} - R_1 D_1 - \cdots - R_i D_i \right] \tag{3.10}$$

Rearranging the terms $q(\vec{\lambda})$ can be written as:

$$q(\vec{\lambda}) = \min_{D_1 \leq \cdots \leq D_V} \sum_{i=1}^{V} \left[C_i(D_i) - (\lambda_i + \cdots + \lambda_V) R_i D_i \right] + K(\vec{\lambda}) \tag{3.11}$$

where we have defined for convenience:

$$K(\vec{\lambda}) = \sum_{i=1}^{V} \lambda_i \left(\frac{R_1 + \cdots + R_i}{\mu - R_1 - \cdots - R_i} \right) \tag{3.12}$$

It is not difficult to see that $\vec{\lambda}$ achieving the maximum in the right hand side of equation (3.9) is unique. Also, as the function to be minimized in the left hand side of equation (3.9) is strictly convex, it is clear that the optimal delay assignment is unique.

A considerable insight on the properties of the optimal delay assignment can be gained by using a dynamic programming argument which we now present.

Equation (3.11) can be written as:

$$\begin{aligned} q(\vec{\lambda}) = \min_{D_1 \leq \cdots \leq D_{V-1}} & \left(\sum_{i=1}^{V-1} \left[C_i(D_i) - (\lambda_i + \cdots + \lambda_V) R_i D_i \right] \right. \\ & \left. + \min_{D_{V-1} \leq D_V} \left[C_V(D_V) - \lambda_V R_V D_V \right] \right) + K(\vec{\lambda}) \end{aligned} \tag{3.13}$$

Note that D_V appears only in the second term under square brackets. Define:

$$A_V = \operatorname*{argmin}_{D} \left[C_V(D) - \lambda_V R_V D \right] \tag{3.14}$$

It follows from the definition of A_V that the D_V which achieves the minimum in equation (3.11) is given by (c.f. Figure 3.1):

$$D_V = \max(A_V, D_{V-1}) \tag{3.15}$$

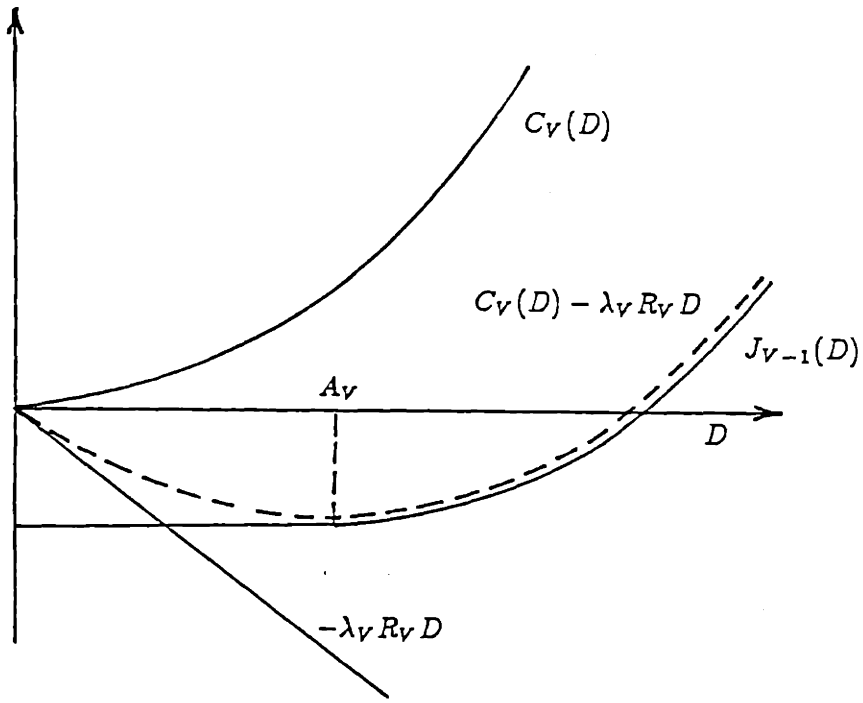


Figure 3.1

Now, define:

$$J_{V-1}(D_{V-1}) = \min_{D_{V-1} \le D_V} \left[C_V(D_V) - \lambda_V R_V D_V \right] \tag{3.16}$$

Using equations (3.14) and (3.15) $J_{V-1}(\cdot)$ can be written as:

$$J_{V-1}(D_{V-1}) = \begin{cases} C_V(A_V) - \lambda_V R_V A_V, & \text{if } D_{V-1} < A_V \\ C_V(D_{V-1}) - \lambda_V R_V D_{V-1}, & \text{if } D_{V-1} \ge A_V \end{cases} \tag{3.17}$$

Note that $J_{V-1}(\cdot)$ is a convex non–decreasing function. Moreover $J_{V-1}(D)$ is constant for $D \le A_V$ and strictly convex for $D \ge A_V$.

Using equation (3.16), equation (3.13) becomes:

$$\begin{aligned} q(\vec{\lambda}) &= \min_{D_1 \le \cdots \le D_{V-1}} \left(\sum_{i=1}^{V-1} \left[C_i(D_i) - (\lambda_i + \cdots + \lambda_V) R_i D_i \right] + J_{V-1}(D_{V-1}) \right) + K(\vec{\lambda}) \\ &= \min_{D_1 \le \cdots \le D_{V-2}} \left(\sum_{i=1}^{V-2} \left[C_i(D_i) - (\lambda_i + \cdots + \lambda_V) R_i D_i \right] \right. \\ &\quad \left. + \min_{D_{V-2} \le D_{V-1}} \left[C_{V-1}(D_{V-1}) - (\lambda_{V-1} + \lambda_V) R_{V-1} D_{V-1} + J_{V-1}(D_{V-1}) \right] \right) \\ &\quad + K(\vec{\lambda}) \end{aligned} \tag{3.18}$$

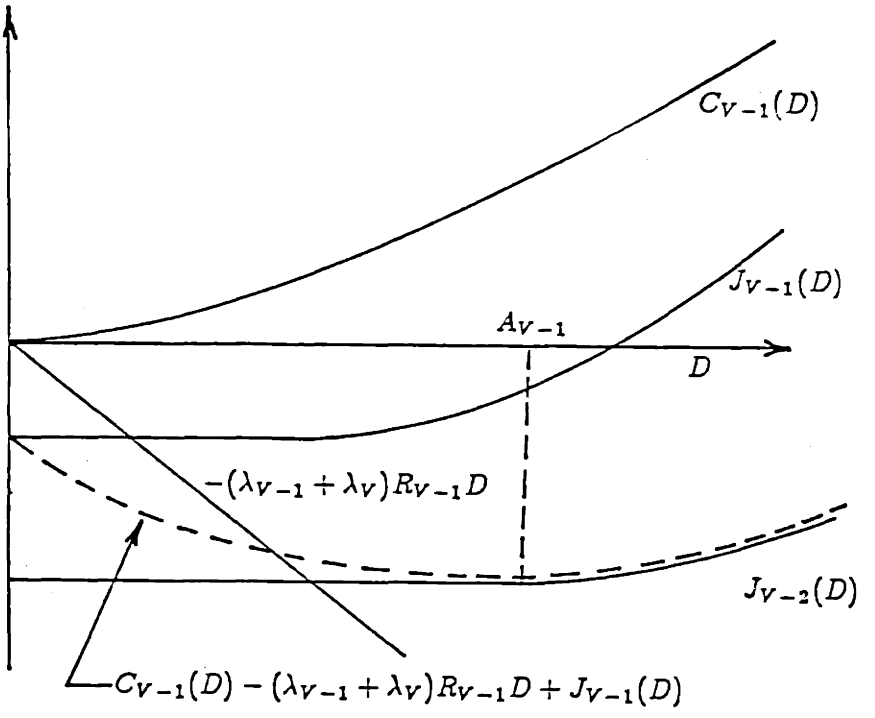


Figure 3.2

It turns out because of the properties of $J_{V-1}(\cdot)$ that the argument developed above can be applied to equation (3.18). Namely, defining:

$$A_{V-1} = \operatorname*{argmin}_D \Big[C_{V-1}(D) - (\lambda_{V-1}+\lambda_V)R_{V-1}D + J_{V-1}(D)\Big] \tag{3.19}$$

and:

$$J_{V-2}(D_{V-2}) = \min_{D_{V-2}\le D_{V-1}} \Big[C_{V-1}(D_{V-1}) - (\lambda_{V-1}+\lambda_V)R_{V-1}D_{V-1} + J_{V-1}(D_{V-1})\Big] \tag{3.20}$$

we obtain (c.f. Figure 3.2):

$$D_{V-1} = \max(A_{V-1}, D_{V-2}) \tag{3.21}$$

and:

$$J_{V-2}(D) = \begin{cases} C_{V-1}(A_{V-1}) - (\lambda_{V-1}+\lambda_V)R_{V-1}A_{V-1} + J_{V-1}(A_{V-1}), & \text{if } D < A_{V-1} \\ C_{V-1}(D) - (\lambda_{V-1}+\lambda_V)R_{V-1}D + J_{V-1}(D), & \text{if } D \ge A_{V-1} \end{cases} \tag{3.22}$$

Note that similarly as before $J_{V-2}(\cdot)$ is convex and non–decreasing. Moreover $J_{V-2}(D)$ is constant for $D \le A_{V-1}$ and is strictly convex for $D \ge A_{V-1}$.

This discussion can be generalized to obtain $D_1, \ldots, D_V$ and $J_1(\cdot), \ldots, J_{V-1}(\cdot)$. This is formalized in the following lemma.

Lemma 3.1: The delay assignment $\vec{D}$ achieving the minimum in the right hand side of equation (3.11) is unique and is given by:

$$D_i = \max(A_i, D_{i-1}) \qquad i = 1, \ldots, V, \quad D_0 = 0$$

where:

$$A_i = \underset{D}{\operatorname{argmin}}\Big[C_i(D) - (\lambda_i + \cdots + \lambda_V)R_i D + J_i(D)\Big] \quad i = 1, \ldots, V$$

and where $J_V(D) = K(\vec{\lambda})$, and for $i = 1, \ldots, V-1$;

$$J_i(D) = \begin{cases} C_{i+1}(A_{i+1}) - (\lambda_{i+1} + \cdots + \lambda_V)R_{i+1}A_{i+1} + J_{i+1}(A_{i+1}), & \text{if } D < A_{i+1} \\ C_{i+1}(D) - (\lambda_{i+1} + \cdots + \lambda_V)R_{i+1}D + J_{i+1}(D), & \text{if } D \geq A_{i+1} \end{cases}$$

Moreover the corresponding value of the dual functional is:

$$q(\vec{\lambda}) = C_1(A_1) - (\lambda_1 + \cdots + \lambda_V)R_1 A_1 + J_1(A_1)$$

Proof:

The validity of the recursion has been established in the preceeding discussion. The uniqueness follows from the fact that the function to be minimized in the right hand side of equation (3.11) is strictly convex. Finally the last result of the lemma follows from the construction of the functions $J_i(\cdot)$, $i = 1, \ldots, V$.

Q.E.D.

For a given $\vec{\lambda}$ the delay assignment prescribed by lemma 3.1 is independent of the right hand side of the equations defining H. In other words varying the right hand side of the equations defining H away from 0 does not affect the assignment generated by lemma 3.1. This is because varying the right hand side of the constraints affects only the function $K(\vec{\lambda})$ which has no impact on the assignment generated by lemma 3.1. This may seem surprising at the first glance. Indeed since the optimal assignment is strictly feasible, it is clear that it must somehow depend on the right hand side of the constraints defining H.

The key words in the preceeding paragraph are "for a given $\vec{\lambda}$". The precise form of $K(\vec{\lambda})$ influences the optimal delay assignment because of its impact on the optimal $\vec{\lambda}$. That

is if $K(\vec{\lambda})$ varies the optimal $\vec{\lambda}$ in equation (3.9) will vary. This will cause the optimal delay assignment to vary as it depends on $\vec{\lambda}$.

From lemma 3.1 we see that if $A_i > A_{i+1}$ for some i then $D_i = D_{i+1}$ while if $A_i \leq A_{i+1}$ then A_i satisfies:

$$(\lambda_i + \cdots + \lambda_V) = \frac{1}{R_i} C_i'(A_i) \tag{3.23}$$

In words this means that v.c. i impacts v.c.'s ranked behind it as long as its threshold A_i is higher than theirs while if v.c. i does not impact v.c.'s ranked behind it then its threshold is independent of these v.c.'s. Also it may be seen that in the former case i's threshold depends on the cost function of the v.c.'s ranked behind it whose thresholds are lower than i's. This can be illustrated as follows. Suppose that $A_{i+2} > A_i > A_{i+1}$. Suppose also for simplicity that $D_{i-1} \leq A_{i+1}$. From lemma 3.1 A_i satisfies:

$$C_i'(A_i) - (\lambda_i + \cdots + \lambda_V) R_i + J_i'(A_i) = 0 \tag{3.24}$$

Because it is assumed that $D_{i-1} \leq A_{i+1}$ we have $D_i = A_i$. Also because $A_{i+1} < A_i$ we have $D_{i+1} = D_i = A_i$. Accordingly it follows that:

$$J_i'(A_i) = C_{i+1}'(A_i) - (\lambda_{i+1} + \cdots + \lambda_V) R_{i+1} + J_{i+1}'(A_i) \tag{3.25}$$

Also as $A_i < A_{i+2}$ it follows that $J_{i+1}'(A_i) = 0$. Using this fact and equations (3.24) and (3.25), we obtain:

$$C_i'(A_i) + C_{i+1}'(A_i) - (\lambda_i + \cdots + \lambda_V) R_i - (\lambda_{i+1} + \cdots + \lambda_V) R_{i+1} = 0 \tag{3.26}$$

Clearly A_i depends on the cost function of v.c. $i+1$ which has a lower threshold than v.c. i but not on v.c. $i+2$ which has a higher threshold, as expected.

Suppose that the $\vec{\lambda}$ achieving the maximum in the right hand side of equation (3.9) is such that $A_i > A_{i+1}$ for some i. Then lemma 3.1 guarantees that $D_i = D_{i+1}$. This indicates that if we interchange the positions of v.c. i and $i+1$, thereby obtaining a new (A_s) problem, the optimal value of the new (A_s) problem cannot be higher than the optimal value of the initial (A_s) problem. Indeed we may at most again obtain $D_i = D_{i+1}$ which would indicate that ordering i in front of $i+1$ or vice–versa does not matter. However in

general we may expect that the new assignment will be such that $D_{i+1} < D_i$, in which case a strict improvement will have been obtained. This intuitively leads one to believe that the optimal solution $\vec{D}^*$ to the problem (P_s) must be such that in an (A_s) problem defined based on an ordering valid for the assignment $\vec{D}^*$ the condition $A_1 \leq \cdots \leq A_V$ should be satisfied for the optimal $\vec{\lambda}$. This intuition is more accurately described in the following theorem.

Theorem 3.3: Let $\vec{D}^*$ be a delay assignment satisfying $D_1^* \leq \cdots \leq D_V^*$. $\vec{D}^*$ is the optimal solution to the problem (P_s) if and only if in the problem (A_s) defined based on the ordering $D_1 \leq \cdots \leq D_V$;

1) $\vec{\lambda}^* \geq 0$.
2) $\vec{D}^*$ satisfies lemma 3.1 when $\vec{\lambda} = \vec{\lambda}^*$.
3) $\vec{D}^*$ is weakly feasible; i.e. $\vec{D}^* \in H$.
4) Complementary slackness is satisfied; i.e.:

$$\lambda_i^* \left[\frac{R_1 + \cdots + R_i}{\mu - R_1 - \cdots - R_i} - R_1 D_1^* - \cdots - R_i D_i^* \right] = 0 \qquad i = 1, \ldots, V$$

5) $A_1 \leq \cdots \leq A_V$.

This theorem is proven in the section 1 of Appendix B.

By slight abuse of language we refer to λ_j^* as the Lagrange multiplier associated with the j^{th} feasibility constraint in (P_s). It should be understood that in fact λ_j^* is the Lagrange multiplier associated with the j^{th} feasibility constraint in the problem (A_s) defined based on the ordering that $\vec{D}^*$ satisfies (namely in the theorem this ordering is $D_1 \leq \cdots \leq D_V$).

An immediate consequence of theorem 3.3 is the following result.

Corollary 3.1: Let $\vec{D}^*$ and $\vec{\lambda}^*$ satisfy theorem 3.3. Then for all i, $i = 1, \ldots, V$:

$$\frac{1}{R_i} C_i'(D_i^*) = \sum_{j=i}^{V} \lambda_j^*$$

This corollary is proven in the section 2 of Appendix B.

Based on the proof of theorem 3.3 we see that if for some ordering $D_1 \leq \cdots \leq D_V$ we obtain $A_i > A_{i+1}$ for some i in the associated (A_s) problem, then interchanging the positions of v.c.'s i and $i+1$ leads to a new (A_s) problem whose optimal value is strictly smaller than that of the initial (A_s) problem. This confirms our earlier intuition. This also provides a way of solving the problem (P_s)*. Indeed after a finite number of iterations an (A_s) problem satisfying $A_1 \leq \cdots \leq A_V$ must be found. At this point the problem (P_s) is solved since theorem 3.3 guarantees that the solution to the (A_s) problem is also the solution to the problem (P_s).

An efficient way of solving an (A_s) problem consists of using the dynamic programming approach (see for example [Be76]). The variables are discretized and the functions $J_i(\cdot)$, $i = 1, \ldots, V-1$ are computed recursively, starting from $J_{V-1}(\cdot)$. This involves a substantial amount of computation but it is still practical if the number of v.c.'s is not large. It may be noted that the dynamic programming problem is particularly nice; perfect state information prevails, there is no noise and the state contains only one variable.

Solving a (P_s) problem via solving a succession of (A_s) problems is however not very attractive. The main drawback of this approach is that the number of (A_s) problems to be solved may grow exponentially with the number of v.c.'s. In fact, since the work required for solving an (A_s) problem is still considerable, this approach would rapidly become inpractical. It turns out that solving directly a (P_s) problem is easier than solving an (A_s) problem. This fact is very surprising. Intuitively we should have expected that specifying the ordering would simplify the problem. What is happening is that in general we will specify a wrong ordering, so that we will not obtain $A_1 \leq \cdots \leq A_V$. When $A_i > A_{i+1}$ it is more difficult to compute A_i than when $A_i < A_{i+1}$ because the term $J_i'(A_i)$ does not vanish. Potentially if our guess for the ordering is very poor we can obtain $A_1 > A_2 > \cdots > A_V$, in which case the amount of computation required to determine $A_1, \ldots, A_V$ can become prohibitive. By solving directly the problem (P_s) the computational burden resulting from a wrong choice of ordering can be eliminated.

* This may be viewed as a simplex–like method where orderings correspond to extreme points.

In the linear cost case: i.e., when $C_i(D_i) = c_i D_i, \quad i = 1, \cdots, V$, it has been found (see for example [Ja68]) that the optimal discrete priority assignment consists of assigning the priorities in order of decreasing unit cost to rate ratio. That is for some re–labelling of the v.c's the sequence of increasing delays $D_1 \leq \cdots \leq D_V$ (D_1 corresponding to the v.c. having the highest priority, D_V to the v.c. having the lowest) corresponds to the sequence of decreasing ratios:

$$\frac{c_1}{R_1} \geq \cdots \geq \frac{c_V}{R_V} \tag{3.27}$$

On the other hand it follows from condition (1) of theorem 3.3 and from corollary 3.1 that the optimal solution to the problem (P_s) satisfies:

$$\frac{1}{R_1} C_1'(D_1^*) \geq \frac{1}{R_2} C_2'(D_2^*) \geq \cdots \geq \frac{1}{R_V} C_V'(D_V^*) \tag{3.28}$$

It is a well–known result of optimization that if $\vec{D}^*$ minimizes $S(\vec{D}^*)$ it must minimize the linearization of $S(\vec{D}^*)$ around $\vec{D}^*$. Comparing equations (3.27) and (3.28) we see that, indeed, $\vec{D}^*$ satisfies this condition.

We now introduce the notion of system–functional priority group. A system–functional priority group, denoted S_γ, is a collection of v.c.'s and is characterized by a number "γ". A v.c., say i, belongs to the system–functional priority group S_γ if $\frac{1}{R_i} C_i'(D_i) = \gamma$. We say that a system–functional priority group is higher, or of higher priority, than another one if its characteristic number is greater. System–functional priority groups defined based on the optimal solution to the problem (P_s) have several properties. Some of these properties are summarized below.

Let $\vec{D}^*$ and $\vec{\lambda}^*$ satisfy theorem 3.3 and consider the system–functional priority groups defined based on the assignment $\vec{D}^*$. Then the following holds:

1) All the v.c.'s in a system–functional priority group have full preemptive priority over the v.c.'s in lower system–functional priority groups and are of lowest priority compared to the v.c.'s in higher system–functional priority groups.
2) If i and k belong to the same system–functional priority group and j is such that $D_i^* \leq D_j^* \leq D_k^*$, then j also belongs to the group.

3) If a system–functional priority group is composed of v.c.'s $i,\ldots,l$ (with, of course, $D_i^* \leq \cdots \leq D_l^*$) then $\lambda_i^* = \cdots = \lambda_{l-1}^* = 0$ and $\lambda_l^* > 0$.

4) Let γ be the characteristic number of the system functional priority group of v.c. i. then: $\gamma = \sum_{j=i}^{V} \lambda_j^*$.

These properties are easily established. We leave this task to the reader.

Note that the property (1) does not imply that within a system–functional priority group a v.c. cannot have full preemptive priority over another v.c. Indeed this may happen in a degenerate situation.

The characteristic number is a very useful figure. It indicates the importance that the v.c.'s in a system–functional priority group attach to their delay. In fact we can view the characteristic number of a system–functional priority group as the marginal delay cost per unit rate for the v.c.'s in the group.

3.3.2 An algorithm solving the problem in the system–oriented approach.

Suppose that we guess that the optimal solution to a given problem (P_s) has only one system–functional priority group. Accordingly it follows that $\lambda_i^* = 0, \quad i = 1,\ldots,V-1$, $\lambda_V^* > 0$ and that the optimal assignment satisfy:

$$\frac{1}{R_i} C_i'(D_i^*) = \lambda_V^* \quad i = 1,\ldots,V \tag{3.29}$$

Based on equation (3.29) we can define D_i, $i = 1,\ldots,V$, as a function of λ_V as follows:

$$D_i = [C_i']^{-1}(R_i \lambda_V) \tag{3.30}$$

where for $i = 1,\ldots,V$ the function $[C_i']^{-1}(\cdot)$, the inverse of $C_i'(\cdot)$, is well–defined because $C_i(\cdot)$ is strictly convex and non–decreasing.

Since the optimal solution is strictly feasible we know that it satisfies Kleinrock's conservation equation. This suggests the following method for finding the optimal delay assignment. Defining the delay asssignment as a function of λ_V as in equation (3.30), and starting with $\lambda_V > 0$ sufficiently small to insure that Kleinrock's conservation is initially violated, we can simply increase λ_V until equality is achieved. Note that the existence of a

λ_V for which equality is achieved is guaranteed by the assumption that the functions $C_i(\cdot)$, $i = 1, \ldots, V$, are strictly convex and non–decreasing over $[0, \infty[$.

Let λ_V^* be the λ_V for which equality is achieved in Kleinrock's conservation equation and let $\vec{D}^*$ be the corresponding delay assignment. Given $\vec{D}^*$, any ordering valid for it can be used to define the remaining $V-1$ feasibility constraints. For example assuming w.l.o.g. that $\vec{D}^*$ satisfies $D_1^* \leq \cdots \leq D_V^*$ (this assumption can always be enforced by re–labelling appropriately the v.c.'s) we can define the $V-1$ remaining feasibility constraints using the ordering $D_1 \leq \cdots \leq D_V$; namely these constraints would then be:

$$\begin{aligned} R_1 D_1 &\geq \frac{R_1}{\mu - R_1} \\ \vdots \quad &\geq \quad \vdots \\ R_1 D_1 + \cdots + R_{V-1} D_{V-1} &\geq \frac{R_1 + \cdots + R_{V-1}}{\mu - R_1 - \cdots - R_{V-1}} \end{aligned} \tag{3.31}$$

If $\vec{D}^*$ satisfies these constraints we can conclude that our initial guess was correct and that $\vec{D}^*$ is the optimal solution to the problem (P_o). Indeed this follows from the fact that $\vec{D}^*$ and $\vec{\lambda}^* = (0, \ldots, 0, \lambda_V^*)$ then satisfy all the conditions of theorem 3.3. This can be easily checked. Condition (1) is satisfied because $\lambda_i^* = 0$, $i = 1, \ldots, V-1$, and $\lambda_V^* > 0$. Condition (3) is satisfied by assumption. Also condition (4) is satisfied since only $\lambda_V^* \neq 0$ and since the equation corresponding to λ_V^* is by construction satisfied with equality. By definition of A_V we must have:

$$C_V'(A_V) - \lambda_V^* R_V = 0 \tag{3.32}$$

As $C_V'(D_V^*) = R_V \lambda_V^*$ it follows from the preceeding equation that $A_V = D_V^*$. Similarly A_{V-1} satisfies:

$$C_{V-1}'(A_{V-1}) - (\lambda_{V-1}^* + \lambda_V^*) R_{V-1} + J_{V-1}'(A_{V-1}) = 0 \tag{3.33}$$

It is easily checked that the preceeding equation, together with the facts that $\lambda_{V-1}^* = 0$ and that $J_{V-1}'(D_{V-1}) = 0$ (because $D_{V-1} \leq D_V$), implies that $A_{V-1} = D_{V-1}$. Continuing this way we get:

$$A_1 = D_1^* \leq \cdots \leq A_V = D_V^* \tag{3.34}$$

It follows immediately from this equation that conditions (2) and (5) of theorem 3.3 are also satisfied.

In general however $\vec{D}$ as determined above will not satisfy all the feasibility constraints in equations (3.31). This will happen when the optimal solution to the problem (P_s) contains more than one system–functional priority group. One possibility of insuring that $\vec{D}$ satisfies all the feasibility constraints consists of increasing λ_V until all these constraints are satisfied. In this case equality will be achieved in some constraint $i < V$ while the V^{th} constraint will be satisfied with strict inequality. Of course the snag in this approach is that as $\lambda_V > 0$, and as the V^{th} constraint is not satisfied with equality, complementary slackness is not maintained. Intuitively the solution is, for the re–labelling of the v.c.'s corresponding to the sequence of increasing delays, to let $D_1, \ldots, D_i$ unchanged but to reduce $D_{i+1}, \ldots, D_V$ until the V^{th} constraint is satisfied with equality. This can be done by using λ_i. Indeed if only λ_i and λ_V are non–zero it follows from corollary 3.1 that the optimal assignment satisfies:

$$\frac{1}{R_j} C_j'(D_j^*) = \begin{cases} \lambda_i^* + \lambda_V^*, & j \leq i \\ \lambda_V^*, & j > i \end{cases} \tag{3.35}$$

Based on this equation we can define similarly as before D_j, $j = 1, \ldots, V$, as a function of λ_i and λ_V as follows:

$$D_j(\lambda_i, \lambda_V) = \begin{cases} [C_j']^{-1}\big(R_j(\lambda_i + \lambda_V)\big), & j \leq i \\ [C_j']^{-1}(R_j \lambda_V), & j > i \end{cases} \tag{3.35}$$

Clearly by selecting λ_i such that $\lambda_V^{old} = \lambda_i + \lambda_V^{new}$, $D_1, \ldots, D_i$ are unchanged. However by selecting $\lambda_V^{new} < \lambda_V^{old}$ it is possible to achieve equality in the last constraint. Note that increasing λ_i is allowed since the i^{th} constraint is satisfied with equality. Also it is not hard to guess that the highest system–functional priority group is the set containing the v.c.'s $1, \cdots, i$.

It may well be that the assignment resulting from the preceeding procedure is still unfeasible because nothing guarantees that this assignment satisfies the feasibility constraints $i+1, \ldots, V-1$. However one may see that this situation can be handled in a similar way as before. Namely we can, in a first step, increase λ_V until feasibility is achieved while adjusting λ_i so that $D_1, \ldots, D_i$ stay unchanged and then, in a second step, increase $\lambda_{i'}$ and

decrease λ_V until the V^{th} constraint is satisfied with equality, where i', $i < i' < V$, is the index of the highest constraint for which equality was achieved in the first step.

We now present an algorithm solving the problem (P_s). This algorithm, called Alg_P_s, is essentially the generalization of the preceeding discussion.

Alg_P_s: This algorithm produces the optimal solution to the problem (P_s).

Let $T \in N$ be a given number.

1) Initially arbitrarily label the v.c.'s $1, \cdots, V$ and set $T = 1$, $\vec{\lambda} = 0$, $\vec{D} = 0$.
2) For $i < T$ let D_i stay constant.

   For $i \geq T$ let:

$$D_i = [C_i']^{-1}(R_i\gamma)$$

   provided that this expression is well–defined. Otherwise if γ is such that $[C_i']^{-1}(R_i\underline{\gamma}) = 0$ for some $\underline{\gamma} > \gamma$ we set $D_i = 0$ and if $[C_i']^{-1}(R_i\overline{\gamma}) = \infty$ for some $\overline{\gamma} < \gamma$ we set $D_i = \infty$.
3) Find the minimum γ such that the following constraints are satisfied, where, for each γ, the labelling of the v.c.'s $T, \ldots, V$ is defined by the ordering of the delays:

$$\begin{aligned} R_1 D_1 + \ldots + R_T D_T &\geq \frac{R_1 + \cdots + R_T}{\mu - R_1 - \cdots - R_T} \\ \vdots \qquad &\geq \qquad \vdots \\ R_1 D_1 + \cdots + R_V D_V &\geq \frac{R_1 + \cdots + R_V}{\mu - R_1 - \cdots - R_V} \end{aligned}$$

4) Let i' be the highest constraint satisfied with equality. Set:
   a) $\lambda_{i'} \leftarrow \gamma$
   b) $\lambda_{T-1} \leftarrow \lambda_{T-1} - \gamma$
   c) $D_j \leftarrow [C_j']^{-1}(R_j\gamma) \quad j = T, \ldots, i'$
   d) $T \leftarrow i' + 1$
5) if $T > V$ stop, else go to step 2.

The proof of correctness of this algorithm is given in the section 3 of Appendix B.

Since T is initialized at 1 and since it increases by at least 1 each time step (4) is executed, it follows that the algorithm terminates in at most V steps. In fact it is not difficult to see that the algorithm requires one iteration for each system–functional priority group of the assignment that it produces.

Most of the computational burden of the algorithm occurs in step (3). Indeed it is clear that the work required in the other steps is very small. We now investigate further the work involved in step (3).

For a given T, $1 \leq T \leq V$, Define:

$$i_T(\gamma) = \underset{i \in \{T,\ldots,V\}}{\operatorname{argmin}} [C_i']^{-1}(R_i\gamma) \tag{3.37}$$

Also define for $l = T+1,\ldots,V$:

$$i_l(\gamma) = \underset{i \in \{T,\ldots,V\},\ i \neq i_T(\gamma),\ldots,i_{l-1}(\gamma)}{\operatorname{argmin}} [C_i']^{-1}(R_i\gamma) \tag{3.38}$$

and:

$$f_l(\gamma) = R_1 D_1 + \cdots + R_{T-1}D_{T-1} + R_{i_T(\gamma)}[C'_{i_T(\gamma)}]^{-1}(R_{i_T(\gamma)}\gamma) + \cdots + R_{i_l(\gamma)}[C'_{i_l(\gamma)}]^{-1}(R_{i_l(\gamma)}\gamma) - \frac{R_1 + \cdots + R_{T-1} + R_{i_T(\gamma)} + \cdots + R_{i_l(\gamma)}}{\mu - R_1 - \cdots - R_{T-1} - R_{i_T(\gamma)} - \cdots - R_{i_l(\gamma)}} \tag{3.39}$$

In words, equations (3.37) and (3.38) rank the v.c.'s $T,\ldots,V$ in order of increasing delay, where the delays are given as a function of γ as in step (2) of Alg_P_s. By construction it follows that for any γ, $i_T(\gamma),\ldots,i_V(\gamma)$ satisfy:

$$D_{i_T(\gamma)} \leq \cdots \leq D_{i_V(\gamma)} \tag{3.40}$$

The $f_l(\cdot)$, $l = T,\ldots,V$, correspond to the slackness in the equations of step (3). Indeed it is not difficult to see that for $l = T,\ldots,V$, $f_l(\cdot)$ is the difference between the left hand side and the right hand side of the constraint in step (3) whose left hand side contains l terms.

Using these definitions it follows that we can formulate the objective of step (3) as that of finding γ such that for some $\tilde{l}$, $T \leq \tilde{l} \leq V$:

$$f_{\tilde{l}}(\gamma) = 0 \tag{3.41}$$

while for $l \neq \tilde{l}$, $T \leq l \leq V$:

$$f_l(\gamma) \geq 0 \tag{3.42}$$

or in a more compact way defining:

$$G_s(\gamma) = \min_{l \in \{T,\ldots,V\}} f_l(\gamma) \tag{3.43}$$

step (3) can be stated as: find γ such that $G_s(\gamma) = 0$. From this viewpoint step (3) deals with the classical problem of finding the zeros of a function. Moreover it is not difficult to see that $G_s(0) < 0$, $G_s(\infty) = \infty$ and that $G_S(\cdot)$ is continuous and increasing. It follows from these facts that $G_s(\cdot)$ has exactly one zero, as must be the case since the γ achieving the minimum in step (3) is clearly unique.

The computation of $G_s(\gamma)$ for a given γ requires the evaluation of equations (3.37), (3.38), (3.39) and (3.43). The ordering task implicit in equations (3.37) and (3.38) can be accomplished in $O(V\log(V))$ computations. This ordering being done, $O(V)$ computations are required to evaluate equation (3.39). Also equation (3.43) requires at most V comparisons. Hence assuming that the assignment produced by Alg_P_s contains p system-functional priority groups, and that finding the zero of $G_s(\cdot)$ requires G evaluations of the function, it follows that the overall number of computations required by Alg_P_s is proportional to $VpG\log(V)$.

This is pleasantly surprising. From chapter 2 we know that the feasible delay set is the convex hull of V! distinct extreme points. Thus we could have suspected that the computations would grow exponentially with the size of the problem. This does not seem to be the case.

The factor G may not always be available. However, this does not infirm the preceeding conclusion. Indeed this can be justified as follows. Because $G_s(\cdot)$ is continuous and increasing, and because $G_s(\infty) = \infty$ it follows that we can always find γ_{max} such that $G_s(\gamma_{max}) > 0$. γ_{max} depends on the rate assignment and on the precise form of the cost functions but not on the number of v.c.'s. As $G_s(0) < 0$ this means that the zero of $G_s(\cdot)$ lies in the interval $[0, \gamma_{max}]$. The zero can be located using for example the well-known secant method. The rate of convergence of this method is linear but it does not depend

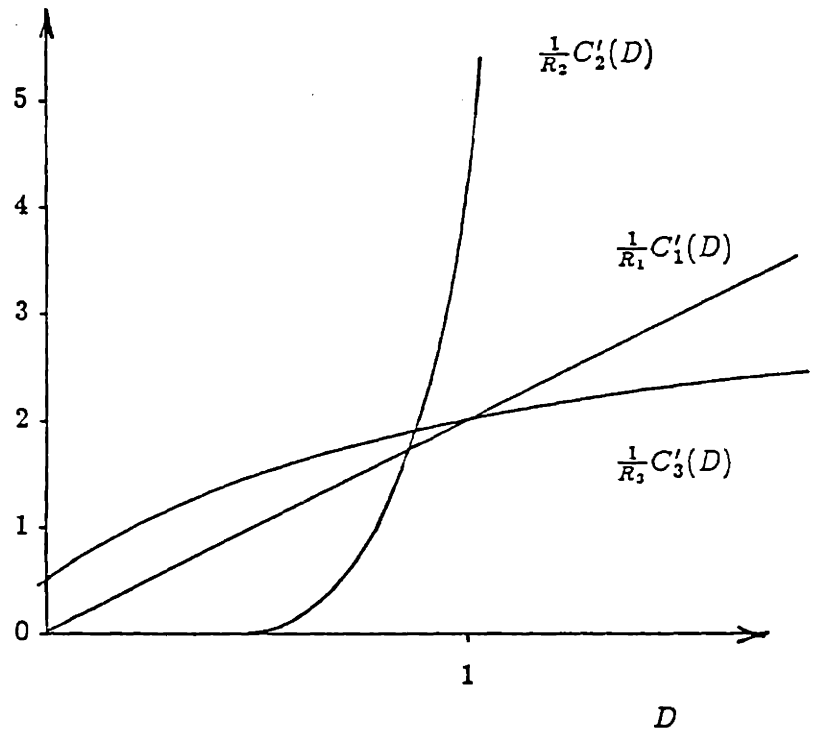


Figure 3.3

on the number of v.c.'s. This means that the work required to solve the problem is still proportional to $VpG\log(V)$, where now G characterizes the number of iterations required by the secant method to locate with a given accuracy a zero in the interval $[0, \gamma_{max}]$.

We conclude this section with an example illustrating the functioning of Alg_P_s.

Example 3.1: Suppose that there are three v.c.'s, characterized as follows:

$$C_1(D_1) = 0.1D_1^2, \qquad R_1 = 0.1$$
$$C_2(D_2) = 0.1\exp\big[10(D_2 - 1)\big], \qquad R_2 = 0.2$$
$$C_3(D_3) = 0.3D_3(.5 + \sqrt{D_3}), \qquad R_3 = 0.3$$

The marginal delay cost per unit rate of the v.c.'s for this choice of cost functions are depicted in Figure 3.3. A little calculus gives:

$$[C_1']^{-1}(R_1\gamma) = 0.5\gamma$$
$$[C_2']^{-1}(R_2\gamma) = 1 + 0.1\ln(0.2\gamma)$$
$$[C_3']^{-1}(R_3\gamma) = \left(\frac{\gamma - 0.5}{1.5}\right)^2$$

We first assume that $\mu = 2$ and we go manually through Alg_P_s.

In the first iteration we obtain for $\gamma = 1.6504$:

$$D_1 = 0.82522 \quad D_2 = 0.88916 \quad D_3 = 0.58824 \tag{3.44}$$

The ordering is $D_3 \leq D_1 \leq D_2$ and, correspondingly, the constraints in step (3) are:

$$R_3 D_3 \geq \frac{R_3}{\mu - R_3} \tag{3.45}$$

$$R_1 D_1 + R_3 D_3 \geq \frac{R_1 + R_3}{\mu - R_1 - R_3} \tag{3.46}$$

$$R_1 D_1 + R_2 D_2 + R_3 D_3 \geq \frac{R_1 + R_2 + R_3}{\mu - R_1 - R_2 - R_3} \tag{3.47}$$

It is easily checked that for the assignment (3.44) equality is achieved in equation (3.45) while equations (3.46) and (3.47) are satisfied with strict inequality. Correspondingly in step (4) we set:

$$D_3 = 0.58824$$

$$\lambda_1 = 1.6504 \tag{3.48}$$

$$T = 2$$

In the second iteration only the v.c.'s 1 and 2 are considered. We find for $\gamma = 1.5187$:

$$D_1 = 0.75933 \quad D_2 = 0.88084 \tag{3.49}$$

The ordering is $D_1 \leq D_2$. Accordingly the constraints in step (3) are:

$$R_1 D_1 + R_3 D_3 \geq \frac{R_1 + R_3}{\mu - R_1 - R_3} \tag{3.50}$$

$$R_1 D_1 + R_2 D_2 + R_3 D_3 \geq \frac{R_1 + R_2 + R_3}{\mu - R_1 - R_2 - R_3} \tag{3.51}$$

where D_3 is as obtained in the last iteration. It is easily checked that the assignment (3.49) satisfies the two preceeding equations and that the last constraint is satisfied with equality. Consequently we set in step (4):

$$D_1 = 0.75933$$

$$D_2 = 0.88084$$

$$\lambda_1 = 1.6504 - 1.5187 = 0.1317$$

$$\lambda_3 = 1.5187$$

$$T = 4$$

Since $T > 3$ the algorithm stops.

In summary the optimal assignment is:

$$D_1 = 0.75933 \quad D_2 = 0.88084 \quad D_3 = 0.58824$$

$$\lambda_1 = 0.1317 \quad \lambda_2 = 0 \quad \lambda_3 = 1.5187$$

There are two system–functional priority groups. V.c. 3 is the only member of the first group while the second group contains v.c.'s 1 and 2.

Suppose now that $\mu = 1$. Going through the same sort of computation we obtain:

$$D_1 = 1.7857 \quad D_2 = 1.2500 \quad D_3 = 2.2781$$

$$\lambda_1 = 57.3401 \quad \lambda_2 = 0.5893 \quad \lambda_3 = 2.9821$$

Now there are three system–functional priority groups. V.c. 2 has full priority over v.c.'s 1 and 3 while v.c. 1 has full priority over v.c. 3.

When $\mu = 2$ the overall average delay is approximately 0.7. In this region the marginal delay cost per unit rate of v.c. 3 is larger than that of v.c.'s 1 and 2. This justifies why in this case v.c. 3 is given full priority over v.c.'s 1 and 2. When $\mu = 1$ the overall average delay increases to 1.5. In this region the marginal delay cost per unit rate of v.c. 2 becomes extremely important. This is why this v.c. is then given full priority over v.c.'s 1 and 2. We may also note that when $\mu = 1$ v.c. 3 is given the lowest priority. This results from the fact that although the marginal delay cost per unit rate of v.c. 3 is relatively high for modest delays, it does not increase as rapidly as the other ones.

3.4 The user–oriented approach.

As in the preceeding section we of course assume that assumptions (A.2.1)–(A.2.3) hold. However we now make different assumptions regarding the cost functions. Specifically we assume that:

(A.3.2) For all i, $i = 1, \ldots, V$, $C_i(\cdot)$ is non–negative, continuous and increasing over the interval $[0, \infty[$.

The assumptions that the cost functions are continuous and increasing have already been discussed in the preceeding section. The assumption that the cost functions are non–negative is not restrictive because it is in practice easy to enforce this assumption. Indeed the cost functions are usually bounded from below. This implies that by adding a sufficiently large constant to all the cost functions we can insure that the assumption is satisfied. Moreover it is not difficult to see that this artifice does not affect the optimal assignment, so that the problem with the modified cost functions is equivalent to the initial problem.

The main result that we present concerning the user–oriented approach is an algorithm solving the problem (P_u). This algorithm, called Alg_P_u, will be presented shortly. A surprising fact is that this algorithm is very similar to Alg_P_s. This is the consequence of some very fundamental relationships between the problems (P_s) and (P_u). In fact although the philosophies on which the problems (P_s) and (P_u) are based are different, these problems are very similar.

Our aim in the following discussion is to provide some insight on the relationships between the problems (P_s) and (P_u). For this purpose we temporarily assume that the cost functions are convex. This will allow us to utilize theorem 3.1 and 3.2. This assumption will be removed later.

The problem (P_u) can be solved by solving an hierarchy of nested problems. The first problem in the hierarchy, which will be called $(P_{u,1})$, minimizes the maximum cost. It can

be written as follows.

$$(P_{u,1}) \qquad \begin{array}{l} \min \quad \gamma_1 \\ C_i(D_i) \le \gamma_1 \quad i = 1, \ldots, V \\ \vec{D} \text{ weakly feasible} \end{array}$$

Note that we only require in this problem that the delay assignment be weakly feasible. This is not restrictive because given any weakly feasible delay assignment satisfying the constraints of $(P_{u,1})$ for some γ_1 we can always construct a strictly feasible delay assignment satisfying these constraints for the same γ_1 simply by decreasing the delay of some of the v.c.'s.

Because we assume that the cost functions are convex it follows that the set defined by the constraints $C_i(D_i) \le \gamma_1$, $i = 1, \ldots, V$, is convex. Indeed for any assignments $(\tilde{\vec{D}}, \tilde{\gamma}_1)$ and $(\hat{\vec{D}}, \hat{\gamma}_1)$ satisfying these constraints we have for any $\alpha \in [0,1]$, and for all i, $i = 1, \ldots, V$:

$$C_i(\alpha \tilde{D}_i + (1-\alpha)\hat{D}_i) \le \alpha C_i(\tilde{D}_i) + (1-\alpha) C_i(\hat{D}_i) \le \alpha \tilde{\gamma}_1 + (1-\alpha)\hat{\gamma}_1 \tag{3.53}$$

Also it is not difficult to see that there exists a weakly feasible delay assignment $\vec{D}$ and a γ_1 such that $C_i(D_i) < \gamma_1$ for all i, $i = 1, \ldots, V$ (for example it is readily verified that the assignment $D_1 = \cdots = D_V = 1/(\mu - R_1 - \cdots - R_V)$ and $\gamma_1 = 2 \max_{1 \le i \le V} C_i(D_i)$ satisfies these conditions). It follows from these facts and from theorem 3.1 that $(P_{u,1})$ fits the framework of (CPP) and that there exists at least one Lagrange multiplier vector for this problem. In addition it also follows from theorem 3.1 that $\gamma_1^* = q^*$, where γ_1^* is the optimal value of $(P_{u,1})$ and q^* is the optimal value of the problem:

$$\begin{array}{l} \max \quad q(\vec{\psi}) \\ \vec{\psi} \ge 0 \end{array} \tag{3.54}$$

where $q(\vec{\psi})$, the dual functional, is:

$$\begin{aligned} q(\vec{\psi}) &= \min_{\vec{D} \text{ weakly feasible},\ \gamma_1} \gamma_1 + \sum_{i=1}^{V} \psi_i [C_i(D_i) - \gamma_1] \\ &= \min_{\vec{D} \text{ weakly feasible},\ \gamma_1} \gamma_1 [1 - \sum_{i=1}^{V} \psi_i] + \sum_{i-1}^{V} \psi_i C_i(D_i) \end{aligned} \tag{3.55}$$

From this expression we can obtain a set of optimality conditions for the problem $(P_{u,1})$ (see [Be83] p. 4.43). These conditions are listed in the following lemma.

Lemma 3.2: $(\vec{D}^*, \gamma_1^*)$ and $\vec{\psi}^*$ are respectively an optimal solution to $(P_{u,1})$ and a Lagrange multiplier vector if and only if:

1) $\vec{\psi}^* \geq 0$.

2) $\vec{D}^*$ is weakly feasible.

3) $C_i(D_i^*) \leq \gamma_1^*$.

4) $\psi_i^* = 0$ if $C_i(D_i^*) < \max_{1 \leq j \leq V} C_j(D_j^*)$.

5) $\vec{D}^* = \text{argmin}_{\vec{D} \text{ weakly feasible}} \sum_{i=1}^{V} \psi_i^* C_i(D_i^*)$.

6) $\sum_{i=1}^{V} \psi_i^* = 1$.

This result is proven in the section 4 of Appendix B.

An immediate consequence of the preceeding lemma is that if $\vec{D}^*$ in an optimal solution of the problem $(P_{u,1})$ it must also be an optimal solution of the following (P_s) problem:

$$\min_{\vec{D} \text{ weakly feasible}} \sum_{i=1}^{V} \psi_i^* C_i(D_i) \tag{3.56}$$

This (P_s) problem has the particularity that only the v.c.'s having the worst delay cost are considered. This is because condition (4) insures that for all i, $i = 1, \ldots, V$, if v.c. i does not have the worst delay cost then the Lagrange multiplier ψ_i^* is zero. Hence we can conclude that an optimal solution to $(P_{u,1})$ is also an optimal solution to a (P_s) problem in which only the v.c.'s experiencing the worst delay cost are considered.

Let $\vec{D}^*$ be the optimal assignment for the problem (P_u) and let γ_1^* be the worst delay cost in this assignment (i.e., $\gamma_1^* = \max_{1 \leq i \leq V} C_i(D_i^*)$). We define $\mathcal{U}_{\gamma_1^*}$ as the set of v.c.'s having the worst delay cost in the assignment $\vec{D}^*$. That is v.c. i, $i = 1, \ldots, V$, is in $\mathcal{U}_{\gamma_1^*}$ if $C_i(D_i^*) = \gamma_1^*$.

It is not difficult to see that the v.c.'s in $\mathcal{U}_{\gamma_1^*}$ must have full priority over the other v.c.'s. Indeed if a v.c. $i \in \mathcal{U}_{\gamma_1^*}$ does not have full priority over a v.c. $j \notin \mathcal{U}_{\gamma_1^*}$ it follows that we can decrease D_i^* and increase D_j^* without violating the feasibility constraints. As the cost functions are increasing this means that for a sufficiently small variation we can

guarantee that the delay cost of v.c.'s i and j in the resulting assignment are strictly less than γ_1^*. As the other delay costs do not vary this implies that the lexicographic ordering of the delay costs in the resulting assignment is strictly less than that of $\vec{D}^*$, contradicting the optimality of $\vec{D}^*$. Note also that this argument does not depend on the convexity assumption.

Now let $\tilde{D}^*$ be an optimal solution to the problem $(P_{u,1})$ and let $\tilde{\gamma}_1$ be the corresponding value of the problem. Clearly by definition of the problem $(P_{u,1})$ we have $\tilde{\gamma}_1 = \gamma_1^*$. Also it is not difficult to see that $U_{\gamma_1^*}$ is the set of v.c.'s i satisfying $C_i(\tilde{D}_i) = \tilde{\gamma}_1$ and such that it is not possible to reduce $C_i(\tilde{D}_i)$ without either violating feasibility or increasing the delay another v.c. whose delay cost is also $\tilde{\gamma}_1$.

The second problem in the hierarchy, which is called $(P_{u,2})$, is defined as follows:

$$(P_{u,2}) \qquad \begin{aligned} &\min \quad \gamma_2 \\ &C_i(D_i) \leq \gamma_2 \quad i \notin U_{\gamma_1^*} \\ &C_i(D_i) = \gamma_1^* \quad i \in U_{\gamma_1^*} \\ &\vec{D} \text{ weakly feasible} \end{aligned}$$

That is $(P_{u,2})$ minimizes the second largest delay cost subject to feasibility and subject to not increasing the delay cost of any v.c. in $U_{\gamma_1^*}$. Note that because $U_{\gamma_1^*}$ and γ_1^* are unique $(P_{u,2})$ is well–defined.

Clearly the form of problem $(P_{u,2})$ is identical to that of problem $(P_{u,1})$. Indeed the only difference is that the v.c.'s in $U_{\gamma_1^*}$ are not considered anymore in problem $(P_{u,2})$ because their delay is known. Accordingly it is not difficult to see that the optimal solutions of problem $(P_{u,2})$ satisfy a very similar set of conditions as the optimal solutions of problem $(P_{u,1})$. Specifically similarly as in lemma 3.2 we find that any optimal assignment for the problem $(P_{u,2})$ minimizes $\sum_{i \notin U_{\gamma_1^*}} \psi_i^* C_i(D_i)$ over the set of delay assignments in which the v.c.'s not in $U_{\gamma_1^*}$ are of lowest priority as compared to the v.c.'s in $U_{\gamma_1^*}$, and where ψ_i^*, $i \notin U_{\gamma_1^*}$, is non–zero only if i experiences the second worst delay cost. Hence we can conclude similarly as in the case of the problem $(P_{u,1})$ that an optimal solution to the problem $(P_{u,2})$ is also an optimal solution to a (P_s) problem in which only the v.c.'s

experiencing the second largest delay cost are considered, and in which the v.c.'s in $\mathcal{U}_{\gamma_1^*}$ are of full priority over the v.c.'s not in $\mathcal{U}_{\gamma_1^*}$.

It should be clear that the preceeding discussion concerning $(P_{u,2})$ can be generalized into an inductive step. We leave this exercice to the reader. The important thing to notice is that the problems in the hierarchy are in fact (P_s) problems. The k^{th} problem is a (P_s) problem is which the v.c.'s experiencing the k^{th} worst delay cost are considered and in which the v.c.'s whose delay was assigned in the previous problems have full priority over the v.c.'s whose delay has not yet been assigned.

We now present Alg_P_u. We define $C_i^{-1}(\cdot)$, $i = 1,\ldots,V$, as the inverse function of $C_i(\cdot)$. Clearly because by assumption $C_i(\cdot)$, $i = 1,\ldots,V$, is continuous and increasing $C_i^{-1}(\cdot)$ is well–defined.

Alg_P_u: This algorithm produces the optimal solution to the problem (P_u).

Let $T \in N$ be a given number.

1) Initially arbitrarily label the v.c.'s $1,\ldots,V$ and set $T = 1$, $\gamma = 0$, $\vec{D} = 0$.
2) For $i < T$ let D_i stay constant.

   For $i \geq T$ let:

$$D_i = C_i^{-1}(\gamma)$$

   provided that this expression is well–defined. Otherwise set $D_i = 0$.
3) Find the minimum γ such that the following constraints are satisfied, where the labelling of the v.c.'s $T,\ldots,V$ is defined by the ordering of the delays.

$$\begin{aligned} R_1 D_1 + \cdots + R_T D_T &\geq \frac{R_1 + \cdots + R_T}{\mu - R_1 - \cdots - R_T} \\ \vdots \qquad &\geq \qquad \vdots \\ R_T D_T + \cdots + R_V D_V &\geq \frac{R_1 + \cdots + R_V}{\mu - R_1 - \cdots - R_V} \end{aligned}$$

4) Let i' be the highest constraint satisfied with equality. Set:

   a) $D_j = C_j^{-1}(\gamma), \quad j = T, \ldots, i'$

   b) $T = i' + 1$

5) if $T > V$ stop, else go to step 2.

The proof of correctness of this algorithm is given in the section 5 of Appendix B.

The similarity between Alg_P_s and Alg_P_u is striking. Intuitively this can be explained as follows. Consider the kth iteration of Alg_P_u. Clearly this iteration solves the kth problem in the hierarchy. We know that this problem is a (P_s) problem in which only the v.c.'s having the kth worst delay cost are considered. Step (2) insures that all the v.c.'s whose delay has not yet been assigned are considered in this (P_s) problem because by construction the delay cost of all the v.c.'s are forced to be equal. Then as in Alg_P_s we increase in step (3) the delay of the v.c.'s until weak feasibility is achieved. However because all the costs are forced to be equal, some v.c.'s will have a delay needlessly high. These are the v.c.'s which are only involved in constraints satisfied with strict inequality. Clearly as in Alg_P_s the delay of these v.c.'s should not be determined by the current iteration, which is indeed the case as these v.c.'s are not considered in step (4). Also because of the form of the equations in step (3) we are guaranteed that the delays assigned in the kth iteration are of lowest priority as compared to the delays assigned in previous iterations, as expected.

Similarly as in the case of the system–oriented approach we may define the notion of user–functional priority group. A user–functional priority group, denoted $\mathcal{U}_\gamma$, is a collection of v.c.'s and is characterized by a number "γ". A v.c., say i, belong to the user–functional priority group $\mathcal{U}_\gamma$ if $C_i(D_i) = \gamma$. It is not difficult to see that the user–functional priority groups defined based on the optimal solution of the problem (P_u) share the properties (1) and (2) of the system–functional priority groups defined based on the optimal solution of the problem (P_s) listed on p. 46. Note also that the set $\mathcal{U}_{\gamma_1^*}$ that we have previously defined is in fact the highest user–functional priority group in the optimal solution of the problem (P_u).

It is easy to see that Alg_P_u requires one iteration for each user–functional priority group in the optimal solution of the problem (P_u). Also from the similarity between Alg_P_s

and Alg_P_u it is clear that the overall number of computations required by Alg_P_u is proportional to $VpG\log(V)$, p being the number of user–functional priority groups in the optimal solution to the problem (P_u) and G being the number of evaluations of the function $G_u(\cdot)$ required to find its zero. $G_u(\cdot)$ is defined in a completely analogous way as $G_s(\cdot)$.

3.5 Comments.

We have seen in the preceeding sections that the problems (P_s) and (P_u) are intimately related. We now present a result that underlines a fundamental relationship between these problems.

Consider the following (P_s) and (P_u) problems:

$$\min \sum_{i=1}^{V} C_i(D_i) \qquad \vec{D} \text{ weakly feasible} \tag{3.59}$$

$$\min \Im\Big(\frac{1}{R_1}C_1'(D_1),\ldots,\frac{1}{R_V}C_V'(D_V)\Big) \qquad \vec{D} \text{ weakly feasible} \tag{3.60}$$

We have the following result:

Theorem 3.4: Assume that the functions $C_i(\cdot)$, $i = 1,\ldots,V$, satisfy assumptions (A.3.1) and that the functions $C_i'(\cdot)$, $i = 1,\ldots,V$, satisfy assumptions (A.3.2). Then $\vec{D}^*$ is the optimal solution of the problem (P_s) given in (3.59) if and only if it is the optimal solution of the problem (P_u) given in (3.60).

The proof of this result is immediate in view of the fact that Alg_P_s and Alg_P_u respectively solve the problems (P_s) and (P_u) and in view of the similarity between the algorithms.

Note that it is not difficult to construct the cost functions $C_i(\cdot)$, $i = 1,\ldots,V$, so that $C_i(\cdot)$ satisfies assumptions (A.3.1) and $C_i'(\cdot)$ satisfies assumptions (A.3.2). In the remaining of this work we concentrate on generalizing the problem (P_s). Theorem 3.4 can always be used to adapt the results to a (P_u) formulation.

Finally it is worth noting that the results established in this chapter can be re–derived in the context in which only non–preemptive queuing is allowed. These modified results are presented without proof in the section 6 of Appendix B. The proofs are almost identical to those of the results presented in the chapter.

4 Delay assignment in an integrated services network.

In this chapter we investigate the problem of assigning delays to v.c.'s in an integrated services network. The chapter is divided in four sections. The goal of the first section is to define the problem precisely. In the second section we derive a set of conditions necessary and sufficient to guarantee the optimality of a delay assignment. Based on these conditions we construct in section 3 two distributed algorithms. The first algorithm is approximate in the sense that the delay assignment that it produces is not necessarily optimal. However, via a suitable choice of the parameters, the assignment produced can be brought as close to optimality as desired. The second algorithm always produces an optimal assignment but it requires more coordination than the first algorithm. Finally section 4 contains some comments on possible improvements and generalizations.

4.1 Problem Formulation.

We first introduce some terminology. A network is defined as a directed graph on which a collection of paths has been defined. The nodes of the network are labelled $1,\ldots,N$. They typically correspond to the switching machines. The links are the transmisssion facilities permitting the exchange of information between the nodes. The links are labelled $1,\ldots,L$. The network users are the v.c.'s. Each v.c. is an oriented path established between two different nodes. The first node on the path of a v.c. is called its home and the last node its destination. The home of v.c. i is abreviated $H(i)$ and its destination $D(i)$. It is assumed that the network contains V v.c.'s, labelled $1,\ldots,V$. $\mathcal{L}_i$ is defined as the set of links on the path of v.c. i, and $\mathcal{V}^l$ is defined as the set of v.c.'s sharing link l. Also, V^l is the total number of v.c.'s on link l, and L_i is the total number of links on the path of v.c. i*.

* In general a subscript is used to refer to a v.c. attribute while a superscript is used to refer to a link attribute. For example if x is a given variable x_i and x^l respectively refer to the variable in the context of v.c. i or of link l.

A quantity of paramount importance in this chapter is the end–to–end delay. The end–to–end delay of v.c. i, denoted D_i, is defined as follows:

$$D_i = \sum_{l \in \mathcal{L}_i} D_i^l \tag{4.1}$$

where D_i^l is the delay experienced by the v.c. on link l. In this equation the delays can be interpreted as instantaneous values or as long term averages. As in the preceeding chapters we will adhere to the latter interpretation.

We also straightforwardly extend the notation introduced in the preceeding chapters to the multiple link case simply by adding a superscript to the variables indicating to what link they belong. For example $\vec{w}^l$ denotes an ordering on link l, e_i^l denotes the priority group of v.c. i on link l, etc. We also define the functions $B^l(\cdot,\cdot,\cdot)$, $l = 1,\ldots,L$, similarly as the function $B(\cdot,\cdot,\cdot)$ as follows:

$$B^l(i,\vec{w}^l,\vec{R}) = \frac{\sum_{p \mid w_p^l \leq_i R_p}}{\mu^l - \sum_{p \mid w_p^l \leq_i R_p}} \quad 1 \leq i \leq V^l \tag{4.2}$$

Finally we again use the cost function:

$$S(\vec{D}) = \sum_{i=1}^{V} C_i(D_i) \tag{4.3}$$

where the functions $C_i(\cdot)$, $i = 1,\ldots,V$, are exactly as in the preceeding chapter in the system–oriented approach. In particular we assume that the cost functions still satisfy the assumptions (A.3.1).

The problem that will be investigated is the following. We refer to it as (NP_s).

$$(NP_s) \qquad \min_{\vec{D} \text{ feasible}} S(\vec{D})$$

This problem is a generalization of (P_s). Indeed if the network contains only one link the problem is clearly a (P_s) problem.

In the single link case we know what the set of feasible delays is. However we do not know what the set of feasible delays is in (NP_s). Of course our hope is to be capable of partitioning the feasible set on a link basis into smaller, mutually independent sets. In fact

what we expect is to be capable of decomposing the overall feasible set into a cartesian product in which each term would represent the set of delays realizable by a particular link; i.e., obtain a description of the feasible set of the form:

$$\left\{\vec{D} \text{ feasible}\right\} = \left\{D_i^1 \text{ feasible, } i \in \mathcal{V}^1\right\} \times \cdots \times \left\{D_i^L \text{ feasible, } i \in \mathcal{V}^L\right\} \qquad (4.4)$$

If equation (4.4) is true it means that, as far as the average delays are concerned, the links behave as if they were alone. Accordingly the feasible set of each link can be characterized using the results of the single link case. That is for a given link, say l, the feasibility constraints are, given a valid ordering on the link $\vec{w}^l$:

$$\sum_{p \mid w_p^l \leq i} R_p D_p^l \geq B^l(i, \vec{w}^l, \vec{R}) \quad i = 1, \ldots, V^l \qquad (4.5)$$

Unfortunately equation (4.4) is false. This is so because when priority queuing is allowed the output process of a single server queuing system is generally not Poisson. The consequence is that the input processes at the links which are fed by other links are generally not Poisson. The Poisson assumption is, however, essential in the derivation of the results of chapter 2.

Because of the dependency between the links it is extremely difficult to obtain a useful, yet accurate, characterization of the set of feasible delays in (NP_s). In fact even the simpler problem in which priorities are prohibited is hopeless. To overcome this difficulty we make the following approximation, which in fact is especially tailored toward allowing the use of equations (4.4) and (4.5).

(Ap.4.1) The arrival processes at each link are Poisson processes independent of each other and independent of the state of the link.

This approximation has also been made by Wong et al. [Wo82]. They have shown by simulation that, at least in the case of two very dissimilar priority schemes, the impact of the approximation on the estimation of the delay is minor. We believe that this result is typical of most priority schemes. Some other results that tend to support this assumption

have also been recently reported in [Ki83]. However further simulation should be carried out to verify this approximation more thoroughly in our context.

The preceeding assumption is seldom made in the literature. In general the celebrated Kleinrock's independence approximation ([Kl76] p. 322) is, for good reasons, much more popular. This approximation consists of pretending that each time a packet is received by a link its length is drawn from the exponential distribution independently of the state of the network and of the lengths that the packet may already have had. Of course this is absurd as the length of a packet is constant. From a practical point of view, however, this approximation has proven to be reasonable. The popularity of this approximation is justified by the fact that, together with the modest additional assumption of Poisson arrival at the input nodes, it is in general sufficient to obtain a network of quasi–reversible queues [Ke79]. At this point some extremely powerful results can be invoked. In particular, although the states of the queues are in general correlated, the probability distribution function for the overall state of the network takes under mild assumptions a particularly simple form.

We have departed from the tradition and not used Kleinrock's approximation for two reasons. The first reason is that when the use of priorities is allowed Kleinrock's approximation is not sufficient to guarantee the obtention of a network of quasi–reversible queues. Guaranteeing this still requires that an additional approximation, such as (Ap.4.1), be made. In fact it seems that in our formulation an approximation such as (Ap.4.1) is unavoidable. The second reason is that since our formulation depends only on average values the knowledge of the overall probability distribution function is anyway not needed. Indeed given the Poisson approximation the average delays on a link can be computed as if the link were alone, ignoring the correlation that may exist between the state of the link and the state of the other links. In this context Kleinrock's approximation loses much of its appeal.

In the remainder of this work we assume that each link satisfies assumptions (A.2.1)–(A.2.3), so that as a consequence the feasible set can be decomposed on a link basis, as in equations (4.4) and (4.5). Of course, as justifying assumption (A.2.1.1) requires that approximation (Ap.4.1) be made, it should always be remembered that our description of the set of feasible delays is only approximate.

4.2 Optimality Conditions.

In this section we derive a set of conditions necessary and sufficient to guarantee the optimality of a delay assignment for (NP_s). These conditions follow immediately from a simple general result which is now presented.

Theorem 4.1: Let X_i, $i = 1, \ldots, n$, be a convex set and let $f(\vec{x}_1, \ldots, \vec{x}_n) : X_1 \times \cdots \times X_n \rightarrow R$ be a convex differentiable function. Then $(\vec{x}_1^*, \ldots, \vec{x}_n^*)$ is an optimal solution of the problem:

$$\min f(\vec{x}_1, \ldots, \vec{x}_n)$$

$$x_i \in X_i, \ i = 1, \ldots, n$$

if and only if for each i, $i = 1, \ldots, n$, $\vec{x}_i^*$ is an optimal solution of the problem:

$$\min \ f(\vec{x}_1^*, \ldots, \vec{x}_{i-1}^*, \vec{x}_i, \vec{x}_{i+1}^*, \ldots, \vec{x}_n^*)$$

$$x_i \in X_i$$

This result is proven in the section 1 of appendix C.

For a given delay assignment, say $\vec{\tilde{D}}$, define:

$$C_i^l(D_i^l) = C_i\Big(\sum_{l' \in \mathcal{L}_i, \ l' \neq l} \tilde{D}_i^{l'} + D_i^l\Big) \tag{4.6}$$

$C_i^l(D_i^l)$ represents the cost of assigning the delay D_i^l to v.c. i on link l when, on the other links, the delay assignment is $\tilde{D}_i^{l'}$, $i \in \mathcal{V}^{l'}$, $l' = 1, \ldots, L$, $l' \neq l$.

Define, for $l = 1, \ldots, L$:

$$(P_s^l) \qquad \min \sum_{i \in \mathcal{V}^l} C_i^l(D_i^l)$$

$$D_i^l, \ i \in \mathcal{V}^l, \text{ feasible}$$

Because of the assumptions on $C_i(\cdot)$, $i = 1, \ldots, V$, we have that for $\vec{\tilde{D}} \geq 0$, $C_i^l(\cdot)$, $l = 1, \ldots, L$, $i \in \mathcal{V}^l$, is strictly convex and non–decreasing. Also because of approximation (Ap.4.1) the feasibility constraints on link l are the same as if the link was considered in isolation. These considerations imply that (P_s^l) is in fact a (P_s) problem. Indeed it is the

(P_s) problem that results when we focus our attention on optimizing the delays on link l, assuming that the delays elsewhere in the network are constant.

Our result ties an overall optimal solution of (NP_s), say $\vec{D}^*$, to the optimal solutions of the (P_s^l) problems that can be defined using it. The result states that to assess the overall optimality of an assignment $\vec{D}^*$ we may equivalently look at the (P_s^l) problems that can be defined using $\vec{D}^*$. The advantage is that it is simpler to look at several (P_s^l) problems than to look at the much bigger (NP_s) problem. This result can be summarized as follows.

Corollary 4.1: $\vec{D}^*$ is optimal for (NP_s) if and only if, for $l = 1, \ldots, L$, $D_i^{*\,l}$, $i \in \mathcal{V}^l$, is an optimal solution of the (P_s^l) problem in which the cost functions $C_i^l(\cdot)$, $i \in \mathcal{V}^l$, are defined using $\vec{D}^*$ via equation (4.6).

The proof follows immediately from theorem 4.1.

An immediate consequence of corollary 4.1 is the following corollary.

Corollary 4.2: Let $\vec{D}^*$ be an optimal solution of (NP_s). Let $\lambda_j^{*\,l}$, $j = 1, \ldots, V^l$, be (as in theorem 3.3) the Lagrange multiplier associated with the j^{th} feasibility constraint in the (P_s^l) problem defined on link l using $\vec{D}^*$. Then, for $i = 1, \ldots, V$, and all $l \in \mathcal{L}_i$:

$$\frac{1}{R_i} C_i'(D_i^*) = \lambda_{w_i^l}^{*\,l} + \cdots + \lambda_{V^l}^{*\,l}$$

The proof of this result follows immediately from the fact that, for all $l \in \mathcal{L}_i$, $D_j^{*\,l}$, $j \in \mathcal{V}^l$, is the optimal solution of the (P_s^l) problem that can be defined on link l using $\vec{D}^*$ and from corollary 3.1.

This corollary underlines the fact that the Lagrange multiplier vectors on different links are intimately related and, as a consequence, that the priority groups to which a v.c. belongs on different links are intimately related.

4.3 A nearly optimal distributed asynchronous algorithm for the problem (NP_s).

In this section we present a distributed asynchronous algorithm whose objective is to solve approximately the problem (NP_s). The assignment produced is in general not optimal. However by choosing the parameters of the algorithm appropriately it can be brought as close to optimality as desired.

One may immediately ask why distributed? Distributed algorithms typically require more work than their centralized counterpart. However in distributed algorithms the work can in general be apportioned intelligently among the processors and be carried out in parallel, resulting in a faster convergence.

Another advantage of distributed algorithms is that they are often well–adapted to respond to local perturbations. Typically the situation in a network is not static but evolves as new v.c.'s are introduced or as old v.c.'s are deleted. Of course we do not want the algorithm to solve from scratch a new (NP_s) problem every time a v.c. is added or deleted. Rather, it is much more efficient to only detect perturbations and just modify slightly the controls accordingly. Distributed algorithms are especially suited in this context because the controls are adjusted locally; i.e. where the perturbations are detected.

Another immediate question is why asynchronous? In a communication network the nodes are physically separated from each other. For this reason coordinating an action involving several nodes is costly both in terms of speed because of the time required for setting up the action, and in terms of overhead because of the control messages which must be communicated by the nodes and which use some transmission capacity that would otherwise be available for transmitting data. In an asynchronous algorithm the nodes act independently of each other, which avoids the above problems.

We will assume that the status of the network evolves very slowly as compared to the speed of convergence of the algorithm. This assumption is called the quasi–static assumption. It insures that the algorithm tracks well the evolution of the network, which must be the case if the algorithm is to be of any use. It also allows us to analyze the algorithm assuming a static environment, which is a major simplification. This assumption cannot in general be guaranteed a priori as the speed of convergence is unknown. Typically, once

an algorithm is designed, simulation studies should be conducted to establish how well the assumption holds.

4.3.1 Description of the algorithm.

The computation centers are the links of the network. Each link, say l, has full control over its local delays; i.e., the D_i^l, $i \in \mathcal{V}^l$.

Each link maintains an estimate of the end–to–end delay of its v.c.'s. The estimate of the end–to–end delay of v.c. i, $i \in \mathcal{V}^l$, maintained by link l is called $d_i^{[l]}$. At all times and for all l, $i \in \mathcal{V}^l$, it is required that:

$$|D_i - d_i^{[l]}| \leq \epsilon_d \left(1 - \frac{1}{L}\right) \tag{4.7}$$

where $\epsilon_d > 0$ is a parameter of the algorithm *. Enforcing equation (4.7) presuppose the existence of a mechanism via which the $d_i^{[l]}$ can be updated as the D_i^l are updated. We will describe such a mechanism later, in the discussion concerning the implementation of the algorithm.

The operation of the algorithm is roughly as follows. Each link, say l, is responsible for its D_i^l, $i \in \mathcal{V}^l$, which it updates occasionally. An update always involves exactly two v.c.'s. Basically the update consists of reducing the delay of the v.c. with the highest marginal delay cost per unit rate while increasing proportionately the delay of the other v.c., so as to maintain strict feasibility. Namely if an update between v.c.'s i and j occurs on some link l, D_i^l is decreased if $\frac{1}{R_i} C_i'(D_i) > \frac{1}{R_j} C_j'(D_j)$ while D_j^l is decreased if the inverse condition holds.

An update between two v.c.'s on a link is only authorized when an associated update condition is satisfied. However the links are capable, using their estimates of the end–to–end delays, to locally detect when the update condition holds. Periodically the estimates of the end–to–end delays are updated to reflect the evolution or the true end–to–end delays.

* The term $1 - 1/L$ in the preceeding equation is not required in the arguments developped in chapter 4. It is included because it is required in Appendix C.4 for generalizing a result presented in the chapter.

The condition for a delay update between v.c.'s i and j on link l, in the case where D_i^l is to be reduced, is:

$$\frac{1}{R_i}C_i'(d_i^{[l]} - \varepsilon_d - \frac{\bar{\Delta}_n}{R_i}) - \frac{1}{R_j}C_j'(d_j^{[l]} + \varepsilon_d + \frac{\bar{\Delta}_n}{R_j}) > \gamma_d \tag{4.8}$$

where $\bar{\Delta}_n$ and γ_d are strictly positive parameters of the algorithm *.

When equation (4.8) holds, the delays of v.c.'s i and j are updated as follows:

$$\begin{aligned} D_i^l &\leftarrow D_i^l - \frac{\Delta_n}{R_i} \\ D_j^l &\leftarrow D_j^l + \frac{\Delta_n}{R_j} \end{aligned} \tag{4.9}$$

where Δ_n is the largest variation satisfying $\Delta_n \in [0, \bar{\Delta}_n]$ and such that the assignment resulting from equations (4.9) is strictly feasible.

In the remainder of this discussion we refer to the updating of a delay assignment as in equations (4.9) with $\Delta_n > 0$ simply as an update and we call equations (4.9) the update rule.

Now we can define the algorithm, which has been called Alg_NP_s_a, as follows. The "a" in Alg_NP_s_a stands for approximate and is justified by the fact that the algorithm does not solve exactly (NP_s) but only produces an approximate solution.

Alg_NP_s_a: Find a link l and a pair of v.c.'s i, $j \in \mathcal{V}^l$ satisfying equation (4.8) and such that a delay update can be performed between v.c.'s i and j on link l and perform the delay update.

This definition is purposely vague. In particular we do not impose any specific rule regarding the choice of the link and pair of v.c.'s among the potential candidates.

An implicit restriction in the definition of Alg_NP_s_a is that only one update can occur at any given time. This restriction is not essential. Indeed it is not difficult to modify the definition of Alg_NP_s_a so as to avoid it. However this restriction allows us to simplify considerably the notation, which is in fact the reason behind the particular formulation of Alg_NP_s_a used above.

* Actually, $\gamma_d = 0$ is also possible. This case is treated in Appendix C.

We conclude this section with an example illustrating the update rule.

Example 4.1: Let l be a link of unit capacity supporting three v.c.'s whose initial assignment is as follows:

$$\begin{aligned} R_1 = 0.1 \quad R_2 = 0.1 \quad R_3 = 0.2 \\ D_1^l = 1.2 \quad D_2^l = 1.7 \quad D_3^l = 1.9 \end{aligned} \tag{4.10}$$

Also let $\bar{\Delta}_n = 0.02$.

We go manually through an update between v.c.'s 1 and 2. First suppose that D_1^l is to be increased. Trying $\Delta_n = \bar{\Delta}_n$, we get:

$$D_1^l = 1.4 \quad D_2^l = 1.5 \quad D_3^l = 1.9 \tag{4.11}$$

It is easily checked that this assignment is strictly feasible so that it is indeed the assignment that should result from the update.

Now suppose that starting again from the assignment (4.10) we want to reduce D_1^l. Consider the variation $\Delta_n = 8.88 \times 10^{-3}$, for which we obtain:

$$D_1^l = 1.111 \quad D_2^l = 1.788 \quad D_3^l = 1.9 \tag{4.12}$$

It is again easily checked that this assignment is strictly feasible. Note that in this case:

$$R_1 D_1^l = \frac{R_1}{1 - R_1} \tag{4.13}$$

Since $D_1^l \leq D_2^l \leq D_3^l$ this means that v.c. 1 has full priority over v.c.'s 2 and 3. Consequently it is impossible to further reduce D_1^l while maintaining feasibility, so that $\Delta_n = 8.88 \times 10^{-3}$ is indeed the largest feasible variation. Thus in this case the update should return the assignment (4.12).

4.3.2 Convergence.

In the analysis of the algorithm we will often add a time dimension to the variables. For example $D_i^l(k)$ will denote the delay of v.c. i on link l at time k. Also, for convenience, we assume that the algorithm starts at time $k = 0$.

We associate a time index, say k, with each update. By convention the value of the variables immediately before the update is referred to as their value at time k^-. At time k it is assumed that the variables have taken their new value.

We make the following assumptions. It is not difficult to see that any sensible assignment must satisfy them.

(A.4.1.1) There exist strictly positive constants K_1 and K_2 such that:

$$R_i \geq K_1 \quad \text{for all } i,\ i = 1, \ldots, V$$

$$\mu^l - \sum_{i \in \mathcal{V}^l} R_i \geq K_2 \quad \text{for all } l,\ l = 1, \ldots, L$$

(A.4.1.2) For each link l, $l = 1, \ldots, L$, the initial asignment $D_i^l(0)$, $i \in \mathcal{V}^l$, is strictly feasible.

Let $H_i = \{y \mid y \geq 0,\ C_i(y) \leq \sum_{i=1}^{V} C_i(D_i(0) + \epsilon_d)\}$. If follows from assumptions (A.4.1) and from the form of the cost functions that there exists K_3 such that for all i and $y, \tilde{y} \in H_i$:

$$\left|C_i'(y) - C_i'(\tilde{y})\right| \leq K_3 |y - \tilde{y}| \tag{4.14}$$

Before presenting the main results of this section we introduce a simple technical lemma.

Lemma 4.1: Let $(\vec{R}, \vec{D})$ be a strictly feasible assignment such that for some v.c's i, j and link l, v.c. i has full priority over v.c. j on link l. Assume also that assumption (A.4.1.1) holds. Then there exists $K_4 > 0$ depending only on K_1 and K_2 such that, for all $\Delta \in [0, K_4]$, the assignment:

$$\bar{D}_i^l = D_i^l + \frac{\Delta}{R_i}$$

$$\bar{D}_j^l = D_j^l - \frac{\Delta}{R_j}$$

$$\bar{D}_k^l = D_k^l \quad \text{for all } k \neq i,\ j$$

is strictly feasible.

The proof of this lemma is given in the section 2 of Appendix C.

Our first result shows that Alg_NP_s_a converges in finite time to a near–optimal solution.

Theorem 4.2: Let assumptions (A.4.1) hold. Then there exists a finite k_f such that the sequence of assignments $\{\vec{D}(k)\}_{k=0}^{k_f}$ generated by the repeated application of Alg_NP_s_a satisfies:

1) For each l, $l = 1, \ldots, L$, $D_i^l(k)$, $i \in \mathcal{V}^l$ is strictly feasible for all k, $0 \leq k \leq k_f$.
2) $S(\vec{D}(k))$ is decreasing in k for all k, $0 \leq k < k_f$.
3) The algorithm stops at time k_f. Moreover the assignment at time k_f is such that for any link l and pair of v.c.'s i, $j \in \mathcal{V}^l$, if:

$$\frac{1}{R_i}C_i'(d_i^{[l]}(k_f) - \varepsilon_d - \frac{\bar{\Delta}_n}{R_i}) - \frac{1}{R_j}C_j'(d_j^{[l]}(k_f) + \varepsilon_d + \frac{\bar{\Delta}_n}{R_j}) \geq \gamma_d$$

then v.c. i has full priority over v.c. j on all their common links.

Proof: Condition (1) is obvious from the assumption that the initial assignment is strictly feasible and from the fact that each iteration of Alg_NP_s_a maintains strict feasibility.

To prove condition (2) consider an arbitrary update, say between v.c.'s i and j on link l. Assume that the update occurs at time k and that D_i^l decreases. Then we must have:

$$\frac{1}{R_i}C_i'(d_i^{[l]}(k^-) - \varepsilon_d - \frac{\bar{\Delta}_n}{R_i}) - \frac{1}{R_j}C_j'(d_j^{[l]}(k^-) + \varepsilon_d + \frac{\bar{\Delta}_n}{R_j}) \geq \gamma_d \tag{4.15}$$

Using equation (4.7) and the fact that the functions $C_i'(\cdot)$ and $C_j'(\cdot)$ are non–decreasing this implies that for all $\Delta \in [0, \bar{\Delta}_n]$;

$$\frac{1}{R_i}C_i'(D_i(k^-) - \frac{\Delta_n}{R_i}) - \frac{1}{R_j}C_j'(D_j(k^-) + \frac{\Delta_n}{R_j}) \geq \gamma_d \tag{4.16}$$

We can also write:

$$C_i(D_i(k)) - C_i(D_i(k^-)) = \int_0^{-\Delta_n/R_i} C_i'(D_i(k^-) + y)\, dy \tag{4.17}$$

Using the transformation $z = -R_i y$, this gives:

$$C_i(D_i(k)) - C_i(D_i(k^-)) = -\int_0^{\Delta_n} \frac{1}{R_i}C_i'(D_i(k^-) - \frac{z}{R_i})\, dz \tag{4.18}$$

Similarly we obtain for j:

$$C_j(D_j(k)) - C_j(D_j(k^-)) = \int_0^{\Delta_n} \frac{1}{R_j} C_j'(D_j(k^-) + \frac{z}{R_j})\, dz \tag{4.19}$$

Adding equations (4.18) and (4.19), and using equation (4.16), we get:

$$S(\vec{D}(k)) - S(\vec{D}(k^-)) \leq -\gamma_d \Delta_n \tag{4.20}$$

which proves the second condition.

As $S(\cdot)$ is bounded below equation (4.20) also implies that the sum of all the Δ_n is finite. Accordingly this means from equations (4.9) that all the D_i^l converge.

Let k_1 be a time such that for all i, $i = 1, \ldots, V$, $l \in \mathcal{L}_i$ and $k \geq k_1$;

$$\left| D_i^l(k) - D_i^l(k_1) \right| \leq \frac{1}{3R_i} \min(\bar{\Delta}_n, K_4) \tag{4.21}$$

It is easy to see from equations (4.9) that in any update involving a v.c., say i, which does not result in one v.c. acquiring full priority over the other, the variation of i's delay is exactly $\bar{\Delta}_n / R_i$. In view of this fact it follows from equation (4.21) that any update occuring after k_1 must result in one v.c. acquiring full priority over the other.

Suppose that the updates never stop. Then there must exist v.c.'s i and j and a link $l \in \mathcal{L}_i \cap \mathcal{L}_j$ such that at least two updates occur between v.c. i and j on link l after k_1 , and such that in each of these updates v.c. i acquires full priority over v.c. j on the link. It follows from this fact that there must occur at least one update after k_1 which results in i losing its full priority over j on link l.

There is only one way via which i can lose its full priority over j on link l. It must be that a v.c. over which i does not have full priority, say p (we may have $p = i$), and a v.c. which has not full priority over j, say q (we may have $q = j$), perform an update together and that, as a result, D_p^l increases. This condition implies that p has initially full priority over q on the link. Thus as D_p^l increases we may conclude using lemma 4.1 that D_p^l can be updated by at least $\frac{1}{R_p} \min(\bar{\Delta}_n, K_4)$. However this violates equation (4.21), and thus produces the required contradiction.

This shows that the algorithm terminates after a finite time, which is called k_f in the theorem. It follows immediately from this fact that the last condition given in the theorem

must hold. Indeed if this condition does not hold updates will continue to occur after k_f, which is impossible.

Q.E.D.

Intuitively it is not difficult to see that the assignment at time k_f is close to optimality. Indeed suppose that the assignment is such that for some link l and v.c.'s $i, j \in \mathcal{V}^l$, v.c. i does not have full priority over v.c. j on the link. Then we must have:

$$\frac{1}{R_i}C_i'(d_i^{[l]} - \varepsilon_d - \frac{\bar{\Delta}_n}{R_i}) - \frac{1}{R_j}C_j'(d_j^{[l]} + \varepsilon_d + \frac{\bar{\Delta}_n}{R_j}) \leq \gamma_d \tag{4.22}$$

from which it can easily be concluded using equation (4.14) that:

$$\frac{1}{R_i}C_i'(D_i) - \frac{1}{R_j}C_j'(D_j) \leq \gamma_d + \frac{2K_3}{R_i}(\varepsilon_d + \frac{\bar{\Delta}_n}{R_i}) + \frac{2K_3}{R_j}(\varepsilon_d + \frac{\bar{\Delta}_n}{R_j}) \tag{4.23}$$

Suppose also that on some link $\tilde{l} \in \mathcal{L}_i \cap \mathcal{L}_j$ ($\tilde{l}$ is not necessary equal to l), j does not have full priority over i. Then similarly as above, we get:

$$\frac{1}{R_j}C_j'(D_j) - \frac{1}{R_i}C_i'(D_i) \leq \gamma_d + \frac{2K_3}{R_i}(\varepsilon_d + \frac{\bar{\Delta}_n}{R_i}) + \frac{2K_3}{R_j}(\varepsilon_d + \frac{\bar{\Delta}_n}{R_j}) \tag{4.24}$$

Thus whenever two v.c.'s i and j are such that neither v.c. has full priority over the other on all their common links we have:

$$\left|\frac{1}{R_i}C_i'(D_i) - \frac{1}{R_j}C_j'(D_j)\right| \leq K_5(\gamma_d + \varepsilon_d + \bar{\Delta}_n) \tag{4.25}$$

where K_5 is some constant depending only on K_3 and on the rate assignment. It follows from this equation that by choosing the parameters γ_d, ε_d and $\bar{\Delta}_n$ appropriately we can insure that the marginal delay costs per unit rate of v.c.'s of comparable priority are as close as desired. This is very reminiscent of an important property of the optimal solutions of (NP_s), namely that in any optimal solution to (NP_S) the marginal delay cost per unit rate of v.c.'s of comparable priority are identical. Basically equation (4.25) insures that this condition is approximately satisfied. Thus the solution generated by the algorithm is in some sense "close" to an optimal solution. The purpose of the next theorem is to quantify more precisely what this "close" means.

Theorem 4.3: For a given $\gamma > 0$, let $\vec{D}$ be an assignment satisfying:

1) For $l = 1, \ldots, L$, $\hat{D}_i^l$, $i \in \mathcal{V}^l$ is strictly feasible.

2) The set of delays $\{\vec{D} \mid S(\vec{D}) \leq S(\vec{\hat{D}})\}$ is bounded *.

3) For all v.c.'s i and j, if the equation;

$$\frac{1}{R_i} C_i'(\hat{D}_i) - \frac{1}{R_j} C_j'(\hat{D}_j) \geq \gamma$$

holds, v.c. i has full priority over v.c. j on all their common links.

Then $\vec{\hat{D}}$ also satisfies:

$$S(\vec{\hat{D}}) - S^* \leq K_6 \gamma$$

where S^* it the optimal value of the problem and K_6 is a constant depending only on the set defined in condition (2).

This is proven in the section 3 of Appendix C.

Consider an assignment, say $\vec{\hat{D}}$, produced by Alg_NP_s_a. Theorem 4.2 guarantees that $\vec{\hat{D}}$ is strictly feasible. Also in view of the form of the cost functions, assumptions (A.4.1.), and of the fact that Alg_NP_s_a is non–increasing, it is clear that $\vec{\hat{D}}$ satisfies the second condition of theorem 4.3. Finally, setting:

$$\gamma = K_5(\gamma_d + \bar{\Delta}_n + \varepsilon_d) \tag{4.26}$$

the third condition is also satisfied. Consequently it follows from theorem 4.3 that $\vec{\hat{D}}$ also satisfies:

$$S(\vec{\hat{D}}) - S^* \leq K_5 K_6(\gamma_d + \bar{\Delta}_n + \varepsilon_d) \tag{4.27}$$

It is obvious from this equation that by choosing γ_d, $\bar{\Delta}_n$ and ε_d small enough assignments as close to optimality as desired can be generated.

Sometimes the term $K_5 K_6$ may be very large. In these cases it may be unrealistic to control the accurary of the solution via equation (4.27) as it could force unacceptably small values for γ_d, $\bar{\Delta}_n$ and ε_d. In a practical situation some experimentation should be carried out to determine appropriate values for the parameters. Also it is possible to reduce the

* Note that the bound can be made relatively tight by using the fact that the delays are strictly feasible.

magnitude of the term K_5K_6 by using a more refined argument in the proof of theorem 4.3. We leave this possibility to the reader.

4.3.3 Implementation.

In the description and analysis of Alg_NP_s_a we have implicitly assumed the existence of a mechanism for updating the end–to–end delay estimates. In particular we have assumed that equation (4.7) was maintained at all times. Our purpose in this section is to construct an update mechanism via which equation (4.7) can be enforced.

Define:

d_i^l: Reference delay of v.c. i on link l.

Δd_i^l: Variation of the current delay of v.c. i on link l from its reference; i.e.,

$$\Delta d_i^l = D_i^l - d_i^l \tag{4.28}$$

$du_i^{[l]}$: Estimate maintained by link l of the overall delay of v.c. i on the links on i's path upstream to l.

$dd_i^{[l]}$: Estimate maintained by link l of the overall delay of v.c. i on the links on i's path downstream to l.

For any link $l \in \mathcal{L}_i$ except the first link on i's path we define us_i^l as the link on i's path immediately upstream to link l, and Us_i^l as the set of links on i's path upstream to link l. Similarly for any link $l \in \mathcal{L}_i$ except the last link on i's path we define ds_i^l as the link on i's path immediately downstream to link l, and Ds_i^l as the set of links on i's path downstream to link l.

At all times it is required that:

$$|\Delta d_i^l| \leq \frac{\varepsilon_d}{8L} \qquad \text{for all } i = 1, \ldots, V,\ l \in \mathcal{L}_i \tag{4.29}$$

Further, the estimates of the end–to–end delays are constructed using the equation:

$$d_i^{[l]} = du_i^{[l]} + D_i^l + dd_i^{[l]} \tag{4.30}$$

Note that equations (4.29) and (4.30) can be enforced locally by the links.

To control the updating of the d_i^l, $du_i^{[l]}$ and $dd_i^{[l]}$ a message is associated with each v.c. The message associated with v.c. i is called update_i. It is responsible for updating the d_i^l, $du_i^{[l]}$ and $dd_i^{[l]}$ for all $l \in \mathcal{L}_i$.

The update_i message cycles along i's path, going from $H(i)$ to $D(i)$ and then back to $H(i)$, etc. As it travels downward from $H(i)$ to $D(i)$, the update_i message updates the d_i^l and $du_i^{[l]}$. The update of a d_i^l consists of setting it equal to the current value of D_i^l while the update of a $du_i^{[l]}$ consists of setting it equal to $\sum_{l \in Us_i^l} d_i^l$. Clearly, in view of equation (4.29), $\sum_{l \in Us_i^l} d_i^l$ is an estimate of i's delay on the links upstream to l whose accuracy can be controlled via the parameter ϵ_d. The update_i message contains a register, called update_i_est, used to keep track of the current value of $\sum_{l \in Us_i^l} d_i^l$. As the update_$i$ message travels downward along i's path the reference delay of v.c. i on the links visited by the message are summed into update_i_est, so that update_i_est always reflects the total delay on the links upstream to the link at which the message currently is. An identical mechanism is used when the update_i message travels from $D(i)$ to $H(i)$ except that in this case, instead of updating the $du_i^{[l]}$, the $dd_i^{[l]}$ are updated. The update mechanism is illustrated in Figure 4.1.

Using this update mechanism we can formulate a version of Alg_NP_s_a that can be readily implemented.

Alg_NP_s_a:

Update of estimates:

Upon reception of update_i from link us_i^l, link l does:

1) $d_i^l \leftarrow D_i^l$.
2) $du_i^{[l]} \leftarrow$ update_i_est.
3) If link l is the last link on i's path then set update_i_est $\leftarrow 0$ and send update_i to link us_i^l, otherwise set update_i_est $\leftarrow$ update_i_est $+ d_i^l$ and send update_i to link ds_i^l.

Upon reception of update_i from link ds_i^l, link l does:

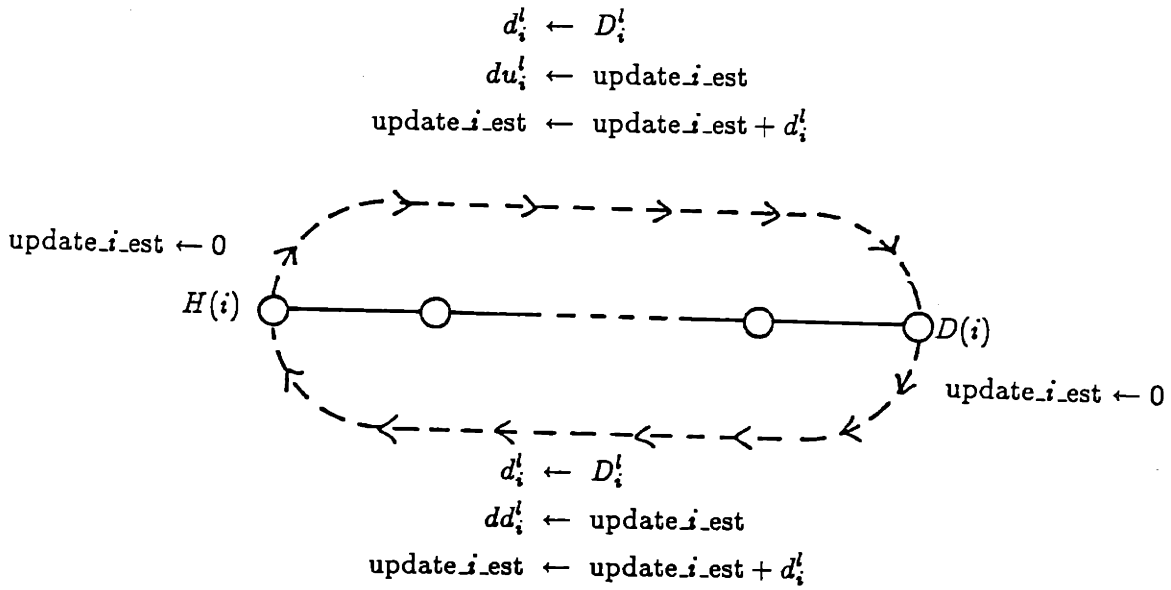


Figure 4.1: Update of the estimates maintained for v.c. i.

1) $d_i^l \leftarrow D_i^l$.
2) $dd_i^{[l]} \leftarrow$ update_i_est.
3) If link l is the first link on i's path then set update_i_est $\leftarrow 0$ and send update_i to link ds_i^l, otherwise set update_i_est $\leftarrow$ update_i_est $+ d_i^l$ and send update_i to link us_i^l.

Delay update:

For all l, $l = 1, \ldots, L$, and i, $j \in \mathcal{V}^l$, whenever the update condition (4.8) holds update D_i^l and D_j^l as in equations (4.9), and where Δ_n is such that equation (4.29) is maintained.

Let $t_0 \geq 0$ be such that for all i, $i = 1, \ldots, V$, the update_i message has visited all the links $l \in \mathcal{L}_i$ at least twice in the interval $[0, t_0[$. Assume that the update_i message is at link l at time $t \geq t_0$ and that it was received from link us_i^l. It is easy to see that:

$$du_i^{[\tilde{l}]} = \sum_{\hat{l} \in Us_i^{\tilde{l}}} d_i^{\hat{l}} \quad \text{for all } \tilde{l} \in Us_i^l \cup l \tag{4.31}$$

$$\left|du_i^{[\tilde{l}]} - \sum_{\hat{l} \in U s_i^{\tilde{l}}} d_i^{\hat{l}}\right| \leq \frac{\varepsilon_d}{4}\left(1 - \frac{1}{L}\right) \quad \text{for all } \tilde{l} \in Ds_i^l \tag{4.32}$$

$$\left|dd_i^{[\tilde{l}]} - \sum_{\hat{l} \in D s_i^{\tilde{l}}} d_i^{\hat{l}}\right| \leq \frac{\varepsilon_d}{4}\left(1 - \frac{1}{L}\right) \quad \text{for all } \tilde{l} \in \mathcal{L}_i \tag{4.33}$$

and, obviously, a completely analogous set of equations is obtained when we suppose that the message is received from ds_i^l.

It follows from equations (4.29), (4.30) and (4.31)-(4.33) that for all $t \geq t_0$ and all i and l :

$$|D_i^l - d_i^{[l]}| \leq \varepsilon_d\left(1 - \frac{1}{L}\right) \tag{4.34}$$

as desired.

To guarantee the convergence of the version of Alg_NP_s_a that was just described we need the following assumption.

(A.4.2) For $i = 1, \ldots, V$, the update_i message cycles infinitely often along i's path.

We will also impose that:

$$\bar{\Delta}_n \leq \frac{\varepsilon_d}{16L(\max_l \mu^l)} \tag{4.35}$$

There is a small difference between the manner in which a delay update was handled in the version of Alg_NP_s_a presented in section 4.3.1 and the manner in which it is handled in the version of Alg_NP_s_a presented in this section. In section 4.3.1 we assumed that unless one v.c. was acquiring full priority over the other Δ_n could always be increased up to $\bar{\Delta}_n$. This implicitly assumed the existence of some ideal mechanism for updating instantaneously the $d_i^{[l]}$, so that equation (4.7) would always hold. In the version of Alg_NP_s_a presented in this section insuring that equation (4.7) always holds requires that the additional condition (4.29) be imposed. This condition may sometimes force Δ_n to be smaller than $\bar{\Delta}_n$, even if the update does not result in one v.c. acquiring full priority over the other.

It is, however, not difficult to see that the convergence results of the last section are still valid for the version of Alg_NP_s_a presented in this section. Indeed, because of equation (4.35), equation (4.29) makes a difference only when the delay of one of the v.c.'s involved

in the update is more than $\varepsilon_d/16L(\max_l \mu^l)$ away from its reference value. However if this is the case the v.c. must have participated recently (i.e., since the last passage of its update message) to updates, and these updates must have caused the objective function to be reduced by at least $\gamma_d\varepsilon_d/16L(\max_l \mu^l)$ (c.f. equation (4.20)). Clearly since the objective function is bounded below this must eventually stop. At this point the true delays will always be close to their reference value, so that the additional condition will never be binding. Accordingly the situation will be exactly as in the preceeding section, which implies that the results presented there hold for the version of Alg_NP_s_a presented in this section.

Hereafter, we refer to the version of Alg_NP_s_a defined in this section simply as Alg_NP_s_a. When we will want to refer to the version of Alg_NP_s_a defined in section 4.3.1, we shall explicitly indicate it.

4.3.4 Comments on Alg_NP$_s$_a.

$\bar{\Delta}_n$ should in general be selected as large as possible. This in view of equation (4.35) implies that in most cases $\bar{\Delta}_n = \varepsilon_d/16L(\max_l \mu^l)$. The reason is that increasing $\bar{\Delta}_n$ allows the links to make bigger, more profitable updates.

γ_d controls the profitability of the updates. The larger the value of γ_d, the larger the progress that can be guaranteed per update. This implies that γ_d influences the speed of convergence of the algorithm. Indeed it is not difficult to see that increasing γ_d increases the speed of convergence. This is because increasing γ_d prevents many marginally profitable updates to occur. Otherwise these updates would take some of the delay variations allowed to the links, creating the possibility that a more profitable update be missed because one of the delay variation would already be as large as permitted.

On the other hand, it is clear from equation (4.27) that γ_d impacts the quality of the solution produced. As the assignments approach optimality the profitability of the updates decreases. Thus a large γ_d will cause the algorithm to stop sooner, in general further away from optimality, than if a smaller value had been used.

As already mentioned the algorithm also works when $\gamma_d = 0$. However the proof of theorem 4.2 does not hold when $\gamma_d = 0$ because it cannot be concluded from equation (4.20)

that the delays converge. Essentially the argument in this case relies on the fact that the quantity:

$$\max_i \frac{1}{R_i} C_i'(D_i(t)) \tag{4.36}$$

is non–increasing. Indeed this is obvious since whenever a v.c. achieves the maximum, an update involving it must result in a reduction of its delay. If the quantity in (4.36) is non–increasing it will eventually converge somewhere. It can be shown that this fact guarantees the convergence of the delay of at least one v.c. In fact, strengthening the argument, we can show that one of the v.c.'s will eventually stop participating to the updates. At this point we may forget this v.c. and consider a smaller network containing only the $V-1$ remaining v.c.'s. Repeating the same argument one of the v.c.'s in this smaller network must eventually stop participating to the updates. Thus, generalizing the argument, we will be able to show that all the delays converge. Then we can continue using essentially the same arguments as those developped in the chapter. The proof that the algorithm works when $\gamma_d = 0$ is given in the section 4 of Appendix C.

ε_d is probably the most important parameter of the algorithm. Distributed algorithms in which the updates are based on local information on the cost functions, often most importantly on their first derivative, are frequent (an excellent example is the routing problem, see [Ga77] or [Go80]). In general, as the cost functions depend on the overall situation, the links are only allowed to make small updates. Otherwise, if large updates were permitted, the local estimates of the cost functions maintained by the links would become very inaccurate. As a consequence it would not be possible to insure that the updates are actually profitable. ε_d provides a handle for controlling precisely the accuracy of the estimates of the cost functions. In fact it is obvious to see that the basic idea behind Alg_NP_s_a consists of exploiting this built–in accuracy. ε_d also impacts strongly the speed of convergence of Alg_NP_s_a because it directly affects the flexibility given to the links. This is important from a practical point of view because it essentially allows us to trade–off accuracy for speed.

It is worth noting that the communication overhead imposed by Alg_NP_s_a is very small. Only one message per v.c. is required. Moreover the length of the messages is small

as they carry only two numbers. In fact, in view of these facts, piggybacking the update messages on regular messages seems to be a particularly attractive solution. This would eliminate the need for a special update message and reduce the processing effort associated with the scheduling of the transmissions.

In a delay update most of the computational burden arises in determining Δ_n in equations (4.9). A little thought reveals that the amount of computation required by a link, say l, to determine Δ_n is in the worst case proportional to V^l. As an update message can at most cause the link to iterate V^l times through equations (4.9) the overall amount of computation, per update message, is $O\big((V^l)^2\big)$. This is very reasonable. Moreover it is an extremely unlikely worst case. It is our belief that in general the amount of computation is much less; approximately proportional to V^l.

It is not difficult to generalize Alg_NP_s_a so as to guarantee convergence to an optimal assignment. For example, one possibility consists of systematically reducing the parameters γ_d, ϵ_d and Δ_n when no further update can be made with the current set of parameters, so that these parameters all eventually vanish. Although it is clear that algorithms of this sort converge to an optimal assignment we believe that these algorithms would perform poorly because to their slow rate of convergence.

The purpose of controlling the accuracy of the delay estimates in Alg_NP_s_a is to insure that the links have accurate estimates of the marginal delay cost of their v.c.'s. This suggests that, instead of controlling the accuracy of the end–to–end delay estimates, we could directly control the accuracy of the marginal delay cost of the v.c.'s*. We believe that this has several advantages. For example it allows a link to make large variations to the delay of a v.c. when the marginal delay cost of the v.c. does not vary rapidly, but restricts the magnitude of the variations as the sensitivity increases. This is obviously desirable. Another advantage is that it would allow us to better control how far from optimality the assignment produced by the algorithm could be.

* This possibility was brought to my attention by Prof. Tsitsiklis.

4.4 A distributed algorithm solving the problem (NP_s).

In this section we develop a distributed syschronous algorithm which solves exactly the problem (NP_s). The main idea of the algorithm is to use the partial derivatives of the cost function to modulate the variation made to the current assignment. This is a very well–known idea in the literature, especially in the context of centralized algorithms. We refer the reader to [Av76,Be83] and [Lu84] for extensive discussions on this topic.

Another important idea of the algorithm is to only allow the links to make small variations to their current delay assignment. The motivation is to insure that the estimates of the first derivative of the cost functions maintained locally by the links are accurate, so that a reduction of the objective function can be guaranteed when the links locally update their delay assignment. Making small variations is typical in algorithms in which the magnitude of the variations depends on the partial derivatives of the cost functions. We refer the reader to [Gal77,Go80] and [Ts84] for examples of algorithms of this sort.

It is our belief that the algorithm presented in this section is representative of a broad class. It is, however, not difficult to find improvements to this algorithm. One of the main concerns in the construction of the algorithm was to keep the proof of convergence and the notation relatively simple. For this reason the algorithm has been built with a minimum of complexity. We will comment at the end of this section on the limitations of the algorithm and on several possibilities for improving it.

4.4.1 Description of the algorithm.

We first state the algorithm and then follow with a discussion motivating the ideas behind it.

Let i and j be two v.c's on some link l. For a given delay assignment $\vec{D}$, define:

$$\Delta C_{ij} = \frac{1}{R_i} C_i'(D_i) - \frac{1}{R_j} C_j'(D_j) \tag{4.37}$$

Note that although ΔC_{ij} depends on $\vec{D}$ this dependency has been dropped in the notation. We will explicitly indicate the dependency of ΔC_{ij} on $\vec{D}$ only in the cases in which a

confusion is possible. In general, however, the delay assignment on which ΔC_{ij} depends will be obvious from the context.

Let τ_{ij}^l be the maximum value that τ can take in the interval $[0, \bar{\tau}]$ such that the assignment:

$$\begin{aligned} D_i^l &\leftarrow D_i^l - \frac{\tau}{R_i}\Delta C_{ij} \\ D_j^l &\leftarrow D_j^l + \frac{\tau}{R_j}\Delta C_{ij} \end{aligned} \tag{4.38}$$

is strictly feasible. Note that τ_{ij}^l depends on $\vec{D}$ but, similarly as for ΔC_{ij}, we will in general drop this dependency in the notation. $\bar{\tau}$ is a strictly positive parameter of the algorithm which will be discussed in more detail later.

In this discussion we refer to the updating of a delay assignment as in equations (4.38) simply as an update and we call the equations (4.38) the update rule.

The algorithm has been called Alg_NP_s_e, where the "e" stands for "exact". It is defined as follows.

Alg_NP_s_e: For all l, $l = 1, \ldots, L$, do:

1) Find a pair of v.c.'s $i, j \in \mathcal{V}^l$, $i \neq j$, satisfying:

$$\tau_{ij}^l (\Delta C_{ij})^2 = \max_{p,q \in \mathcal{V}^l} \tau_{pq}^l (\Delta C_{pq})^2 \tag{4.39}$$

2) Using $\tau = \tau_{ij}^l$, update D_i^l and D_j^l as in equations (4.38).

3) Next l.

The update rule of Alg_NP_s_e is very similar to the update rule of Alg_NP_s_a. However there is one major difference. The fixed parameter Δ_n used in Alg_NP_s_a is replaced by ΔC_{ij} in the update rule of Alg_NP_s_e. This has two main advantages. First the update rule of Alg_NP_s_e always causes the assignment to be modified in a way that reduces the total cost (this is true only if $\bar{\tau}$ is sufficiently small, we will make this statement more rigorous later). For example if $\frac{1}{R_i}C_i'(D_i)$ is larger than $\frac{1}{R_j}C_j'(D_j)$ it is easy to see that decreasing D_i^l and correspondingly increasing D_j^l (so as to maintain strict feasibility) reduces the total cost. This is precisely what the update rule of Alg_NP_s_e does in this case.

The second advantage of using ΔC_{ij} is that it provides a mean of automatically scaling down the magnitude of the variations as the assignment approaches optimality. Indeed if $\frac{1}{R_i}C_i'(D_i) \approx \frac{1}{R_j}C_j'(D_j)$ the assignment of v.c.'s i and j is, as far as i and j alone are concerned, nearly optimal. In this case the update should not result in large variations of the delay of v.c.'s i and j, which is automatically insured by the fact that ΔC_{ij} is then very small.

The main property of the update rule of Alg_NP_s_e is that when this rule is used it is possible to lower bound away from zero the decrease of the objective function resulting from a given update. Indeed it may be proven that if an update occurs between v.c.'s i and j on link l, the objective function decreases by at least $\frac{1}{2}\tau_{ij}^l(\Delta C_{ij})^2$. This result is the main motivation behind Alg_NP_s_e. If fact the design of Alg_NP_s_e was expressly based on the exploitation of this result.

Consider an arbitrary link l and a pair of v.c.'s $i, j \in \mathcal{V}^l$. An immediate consequence of the preceeding paragraph is that $\tau_{ij}^l(\Delta C_{ij})^2$ must converge to zero as Alg_NP_s_e is repeated. Accordingly it follows that if ΔC_{ij} does not converge to zero τ_{ij}^l must converge to zero and, conversely, if τ_{ij}^l does not converge to zero ΔC_{ij} must converge to zero. These facts basically guarantee that the assignments generated by Alg_NP_s_e become increasingly close to being optimal as the algorithm is repeated. Indeed in any optimal assignment whenever two v.c.'s, say i and j, have different marginal costs per unit rate, the v.c. with the highest marginal cost per unit rate must have full priority over the other v.c. on all their common links. In other words this means that in any optimal solution if $\Delta C_{ij} \neq 0$ for some v.c.'s i and j, then $\tau_{ij}^l = 0$ for all the links $l \in \mathcal{L}_i \cap \mathcal{L}_j$. Clearly the assignments generated by Alg_NP_s_e become increasingly close to satisfying this condition as the algorithm is repeated. Similarly if $\tau_{ij}^l \neq 0$ for some v.c.'s i, j and link l in an optimal assignment it must be that v.c.'s i and j have the same marginal delay cost per unit rate. Accordingly we then have $\Delta C_{ij} = 0$. Clearly the assignments generated by Alg_NP_s_e also become increasingly close to satisfying this condition as the algorithm is repeated.

4.4.2 Convergence.

We again assume that assumptions (A.4.1) hold. We have the following result.

Theorem 4.4: For each $\vec{D}(0)$ there exists $\bar{\tau} > 0$ (depending only on $\vec{D}(0)$) such that for this $\bar{\tau}$ the sequence of assignments $\{\vec{D}(k)\}_{k=0}^{\infty}$ generated by the repeated application of Alg_NP_s_e satisfies:

1) $\{D_i^l(k),\ i \in \mathcal{V}^l\}$ is strictly feasible, for all $l = 1, \ldots, L$, and $k \geq 0$.

2)

$$\lim_{k \to \infty} S(\vec{D}(k)) = S^*$$

where S^* is the optimal value of (NP_s).

This is proven in the section 5 of Appendix C.

4.4.3 Implementation.

There are two main problems associated with the implementation of Alg_NP_s_e. The first problem is the synchronization required for performing an update of Alg_NP_s_e in a distributed environment. Indeed the links must know the end–to–end delay of their v.c.'s to determine the ΔC_{ij}. In a distributed environment this means that before an iteration of Alg_NP_s_e can be performed some update mechanism insuring that all the links know the end–to–end delay of their v.c.'s must first be executed. Also, as an iteration of Alg_NP_s_e may potentially result in an update on every link, the distributed implementation of Alg_NP_s_e requires some sort of supervision mechanism for insuring that all the links participate when an iteration of Alg_NP_s_e is initiated. It is not difficult to design such mechanisms. However as all the links are involved these mechanisms may well impose a substantial overhead on the algorithm, especially in terms of time. In fact the essence of the problem is that Alg_NP_s_e is a synchronous algorithm. Implementing it in a distributed environment requires some mechanism for mimicking the synchronous operation, and whose overhead can be significant.

We believe that in practice it is not necessary nor desirable to fully respect the synchronization required by Alg_NP_s_e in the implementation. In particular the links should be allowed to update their delay assignment independently from each other and should not

be required to have a perfect knowledge of the end–to–end delay of their v.c.'s for doing so. Of course convergence is not guaranteed in this context. However we believe that if the estimates of the end–to–end delays maintained by the links are updated often as compared to the frequency at which updates of the assignment are made the lack of synchronization should not result in a significant difference in the quality of the assignment produced. In fact relaxing the synchronization requirement of Alg_NP_s_e in a quasi–static environment is likely to improve the performance because the links would then be able to respond much more quickly to local perturbations.

The implementation of Alg_NP_s_e becomes very simple when the synchronization requirement is relaxed. Indeed we then only need a simple asynchronous mechanism for updating the estimates of the end–to–end delays maintained by the links. In this context an obvious possibility is to use the same scheme as in Alg_NP_s_a; namely to dedicate a special message to each v.c. responsible for updating the estimates of the end–to–end delay of the v.c.

One more subtle problem of Alg_NP_s_e results from the fact that $\bar{r}$ depends on the initial assignment. In a quasi–static environment the goal of an algorithm is not to solve one problem starting from some given initial assignment but it is rather to constantly adjust the assignment so as to solve (or, more accurately, nearly solve) the current problem, which may often change as new v.c.'s are established or as old ones are deleted. In this context enforcing the conditions on $\bar{r}$ under which convergence can be guaranteed may be difficult. We believe that in practice one should not try to enforce these conditions at all times. However $\bar{r}$ should be sufficiently small to guarantee convergence under most circumstances.

4.4.4 Comments on Alg_NP_s_e.

In one iteration of Alg_NP_s_e a link, say l, updates only the delays of the pair of v.c.'s for which $r^l_{ij}(\Delta C_{ij})^2$ is maximum. An obvious improvement would be to let the links update the delay of their other v.c.'s in a similar manner in each iteration. The proof of convergence can easily be extended to this case. It only requires a simple modification to lemma C.2.

Another possible improvement would be to simplify the work done by the links in each iteration. As most of the computational burden results from finding the pair of v.c.'s with the maximum $\tau_{ij}^l(\Delta C_{ij})^2$, one possibility would be to let the links arbitrarily select one v.c. and only carry the maximization over the remaining v.c., or even to let the links choose arbitrarily both v.c.'s. However we have not been able to prove the convergence of these variations of Alg_NP_s_e.

In each iteration of Alg_NP_s_e a link, say l, must evaluate $\tau_{ij}^l(\Delta C_{ij})^2$ for all the pairs of v.c.'s $i, j \in \mathcal{V}^l$. As there are $V^l(V^l-1)/2$ pairs of v.c's on link l, and as the determination of a $\tau_{ij}^l(\Delta C_{ij})^2$ requires $O\big((V^l)^2\big)$ computations, it follows that the amount of computation performed by link l, per iteration of Alg_NP_s_e, is $O\big((V^l)^4\big)$. This, however, is an extremely unlikely case. In most circumpstances the amount of computations should be much less; approximately $O\big((V^l)^2\big)$ or $O\big((V^l)^3\big)$ depending on $\bar{\tau}$.

4.5 Comments.

Essentially the difference between Alg_NP_s_a and Alg_NP_s_e is a matter of philosophy. Alg_NP_s_a is an approximate algorithm in the sense that it cannot in general produce an optimal assignment. However the quality of the assignment produced by the algorithm is fully controllable and the algorithm is designed for working in a completely uncoordinated manner. On the other hand, provided that the iterations are synchronized and that the initial assignment is known, Alg_NP_s_e is guaranteed to produce an optimal solution. In practice, however, synchronizing the updates and keeping track of an initial assignment is in general too costly to be justifiable. In this context Alg_NP_s_e also becomes an approximate algorithm. In fact controlling the accuracy in an uncoordinated and evolving environment is the major reason that has lead us to Alg_NP_s_a.

We believe that the ideas behind Alg_NP_s_a are not specific to our problem but that they can be used in a more general context. Indeed the fundamental feature of (NP_s) on which Alg_NP_s_a is based is that (NP_s) fits the framework of theorem 4.1. When a problem fits this framework the feasible set can be decomposed into subsets which, except for their impact on the objective function, are completely independent. Then associating one subset with each computation center the problem becomes that of insuring that the update directions locally perceived as good by the computation centers are actually good. This is basically what Alg_NP_s_a does*.

Consider the following problem.

$$(NP_u) \qquad \min_{\vec{D} \text{ feasible}} \Im(C_1(D_1), \ldots, C_V(D_V))$$

Clearly this is the counterpart of problem (NP_s) in the user–oriented formulation. Although it will not be done here, it can be proven that with a trivial modification to their respective update condition (c.f. theorem 3.4), Alg_NP_s_a and Alg_NP_s_e have the same properties in the user–oriented formulation as in the system–oriented formulation.

* The possibility of generalizing Alg_NP_s_a was suggested by Prof. Humblet.

Finally it is also worth mentioning that all the algorithms presented in this chapter, including the generalization to the user–oriented formulation, have a straightforward equivalent in the context in which only non–preemptive queuing is allowed.

5 An Integrated Approach to Rate and Delay Management.

At the beginning of this thesis we have mentioned the fact that the vast majority of the flow control schemes do not consider the interactions between rate and delay in the determination of the rate assignment. Indeed apart from the flow control schemes based on the notion of power we do not know of any end–to–end flow control scheme in which the impact of the rate on the delay is considered in the selection of the rate assignment. In fact the motivation of the delay assignment problem studied in chapter 4 is to partially compensate this weakness of most flow control schemes by fitting as well as possible the delays to the rates. However, a posteriori adjusting the delays to the rates can only be a partial remedy. Indeed the tight coupling between rate and delay may force in all cases a poor delay assignment when not enough attention is paid to the interactions between rate and delay in the selection of the rate assignment. In this chapter we present a means for overcoming this problem. More specifically we develop a formulation of the flow control problem in which the interactions between rate and delay are fully considered. In fact rate and delay are equally important factors in the formulation. They are determined simultaneously so as to produce the best overall assignment.

We follow an approach similar to that of chapter 4. In the first section we define precisely the problem. Then in section 2 a set of optimality conditions for the problem is derived. In section 3 we exploit the optimality conditions for constructing a distributed algorithm solving the problem. Finally in section 4 some comments on possible generalizations and improvements are made.

5.1 Problem formulation.

For all i, $i = 1,\ldots,V$, let N_i be the average number of packets (a.n.o.p.) of v.c. i present in the network. By abuse of language we will often say that N_i is the a.n.o.p. of v.c. i "stored" in the network. Similarly for all i, $i = 1,\ldots,V$ and $l \in \mathcal{L}_i$, let N_i^l be the a.n.o.p. of v.c. i stored on link l. Let $\vec{N}$ be the vector whose components are the N_i^l, for all i, $i = 1,\ldots,V$, and $l \in \mathcal{L}_i$. Also for all l, $l = 1,\ldots,L$, let $\vec{N}^l$ be the vector containing the N_i^l for all $i \in \mathcal{V}^l$.

Clearly for all i, $i = 1, \ldots, V$:

$$N_i = \sum_{l \in \mathcal{L}_i} N_i^l \tag{5.1}$$

Also using the well–known Little's result we have for all i, $i = 1, \ldots, V$, and $l \in \mathcal{L}_i$:

$$N_i^l = R_i D_i^l \tag{5.2}$$

We straightforwardly extend the definition of weak and strict feasibility as follows. An assignment $(\vec{R}, \vec{N})$ is weakly feasible on link l if:

$$\begin{aligned} R_i &\geq 0 \quad \text{for all } i \in \mathcal{V}^l \\ \sum_{i \in \mathcal{V}^l} R_i &\leq \mu^l \end{aligned} \tag{5.3}$$

and if for some ordering $\vec{w}^l$ on link l valid for the assignment:

$$\sum_{p | w_p^l \leq k} N_p^l \geq B^l(k, \vec{w}^l, \vec{R}) \qquad \text{for all } k = 1, \ldots, V^l \tag{5.4}$$

$(\vec{R}, \vec{N})$ is strictly feasible on link l if, in addition to being weakly feasible, it also satisfies:

$$\sum_{p=1}^{V^l} N_p^l = B^l(V^l, \vec{w}^l, \vec{R}) \tag{5.5}$$

Finally $(\vec{R}, \vec{N})$ is respectively weakly feasible or strictly feasible if it is weakly feasible or strictly feasible on all links.

Let $(\vec{R}, \vec{N})$ be a given assignment and, for $l = 1, \ldots, L$, let $\vec{w}^l$ be an ordering on link l valid for this assignment. For all l, $l = 1, \ldots, L$, we define:

$$B^{l\prime}(i, \vec{w}^l, \vec{R}) = \frac{\mu^l}{\left(\mu^l - \sum_{p | w_p^l \leq i} R_p\right)^2} \qquad 1 \leq i \leq V^l \tag{5.6}$$

It is not difficult to see that for all l, $l = 1, \ldots, L$:

$$\frac{\partial}{\partial R_i} B^l(i, \vec{w}^l, \vec{R}) = B^{l\prime}(i, \vec{w}^l, \vec{R}) \qquad \text{for all } q,\ 1 \leq q \leq i \tag{5.7}$$

From this equation we see that $B^{l\prime}(k, \vec{w}^l, \vec{R})$ represents the variation per unit rate (in the limit of a small rate variation) of the right hand side of the k^{th} feasibility constraint on link l as the rate of one of the v.c.'s involved in the constraint is varied.

As in the preceeding chapters a cost function $G_i(\cdot,\cdot)$ is associated with each v.c. i, $i = 1,\ldots,V$. However the cost function of a v.c. now depends on two parameters: the rate and the a.n.o.p. of the v.c. Indeed $G_i(R_i, N_i)$ quantifies the dissatisfaction of v.c. i when its rate is R_i and its a.n.o.p. is N_i. Using Little's result the cost function of a v.c. can also be expressed as a function of its rate and delay. Namely;

$$G_i(R_i, N_i) = G_i(R_i, R_i D_i) \tag{5.8}$$

We make the following assumptions.

(A.5.1) For all i, $i = 1,\ldots,V$, $G_i(\cdot,\cdot)$ is a convex, twice continuously differentiable function. Also for every fixed $R_i > 0$, $G_i(R_i, R_i D_i)$ is a strictly convex and non–decreasing function of D_i over the interval $[0, \infty[$. Moreover for every fixed $N_i \geq 0$, $G_i(R_i, N_i)$ is a non–increasing function of R_i and satisfies:

$$\lim_{R_i \to 0} G_i(R_i, N_i) = \infty \tag{5.9}$$

The motivation behind the assumptions on $G_i(R_i, R_i D_i)$ for fixed $R_i > 0$ is to insure that the cost function fits the framework of the previous chapters. Indeed these assumptions guarantee that the function $G_i(R_i, R_i D_i)$ for fixed R_i is exactly as the function $C_i(D_i)$ used in the preceeding chapters in the system–oriented approach.

The assumption that $G_i(R_i, N_i)$ for fixed N_i is a non–increasing function of R_i is not restrictive. Indeed, for fixed N_i, increasing R_i implies that D_i decreases. Clearly the dissatisfaction of v.c. i cannot increase as its rate is increased and as its delay is reduced. The second assumption on $G_i(R_i, N_i)$ for fixed N_i, namely equation (5.9), is slightly more restrictive. It is made to insure that no user can be shut off from the network.

One may wonder why we use the a.n.o.p. instead of the delay in the cost functions. At the first glance using the delay seems to be more consistent with the approach taken in the previous chapters. The reason is that the set of feasible rates and delays is not convex. The reader is referred to p. 40 for an explanation of this fact. However the set of weakly feasible rates and a.n.o.p.'s is convex. This fact is the subject of the following lemma.

Lemma 5.1: For each l, $l = 1, \ldots, L$, the set of weakly feasible rate and a.n.o.p. assignments on link l is convex.

This is proven in the section 1 of Appendix D.

Our approach to the flow control problem is based on the following problem, which will be referred to as (FC_s).

$$(FC_s) \qquad \begin{array}{c} \min S(\vec{R}, \vec{N}) \\ (\vec{R}, \vec{N}) \text{ feasible} \end{array}$$

where:

$$S(\vec{R}, \vec{N}) = \sum_{i=1}^{V} G_i(R_i, N_i) \tag{5.10}$$

Clearly this problem is the generalization of problem (NP_s) to the case in which the rates are not fixed.

As in chapter 4 we make approximation (Ap.4.1) in order to obtain a simple characterization of the feasible set. Under this approximation, (FC_s) can be more conveniently written as:

$$\begin{array}{c} \min S(\vec{R}, \vec{N}) \\ \left\{R_i,\ N_i^l,\ i \in \mathcal{V}^l, \text{ feasible}\right\} \quad l = 1, \ldots, L \end{array} \tag{5.11}$$

5.2 Optimality Conditions.

In this section we first give a set of necessary and sufficient conditions that any optimal solution of (FC_s) must satisfy. These conditions are stated in the theorem presented below. We follow with a discussion motivating the intuition behind these conditions. We believe this discussion will provide an useful insight in the algorithm to be presented in the next section.

Theorem 5.1: $(\vec{R}^*, \vec{N}^*)$ is an optimal solution of (FC_s) if and only if the following conditions are satisfied.

1) $(\vec{R}^*, \vec{N}^*)$ is strictly feasible.
2) $\vec{D}^*$ is an optimal solution of the (NP_s) problem obtained from (FC_s) by setting $\vec{R} = \vec{R}^*$.
3) for $i = 1, \ldots, V$;

$$\frac{\partial}{\partial R} G_i(R_i^*, N_i^*) = -\sum_{l \in \mathcal{L}_i} \left\{ B^{l\prime}(w_i^l, \vec{w}^l, \vec{R}^*) \frac{\partial}{\partial N} G_i(R_i^*, N_i^*) + \sum_{j | w_j^l > w_i^l} \left(B^{l\prime}(w_j^l, \vec{w}^l, \vec{R}^*) - B^{l\prime}(w_j^l - 1, \vec{w}^l, \vec{R}^*) \right) \frac{\partial}{\partial N} G_j(R_j^*, N_j^*) \right\}$$

where, for $l = 1, \ldots, L$, $\vec{w}^l$ is an ordering on link l valid for the assignment $(\vec{R}^*, \vec{N}^*)$.

This theorem is proven in the section 2 of Appendix D.

Since by definition any solution of (FC_s) must be strictly feasible it is clear that condition (1) must hold. Also, if $(\vec{R}^*, \vec{N}^*)$ is an optimal assignment, any other strictly feasible assignment having the rate assignment $\vec{R}^*$ cannot have a lower objective function value than $(\vec{R}^*, \vec{N}^*)$. The second condition follows immediately from this fact.

Essentially condition (3) states that at optimality the marginal rate cost of a v.c. must equal the marginal cost for adjusting the a.n.o.p. assignment to the rate of the v.c. Clearly, if some v.c. does not satisfies this condition, then we can adjust the rate of the v.c. and the a.n.o.p. assignment so as to reduce the objective function value. This is impossible if the assignment is optimal.

The equation in condition (3) provides a considerable insight into the rate versus delay trade–off. We now go through an example which we hope will help in understanding some of the implications of this equation.

Let $(\vec{R}, \vec{N})$ be a given strictly feasible assignment and assume that the a.n.o.p. assignment is optimal for the rate assignment (namely, $(\vec{R}, \vec{N})$ satisfies the first two conditions of the theorem). Also, for $l = 1, \ldots, L$, let $\vec{w}^l$ be an ordering on link l valid for this assignment. Our aim is to evaluate the net cost for increasing the rate of a v.c., assuming that the assignment is adjusted in the cheapest feasible way that maintains strict feasibility.

Specifically assume that the rate of a given v.c., say i, is increased by a very small amount ΔR_i. As this causes, on all the links $l \in \mathcal{L}_i$, the right hand side of the feasibility constraints containing i to increase, it is necessary to adjust the a.n.o.p. assignment to maintain strict feasibility.

Consider a particular link on i's path, say link l. Assuming that the initial ordering $\vec{w}^l$ remains valid (we will soon see that this assumption is not restrictive), the increase of R_i causes only an increase in the right hand side of the constraints $w_i^l, \ldots, V^l$. Specifically the increase in the right hand side of constraint k, $k = w_i^l, \ldots, V^l$, is:

$$B^l(k, \vec{w}^l, \vec{R} + \overrightarrow{\Delta R_i}) - B^l(k, \vec{w}^l, \vec{R}) \tag{5.12}$$

where $\overrightarrow{\Delta R_i}$ denotes the vector whose i^{th} entry is ΔR_i and whose other entries are zero.

For small ΔR_i the last expression is approximately equal to:

$$B^{l\prime}(k, \vec{w}^l, \vec{R}) \Delta R_i \tag{5.13}$$

We assume that the variations of R_i and of the a.n.o.p. assignment are so small that the constraints initially satisfied with strict inequality stay satisfied with strict inequality (again, we will soon see that this assumption is not restrictive). In this context we only have to increase the left hand side of the constraints initially satisfied with equality just enough to balance the increase of their right hand side to insure that strict feasibility is maintained. It is easy to see that the most economical way of increasing the left hand side of a constraint consists of increasing the a.n.o.p. of a v.c. involved in the constraint having

the lowest marginal a.n.o.p. cost. In view of the fact that the a.n.o.p. assignment is initially optimal for the rate assignment this means that we can always compensate at the cheapest cost an increase in the right hand side of a constraint by increasing the a.n.o.p. of the v.c. involved in the constraint with the lowest (i.e., rightmost) position in the ordering. Indeed if the a.n.o.p. assignment is optimal for the rate assignment, the v.c.'s must be ranked in order of decreasing marginal a.n.o.p. cost in the ordering $\vec{w}^l$. This implies that, in each constraint, the v.c. involved in the constraint with the lowest position in the ordering is always among the v.c.'s involved in the constraint with the lowest marginal a.n.o.p. cost, so that increasing the a.n.o.p. of this v.c. leads to the smallest additional cost. (We are neglecting here the second order effects of the variation of the assignment on the cost functions.)

The first constraint on link l containing i and satisfied with equality is the constraint defining the priority group to which i belongs. Namely this is the constraint $w^l_{\underline{i}(e^l_i)}$. From (5.13) the increase in the right hand side of this constraint is:

$$B^{l\prime}(w^l_{\underline{i}(e^l_i)}, \vec{w}^l, \vec{R})\Delta R_i \tag{5.14}$$

As motivated above the v.c. whose a.n.o.p. should be increased to compensate for the increase of the right hand side is the v.c. in position $w^l_{\underline{i}(e^l_i)}$. Accordingly, this results in an additional cost of:

$$B^{l\prime}(w^l_{\underline{i}(e^l_i)}, \vec{w}^l, \vec{R})\Delta R_i \frac{\partial}{\partial N} G_{\underline{i}(e^l_i)}(R_{\underline{i}(e^l_i)}, N_{\underline{i}(e^l_i)}) \tag{5.15}$$

Let e be the priority group on link l immediately following the priority group containing i (namely, $e-1 = e^l_i$). Then by definition the next constraint satisfied with equality after the constraint $w^l_{\underline{i}(e^l_i)}$ is the constraint $w^l_{\underline{i}(e)}$. From equation (5.13) the variation in the right hand side of this constaint is :

$$B^{l\prime}(w^l_{\underline{i}(e)}, \vec{w}^l, \vec{R})\Delta R_i \tag{5.16}$$

As before the v.c. whose delay should be increased to compensate for the increase in the right hand side is the v.c. in position $w^l_{\underline{i}(e)}$. However in this case the a.n.o.p. of this v.c. only has to be increased by :

$$\left(B^{l\prime}(w^l_{\underline{i}(e^l_i)}, \vec{w}^l, \vec{R}) - B^{l\prime}(w^l_{\underline{i}(e)}, \vec{w}^l, \vec{R})\right)\Delta R_i \tag{5.17}$$

since the remaining contribution needed to balance (5.16) is already provided by the v.c. $\underline{i}(e_i^l)$ (remember that the left hand side of the constraint $w^l_{\underline{i}(e)}$ contains both the v.c.'s $\underline{i}(e_i^l)$ and $\underline{i}(e)$). Thus the additional cost incurred for maintaining equation $w^l_{\underline{i}(e)}$ is:

$$\left(B^{l\prime}(w^l_{\underline{i}(e_i^l)}, \vec{w}^l, \vec{R}) - B^{l\prime}(w^l_{\underline{i}(e)}, \vec{w}^l, \vec{R})\right)\Delta R_i \frac{\partial}{\partial N} G_{\underline{i}(e)}(R_{\underline{i}(e)}, N_{\underline{i}(e)}) \tag{5.18}$$

Generalizing this argument it is not difficult to see that the total additional cost incurred on link l for maintaining strict feasibility is:

$$B^{l\prime}(w^l_{\underline{i}(e_i^l)}, \vec{w}^l, \vec{R}) \frac{\partial}{\partial N} G_{\underline{i}(e_i^l)}(R_{\underline{i}(e_i^l)}, N_{\underline{i}(e_i^l)}) + \sum_{e \in E^l,\ e > e_i^l} \left(B^{l\prime}(w^l_{\underline{i}(e)}, \vec{w}^l, \vec{R}) - B^{l\prime}(w^l_{\underline{i}(e-1)}, \vec{w}^l, \vec{R})\right)\Delta R_i \frac{\partial}{\partial N} G_{\underline{i}(e)}(R_{\underline{i}(e)}, N_{\underline{i}(e)}) \tag{5.19}$$

We are going to show that this expression is equal to the term corresponding to link l in the sum in the right hand side of the equation in condition (3) of the theorem. However we first make a short pause to verify the validity of the two assumptions that we made earlier in the discussion.

Let $D^l_j(\Delta R_i)$ denote the delay of v.c. j on link l for a given variation ΔR_i when the a.n.o.p. assignment is updated as a function of ΔR_i in the manner described above. Let j and k be two v.c's on link l and assume that $D^l_j(0) < D^l_k(0)$. As the variations of the a.n.o.p. are continuous functions of ΔR_i it is clear that we can choose $\Delta R_i > 0$ sufficiently small to insure that $D^l_j(\Delta R_i) < D^l_k(\Delta R_i)$. Accordingly it follows that we can choose $\Delta R_i > 0$ sufficiently small to insure that the initial ordering $\vec{w}^l$ ranks correctly all the v.c.'s whose delay are initially different. Now assume that $D^l_j(0) = D^l_k(0)$. Assume also w.l.o.g. that $w^l_j < w^l_k$. Since $D^l_j(0) = D^l_k(0)$ j and k belong initially to the same priority group, which immediately implies that only k's a.n.o.p. may be increased. Suppose first that $k \neq i$. Then, as R_k is constant, the increase in k's a.n.o.p. implies a corresponding increase in k's delay. As j's delay is constant this means that we can only obtain $D^l_j(\Delta R_i) \leq D^l_k(\Delta R_i)$, so that the ordering $w^l_j < w^l_k$ remains valid. Next suppose that $k = i$. This case presents a special difficulty because i's a.n.o.p. increase tends to increase i's delay while i's rate

increase tends to decrease i's delay, so that it is not clear whether the net variation is positive or negative. It is easy to see that i's delay as a function of ΔR_i is given by:

$$D_i^l(\Delta R_i) = \frac{1}{R_i + \Delta R_i}\left(N_i^l + B^l(w_i^l, \vec{w}^l, \vec{R} + \overrightarrow{\Delta R_i}) - B^l(w_i^l, \vec{w}^l, \vec{R})\right) \tag{5.20}$$

Evaluating the derivative of this expression at $\Delta R_i = 0$ we obtain:

$$\frac{d}{d\Delta R_i} D_i^l(0) = \frac{1}{R_i}\left(-D_i^l(0) + B^{l\prime}(w_i^l, \vec{w}^l, \vec{R})\right) \tag{5.21}$$

If i's a.n.o.p. is increased i must be the v.c. in the priority group e_i^l with the lowest position in the ordering. By definition of the priority groups this implies that i must have full priority over the v.c.'s in position $w_i^l + 1, \ldots, V^l$. As i can be at most of lowest priority as compared to the v.c.'s in position $1, \ldots, w_i^l - 1$, it follows that the initial delay of v.c.i is upper bounded by:

$$D_i^l(0) \leq \frac{1}{R_i}\left(B^l(w_i^l, \vec{w}^l, \vec{R}) - B^l(w_i^l - 1, \vec{w}^l, \vec{R})\right) \tag{5.22}$$

Using equation (5.22) in equation (5.21) we get:

$$\frac{d}{d\Delta R_i} D_i^l(0) \geq \frac{\mu^l}{R_i\left(\mu^l - \sum_{p|w_p^l \leq w_i^l} R_p\right)}\left[-\frac{1}{\mu^l - \sum_{p|w_p^l < w_i^l} R_p} + \frac{1}{\mu^l - \sum_{p|w_p^l \leq w_i^l} R_p}\right] \tag{5.23}$$

This expression is strictly positive, which means that for sufficiently small ΔR_i the delay of v.c. i increases. As D_j^l is constant it follows that for sufficiently small $\Delta R_i > 0$ we obtain $D_j^l(\Delta R_i) \leq D_i^l(\Delta R_i)$, so that the initial ordering $w_i^l < w_j^l$ remains valid.

This shows that by choosing $\Delta R_i > 0$ sufficiently small we can insure that the initial ordering $\vec{w}^l$ remains valid, which was our first assumption. Also as the variations of the a.n.o.p. are continuous functions of ΔR_i we can always choose ΔR_i sufficiently small to insure that all the constraints initially satisfied with strict inequality remain satisfied with strict inequality, which was our second assumption.

Now, having established the validity of our assumptions, we return to equation (5.19). Since the a.n.o.p. assignment is optimal for the rate assignment, the v.c's in each priority group must have the same marginal a.n.o.p. cost. Using this fact it is easy to see that:

$$\begin{aligned} B^{l\prime}(w^l_{\underline{i}(e_i^l)}, \vec{w}^l, \vec{R})\frac{\partial}{\partial N} G_{\underline{i}(e_i^l)}(R_{\underline{i}(e_i^l)}, N_{\underline{i}(e_i^l)}) &= B^{l\prime}(w_i^l, \vec{w}^l, \vec{R})\frac{\partial}{\partial N} G_i(R_i, N_i) \\ &+ \sum_{j|w_i^l < w_j^l \leq w^l_{\underline{i}(e_i^l)}} \left(B^{l\prime}(w_j^l, \vec{w}^l, \vec{R}) - B^{l\prime}(w_j^l - 1, \vec{w}^l, \vec{R})\right)\frac{\partial}{\partial N} G_j(R_j, N_j) \end{aligned} \tag{5.24}$$

Similarly for any priority group $e \in E^l$;

$$\left(B^{l\prime}(w^l_{\underline{i}(e)}, \vec{w}^l, \vec{R}) - B^{l\prime}(w^l_{\underline{i}(e-1)}, \vec{w}^l, \vec{R})\right) \frac{\partial}{\partial N} G_{\underline{i}(e)}(R_{\underline{i}(e)}, N_{\underline{i}(e)})$$
$$= \sum_{j | w^l_{\underline{i}(e-1)} < w^l_j \leq w^l_{\underline{i}(e)}} \left(B^{l\prime}(w^l_j, \vec{w}^l, \vec{R}) - B^{l\prime}(w^l_j - 1, \vec{w}^l, \vec{R})\right) \frac{\partial}{\partial N} G_j(R_j, N_j) \quad (5.25)$$

Using these equations in equation (5.19) the total additional cost on link l for increasing R_i by ΔR_i is:

$$\Bigg\{ B^{l\prime}(w^l_i, \vec{w}^l, \vec{R}) \frac{\partial}{\partial N} G_i(R_i, N_i)$$
$$+ \sum_{j | w^l_j > w^l_i} \left(B^{l\prime}(w^l_j, \vec{w}^l, \vec{R}) - B^{l\prime}(w^l_j - 1, \vec{w}^l, \vec{R})\right) \frac{\partial}{\partial N} G_j(R_j, N_j) \Bigg\} \Delta R_i \quad (5.26)$$

To get the total additional cost per unit increase of ΔR_i (in the limit of small ΔR_i) we only have to sum the last equation over all the links $l \in \mathcal{L}_i$ and divide by ΔR_i. That is this cost is :

$$\sum_{l \in \mathcal{L}_i} \Bigg\{ B^{l\prime}(w^l_i, \vec{w}^l, \vec{R}) \frac{\partial}{\partial N} G_i(R_i, N_i)$$
$$+ \sum_{j | w^l_j > w^l_i} \left(B^{l\prime}(w^l_j, \vec{w}^l, \vec{R}) - B^{l\prime}(w^l_j - 1, \vec{w}^l, \vec{R})\right) \frac{\partial}{\partial N} G_j(R_j, N_j) \Bigg\} \quad (5.27)$$

But this is precisely equal to the right hand side of the equation in the condition (3) of theorem 5.1.

It may be shown using a similar argument that (5.27) also represents the savings per unit decrease of R_i that can be realized by adjusting the a.n.o.p. assignment when R_i is decreased. This confirms the interpretation of the condition (3) of theorem 5.1 that we gave earlier. Namely if the rate assignment is optimal, increasing (resp. reducing) the rate of a v.c., say i, is not profitable because the savings (resp. costs) directly resulting from the rate variation; i.e., $-\frac{\partial}{\partial R} G_i(R_i, N_i) \Delta R_i$, are exactly compensated by the additional costs (resp. savings) incurred for adjusting the a.n.o.p. assignment.

5.3 A distributed algorithm solving the problem (FC_s).

In this section we develop a distributed algorithm solving the problem (FC_s). The algorithm, called Alg_FC_s, is based on similar ideas as Alg_NP_s_e. As in Alg_NP_s_e a major preocupation in the construction of the algorithm is to keep the notation and the proof of convergence relatively simple. For these reasons the algorithm has been built as simply as possible. Also some hypotheses regarding the operation of the algorithm that may in practice be difficult to enforce have been made. We will comment at the end of this chapter on the implications of these restrictions and on how, in practice, they can be avoided.

Before concentrating on Alg_FC_s we introduce some notation. Let $(\vec{R}, \vec{N})$ be a strictly feasible assignment. For $i \in \mathcal{V}^l$, define:

$$f_i^l = \{i\} \cup \{j \mid j \in \mathcal{V}^l,\ |D_j^l - D_k^l| \leq \nu \text{ for some } k \in f_i^l\} \tag{5.28}$$

where $\nu > 0$ is a fixed parameter. Also, define:

$$F^l = \{f_i^l,\ i \in \mathcal{V}^l\} \tag{5.29}$$

We call f_i^l the delay group of v.c. i on link l. It is easy to see that for any $i,\ j \in \mathcal{V}^l$, if $j \in f_i^l$ then $f_i^l = f_j^l$. Moreover for any delay groups $f,\ \tilde{f} \in F^l$, $f \neq \tilde{f}$:

$$f \cap \tilde{f} = \phi \tag{5.30}$$

and either:

$$D_i^l < D_j^l - \nu \quad \text{for all } i \in f,\ j \in \tilde{f} \tag{5.31}$$

or:

$$D_j^l < D_i^l - \nu \quad \text{for all } i \in f,\ j \in \tilde{f} \tag{5.32}$$

Equations (5.31) and (5.32) can be used to order the delay groups on each link. Indeed we will say that $f < \tilde{f}$ whenever f and $\tilde{f}$ satify equation (5.31) and conversely we will say that $f > \tilde{f}$ whenever f and $\tilde{f}$ satisfy equation (5.32). Also if f is a delay group on link l such that $f > \tilde{f}$ for some delay group $\tilde{f} \in F^l$, we define $f - 1$ to be the delay group on link l

immediately preceeding f. Namely, $f-1$ is the delay group on link l satisfying $f-1<f$ and $f-1\geq \hat{f}$ for all $\hat{f}\in F^l$, $\hat{f}<f$.

Let $\vec{w}^l$ be an ordering on link l valid for the assignment $(\vec{R},\vec{N})$ and let $f\in F^l$ be a delay group on link l. Given the ordering $\vec{w}^l$ we denote by $\underline{i}(f)$ and $\bar{i}(f)$ the v.c.'s in f having respectively the lowest and the highest positions in the ordering. Namely $\underline{i}(f)$ and $\bar{i}(f)$ satifies:

$$w^l_{\underline{i}(f)}\geq w^l_j \text{ for all } j\in f \tag{5.33}$$

$$w^l_{\bar{i}(f)}\leq w^l_j \text{ for all } j\in f \tag{5.34}$$

Clearly the delay groups are similar to the priority groups. It may also be noted that although the variables defined above depend on $\vec{R}$, $\vec{N}$, $\vec{w}$ and ν, this dependency has been dropped in the notation. We will indicate explicitely the dependency only when a confusion is possible. However the dependency will in general be obvious from the context.

Now for given $\vec{R}$, $\vec{N}$, $\vec{w}$ and ν we define the following variables which are intimately related to the right hand side of the equation in condition (3) of theorem 5.1. We will discuss later the role of these variables.

$$z^l_i = B^{l\prime}(w^l_i,\vec{w}^l,\vec{R})\frac{\partial}{\partial N}G_i(R_i,N_i) + \sum_{j|w^l_j>w^l_i}\left(B^{l\prime}(w^l_j,\vec{w}^l,\vec{R})-B^{l\prime}(w^l_j-1,\vec{w}^l,\vec{R})\right)\frac{\partial}{\partial N}G_j(R_j,N_j) \tag{5.35}$$

$$\underline{z}^l_i = B^{l\prime}(w^l_{\underline{i}(f^l_i)},\vec{w}^l,\vec{R})\frac{\partial}{\partial N}G_{\underline{i}(f^l_i)}(R_{\underline{i}(f^l_i)},N_{\underline{i}(f^l_i)}) + \sum_{f\in F^l,\ f>f^l_i}\left(B^{l\prime}(w^l_{\underline{i}(f)},\vec{w}^l,\vec{R})-B^{l\prime}(w^l_{\underline{i}(f-1)},\vec{w}^l,\vec{R})\right)\frac{\partial}{\partial N}G_{\underline{i}(f)}(R_{\underline{i}(f)},N_{\underline{i}(f)}) \tag{5.36}$$

$$\bar{z}^l_i = B^{l\prime}(w^l_{\underline{i}(f^l_i)},\vec{w}^l,\vec{R})\frac{\partial}{\partial N}G_{\bar{i}(f^l_i)}(R_{\bar{i}(f^l_i)},N_{\bar{i}(f^l_i)}) + \sum_{f\in F^l,\ f>f^l_i}\left(B^{l\prime}(w^l_{\underline{i}(f)},\vec{w}^l,\vec{R})-B^{l\prime}(w^l_{\underline{i}(f-1)},\vec{w}^l,\vec{R})\right)\frac{\partial}{\partial N}G_{\bar{i}(f)}(R_{\bar{i}(f)},N_{\bar{i}(f)}) \tag{5.37}$$

We also define:

$$z_i=\sum_{l\in\mathcal{L}_i} z^l_i \tag{5.38}$$

$$\underline{z}_i = \sum_{l \in \mathcal{L}_i} \underline{z}_i^l \tag{5.39}$$

$$\overline{z}_i = \sum_{l \in \mathcal{L}_i} \overline{z}_i^l \tag{5.40}$$

As before although the variables defined above depend on $\vec{R}$, $\vec{N}$, $\vec{w}$ and ν this dependency is dropped in the notation. We will indicate explicitly the dependency only when a confusion is possible.

The z_i^l and z_i are fundamental variables. In fact it is easy to see that the condition (3) of theorem 5.1 can be expressed as:

$$-\frac{\partial}{\partial R} G_i(R_i, N_i) = z_i \quad \text{for all } i,\ i = 1, \ldots, V \tag{5.41}$$

$\underline{z}_i^l$ and $\overline{z}_i^l$ are approximations of z_i^l having several important properties. First it is not difficult to see that when the a.n.o.p. assignment is optimal for the rate assignment, and when ν is sufficiently small, we have for all $i = 1, \ldots, V$, and $l \in \mathcal{L}_i$:

$$\underline{z}_i^l = \overline{z}_i^l = z_i^l \tag{5.42}$$

A second property of $\underline{z}_i^l$ and $\overline{z}_i^l$ is that they represent the cost per unit variation of R_i for adjusting the a.n.o.p. assignment on link l when the adjustments are made along two directions that will play a central role in the algorithm.

5.3.1 Description of the algorithm.

Each iteration of Alg_FC_s consists of executing in sequence two algorithms. This can be summarized as follows.

Alg_FC_s:

1) Execute A_n.

2) Execute A_r.

3) End.

A_n is an algorithm updating only the a.n.o.p. assignment. A_r updates both the rate and the a.n.o.p. assignment but its primary concern is to improve the rate assignment. In

fact in A_r the a.n.o.p. assignment is updated uniquely as a consequence of the update of the rate assignment, the motivation being to maintain strict feasibility. We first concentrate on describing A_n and then describe A_r.

The objective of A_n is to improve the a.n.o.p. assignment. This is done in a very similar way as in Alg_NP_s_e. In fact the unique difference between A_n and Alg_NP_s_e is that the a.n.o.p. are the main variables in A_n while the delays are the main variables in Alg_NP_s_e. Clearly since the rates are constant in both cases the formulations are equivalent.

For a given assignment $(\vec{R}, \vec{N})$ define:

$$\Delta_n G_{ij} = \frac{\partial}{\partial N} G_i(R_i, N_i) - \frac{\partial}{\partial N} G_j(R_j, N_j) \tag{5.43}$$

Also let τ_{ij}^l be the maximum value that τ can take in the interval $[0, \bar{\tau}]$ for which the assignment:

$$\begin{aligned} N_i^l &\leftarrow N_i^l - \tau \Delta_n G_{ij} \\ N_j^l &\leftarrow N_j^l + \tau \Delta_n G_{ij} \end{aligned} \tag{5.44}$$

is strictly feasible. $\bar{\tau}$ is a strictly positive parameter of the algorithm to be specified later.

A_n is defined as follows:

A_n: for all l, $l = 1, \ldots, L$, do:

1) Find a pair of v.c.'s i, $j \in \mathcal{V}^l$, $i \neq j$ satisfying:

$$\tau_{ij}^l (\Delta_n G_{ij})^2 = \max_{p,q \in \mathcal{V}^l} \tau_{pq}^l (\Delta_n G_{pq})^2$$

2) Using $\tau = \tau_{ij}^l$, update N_i^l and N_j^l as in equations (5.44).

3) Next l.

Now we concentrate on describing A_r. We first state the algorithm and then follow with a discussion explaining the motivation behind it.

Define:

$$\Delta_r G_i = \begin{cases} 0, & \text{if } \underline{z}_i \leq -\frac{\partial}{\partial R} G_i(R_i, N_i) \leq \overline{z}_i \\ \max\left(-\frac{\partial}{\partial R} G_i(R_i, N_i) - \overline{z}_i, \frac{\partial}{\partial R} G_i(R_i, N_i) + \underline{z}_i \right), & \text{otherwise} \end{cases} \tag{5.45}$$

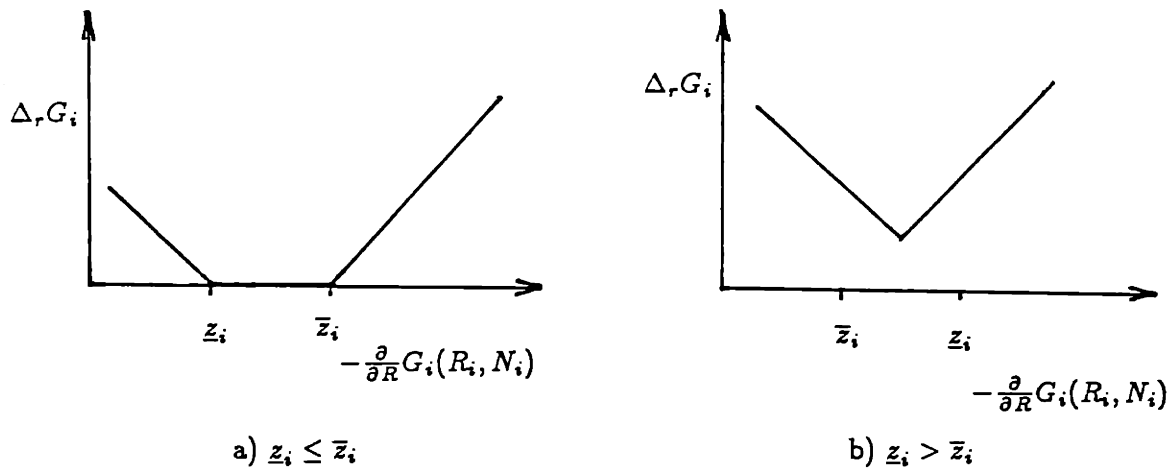


Figure 5.1

Figure 5.1 depicts $\Delta_r G_i$ as a function of $-\frac{\partial}{\partial R}G_i(R_i, N_i)$. Two cases are distinguished depending if $\underline{z}_i \leq \overline{z}_i$ of if $\underline{z}_i > \overline{z}_i$. Note that $\Delta_r G_i$ is always non–negative.

A_r is defined as follows. Here σ is a fixed parameter of the algorithm related to ν which we will discuss later.

A_r:

1) Find a v.c. i satisfying:

$$\Delta_r G_i = \max_j \Delta_r G_j$$

2) If $\Delta_r G_i = 0$ then stop.

3) If $-\frac{\partial}{\partial R}G_i(R_i, N_i) - \overline{z}_i \geq \frac{\partial}{\partial R}G_i(R_i, N_i) + \underline{z}_i$, then set:

$$R_i \leftarrow R_i + \Delta R_i$$

where $\Delta R_i = \sigma \Delta_r G_i$. Also for all $l \in \mathcal{L}_i$ and $f \in F^l$, $f \geq f_i^l$, set:

$$N^l_{i(f)} \leftarrow N^l_{i(f)} + \Delta N^l_{i(f)}$$

where:

$$\Delta N^l_{i(f_i^l)} = B^l(w^l_{i(f_i^l)}, \vec{w}^l, \vec{R} + \overrightarrow{\Delta R_i}) - B^l(w^l_{i(f_i^l)}, \vec{w}^l, \vec{R})$$

and for $f \in F^l,\ f > f_i^l$;

$$\Delta N^l_{\underline{i}(f)} = \left(B^l(w^l_{\underline{i}(f)}, \vec{w}^l, \vec{R} + \overrightarrow{\Delta R_i}) - B^l(w^l_{\underline{i}(f)}, \vec{w}^l, \vec{R})\right) - \left(B^l(w^l_{\underline{i}(f-1)}, \vec{w}^l, \vec{R} + \overrightarrow{\Delta R_i}) - B^l(w^l_{\underline{i}(f-1)}, \vec{w}^l, \vec{R})\right)$$

4) Otherwise (i.e., when $-\frac{\partial}{\partial R}G_i(R_i, N_i) - \overline{z}_i < \frac{\partial}{\partial R}G_i(R_i, N_i) + \underline{z}_i$) set:

$$R_i \leftarrow R_i - \Delta R_i$$

where $\Delta R_i = \sigma \Delta_r G_i$. Also for all $l \in \mathcal{L}_i$ and $f \in F^l,\ f \geq f_i^l$ set:

$$N^l_{\underline{i}(f)} \leftarrow N^l_{\underline{i}(f)} + \Delta N^l_{\underline{i}(f)}$$

where:

$$\Delta N^l_{\underline{i}(f_i^l)} = B^l(w^l_{\underline{i}(f_i^l)}, \vec{w}^l, \vec{R} - \overrightarrow{\Delta R_i}) - B^l(w^l_{\underline{i}(f_i^l)}, \vec{w}^l, \vec{R})$$

and for $f \in F^l,\ f > f_i^l$;

$$\Delta N^l_{\underline{i}(f)} = \left(B^l(w^l_{\underline{i}(f)}, \vec{w}^l, \vec{R} - \overrightarrow{\Delta R_i}) - B^l(w^l_{\underline{i}(f)}, \vec{w}^l, \vec{R})\right) - \left(B^l(w^l_{\underline{i}(f-1)}, \vec{w}^l, \vec{R} - \overrightarrow{\Delta R_i}) - B^l(w^l_{\underline{i}(f-1)}, \vec{w}^l, \vec{R})\right)$$

We now discuss in some details the main ideas behind A_r. We concentrate on the case in which the rate of a v.c. is increased. The case in which the rate of a v.c. is decreased is similar and can be motivated using essentially the same arguments.

Let $(\hat{\vec{R}}, \hat{\vec{N}})$ be the assignment produced by A_r from the assignment $(\vec{R}, \vec{N})$. Also let i be the v.c. whose rate is increased. Clearly in this context:

$$\hat{R}_i = R_i + \sigma \Delta_r G_i$$
$$R_i + \sigma\left[-\frac{\partial}{\partial R}G_i(R_i, N_i) - \overline{z}_i\right] \tag{5.46}$$

Consider the cost incurred for adjusting the a.n.o.p. assignment on link l when the adjustment is made as specified in A_r. For small ΔR_i the cost variation is approximately:

$$\sum_{f \in F^l,\ f \geq f_i^l} \Delta N^l_{\underline{i}(f)} \frac{\partial}{\partial R} G_{\underline{i}(f)}(R_{\underline{i}(f)}, N_{\underline{i}(f)}) \tag{5.47}$$

For small ΔR_i we can also use a linear approximation for estimating the variations of the a.n.o.p. assignment. This gives:

$$\Delta N^l_{\bar{i}(f^l_i)} \approx B^{l\prime}(w^l_{\underline{i}(f^l_i)}, \vec{w}^l, \vec{R})\Delta R_i \tag{5.48}$$

and for $\bar{i}(f)$, $f > f^l_i$:

$$\Delta N^l_{\bar{i}(f)} \approx \left(B^{l\prime}(w^l_{\underline{i}(f)}, \vec{w}^l, \vec{R}) - B^{l\prime}(w^l_{\underline{i}(f-1)}, \vec{w}^l, \vec{R})\right)\Delta R_i \tag{5.49}$$

Using equations (5.48) and (5.49) in equation (5.47) we obtain that the cost of adjusting the a.n.o.p. assignment on link l is approximately;

$$\begin{aligned}\Delta R_i \Bigg\{ & B^{l\prime}(w^l_{\underline{i}(f^l_i)}, \vec{w}^l, \vec{R}) \frac{\partial}{\partial N} G_{\bar{i}(f^l_i)}(R_{\bar{i}(f^l_i)}, N_{\bar{i}(f^l_i)}) \\ & + \sum_{f \in F^l,\ f > f^l_i} \left(B^{l\prime}(w^l_{\underline{i}(f)}, \vec{w}^l, \vec{R}) - B^{l\prime}(w^l_{\underline{i}(f-1)}, \vec{w}^l, \vec{R})\right) \frac{\partial}{\partial N} G_{\bar{i}(f)}(R_{\bar{i}(f)}, N_{\bar{i}(f)}) \Bigg\} \\ & = \overline{z}^l_i \Delta R_i \end{aligned} \tag{5.50}$$

It follows that the overall cost for adjusting the a.n.o.p. assignment is approximately:

$$\begin{aligned} & \sum_{l \in \mathcal{L}_i} \overline{z}^l_i \Delta R_i \\ & = \overline{z}_i \Delta R_i \end{aligned} \tag{5.51}$$

On the other hand increasing i's rate causes a direct variation of i's cost approximately equal to:

$$\frac{\partial}{\partial R} G_i(R_i, N_i)\Delta R_i \tag{5.52}$$

To get the net cost variation we only have to add (5.51) and (5.52). Namely the net cost variation is:

$$\begin{aligned} & \frac{\partial}{\partial R} G_i(R_i, N_i)\Delta R_i + \overline{z}_i \Delta R_i \\ & = -\sigma \left(\frac{\partial}{\partial R} G_i(R_i, N_i) + \overline{z}_i\right)^2 \end{aligned} \tag{5.53}$$

where the last step follows immediately from equation (5.46).

It can be shown using a completely analogous argument that when R_i is decreased the net cost variation is:

$$-\sigma\left(\frac{\partial}{\partial R}G_i(R_i, N_i) + \underline{z}_i\right)^2 \tag{5.54}$$

As (5.53) and (5.54) are always negative we may conclude that A_r always improves the assignment. It may also be noted that the improvement increases as the difference between $-\frac{\partial}{\partial R}G_i(R_i, N_i)$ and $\overline{z}_i$ or $\underline{z}_i$ increases.

A second fundamental property of A_r is that the a.n.o.p. assignment can be updated by substantial amounts without violating feasibility. It is this property that allows us to use a fixed σ. This was one of the main concerns behind our choice of update directions. The reason is that it is in general not sufficient to find a direction in which progress can be made to guarantee the convergence of an algorithm. The direction must be sufficiently good to insure that substantial progress is made when the current assignment is far from optimality. For example we believe that A_r would not converge if the a.n.o.p. assignment was updated in the manner described in section 5.2 in the discussion of the third condition of theorem 5.1. This is because although a strict progress is guaranteed at each iteration, it cannot be guaranteed that the progress does not become arbitrarily small when the current assignment is far from optimality.

The fact that the algorithm uses a fixed σ simplifies considerably its implementation. Indeed a fixed σ means that the processors responsible for updating the rates know a priori that when they update the rates the links can adjust their a.n.o.p. assignment in such a way that strict feasibility is maintained. For example the processor responsible for i's rate knows a priori that if it updates R_i by $\pm\sigma\Delta_r G_i$, the links on i's path can adjust their a.n.o.p. assignment in such a way that the assignment remains strictly feasible. The advantage in this situation is that the processors responsible for the rates do not have to probe the links to determine if a given rate update is feasible.

The fact that a fixed σ can be used follows essentially from the following lemma.

Lemma 5.2: Let $(\vec{R}, \vec{N})$ be a strictly feasible assignment and let $\vec{w}^l$, $l = 1, \ldots, L$, be orderings valid for this assignment. Assume that there exists $K_1 > 0$ and $K_2 > 0$ such that:

$$R_i \geq K_1 \quad \text{for all } i,\ i = 1, \ldots, V$$
$$\mu^l - \sum_{i \in \mathcal{V}^l} R_i \geq K_2 \quad \text{for all } l,\ l = 1, \ldots, L$$

Then, for all $\nu > 0$, there exists $K_3 > 0$ depending only on K_1, K_2 and ν such that for any i, $i = 1, \ldots, V$, and ΔR_i, $|\Delta R_i| \leq K_3$, the assignments:

$$\begin{aligned}
R_i &\leftarrow R_i + \Delta R_i \\
N^l_{\underline{i}(f^l_i)} &\leftarrow N^l_{\underline{i}(f^l_i)} + B^l(w^l_{\underline{i}(f^l_i)}, \vec{w}^l, \vec{R} + \overrightarrow{\Delta R_i}) - B^l(w^l_{\underline{i}(f^l_i)}, \vec{w}^l, \vec{R}) \quad \text{for all } l \in \mathcal{L}_i \\
N^l_{\underline{i}(f)} &\leftarrow N^l_{\underline{i}(f)} + \left[B^l(w^l_{\underline{i}(f)}, \vec{w}^l, \vec{R} + \overrightarrow{\Delta R_i}) - B^l(w^l_{\underline{i}(f)}, \vec{w}^l, \vec{R})\right] \\
&\quad - \left[B^l(w^l_{\underline{i}(f-1)}, \vec{w}^l, \vec{R} + \overrightarrow{\Delta R_i}) - B^l(w^l_{\underline{i}(f-1)}, \vec{w}^l, \vec{R})\right] \text{ for all } f \in F^l,\ f > f^l_i,\ l \in \mathcal{L}_i
\end{aligned}$$

and:

$$\begin{aligned}
R_i &\leftarrow R_i - \Delta R_i \\
N^l_{\underline{i}(f^l_i)} &\leftarrow N^l_{\underline{i}(f^l_i)} + B^l(w^l_{\underline{i}(f^l_i)}, \vec{w}^l, \vec{R} - \overrightarrow{\Delta R_i}) - B^l(w^l_{\underline{i}(f^l_i)}, \vec{w}^l, \vec{R}) \quad \text{for all } l \in \mathcal{L}_i \\
N^l_{\underline{i}(f)} &\leftarrow N^l_{\underline{i}(f)} + \left[B^l(w^l_{\underline{i}(f)}, \vec{w}^l, \vec{R} - \overrightarrow{\Delta R_i}) - B^l(w^l_{\underline{i}(f)}, \vec{w}^l, \vec{R})\right] \\
&\quad - \left[B^l(w^l_{\underline{i}(f-1)}, \vec{w}^l, \vec{R} - \overrightarrow{\Delta R_i}) - B^l(w^l_{\underline{i}(f-1)}, \vec{w}^l, \vec{R})\right] \text{ for all } f \in F^l,\ f > f^l_i,\ l \in \mathcal{L}_i
\end{aligned}$$

are strictly feasible.

This is proven in the section 3 of Appendix D.

We can motivate the intuition behind this lemma via the following example.

Example 5.1: Consider a link of unit capacity shared by two v.c.'s. Let the initial assignment be as follows:

$$\begin{aligned}
R_1 &= 0.1 & R_2 &= 0.1 \\
D_1 &= \frac{10}{9} + \Delta & D_2 &= \frac{25}{18} - \Delta
\end{aligned} \tag{5.55}$$

where Δ is a very small positive number. It is readily verified that this assignment is strictly feasible.

Suppose we want to increase the rate of v.c. 1 and compensate this increase by increasing the a.n.o.p. of v.c. 2. Then it is not difficult to see that the variation of the rate of v.c. 1 can be at most:

$$\frac{.1\Delta + 1/9}{.1\Delta + 10/9} - 0.1 \tag{5.56}$$

For small Δ this is approximately equal to:

$$\frac{1/9}{10/9} + \frac{0.1\Delta}{(10/9)^2} - 0.1$$

$$= \frac{81}{1000}\Delta \tag{5.57}$$

The critical point to observe is that the maximum rate variation is proportional to Δ.

Now suppose that instead of compensating the rate increase of v.c. 1 by increasing the a.n.o.p. of v.c. 2 we compensate by increasing the a.n.o.p. of v.c. 1. Then it is not difficult to see that we can increase R_1 up to the point where the total rate equals the capacity of the link without ever violating the feasibility constraints.

The point of this example is that if we want to compensate an increase in the rate of a v.c. by increasing the a.n.o.p. of a v.c. whose delay is higher that the delay of the v.c. whose rate is increased we may be forced to keep the rate variation very small in order to maintain strict feasibility. This is because in this case the v.c. whose rate is increased may very quickly acquire full priority over the v.c. whose a.n.o.p. is increased. However if the rate increase is compensated by increasing the a.n.o.p. of a v.c. whose delay is initially not larger than the delay of the v.c. whose rate is increased a substantial rate increase can be guaranteed. This is because in this case the a.n.o.p. can be increased by a substantial amount before the v.c. whose a.n.o.p. is increased becomes of lowest priority as compared to the v.c. whose rate is increased. In fact we can guarantee in this case that a minimum rate variation which depends only on the rate assignment is feasible.

The results presented in lemma 5.2 are a generalization of the preceeding discussion. Consider for example the case in which the rate of a v.c. is increased. In this case the v.c. whose a.n.o.p. is increased in each delay group is one of the v.c.'s in the group whose delay is initially minimum. This insures that the a.n.o.p. of this v.c. can be increased substantially

before it becomes of lowest priority as compared to the other v.c.'s in its group. Also, as the delay of v.c.'s in different delay groups differ by at least ν, it is easy to see that a minimum rate increase proportional to ν can be achieved before v.c.'s in different delay groups start interfering. These two facts basically guarantee that a minimum rate increase depending only on the rate assignment and on ν can be achieved, which is what lemma 5.2 asserts.

It is not difficult to see that the sequence of assignments generated by the repeated application of A_n and A_r satisfies the condition of lemma 5.2 for some $K_1 > 0$ and $K_2 > 0$. Indeed if K_1 does not exist it must be that the rate of some v.c. converges to 0. However in view of the form of the cost functions this implies that the objective value converges to ∞. Clearly, under the mild assumption that the initial assignment has a finite objective value, this cannot happen as both A_n and A_r are non–increasing algorithms. Similarly if K_2 does not exist it must be that on some link, say l, $\sum_{i \in \nu^l} R_i$ converges to μ^l. This implies that the a.n.o.p. of some v.c. on the link converges to ∞, which again means that the objective value converges to ∞. Clearly this cannot happen.

If an assignment, say $(\vec{R}, \vec{N})$, satisfies the conditions of lemma 5.2 it follows from the form of the cost functions that the assignment must also satisfy $\Delta_r G_i \leq K_4$, $i = 1, \ldots, V$, for some constant $K_4 > 0$. It is this fact which, together with lemma 5.2, allows us to use a fixed σ. Namely if all the assignments generated by the repeated application of A_n and A_r satisfy the conditions of lemma 5.2 we are guaranteed by the lemma the existence of K_3 such that if $\sigma \Delta_r G_i \leq K_3$ for all i and $\Delta_r G_i$, the assignment resulting from updating R_i by $\pm \sigma \Delta_r G_i$ is strictly feasible. As $\Delta_r G_i \leq K_4$ it follows immediately that we can use a fixed σ as long as $\sigma \leq K_3/K_4$.

The last fundamental fact about A_r is that when it is used in conjunction with A_n it can be shown that each $-\frac{\partial}{\partial R} G_i(R_i, N_i)$ converges to z_i. This means that, eventually, the assignment produced by Alg_FC_s satisfies the third condition of theorem 5.1. Also, using the same arguments as in Alg_NP_s_e, it can be shown that the repeated application of A_n insures that the assignment eventually satisfies the first two conditions of theorem 5.1. It follows from these facts that the assignment produced by Alg_FC_s eventually satisfies all the conditions of theorem 5.1; namely solves the problem (FC_s).

5.3.2 Convergence.

Let $(\vec{R}(0), \vec{N}(0))$ denote the initial assignment. We make the following assumptions.

(A.5.2.1) $S(\vec{R}(0), \vec{N}(0)) < \infty$.

(A.5.2.2) $(\vec{R}(0), \vec{N}(0))$ is strictly feasible.

Let $(\vec{R}(k), \vec{N}(k))$ denote the assignment resulting after k iterations of Alg_FC_s. We have the following result.

Theorem 5.2: For every $(\vec{R}(0), \vec{N}(0))$ satisfying assumptions (A.5.2) there exists $\bar{\tau} > 0$, $\sigma > 0$, and $\nu > 0$ such that:

1) $(\vec{R}(k), \vec{N}(k))$ is strictly feasible, for all $k \geq 0$.

2)

$$\lim_{k \to \infty} S(\vec{R}(k), \vec{N}(k)) = S^*$$

where S^* is the optimal value of (FC_s).

This is proven in section 4 of Appendix D.

5.3.3 Implementation.

The implementation of Alg_FC_s, as described in section 5.3.1, requires a complete coordination among the links and the processors controlling the rate of the v.c.'s. Such a coordination is in practice too costly in terms of time and overhead to be justified. In practice Alg_FC_s should run in an asynchronous manner. Each link should be responsible for carrying the iteration concerning it in A_n, which it should do independently of the other links and nodes. Similarly the home node of each v.c. should update the rate of the v.c. independently of the other nodes and of the links, and the links should adjust asynchronously their a.n.o.p. assignment as a response to a rate update (namely the links should not try to synchronize the update of their a.n.o.p. assignment but rather they should independently update their a.n.o.p. assignment as they become aware of the rate update). Of course convergence is not guaranteed in this context. However we believe that if the

estimates of the $\underline{z}_i$, $\overline{z}_i$ and D_i are updated often as compared to the frequency at which updates are made the convergence properties will be maintained.

From an implementation standpoint the $\underline{z}_i$, $\overline{z}_i$ and D_i are very similar variables. Namely, for each i, D_i is a sum over all the links on i's path of the individual contribution D_i^l of each link l, exactly as $\underline{z}_i$ and $\overline{z}_i$ are sums over the links on i's path of the individual contribution $\underline{z}_i^l$ and $\overline{z}_i^l$ of each link l. In this context we can construct the estimates in Alg_FC_s in essentially the same manner as in Alg_NP_s_a. Namely we can use, for each i, an update_i message cycling along i's path. In addition to constructing an estimate of D_i the message would construct estimates of $\underline{z}_i$ and $\overline{z}_i$ in exactly the same way. Also the message would carry the current value of R_i so that the links would be kept informed of the rate updates made by the home of node i. It may be noted that this implementation scheme is particularly simple and that it imposes a very small overhead.

5.4 Comments.

Basically all the comments made on Alg_NP_s_e still apply to Alg_FC_s. In particular because of the dependency of the parameters σ and $\bar{r}$ on the initial assignment it is in general not practical to enforce in a quasi–static environment the conditions on σ and $\bar{r}$ under which convergence can be guaranteed. However we believe that by choosing these parameter sufficiently small the assignments should track well the evolution of the network. Of course the choice of these parameters depends strongly on the particular application. Simulation would at this point be indicated to tailor the parameters to the application.

In chapter 4 we were able to construct an approximate but completely asynchronous algorithm by exploiting the structure of (NP_s); namely by exploiting the fact that the feasible set could be decomposed on a link basis. This, however, does not seem to be possible in the case of (FC_s). The difficulty is that the rate of each v.c., which is now also variable, ties together the feasible sets on the links on the path of the v.c., thus destroying the decomposition property.

6 Discussion.

The purpose of this chapter is to present some issues related to the results developed in this thesis which we believe are worth investigating. The chapter is divided in several short sections, each section concentrating on a particular issue.

6.1 Choice of the queuing strategy.

We have seen in chapter 3 that in a queuing system satisfying assumptions (A.3.1)–(A.3.3) any feasible delay assignment can be realized by using the cascade scheme. It is however not difficult to see that the cascade scheme is not the only scheme possessing this property*. Hence a natural question that could be raised is whether or not a better universal queuing strategy than the cascade scheme can be found. We believe that, at least in the context of our application, the answer to this question is yes. In particular it would be useful to have a universal queuing strategy minimizing the variance of the delays as such a strategy would result in a smoother operation, which is obviously desirable.

A queuing strategy that deserves more attention is the round robin strategy. In a context similar to our Hahne [Hah86] has shown that round robin scheduling is a particularly simple means via which a min–max fair rate assignment can be obtained. Moreover as round robin scheduling inherently attempts to regularize the transmissions it should result in low variance of the delays. As mentioned in the preceeding paragraph this is clearly desirable.

We do not believe that the classical versions of round robin scheduling (namely the versions described in [Kl76] and [Hah86]) are universal. However we believe that the principle behind round robin scheduling can be generalized to construct a universal queuing strategy inheriting most of the properties of round robin scheduling.

* For example it is readily verified that the scheme which is identical to the cascade scheme except for scheduling in each priority stream the packet transmissions in a last–in–first–out manner is also universal.

6.2 Simulation.

The algorithms developed in chapters 4 and 5 can only be useful if the assumptions and approximations on which their development is based are justified. For this reason a simulation study of the algorithms in a realistic environment should be conducted. Some important issues that this study should address are:

- How severe are the exogenous Poisson arrival and common exponential packet length assumptions, and how severe is approximation (Ap.4.1)? In particular if the cascade scheme and corollary 2.1 are used to convert a target assignment into a queuing strategy that should ideally realize it, how far from the target assignment can the true resulting assignment be in a realistic situation?
- Are the algorithms sufficiently fast to track the evolution of the network in a typical environment?
- How frequently should the local estimates used in Alg_NP_s_e and Alg_FC_s be updated to maintain good convergence properties, and how does this relate to the choice of the parameters of the algorithms?

6.3 Choice of the cost functions.

The preferences of some types of users have been well characterized in the literature. In these cases constructing a cost function reflecting the preferences is relatively straightforward. For example we know that the quality of a packetized voice conversation decreases sharply once the delay has reached a certain threshold, and that the quality decreases smoothly as the rate is reduced. Hence the cost function should, as a function of the delay, have a sharp knee around the threshold, and should, as a function of the rate, be smoothly decreasing.

There are however several types of users whose preferences are currently not well-known. One such type of particular importance is the interactive (or transaction oriented) user. These users generate transactions sequentially, each transaction being generated only after the preceeding transaction has been completed. These users account for a substantial

portion of the traffic in i.s.n. A fundamental characteristic of interactive users is that their rate can only be controlled through controlling their delay. Indeed at low delay the rate is essentially a constant depending only on the nature of the user. In this situation the transactions are processed sufficiently fast so as to not affect the process underlying their generation. At high delay the rate becomes essentially a function of the delay as the users then spend much more time waiting for the completion of the current transaction than in preparing the next transaction once the result of the current transaction is available. Based on this discussion we can conjecture that the cost function of an interactive user should be a strong function of the delay and a weak function of the rate. However the precise form of this function is an open question.

6.4 Generalization of the flow control problem.

In chapter 5 we investigated the flow control problem in the context of the system–oriented approach, and we implicitly assumed that preemptive queuing could be used. We believe that the generalization of the results of chapter 5 to the non–preemptive case is relatively straightforward. Essentially the manner in which the a.n.o.p.'s are updated in a rate update should be modified to account for the new form of the feasible set. The generalization of the results to the user–oriented approach seems, however, more difficult. In particular we believe that constructing an algorithm similar to Alg_FC_s for solving the problem in the user–oriented approach will require some major modifications to the conditions under which the rate updates are authorized.

6.5 Enforcing a rate assignment.

The output of the algorithms presented in chapters 4 and 5 is a desired rate, delay and a.n.o.p. assignment. To be useful this desired assignment must somehow be converted into a practical mechanism via which it can be realized. Of course if the desired rate assignment is realized, then the desired delay and a.n.o.p. assignment can also be realized simply by using at each link the appropriate queuing strategy. Hence the important question is how can the desired rate assignment be realized?

We believe that the well–known end–to–end windowing technique (see for example [Ta81]) is particularly well suited to realize the desired rate assignment in the context of the algorithms developed in chapter 4 and 5. Indeed by selecting the window of each user appropriately we can insure that the desired a.n.o.p. assignment is achieved. In addition we can also select the queuing strategy on each link to insure that the desired rate assignment would result in the desired delay assignment. Assuming that the arrivals can still be considered as Poisson when windows are used, it is not difficult to see that the above choice of windows and queuing strategies must result in the desired assignment being realized. Although the Poisson assumption is clearly unrealistic when windows are used, we believe that the resulting assignment should still be close to the desired assignment. This should be investigated further, especially via simulation.

6.6 Second derivative of the cost functions.

In Alg_NP_s_e and Alg_FC_s the links can use their local estimates of the rate and end–to–end delay of their v.c.'s to determine the second derivative of the cost functions. The second derivative can be used as a means of automatically scaling the step size of the algorithms. In several related applications this feature has been found to improve substantially the speed of convergence.

6.7 Measurements.

It is possible to measure the rate and the delay of the v.c.'s. The measurements can be used in a feedback loop to adjust the window sizes and the parameters of the queuing strategies so as to better realize the desired assignment.

6.8 Integration of routing and flow control.

An important problem in i.s.n. is to determine the routes over which new v.c.'s should be established. Handling this problem is the responsibility of the routing algorithm. Motivated by the fact that routing decisions have a strong impact on the rate and delay of the users, and by the fact that these measures are also central to the flow control problem,

some researchers have proposed approaches integrating routing and flow control in a unified framework [Gaf82,Go80,Ib81]. However these approaches are not very well suited in an i.s.n. because they are not geared toward handling several classes of users with different rate and delay considerations. We believe that integrating routing and the approach to flow control developed in this thesis may be the solution to this problem. In particular we believe that the information used in the flow control algorithm developed in chapter 5 can also be very useful to a routing algorithm.

6.9 Conclusion.

Of course the results presented in this thesis are questionable in many respects. We believe however that they provide some insight on the use of priority queuing in i.s.n. We hope that these results can be useful to stimulate further research.

References

[Av76] M. Avriel, "Nonlinear Programming Analysis and Methods", Prentice Hall Inc., NJ., 1976.

[Be76] D. P. Bertsekas, "Dynamic Programming and Stochastic Control", Academic Press, New–york, 1976.

[Be83] D. P. Bertsekas, "Notes on Non–linear Programming and Discrete–time Optimal Control, LIDS–R–919, Mass. Ins. of Tech., Camb., MA., 1983.

[Bh81] K. Bharath–Kumar, "A New Approach to Performance–Oriented Flow Control", IEEE. Trans. on Comm., Vol. Com–29, No 4, April 1981, pp. 427–435.

[Co80] E. G. Coffman Jr, I. Mitrani, "A Characterization of Waiting Time Performance Realizable by Single Server Queues", Operations Research, Vol. 28, No. 3, Part II, May–June, 1980.

[Gaf82] E. M. Gafni, "The Integration of Routing and Flow Control for Voice and Data in a Computer Communication Network", Ph.D. Dissertation, Dept. of Elect. Eng. and Comp. Science, Mass. Ins. of Tech., Camb., MA., 1982.

[Gal77] R. G. Gallager, "A Minimum Delay Routing Algorithm Using Distributed Computation", IEEE. Trans. on Comm., Vol. Com–25, No. 1, January 1977, pp, 73–85.

[Gal84] R. G. Gallager, D. P. Bertsekas, "Data Communication Network Lecture's Notes", Mass. Ins. of Tech., Camb., MA., 1984.

[Ge80] M. Gerla, L. Kleinrock, "Flow Control, A Comparative Survey", IEEE. Trans. on Comm., Vol. Com.–28, No. 4, April 1980, pp. 553–572.

[Go80] 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. Ins. of Tech., Camb., Ma., 1980.

[Gr81] J. G. Gruber, "Delay Related Issues in Integrated Voice and Data Networks", IEEE. Trans. on Comm., Vol. Com.–29, No. 6, June 1981, pp. 786–800.

[Gr83] J. G. Gruber, N. H. Le, "Performance Requirements for Integrated Voice/Data Networks", IEEE. Journal on Selected Areas in Comm., Vol. Sac.–1, No. 6, December 1983, pp. 981–1005.

[Hah86] E. L. Hahne, "Round Robin Scheduling for Fair Flow Control in Data Communication Networks", To be published at the IEEE International Conference on Communications, Toronto, June 1986. Also report LIDS–P–1537, Mass. Ins. of Tech., Camb., MA., 1986.

[Hay81] H. Hayden, "Voice Flow Control in Integrated Packed Network", M.Eng. Thesis, Dept. of Elect. Eng. and Comp. Science, Mass. Ins. of Tech., Camb., MA., 1981.

[Ib81] 0. C. Ibe, "Flow Control and Routing in an Integrated Voice and Data Communication Network", Ph.D. Dissertation, Dept. of Elect. Eng. and Comp. Science, Mass. Ins. of Tech., Camb., MA., 1981.

[Ir78] M. Irland, "Buffer Management in a Packet Switch", IEEE. Trans. on Comm., Vol. Com.–26, No. 3, March 1978, pp. 328–337.

[Ja80] J. M. Jaffe, "A Decentralized, "Optimal", Multiple–user, Flow Control Algorithm", Proceedings 5^{th} ICCC Conf., Atlanta, GA, Oct. 1980.

[Ja81] J. M. Jaffe, "Flow Control Power is Nondecentralizable", IEEE. Trans. on Comm, Vol. Com–29, No. 9, Sep. 1981, pp. 1301–1306.

[Ja68] N. K. Jaiswal, "Priority Queues", Academic Press, NY., 1968.

[Ke79] F. P. Kelly, "Reversibility and Stochastic Networks", J Wiley, New York, 1979.

[Ki83] B. G. Kim, "Characterization of Arrival Statistics of Multiplexed Voice Packets", IEEE. Journal on Selected Areas in Comm., Vol. Sac.–1, No. 6, December 1983, pp. 1133–1139.

[Kl76] L. Kleinrock, "Queueing Theory", Vol. 2, John Wiley and Sons, 1976.

[Kl78] L. Kleinrock, "On Flow Control in Computer Networks", Proceedings of the International Conference on Communications, Vol. 2, June 1978, pp. 27.2.1–27.2.5.

[Lu84] D. G. Luenberger, "Linear and Nonlinear Programming", Addison Wesley, 1984.

[La77] S. Lam, M. Reiser, "Congestion Control in Store–and–Forward Networks by Input Buffer Limits", National Telecommunications Conference, Los Angeles, December 1977, pp. 12:1–1 12:1–7.

[Mu86] U. Mukherji, "A Schedule–Based Approach for Flow–Control in Data Communication Networks", Ph.D. Dissertation, Dept. of Elect. Eng. and Comp. Science, Mass. Ins. of Tech., Camb., MA., 1986.

[Ra76] E. Raubold, J. Haenle, "A Method of Deadlock–Free Ressources Allocation and Flow Control in Packet Networks", Proceeding IEEE 3^{rd} International Conference on Computer Communications, Toronto, Ontario, Canada, Aug. 1976, pp. 483–487.

[Ro78] F. A. Ros Peran, "Routing to Minimize the Maximum Congestion in a Communication Network", Ph.D. Dissertation, Dept. of Elect. Eng. and Comp. Science, Mass. Ins. of Tech., Camb., MA., 1978.

[So74] M. J. Sobel, "Optimal Operation of Queues", Mathematical Methods in Queuing Theory, Clarke, A. B., (ed.), Springer–Verlag, Berlin, pp. 231–261.

[St74] S. Stidham, N. U. Prabhu, "Optimal Control of Queuing Systems", Mathematical Methods in Queuing Theory, Clarke, A. B., (ed.), Springer–Verlag, Berlin, pp. 263–294.

[Ta81] A. S. Tanenbaum, "Computer Networks", Prentice–hall, N. J., 1981.

[Ti85] Time Magazine, May 20, 1985, pp. 50–51.

[Th84] D. M. Thabit, "Transmission Scheduling to Reduce Convex Delay Costs In Packets Networks", M. S. Thesis, Dept. of Elect. Eng. and Comp. Science, Mass. Ins. of Tech., Camb., MA., 1984.

[Ts84] J. N. Tsitsiklis, D. P. Bertsekas, "Distributed Asynchronous Optimal Routing in Data Networks", LIDS–P–1399, Mass. Ins. of Tech., Camb., MA., 1984.

[We79] C. J. Weinstein, "The Tradeoff Between Delay and TASI Advantage in a Packetized Speech Multiplexer", IEEE. Trans. on Comm., Vol. Com.–27, No. 11, Nov. 1979, pp. 1716–1720.

[Wo82] J. W. Wong, J. P. Sauvé, J. A. Field, "A study of Fairness in Packet–Switching Networks", IEEE. Trans. on Comm., Vol. Com.–39, No. 2, February 1982, pp. 346–353.

Appendix A: Complement to Chapter 2.

A.1 Continuation of the proof of theorem 2.1.

We assume by induction that $k < V$ and that for $i = 1, \ldots, k-1$ there exists a unique $p_i \in [0,1]$ such that:

$$D_i = p_i D_{ps_i} + (1 - p_i) D_{i+1} \tag{A.1}$$

Our aim is to show that there exists a unique $p_k \in [0,1]$ such that:

$$D_k = p_k D_{ps_k} + (1 - p_k) D_{k+1} \tag{A.2}$$

Using equation (2.12) to evaluate D_{ps_k}, we obtain:

$$D_k = \frac{1}{\gamma_k} \left[\frac{p_k \gamma_k + \phi_k}{\mu - p_k \gamma_k - \phi_k} - \frac{\phi_k}{\mu - \phi_k} \right] + (1 - p_k) D_{k+1} \tag{A.3}$$

Define:

$$f_k(p) = \frac{1}{\gamma_k} \left[\frac{p \gamma_k + \phi_k}{\mu - p \gamma_k - \phi_k} - \frac{\phi_k}{\mu - \phi_k} \right] \tag{A.4}$$

It is easily verified that for $p \in [0,1]$ $f_k(p)$ is strictly convex and non–decreasing. Also, $f_k(0) = 0$. Since the equations in proposition (c) hold by assumption, we have:

$$R_1 D_1 + \cdots + R_k D_k \geq \frac{R_1 + \cdots + R_k}{\mu - R_1 - \cdots - R_k} \tag{A.5}$$

Using repeatedly equation (A.1) for $i = 1, \ldots, k-1$ to eliminate $D_1, \ldots, D_{k-1}$ in the left hand side of equation (A.5), we obtain:

$$\frac{p_1 \gamma_1 + \cdots + p_{k-1} \gamma_{k-1}}{\mu - p_1 \gamma_1 - \cdots - p_{k-1} \gamma_{k-1}} + \gamma_k D_k \geq \frac{R_1 + \cdots + R_k}{\mu - R_1 - \cdots - R_k} \tag{A.6}$$

Thus:

$$D_k \geq \frac{1}{\gamma_k} \left[\frac{R_1 + \cdots + R_k}{\mu - R_1 - \cdots - R_k} - \frac{p_1 \gamma_1 + \cdots + p_{k-1} \gamma_{k-1}}{\mu - p_1 \gamma_1 - \cdots - p_{k-1} \gamma_{k-1}} \right] \tag{A.7}$$

Using the following identities which easily follow from conservation:

$$R_1 + \cdots + R_k = p_1 \gamma_1 + \cdots + p_{k-1} \gamma_{k-1} + \gamma_k \tag{A.8}$$

$$\phi_k = p_1 \gamma_1 + \cdots + p_{k-1} \gamma_{k-1} \tag{A.9}$$

equation (A.7) becomes:

$$D_k \geq \frac{1}{\gamma_k}\left[\frac{\gamma_k+\phi_k}{\mu-\gamma_k-\phi_k}-\frac{\phi_k}{\mu-\phi_k}\right] \tag{A.10}$$

This equation implies that:

$$D_k \geq f_k(1) \tag{A.11}$$

Using this equation, the properties of the function $f_k(\cdot)$, and the fact that $D_k \leq D_{k+1}$ we can show in exactly the same manner as in the case of the initial step that there exists at least one $p_k \in [0,1]$ such that equation (A.2) is satisfied.

If strict inequality is achieved in equation (A.11) it follows for exactly the same reasons as in the case of the initial step that p_k is unique (refer to Figure 2.2). If equality is achieved in equation (A.11) and if p_k is not unique we must have (refer to Figure 2.3–a):

$$\left.\frac{df_k(p)}{dp}\right|_{p=1} > \frac{d}{dp}\left[D_k+(p_k-1)D_{k+1}\right] \tag{A.12}$$

Evaluating the derivatives we obtain:

$$\frac{\mu}{(\mu-R_1-\cdots-R_k)^2} > D_{k+1} \tag{A.13}$$

On the other hand, since the set of equations in proposition (c) is satisfied we have:

$$D_{k+1} \geq \frac{1}{R_{k+1}}\left[\frac{R_1+\cdots+R_{k+1}}{\mu-R_1-\cdots-R_{k+1}}-R_1D_1-\cdots-R_kD_k\right] \tag{A.14}$$

For $p_k = 1$ the v.c.'s $1\ldots,k$ have full preemptive priority over the v.c.'s $k+1,\ldots,V$. Accordingly, Kleinrock's conservation equation must apply to the group of v.c.'s $1,\ldots,k$ taken in isolation; namely,

$$R_1D_1+\cdots+R_kD_k = \frac{R_1+\cdots+R_k}{\mu-R_1-\cdots-R_k} \tag{A.15}$$

Replacing in equation (A.14) we get:

$$\begin{aligned} D_{k+1} &\geq \frac{1}{R_{k+1}}\left[\frac{R_1+\cdots+R_{k+1}}{\mu-R_1-\cdots-R_{k+1}}-\frac{R_1+\cdots+R_k}{\mu-R_1-\cdots-R_k}\right] \\ &\geq \frac{\mu}{(\mu-R_1-\cdots-R_{k+1})(\mu-R_1-\cdots-R_k)} \end{aligned} \tag{A.16}$$

This equation contradicts equation (A.13). It follows that p_k is unique, which concludes the proof of the induction step.

To complete the proof of the second implication we must show that equation (2.12) also holds when $i = V$. Since $p_i \in [0,1]$, $i = 1, \ldots, V-1$, the cascade scheme is realizable. Accordingly it follows from lemma 2.1 that:

$$\sum_{i=1}^{V} R_i D_i = \frac{\sum_{i=1}^{V} R_i}{\mu - \sum_{i=1}^{V} R_i} \tag{A.17}$$

Using the same argument that lead to equation (A.7) from equation (A.5), and using equations (A.8) and (A.9), we obtain:

$$\begin{aligned} D_V =& \frac{1}{\gamma_V}\left[\frac{\gamma_V + \phi_V}{\mu - \gamma_V - \phi_V} - \frac{\phi_V}{\mu - \phi_V}\right] \\ =& D_{ps_V} \end{aligned} \tag{A.18}$$

Q.E.D.

A.2 Proof of Corollary 2.1.

Multiplying both sides by $\mu - p_k\gamma_k - \phi_k$ equation (A.3) becomes, for each k, quadratic in p_k. From feasibility the equation admits at least one root in [0,1]. Also, it may easily be seen that a root cannot be negative. The recursive equations given in corollary 2.1 correspond to the smallest root of each of these equations. Since each p_k is unique these roots must be the solution.

Q.E.D.

A.3 Proof of corollary 2.2.

Choose two feasible delay assignments $\vec{\tilde{D}}$ and $\vec{\hat{D}}$. Then theorem 2.1 guarantees that there exist instances of the cascade scheme which respectively realize $\vec{\tilde{D}}$ and $\vec{\hat{D}}$. Pick any $\alpha \in [0,1]$ and let $\vec{D} = (1-\alpha)\vec{\tilde{D}} + \alpha\vec{\hat{D}}$.

Consider the scheme in which at the beginning of each busy period it is decided with probability α, and independently of the state of the system, to use the cascade scheme corresponding to $\vec{\hat{D}}$ for the whole busy period, and with probability $1-\alpha$ to use the cascade scheme corresponding to $\vec{\tilde{D}}$. Clearly, this scheme satisfies assumptions (A.2.2). Since $\rho < 1$ busy periods (of duration and of number served independent of the cascade scheme used) occur infinitely often, implying that this scheme realizes $\vec{D}$.

Q.E.D.

A.4 Results of chapter 2 in the non–preemptive case.

In this section we present the results of chapter 2 as adapted to the situation in which only non–preemptive queuing is allowed. By non–preemptive queuing we mean queuing strategies in which the server must carry to completion the service on the current packet before serving another packet. The results are very similar to those of chapter 2. Indeed, they are obtained using arguments almost indentical to those developped in chapter 2. For this reason the results are given without proof.

We call "non–preemptive version of the cascade scheme" the queuing strategy that result if in the cascade scheme non–preemptive priorities are used instead of preemptive priorities. Apart from allowing only non–preemptive queuing the assumptions made in this section are identical to those of chapter 2.

We have the following results.

Theorem A.1: Let $\vec{D}$ be a given delay assignment satisfying $D_1 \leq \cdots \leq D_V$. Then the following propositions are equivalent:

a) $\vec{D}$ is realizable using a non–preemptive queuing strategy.

b) $\vec{D}$ is realizable using the non–preemptive version of the cascade scheme and the choice of the p_i, $i = 1, \ldots, V-1$, is unique.

c) The following equations are satisfied:

$$\begin{aligned} R_1 D_1 &\geq \frac{R_1(\mu + R_2 + \cdots + R_V)}{\mu(\mu - R_1)} \\ &\vdots \qquad\qquad \vdots \\ R_1 D_1 + \cdots + R_{V-1} D_{V-1} &\geq \frac{(R_1 + \cdots + R_{V-1})(\mu + R_V)}{\mu(\mu - R_1 - \cdots - R_{V-1})} \\ R_1 D_1 + \cdots + R_{V-1} D_{V-1} + R_V D_V &= \frac{R_1 + \cdots + R_V}{\mu - R_1 - \cdots - R_V} \end{aligned}$$

Corollary A.1: Assume that $\vec{D}$ is realizable using a non–preemptive queuing strategy. Also, assume (w.l.o.g.) that $D_1 \leq \cdots \leq D_V$. Then the probabilities $p_1, \ldots, p_{V-1}$ which, in the non–preemptive version of the cascade scheme, realize $\vec{D}$ are given recursively for $k = 1, \ldots, V-1$, by;

$$p_k = \frac{K_k - \sqrt{K_k^2 - 4\Big(D_{k+1} - \frac{1}{\mu}\Big)\gamma_k(D_{k+1} - D_k)(\mu - \phi_k)}}{2\Big(D_{k+1} - \frac{1}{\mu}\Big)\gamma_k}$$

where:

$$K_k = (\mu - \phi_k)\big(D_{k+1} - \frac{1}{\mu}\big) + \gamma_k(D_{k+1} - D_k) - \rho\frac{\mu}{\mu - \phi_k}$$

and where γ_k and ϕ_k are as in equations **(2.4)** and **(2.5)**.

Corollary A.2: The set of delays realizable by non–preemptive queuing strategies is convex.

The only difference between the equations of theorem 2.1–c and of theorem A.1–c is their right hand sides. If fact it may be noted that the right hand side of each equation in theorem A.1–c is strictly bigger than the right hand side of the corresponding equation in theorem 2.1–c. This is a consequence of the fact that by prohibiting preemptive queuing we are restricting ourselves to a smaller feasible delay set.

Appendix B: Complement to Chapter 3.

B.1 Proof of theorem 3.3.

1) Forward implication: If $\vec{D}^*$ is optimal for the problem (P_s), then conditions (1)–(5) of theorem 3.3 are satisfied.

We first show that if the assignment $\vec{D}^*$ is optimal for the (P_s) problem then whenever two v.c.'s i and j satisfy $D_i^* = D_j^*$ these v.c.'s also satisfy $\frac{1}{R_i}C_i'(D_i^*) = \frac{1}{R_j}C_j'(D_j^*)$. We use a contradiction argument.

Assume that $D_i^* = D_j^*$ for some v.c. i and j and assume also w.l.o.g. that $\frac{1}{R_i}C_i'(D_i^*) < \frac{1}{R_j}C_j'(D_j^*)$. If $D_i^* = D_j^*$ it is clear that neither v.c. i or j has full priority over the other v.c. It is not difficult to see that this fact implies that there exists $\Delta D_i > 0$ and $\Delta D_j < 0$ such that the assignment resulting from updating D_i^* by ΔD_i and D_j^* by ΔD_j is strictly feasible. If fact since Kleinrock's conservation equation is maintained it follows that for sufficiently small $\Delta D_i > 0$ and $\Delta D_j < 0$, the resulting assignment is strictly feasible if the variations are made along the direction:

$$R_i\Delta D_i + R_j\Delta D_j = 0 \qquad (B.1)$$

Along this direction the variation of the objective function ΔS corresponding to a given small variation ΔD_i is:

$$\Delta S = \Delta D_i C_i'(D_i^*) + \Delta D_j C_j'(D_j^*) + o(\Delta D_i) + o(\Delta D_j) \qquad (B.2)$$

where $o(\Delta)$ denotes a small order of Δ.

Using equation (B.1) in equation (B.2) we get:

$$\Delta S = R_i\Delta D_i\Big[\frac{1}{R_i}C_i'(D_i^*) - \frac{1}{R_j}C_j'(D_j^*)\Big] + o(\Delta D_i) \qquad (B.3)$$

Since $\frac{1}{R_i}C_i'(D_i^*) < \frac{1}{R_j}C_j'(D_j^*)$ it follows that $\Delta S < 0$ for sufficiently small $\Delta D_i > 0$, which contradicts the optimality of $\vec{D}^*$. Thus we may assume that whenever two v.c.'s i and j satisfy $D_i^* = D_j^*$ they also satisfy $\frac{1}{R_i}C_i'(D_i^*) = \frac{1}{R_j}C_j'(D_j^*)$.

Now we can prove the forward implication. If $\vec{D}^*$ is optimal for the problem (P_s) it is certainly optimal for the (A_s) problem defined based on the ordering $D_1 \leq \cdots \leq D_V$

because the ordering constraint is then redundant by construction. As (A_s) is a convex programming problem this immediately leads to conditions (1)–(4).

It remains to show that condition (5) also holds. We use a contradiction argument.

Suppose that condition (5) does not hold. Then there must be a v.c. i, $1 \leq i \leq V-1$, such that:

$$A_i > A_{i+1} \tag{B.4}$$

W.l.o.g. we assume also that i is such that $A_{p-1} \leq A_p$ for all $p \leq i$ (this is the case if i is the smallest index satisfying equation (B.4)).

Let $j > i$ be the largest index such that the condition $A_p \leq A_i$ holds for all p, $i \leq p \leq j$.

Because condition (2) holds and because of the choice of i and j we must have:

$$D_p^* = A_i \quad \text{for all } p,\ i \leq p \leq j \tag{B.5}$$

It follows from this equation that:

$$\lambda_p^* = 0 \quad \text{for all } p,\ i \leq p \leq j-1 \tag{B.6}$$

Indeed if $\lambda_p^* \neq 0$ for some p, $i \leq p \leq j-1$, condition (4) then guarantees that the p^{th} constraint is satisfied with equality. Since condition (3) also holds this implies that the v.c.'s $1,\ldots,p$ have full priority over v.c. $p+1$. But this means that $D_p^* < D_{p+1}^*$, contradicting equation (B.5).

Now, by definition of A_i we have:

$$0 = C_i'(A_i) - (\lambda_i^* + \cdots + \lambda_V^*)R_i + J_i'(A_i) \tag{B.7}$$

Using equation (B.6), the fact that $A_i \geq A_{i+1}$ and the definition of $J_i(\cdot)$ this equation can be written as:

$$0 = C_i'(A_i) + C_{i+1}'(A_i) - (\lambda_j^* + \cdots + \lambda_V^*)(R_i + R_{i+1}) + J_{i+1}'(A_i) \tag{B.8}$$

Repeating this procedure for $J_{i+1}(\cdot),\ldots,J_{j-1}(\cdot)$, we eventually obtain:

$$0 = \sum_{p=i}^{j} C_p'(A_i) - (\lambda_j^* + \cdots + \lambda_V^*)\sum_{p=i}^{j} R_p + J_j'(A_i) \tag{B.9}$$

Because of the choice of j we have $A_{j+1} > A_i$. Accordingly it follows that $J_j'(A_i) = 0$. Also in view of equation (B.5) and in view of the preliminary result proven at the beginning of this proof we have $\frac{1}{R_p}C_p'(A_i) = \frac{1}{R_q}C_q'(A_i)$ for all $p,\ q,\ i \le p,\ q \le j$. Using these facts in the preceeding equation we obtain:

$$C_i'(A_i) = (\lambda_j^* + \cdots + \lambda_V^*)R_i \qquad (B.10)$$

so that using equations (B.6), (B.7) and (B.10) we can finally conclude that $J_i'(A_i) = 0$.

On the other hand we know that $J_i(D)$ is strictly convex and non–decreasing for $D \ge A_{i+1}$. Since by assumption $A_i > A_{i+1}$ it follows that $J_i'(A_i) > 0$, contradicting the preceeding conclusion. Thus the forward implication is proven.

2) Reverse implication: If $\vec{D}^*$ satisfies conditions (1)–(5) of theorem 3.3 then $\vec{D}^*$ is the optimal solution to the (P_s) problem.

Let $(\vec{D}^*, \vec{\lambda}^*)$ satisfy conditions (1)–(5) of theorem 3.3 and assume that $\vec{D}^*$ is not optimal. Accordingly let $\vec{\tilde{D}}$ be a strictly feasible delay assignment satisfying $S(\vec{D}^*) > S(\vec{\tilde{D}})$. Also, let $\vec{\hat{D}} = (1-\alpha)\vec{D}^* + \alpha\vec{\tilde{D}}$ which in view of the convexity of the set of weakly feasible delay assignments is weakly feasible for all $\alpha \in [0,1]$.

Because conditions (1)–(4) are satisfied, it is known that $\vec{D}^*$ is the optimal solution to the (A_s) problem defined based on the ordering $D_1 \le \cdots \le D_V$. Also, we know from conditions (2) and (5) that $D_1^* = A_1 \le \cdots \le D_V^* = A_V$. Assume first that strict inequality holds throughout; i.e., $D_1^* = A_1 < \cdots < D_V^* = A_V$. Then it follows that there exists $\epsilon > 0$ such that for all $\alpha \in [0,\epsilon]$, the ordering $D_1 \le \cdots \le D_V$ is valid for $\vec{\hat{D}}$. This, together with the convexity of $S(\cdot)$ and the assumption that $S(\vec{D}^*) > S(\vec{\tilde{D}})$, implies that $(\vec{D}^* - \vec{\tilde{D}})$ is a feasible descent direction at $\vec{D}^*$, contradicting the optimality of $\vec{D}^*$ for (A_s). Consequently we may assume that for at least one index i $D_i^* = D_{i+1}^*$ but $\tilde{D}_i > \tilde{D}_{i+1}$ (if $\tilde{D}_i \le \tilde{D}_{i+1}$ for all i, the same convex combination argument leads again to a contradiction).

If $D_i^* = D_{i+1}^*$ it is clear that v.c. i does not have full priority over v.c. $i+1$. Accordingly it follows that the i^{th} feasibility constraint is satisfied with strict inequality. This in view of condition (4) implies that $\lambda_i^* = 0$.

Our initial (A_s) problem is:

$$\begin{aligned} \min & \sum_{j=1}^{V} C_j(D_j) \\ R_1 D_1 & \geq \frac{R_1}{\mu - R_1} \\ \vdots & \quad \geq \quad \vdots \\ R_1 D_1 + \cdots + R_V D_V & \geq \frac{R_1 + \cdots + R_V}{\mu - R_1 - \cdots - R_V} \\ D_1 \leq D_2 & \leq \cdots \leq D_V \end{aligned} \tag{B.11}$$

For this problem define the Lagrangian:

$$\begin{aligned} L(\vec{D}, \vec{\varsigma}, \vec{\beta}) =& \sum_{j=1}^{V} C_j(D_j) + \sum_{j=1}^{V} \varsigma_j \left(\frac{R_1 + \cdots + R_j}{\mu - R_1 - \cdots - R_j} - R_1 D_1 - \cdots - R_j D_j \right) \\ & + \sum_{j=1}^{V-1} \beta_j (D_j - D_{j+1}) \\ =& \sum_{j=1}^{V} \left(C_j(D_j) - R_j D_j (\varsigma_j + \cdots + \varsigma_V) \right) \\ & + \sum_{j=1}^{V} \varsigma_j \left(\frac{R_1 + \cdots + R_j}{\mu - R_1 - \cdots - R_j} \right) + \sum_{j=1}^{V-1} \beta_j (D_j - D_{j+1}) \end{aligned} \tag{B.12}$$

Consider the assignment:

$$D_j = A_j, \quad \varsigma_j = \lambda_j^*, \qquad j = 1, \ldots, V, \quad \beta_j = 0 \quad j = 1, \ldots, V-1 \tag{B.13}$$

It satisfies $D_1 \leq \cdots \leq D_V$, $\vec{D} \in H$, $\vec{\varsigma} \geq 0$, $\vec{\beta} \geq 0$ and complementary slackness. Thus to show that this assignment is optimal it only remains to show that $\vec{D}^*$ achieves the minimum of $L(\vec{D}, \vec{\varsigma}, \vec{\beta})$ when $\vec{\varsigma} = \vec{\lambda}^*$ and $\vec{\beta} = \vec{0}$. Since the Lagrangian is a convex function, a sufficient condition to guarantee that $\vec{D}^*$ achieves the minimum is that all the partial derivatives of the Lagrangian with respect to $\vec{D}$ vanish.

Consider the partial derivative of the Lagrangian with respect to D_j. Evaluating this derivative and simplifying using equation (B.13) we obtain for all j, $j = 1, \ldots, V$;

$$\frac{\partial}{\partial D_j} L(\vec{D}^*, \vec{\lambda}^*, \vec{0}) = C_j'(D_j^*) - (\lambda_j^* + \cdots + \lambda_V^*) R_j \tag{B.14}$$

However it follows immediately from conditions (2) and (5) that the right hand side of this equation is zero. Thus we may conclude that the assignment (B.13) is optimal.

Now consider the new (A_s) problem, say (A'_s), in which the positions of v.c. i and $i+1$ are interchanged. Specifically this problem is:

$$\begin{aligned}
&\min \sum_{j=1}^{V} C_j(D_j) \\
R_1 D_1 &\geq \frac{R_1}{\mu - R_1} \\
\vdots \quad &\geq \qquad \vdots \\
R_1 D_1 + \cdots + R_{i-1} D_{i-1} &\geq \frac{R_1 + \cdots + R_{i-1}}{\mu - R_1 - \cdots - R_{i-1}} \\
R_1 D_1 + \cdots + R_{i-1} D_{i-1} + R_{i+1} D_{i+1} &\geq \frac{R_1 + \cdots + R_{i-1} + R_{i+1}}{\mu - R_1 - \cdots - R_{i-1} - R_{i+1}} \\
R_1 D_1 + \cdots + R_i D_i + R_{i+1} D_{i+1} &\geq \frac{R_1 - \cdots - R_{i-1} - R_i - R_{i+1}}{\mu - R_1 - \cdots - R_{i-1} - R_i - R_{i+1}} \\
\vdots \quad &\geq \qquad \vdots \\
R_1 D_1 + \cdots + R_V D_V &\geq \frac{R_1 + \cdots + R_V}{\mu - R_1 - \cdots - R_V} \\
D_1 \leq D_2 \leq \cdots \leq D_{i-1} \leq D_{i+1} &\leq D_i \leq D_{i+2} \cdots \leq D_V
\end{aligned} \tag{B.15}$$

Similarly as for the initial (A_s) problem define the Lagrangian for the new (A_s) problem as:

$$\begin{aligned}
L'(\vec{D}', \vec{\varsigma}', \vec{\beta}') = &\sum_{j=1}^{V} C_j(D'_j) \\
&+ \sum_{j=1,\ j\neq i}^{V} \varsigma'_j \left(\frac{R_1 + \cdots + R_j}{\mu - R_1 - \cdots - R_j} - R_1 D'_1 - \cdots - R_j D'_j \right) \\
&+ \varsigma'_i \left(\frac{R_1 + \cdots + R_{i-1} + R_{i+1}}{\mu - R_1 - \cdots - R_{i-1} - R_{i+1}} \right. \\
&\left. - R_1 D'_1 - \cdots - R_{i-1} D'_{i-1} - R_{i+1} D'_{i+1} \right) \\
&+ \beta'_1 (D'_1 - D'_2) + \cdots + \beta'_{i-2} (D'_{i-2} - D'_{i-1}) \\
&+ \beta'_{i-1} (D'_{i-1} - D'_{i+1}) + \beta'_i (D'_{i+1} - D'_i) + \beta'_{i+1} (D'_i - D'_{i+2}) \\
&+ \beta'_{i+2} (D'_{i+2} - D'_{i+3}) + \cdots + \beta'_{V-1} (D'_{V-1} - D'_V)
\end{aligned} \tag{B.16}$$

It is easily checked that again the assignment:

$$D'_j = A_j, \quad \varsigma'_j = \lambda^*_j \quad j = 1,\ldots,V, \quad \beta'_j = 0 \quad j = 1,\ldots,V-1 \qquad (B.17)$$

satisfies $D'_1 \leq \cdots \leq D'_{i-1} \leq D'_{i+1} \leq D'_i \leq D'_{i+2} \leq \cdots \leq D'_V$, $\vec{D}' \in H'$ (H' being now the set corresponding to the new ordering), $\vec{\varsigma}' \geq 0$, $\vec{\beta}' \geq 0$ and complementary slackness (because $\lambda^*_i = 0$). Thus to show that this assignment is optimal for the new problem it only remains to show that $\vec{D}^*$ achieves the minimum of $L'(\vec{D}', \vec{\varsigma}', \vec{\beta}')$ when $\vec{\varsigma}' = \vec{\lambda}^*$ and $\vec{\beta}' = \vec{0}$.

Consider the partial derivative of the Lagrangian with respect to D_j, where $j \neq i$. Evaluating this derivative and using equation (B.17) we obtain:

$$\frac{\partial}{\partial D_j} L'(\vec{D}^*, \vec{\lambda}^*, \vec{0}) = C'_j(D^*_j) - (\lambda^*_j + \cdots + \lambda^*_V) R_j \qquad (B.18)$$

Clearly the right hand side of the preceeding equation is identical to that of equation (B.14). Accordingly it follows that the partial derivative of $L'(\vec{D}^*, \vec{\lambda}^*, \vec{0})$ with respect to D_j is zero for all $j \neq i$.

Now consider the partial derivative of the Lagrangian with respect to D_i. Evaluating this derivative we obtain:

$$\frac{\partial}{\partial D_i} L'(\vec{D}^*, \vec{\lambda}^*, \vec{0}) = C'_i(D^*_i) - (\lambda^*_{i+1} + \cdots + \lambda^*_V) R_i \qquad (B.19)$$

However as $\lambda^*_i = 0$ the right hand side of this equation is also equivalent to that of equation (B.14). Accordingly it follows that the partial derivative with respect to D_i also vanishes.

This shows that in the new (A_s) problem in which the positions of v.c.'s i and $i+1$ are interchanged, $\vec{D}^*$ is still the optimal solution. If there is another index i' such that $D^*_{i'} = D^*_{i'+1}$ but $\tilde{D}_{i'} > \tilde{D}_{i'+1}$, the above argument can be repeated; the consequence being that in the new problem where the positions of v.c.'s i' and $i'+1$ are interchanged $\vec{D}^*$ is still optimal. Since there is a finite number of possible interchange (at most $V!$), we will eventually obtain that $\vec{D}^*$ is the optimal solution to the (A_s) problem defined by the ordering corresponding to the assignment $\vec{\tilde{D}}$. This however is a contradiction as we assumed that $S(\vec{D}^*) > S(\vec{\tilde{D}})$.

Q.E.D.

B.2 Proof of corollary 3.1.

Since $A_1 \leq \cdots \leq A_V$ it follows from lemma 3.1 that:

$$D_1^* = A_1 \leq D_2^* = A_2 \leq \cdots \leq D_V^* = A_V \tag{B.20}$$

By definition A_i satisfies:

$$C_i'(A_i) - (\lambda_i^* + \cdots + \lambda_V^*)R_i + J_i'(A_i) = 0 \tag{B.21}$$

Now since $A_i \leq A_{i+1}$ for all i, $i = 1, \ldots, V-1$, it follows that $J_i'(A_i) = 0$ for all i, $i = 1, \ldots, V-1$. Moreover since $J_V(\cdot)$ is a constant we also have $J_V'(\cdot) = 0$. Using these facts in the preceeding equation we obtain that for all i, $i = 1 \ldots, V$:

$$C_i'(D_i^*) - (\lambda_i^* + \cdots + \lambda_V^*)R_i = 0 \tag{B.22}$$

Q.E.D.

B.3 Proof of correctness of Alg_P_s.

We first show that the minimization in step (3) always has a solution and that this solution is positive. Consider the first iteration. Since the functions $C_i(\cdot)$, $i = 1, \ldots, V$, are strictly convex and non–decreasing it follows that for $\gamma \leq 0$ we obtain $D_i = 0$, $i = 1, \ldots, V$. Clearly this assignment does not satisfies the equations in step (3). On the other hand as we also have that the functions $C_i(\cdot)$, $i = 1, \ldots, V$, are twice continuously differentiable in the interval $[0, \infty[$ it follows that $D_i \to \infty$, $i = 1, \ldots, V$, as $\gamma \to \infty$. As the right hand side of the equations in step (3) is bounded this means that there exists a finite γ for which all the equations in step (3) are satisfied. Thus we may conclude that γ solving step (3) in the first iteration exists and is positive.

Now the existence of a solution in the first iteration guarantees that step (3) has also a solution $\gamma > 0$ in the second iteration. Indeed using $\gamma \leq 0$ in the second iteration we obtain $D_j = 0$, $j = i + 1, \ldots, V$, where i is the largest constraint for which equality was achieved in the first iteration. As the right hand side of the i^{th} constraint in the first iteration is strictly smaller than the right hand side of the constraints $i + 1, \ldots, V$ in the second iteration it immediately follows that the preceeding assignment cannot satisfies the constraints of step (3) in the second iteration. This shows that we must still have $\gamma > 0$ in the second iteration. Also by increasing γ in the second iteration until it reaches the value it had in the first iteration the assignment obtained in the first iteration is reproduced. The equations of step (3) in the second iteration are satisfied for this assignment because they are a subset of the equations the assignment had to satisfy in the first iteration. This shows that γ solving step (3) in the second iteration exists and is positive. This argument can easily be generalized.

Next we show that the algorithm terminates in at most V steps. Initially T starts at 1. Since in step (4) $i' \geq T$, it follows that T increases by at least 1 at each passage at step (4). Thus after at most V iterations the condition in step (5) becomes true and the algorithm stops.

It remains to show that the assignment produced by the algorithm solves the problem (P_s). We do this by showing that the assignment $(\vec{D}, \vec{\lambda})$ produced by the algorithm satisfies the conditions (1)–(5) of theorem 3.3.

First we show that condition (1) holds, namely that $\vec{\lambda} \geq 0$. Let $\gamma^{(p)}$ denote the value of γ in the p^{th} iteration. Also let i be the index of the largest constraint for which equality is achieved in the first iteration. We have already shown that $\gamma^{(1)} > 0$. Accordingly it follows that $\lambda_i = \gamma^{(1)} > 0$. If $i = V$ the proposition is true and if $i < V$ λ_i is modified exactly once: in the second iteration. Now in the second iteration it is known because of the choice of i that for $\gamma^{(2)} = \gamma^{(1)}$ the constraints $i+1, \ldots, V$ are all satisfied with strict inequality. Accordingly it follows that the $\gamma^{(2)}$ solving step (3) in the second iteration must satisfy $\gamma^{(2)} \leq \gamma^{(1)}$. This imply that after update $\lambda_i \leftarrow \gamma^{(1)} - \gamma^{(2)}$ is still non–negative. This argument can easily be generalized into an inductive step.

Next we show that in step 5–c the delays are assigned in increasing order. By construction of step (3) it is clear that the delays assigned in an iteration are in increasing order. Thus we may concentrate on the case of the delays assigned in different iterations.

Assume that in the first iteration the first i delays are assigned. By construction of step (3) this implies that the i^{th} constraint must be satisfied with equality. In addition we also have by construction of step (3) that the $i-1^{th}$ constraint is satisfied. Using these facts it follows that :

$$D_i \leq \frac{1}{R_i}\left[\frac{R_1 + \cdots + R_i}{\mu - R_1 - \cdots - R_i} - \frac{R_1 + \cdots + R_{i-1}}{\mu - R_1 - \cdots - R_{i-1}}\right] \qquad (B.23)$$

In the second iteration D_{i+1} is the smallest delay assigned. Since $D_1, \ldots, D_i$ do not change in the second iteration and since the first constraint in step (3) must be satisfied it follows that:

$$D_{i+1} \geq \frac{1}{R_{i+1}}\left[\frac{R_1 + \cdots + R_{i+1}}{\mu - R_1 - \cdots - R_{i+1}} - \frac{R_1 + \cdots + R_i}{\mu - R_1 - \cdots - R_i}\right] \qquad (B.24)$$

Comparing equations (B.23) and (B.24) it is not difficult to see that $D_{i+1} > D_i$. This shows that in the first two iterations the delays are assigned in increasing order. This can straightforwardly be extended.

A direct consequence of the last result is that conditions (3) and (4); i.e., feasibility and complementary slackness, are satisfied. The first iteration produces $D_1, \ldots, D_i$. Because the delays are assigned in increasing order it is known that these delays are the first i smallest delays. Accordingly it follows that we can define the first i feasibility constraints based on the ordering $D_1 \leq \cdots \leq D_i$. Namely these constraints are:

$$\begin{aligned} R_1 D_1 &\geq \frac{R_1}{\mu - R_1} \\ \vdots \quad &\geq \quad \vdots \\ R_1 D_1 + \cdots + R_i D_i &\geq \frac{R_1 + \cdots + R_i}{\mu - R_1 - \cdots - R_i} \end{aligned} \tag{B.25}$$

Clearly by construction of step (3) the delays $D_1, \ldots, D_i$ produced by the first iteration satisfy these constraints. Also we may have $\lambda_i > 0$ but since we also have:

$$R_1 D_1 + \cdots + R_i D_i = \frac{R_1 + \cdots + R_i}{\mu - R_1 - \cdots - R_i} \tag{B.26}$$

it follows that complementary slackness is satisfied up to equation (i).

Similarly in the second iteration the delays $D_{i+1}, \ldots, D_j$ are assigned, where j is the index of the largest constraint satisfied with equality in the second iteration. By construction of step (3) these delays satisfy:

$$\begin{aligned} R_1 D_1 + \cdots + R_{i+1} D_{i+1} &\geq \frac{R_1 + \cdots + R_{i+1}}{\mu - R_1 - \cdots - R_{i+1}} \\ \vdots \quad &\geq \quad \vdots \\ R_1 D_1 + \cdots + R_j D_j &\geq \frac{R_1 + \cdots + R_j}{\mu - R_1 - \cdots - R_j} \end{aligned} \tag{B.27}$$

Since $D_{i+1}, \ldots, D_j$ are the smallest delays after $D_1, \ldots, D_i$, it follows that we can define the feasibility constraints $i+1, \ldots, j$ based on the ordering $D_1 \leq \cdots \leq D_j$. Clearly equations (B.27) insure that these feasibility constraints are satisfied. It is also immediate to see that complementary slackness is now satisfied up to the constraint j as the two equations corresponding to the non–zero multipliers λ_i and λ_j are satisfied with equality. This argument can easily be generalized.

Finally we may see from the fact that the assignment $(\vec{D}, \vec{\lambda})$ produced by the algorithm satisfies $\vec{\lambda} \geq 0$ and $D_1 \leq \cdots \leq D_V$ that the conditions (2) and (5) of theorem 3.3 hold. Indeed A_V satisfies by definition:

$$A_V = \underset{D}{\operatorname{argmin}} \Big[C_V(D) - R_V \lambda_V D \Big] \tag{B.28}$$

That is A_V is such that:

$$C'_V(A_V) = R_V \lambda_V \tag{B.29}$$

Assuming that the algorithm took P iterations to complete D_V satisfies:

$$C'_V(D_V) = R_V \gamma^{(P)} \tag{B.30}$$

As $\gamma^{(P)} = \lambda_V$ it follows that the solution of equation (B.29) is $A_V = D_V$.

Similarly A_{V-1} satisfies by definition:

$$A_{V-1} = \underset{D}{\operatorname{argmin}} \Big[C_{V-1}(D) - R_{V-1}(\lambda_{V-1} + \lambda_V) D + J_{V-1}(D) \Big] \tag{B.31}$$

or, equivalently, A_{V-1} is such that:

$$C'_{V-1}(A_{V-1}) + J'_{V-1}(A_{V-1}) = R_{V-1}(\lambda_{V-1} + \lambda_V) \tag{B.32}$$

If $\lambda_{V-1} = 0$ D_{V-1} and D_V were both determined during the last iteration. Then D_{V-1} satisfies:

$$C'_{V-1}(D_{V-1}) = R_{V-1} \gamma^{(P)} \tag{B.33}$$

so that since $D_{V-1} \leq D_V$ and $\lambda_V = \gamma^{(P)}$ the solution of equation (B.32) is $A_{V-1} = D_{V-1}$.

If $\lambda_{V-1} \neq 0$ D_{V-1} was determined in the next to last iteration. Then it follows that D_{V-1} satisfies:

$$C'_{V-1}(D_{V-1}) = R_{V-1} \gamma^{(P-1)} \tag{B.34}$$

As λ_V is set to $\gamma^{(P)}$ in the last iteration, and as λ_{V-1} is set to $\gamma^{(P-1)}$ in the next to last iteration and is updated to $\gamma^{(P-1)} - \gamma^{(P)}$ in the last iteration it follows from the preceeding equation that:

$$C'_{V-1}(D_{V-1}) = R_{V-1}(\lambda_{V-1} + \lambda_V) \tag{B.35}$$

Using this equation and the fact that $D_{V-1} \leq D_V$ we obtain that $A_{V-1} = D_{V-1}$ is again the solution of equation (B.32). Continuing this way we find that $A_i = D_i$, $i = 1, \ldots, V$. Since $D_1 \leq \cdots \leq D_V$ it follows that $A_1 \leq \cdots \leq A_V$ which is condition (5). It also follows from the facts that $D_i = A_i$, $i = 1, \ldots, V$, and that $D_1 \leq \cdots \leq D_V$ that $\vec{D}$ is the assignment produced by lemma 3.1 when $\vec{\lambda}$ is used, which is condition (2).

Q.E.D.

B.4 Proof of lemma 3.2.

Using theorem 3.2 it follows that $(\vec{D}^*, \gamma_1^*)$ and $\vec{\psi}^*$ are respectively an optimal solution to $(P_{u,1})$ and a Lagrange multiplier vector if and only if the following conditions are satisfied.

1) $\vec{\psi}^* \geq 0$.

2) $\vec{D}^*$ is weakly feasible.

3) $C_i(D_i^*) \leq \gamma_1^* \quad i = 1, \ldots, V$.

4) $\psi_i^* = 0$ if $C_i(D_i^*) < \gamma_1^*$.

5) $(\vec{D}^*, \gamma_1^*)$ achieves the minimum in the dual functional when $\vec{\psi} = \vec{\psi}^*$.

Clearly the conditions (1), (2) and (3) of lemma 3.2 must hold as they are identical to the conditions (1), (2) and (3) given above.

Note that $\gamma_1^* < \max_{1 \leq i \leq V} C_i(D_i^*)$ is impossible because of condition (3) and that $\gamma_1^* > \max_{1 \leq i \leq V} C_i(D_i^*)$ is also impossible for otherwise it would be possible to reduce γ_1^*, contradicting its optimality. It follows that $\gamma_1^* = \max_{1 \leq i \leq V} C_i(D_i^*)$. Using this fact and the condition (4) above we obtain the condition (4) of lemma 3.2.

Now we prove that conditions (5) and (6) of lemma 3.2 are equivalent to the condition (5) given above. We can write the dual functional as:

$$q(\vec{\psi}) = \min_{\gamma_1} \Big(1 - \sum_{i=1}^{V} \psi_i\Big)\gamma_1 + \min_{\vec{D} \text{ weakly feasible}} \sum_{i=1}^{V} \psi_i C_i(D_i) \qquad (B.36)$$

Suppose $\sum_{i=1}^{V} \psi_i \neq 1$. Then because γ_1 is unconstrained in sign the first term in the right hand side can be made arbitrarily small. Also because we assume that the cost functions are increasing over $[0, \infty[$ it follows that the second term can always be bounded from above. Using these facts we obtain that any $\vec{\psi}$ satisfying $\sum_{i=1}^{V} \psi_i \neq 1$ also satisfies $q(\vec{\psi}) = -\infty$. Hence as $q(\vec{\psi}^*) = \gamma_1^* = \max_{1 \leq i \leq V} C_i(D_i^*)$, and as the cost functions are positive it follows that $\vec{\psi}^*$ satisfies:

$$\sum_{i=1}^{V} \psi_i^* = 1 \qquad (B.37)$$

which is condition (6) of lemma 3.2. Also using the preceeding equation in equation (B.36) we obtain that $(\vec{D}^*, \gamma_1^*)$ achieves the minimum in $q(\vec{\psi}^*)$ when $\vec{\psi} = \vec{\psi}^*$ if and only if $\vec{D}^*$

minimizes the second term in the right hand side of equation (B.36) when $\vec{\psi} = \vec{\psi}^*$, which is condition (5) of lemma 3.2.

Q.E.D.

B.5 Proof of correctness of Alg_P_u.

Using the same argument as in the proof of Alg_P_s it is not difficult to see that there always exists a solution in step (3) of Alg_P_u and that the algorighm terminates in at most V iterations.

It remains to show that the assignment produced by the algorithm is the optimal solution to the problem (P_u). We do this by showing that the assignment produced by the algorithm is identical to the one that would be obtained by solving the hierarchy of nested problems.

Let $\vec{D}^{(1)}$ be the assignment produced by the first iteration of Alg_P_u. Let $\gamma^{(1)}$ be the value that γ takes in the first iteration and let $T^{(1)}$ be the value of the parameter T at the end of the first iteration. Suppose $\vec{D}^{(1)}$ is not an optimal solution to the first problem in the hierarchy. Then there exists $\vec{D}^*$ such that:

$$\max_{1\le i\le V} C_i(D_i^*) < C_1(D_1^{(1)}) = \cdots = C_V(D_V^{(1)}) = \gamma^{(1)} \tag{B.38}$$

Let i be an index such that $C_i(D_i^*) = \max_{1\le j\le V} C_j(D_j^*)$ and consider the assignment:

$$\begin{aligned} \tilde{D}_i &= D_i^* \\ \tilde{D}_j &= C_j^{-1}[C_i(D_i^*)] \quad j \neq i \end{aligned} \tag{B.39}$$

Since $C_j^{-1}(\cdot)$, $j = 1,\ldots,V$, is increasing and since $C_i(D_i^*) \ge C_j(D_j^*)$ we have:

$$C_j^{-1}[C_i(D_i^*)] \ge C_j^{-1}[C_j(D_j^*)] = D_j^* \tag{B.40}$$

That is $\vec{\tilde{D}} \ge \vec{D}^*$ or, otherwise stated, $\vec{\tilde{D}}$ is weakly feasible. It follows that $\vec{\tilde{D}}$ satisfies the equations of step (3) since in the first iteration these equations are the feasibility constraints. Also by construction $\vec{\tilde{D}}$ satisfies:

$$C_1(\tilde{D}_1) = \cdots = C_V(\tilde{D}_V) \tag{B.41}$$

It follows that:

$$\tilde{\gamma} = C_1(\tilde{D}_1) = C_i(D_i^*) < C_1(D_1^{(1)}) = \gamma^{(1)} \tag{B.42}$$

solves step (3) in the first iteration of Alg_P_u. This however contradicts the fact that $\gamma^{(1)}$ is minimum. Thus the first iteration of Alg_P_u solves the first problem in the hierarchy.

We now show that the v.c.'s in $\mathcal{U}_{\gamma_1^*}$ are the v.c.'s $1,\ldots,T^{(1)}-1$ (where the labelling of the v.c.'s is defined by the ordering of the delays as in step (3)) and that these v.c.'s are correctly assigned their delay in the first iteration.

By definition of $T^{(1)}$ the largest equation satisfied with equality in step (3) in the first iteration is the equation $T^{(1)}-1$. As the equations in step (3) in the first iteration correspond to the feasibility constraints, and as by construction all the v.c.'s have the same delay cost in the assignment $\vec{D}^{(1)}$ it follows that it is not possible to reduce the delay of a v.c. i, $1 \leq i \leq T^{(1)}-1$, without either violating feasibility or having to increase the delay of another v.c. whose delay cost is as high as i's. Thus the v.c.'s $1,\ldots,T^{(1)}-1$ belong to $\mathcal{U}_{\gamma_1^*}$. However as the constraint $T^{(1)}-1$ is the highest constraint satisfied with equality and as the constraints in the first iteration are the feasibility constraints it follows that it is possible to reduce D_j, $j=T^{(1)},\ldots,V$, without violating feasibility. This implies that the v.c.'s $T^{(1)},\ldots,V$ are not in $\mathcal{U}_{\gamma_1^*}$, so that we must have $\mathcal{U}_{\gamma_1^*}=\{1,\ldots,T^{(1)}-1\}$. Since only the v.c.'s $1,\ldots,T^{(1)}-1$ are assigned their delay in step (4) it follows that the first iteration correctly assign their delay to the v.c.'s in $\mathcal{U}_{\gamma_1^*}$.

This shows that the first iteration of Alg_P_u solves the first problem in the hierarchy and correctly assigns their delay to the v.c.'s experiencing the worst delay cost. The argument developped above can stratghtforwardly be generalized into an induction proof in which the induction step would show that the k^{th} iteration of Alg_P_u solves the k^{th} problem in the hierarchy and correctly assigns their delay to the v.c.'s experiencing the k^{th} worst delay cost. This is left to the reader.

Q.E.D.

B.6 Results of chapter 3 in the non–preemptive case.

We assume that the system satisfies assumptions (A.2.1)–(A.2.3).

The problem in the system–oriented approach, called (P_s^{np}), is defined as follows:

$$(P_s^{np}) \qquad \min \quad \sum_{i=1}^{V} C_i(D_i)$$

$$\vec{D} \text{ realizable using a non–preemptive queuing strategy}$$

where we again assume that the cost functions satisfy assumptions (A.3.1).

Similarly as in chapter 3 a familly of auxiliary problem is defined. Each auxiliary problems is obtained by requiring that the delay assignment in (P_s^{np}) satisfy a particular ordering. The auxiliary problem associated with the ordering $D_1 \leq \cdots \leq D_V$, which is generically called (A_s^{np}), is:

$$(A_s^{np}) \qquad \min_{\vec{D} \in H^{np},\ D_1 \leq \cdots \leq D_V} \sum_{i=1}^{V} C_i(D_i)$$

where:

$$H^{np} = \left\{ \vec{D} \;\middle|\; \frac{(R_1 + \cdots + R_i)(\mu + R_{i+1} + \cdots + R_V)}{\mu(\mu - R_1 - \cdots - R_i)} - R_1 D_1 - \cdots - R_i D_i \leq 0 \quad i = 1, \ldots, V \right\} \qquad (B.43)$$

From duality we have:

$$\min_{\vec{D} \in H^{np},\ D_1 \leq \cdots \leq D_V} \sum_{i=1}^{V} C_i(D_i) = \max_{\vec{\lambda} \geq 0} \; q(\vec{\lambda}) \qquad (B.44)$$

where $q(\vec{\lambda})$, the dual functional, is:

$$q(\vec{\lambda}) = \min_{D_1 \leq \cdots \leq D_V} \sum_{i=1}^{V} \Big[C_i(D_i) - (\lambda_i + \cdots + \lambda_V) R_i D_i \Big] + K^{np}(\vec{\lambda}) \qquad (B.45)$$

and where we have defined for convenience:

$$K^{np}(\vec{\lambda}) = \sum_{i=1}^{V} \lambda_i \left[\frac{(R_1 + \cdots + R_i)(\mu + R_{i+1} + \cdots + R_V)}{\mu(\mu - R_1 - \cdots - R_i)} \right] \qquad (B.46)$$

This leads to the following results.

Lemma B.1: The delay assignment $\vec{D}$ achieving the minimum in the right hand side of equation (B.45) is unique and is given by:

$$D_i = \max(A_i, D_{i-1}) \qquad i = 1 \ldots V, \quad D_0 = 0$$

where:

$$A_i = \underset{D}{\operatorname{argmin}} \Big[C_i(D) - (\lambda_i + \cdots + \lambda_V) R_i D + J_i(D) \Big] \quad i = 1, \ldots, V$$

and where $J_V(D) = K^{np}(\vec{\lambda})$, and for $i = 1, \ldots, V-1$;

$$J_i(D) = \begin{cases} C_{i+1}(A_{i+1}) - (\lambda_{i+1} + \cdots + \lambda_V) R_{i+1} A_{i+1} + J_{i+1}(A_{i+1}), & \text{if } D < A_{i+1} \\ C_{i+1}(D) - (\lambda_{i+1} + \cdots + \lambda_V) R_{i+1} D + J_{i+1}(D), & \text{if } D \geq A_{i+1} \end{cases}$$

Moreover the corresponding value of the dual functional is:

$$q(\vec{\lambda}) = C_1(A_1) - (\lambda_1 + \cdots + \lambda_V) R_1 A_1 + J_1(A_1)$$

Theorem B.1: Let $\vec{D}^*$ be a delay assignment satisfying $D_1^* \leq \cdots \leq D_V^*$. $\vec{D}^*$ is the optimal solution to the problem (P_g^{np}) if and only if in the problem (A_g^{np}) defined based on the ordering $D_1 \leq \cdots \leq D_V$;

1) $\vec{\lambda}^* \geq 0$.

2) $\vec{D}^*$ satisfies lemma B.1 when $\vec{\lambda} = \vec{\lambda}^*$.

3) $\vec{D}^* \in H^{np}$.

4) Complementary slackness is satisfied; i.e.:

$$\lambda_i^* \left[\frac{(R_1 + \cdots + R_i)(\mu + R_{i+1} + \cdots + R_V)}{\mu(\mu - R_1 - \cdots - R_i)} - R_1 D_1^* - \cdots - R_i D_i^* \right] = 0 \quad i = 1, \ldots, V$$

5) $A_1 \leq \cdots \leq A_V$.

Corollary B.1: Let $\vec{D}^*$ and $\vec{\lambda}^*$ satisfy theorem B.1. Then for all i, $i = 1, \ldots, V$:

$$\frac{1}{R_i} C_i'(D_i^*) = \sum_{j=i}^{V} \lambda_j^*$$

Alg_P_s^{np}: This algorithm produces the optimal solution to the problem (P_s^{np}).

Let $T \in N$ be a given number.

1) Initially arbitrarily label the v.c.'s $1, \cdots, V$ and set $T = 1$, $\vec{\lambda} = 0$, $\vec{D} = 0$.

2) For $i < T$ let D_i stay constant.

For $i \geq T$ let:

$$D_i = [C_i']^{-1}(R_i\gamma)$$

provided that this expression is well–defined. Otherwise if γ is such that $[C_i']^{-1}(R_i\underline{\gamma}) = 0$ for some $\underline{\gamma} > \gamma$ we set $D_i = 0$ and if $[C_i']^{-1}(R_i\overline{\gamma}) = \infty$ for some $\overline{\gamma} < \gamma$ we set $D_i = \infty$.

3) Find the minimum γ such that the following constraints are satisfied, where the labelling of the v.c.'s $T, \ldots, V$ is defined by the ordering of the delays:

$$\begin{aligned}
R_1 D_1 + \cdots + R_T D_T &\geq \frac{(R_1 + \cdots + R_T)(\mu + R_{T+1} + \cdots + R_V)}{\mu(\mu - R_1 - \cdots - R_T)} \\
\vdots \qquad &\geq \qquad \vdots \\
R_1 D_1 + \cdots + R_{V-1} D_{V-1} &\geq \frac{(R_1 + \cdots + R_{V-1})(\mu + R_V)}{\mu(\mu - R_1 - \cdots - R_{V-1})} \\
R_1 D_1 + \cdots + R_V D_V &\geq \frac{R_1 + \cdots + R_V}{\mu - R_1 - \cdots - R_V}
\end{aligned}$$

4) Let i' be the highest constraint satisfied with equality. Set:

a) $\lambda_{i'} \leftarrow \gamma$

b) $\lambda_{T-1} \leftarrow \lambda_{T-1} - \gamma$

c) $D_j \leftarrow [C_j']^{-1}(R_j\gamma) \quad j = T, \ldots, i'$

d) $T \leftarrow i' + 1$

5) if $T > V$ stop, else go to step 2.

The problem in the user–oriented approach, called (P_u^{np}), is defined as follows:

$$(P_u^{np}) \qquad \min \quad \Im\Big(C_1(D_1), \ldots, C_V(D_V)\Big)$$
$$\vec{D} \text{ realizable using a non–preemptive queuing strategy}$$

where it is now assumed that the cost functions satisfy assumptions (A.3.2).

Similarly as in the case of the problem (P_u) in chapter 3 the optimal solution of (P_u^{np}) can be obtained by solving an hierarchy of nested problems. The first problem minimizes the

worst delay cost, and in general the kth problem minimizes the kth worst delay cost subject to not increasing the delay costs assigned in the previous problems. The first problem, called $(P^{np}_{u,1})$, can be written as:

$$(P^{np}_{u,1}) \qquad \begin{aligned} &\min \quad \gamma_1 \\ &C_j(D_j) \leq \gamma_1 \\ &\vec{D} \text{ realizable using a non-preemptive queuing strategy} \end{aligned}$$

We have the following result:

Lemma B.2: Assume that the cost functions are convex. Then $(\vec{D}^*, \gamma_1^*)$ and $\vec{\psi}^*$ are respectively an optimal solution to $(P^{np}_{u,1})$ and a Lagrange multiplier vector if and only if:

1) $\vec{\psi}^* \geq 0$.

2) $\vec{D}^*$ is realizable using a non-preemptive queuing strategy.

3) $C_i(D_i^*) \leq \gamma_1^*$.

4) $\psi_i^* = 0$ if $C_i(D_i^*) < \max_{1 \leq j \leq V} C_j(D_j^*)$.

5) $\vec{D}^* = \text{argmin}_{\vec{D} \text{ realizable using a non-preemptive queuing strategy}} \sum_{i=1}^{V} \psi_i^* C_i(D_i^*)$.

6) $\sum_{i=1}^{V} \psi_i^* = 1$.

The kth problem in the hierarchy, which is called $(P^{np}_{u,k})$, is defined as:

$$(P^{np}_{u,k}) \qquad \begin{aligned} &\min \quad \gamma_k \\ &C_i(D_i) \leq \gamma_k \quad i \notin I_k \\ &D_i \; i \in I_k \text{ as determined in one of the problems } (P^{np}_{u,j}),\; j = 1, \ldots, k-1 \\ &\vec{D} \text{ realizable using a non-preemptive queuing strategy} \end{aligned}$$

I_k is the set containing the identity of all the v.c.'s whose delay has been assigned in one of the problems $(P^{np}_{u,j})$, $j = 1, \ldots, k-1$. The delays assigned in the kth problem are the delays of the v.c.'s whose cost is equal to the optimal value of the problem and whose delay cannot be reduced without violating feasibility or having to increase the delay of another v.c. whose cost is at least as high as the optimal value of the problem.

We have the following facts:

1) With a trivial modification lemma B.2 is applicable to the problem $(P^{np}_{u,k})$. The modification is that only the v.c's whose delay has not yet been assigned are actively considered. The v.c.'s whose delay has been determined in previous problems only affect the current problem through their impact on the set of delays realizable by non–preemptive queuing strategies.

2) The delays are assigned in increasing order. For example the delays assigned in the k^{th} problem are all larger than the delays assigned in the $k-1^{th}$ problem but are all smaller than the delays assigned in the $k+1^{th}$ problem.

3) The v.c.'s whose delay are assigned in a given problem are of lowest non–preemptive priority compared to the v.c.'s whose delay was assigned in previous problems but have full non–preemptive priority over the v.c.'s whose delay will be assigned in subsequent problems.

These results demonstrate that the strong coupling that exists between the problems (P_s) and (P_u) in the preemptive case is still present in the non–preemptive case.

The algorithm that solves the problem (P^{np}_u) is obtained from Alg_P_u by accounting for the fact that the feasible delay set is slightly different. This algorithm, called alg_P^{np}_u, is given below.

Alg_P^{np}_u: This algorithm produces the optimal solution to the problem (P^{np}_u).

Let $T \in N$ be a given number.

1) Initially arbitrarily label the v.c.'s $1,\ldots,V$ and set $T = 1$, $\gamma = 0$, $\vec{D} = 0$.

2) For $i < T$ let D_i stay constant.

   For $i \geq T$ let:

$$D_i = C_i^{-1}(\gamma)$$

   provided that this expression is well–defined. Otherwise set $D_i = 0$.

3) Find the minimum γ such that the following constraints are satisfied, where the labelling of the v.c.'s $T,\ldots,V$ is defined by the ordering of the delays.

$$
\begin{aligned}
R_1 D_1 + \cdots + R_T D_T &\geq \frac{(R_1 + \cdots + R_T)(\mu + R_{T+1} + \cdots + R_V)}{\mu(\mu - R_1 - \cdots - R_T)} \\
\vdots \qquad &\geq \qquad \vdots \\
R_1 D_1 + \cdots + R_{V-1} D_{V-1} &\geq \frac{(R_1 + \cdots + R_{V-1})(\mu + R_V)}{\mu(\mu - R_1 - \cdots - R_{V-1})} \\
R_1 D_1 + \cdots + R_V D_V &\geq \frac{R_1 + \cdots + R_V}{\mu - R_1 - \cdots - R_V}
\end{aligned}
$$

4) Let i' be the highest constraint satisfied with equality. Set:

   a) $D_j = C_j^{-1}(\gamma), \quad j = T, \ldots, i'$

   b) $T = i' + 1$

5) if $T > V$ stop, else go to step 2.

Finally from the similarity between Alg_P_s^{np} and Alg_P_u^{np} it is clear that theorem 3.4 can be extended to the non-preemptive case. Namely we have:

Theorem B.2: Assume that the functions $C_i(\cdot), i = 1, \ldots, V$, satisfy assumptions (A.3.1) and that the functions $C_i'(\cdot), i = 1, \ldots, V$, satisfy assumptions (A.3.2). Then $\bar{D}^*$ is the optimal solution of the problem (P_s^{np}) with cost functions $C_i(\cdot)$ $i = 1, \ldots, V$, if and only if it is the optimal solution of the problem (P_u^{np}) with cost function $\frac{1}{R_i} C_i'(\cdot)$ $i = 1, \ldots, V$.

Appendix C: Complement to Chapter 4.

C.1 Proof of theorem 4.1.

1) Forward implication.

We use a contradiction argument. Let i be one index for which $\vec{x}_i^*$ is not optimal. Accordingly, there exists $\vec{\hat{x}}_i \in X_i$ satisfying:

$$f(\vec{x}_1^*, \ldots, \vec{x}_{i-1}^*, \vec{\hat{x}}_i, \vec{x}_{i+1}^*, \ldots, \vec{x}_n^*) < f(\vec{x}_1^*, \ldots, \vec{x}_n^*) \qquad (C.1)$$

from which it immediately follows that $(\vec{x}_1^*, \ldots, \vec{x}_n^*)$ is not optimal, providing the required contradiction.

2) Reverse implication.

We again use a contradiction argument. Assume $\vec{x}^* = (\vec{x}_1^*, \ldots, \vec{x}_n^*)$ is not optimal. Accordingly, there exists $\vec{\hat{x}} = (\vec{\hat{x}}_1, \ldots, \vec{\hat{x}}_n)$ satisfying:

$$f(\vec{x}^*) > f(\vec{\hat{x}}) \qquad (C.2)$$

In view of the convexity of $f(\cdot)$ this means that:

$$\nabla f(\vec{x}^*)(\vec{\hat{x}} - \vec{x}^*) < 0 \qquad (C.3)$$

From this equation we must have for at least one i:

$$\nabla f_i(\vec{x}^*)(\vec{\hat{x}}_i - \vec{x}_i^*) < 0 \qquad (C.4)$$

where $\nabla f_i(\cdot)$ stands for the subgradient of $f(\cdot)$ containing only the coordinates in $\vec{x}_i$.

It follows immediately from equation (C.4) that $\vec{x}_i^*$ is not optimal for its subproblem, which is a contradiction.

Q.E.D.

C.2 Proof of lemma 4.1.

Let $\vec{w}^l$ be any ordering on link l valid for the assignment $\vec{D}^l$. By assumption we have $w_i^l < w_j^l$. Also, there exists k, $w_i^l \leq w_k^l < w_j^l$, such that the feasibility constraint w_k^l is satisfied with equality and such that the feasibility constraints $w_{k+1}^l, \ldots, w_{j-1}^l$ are all satisfied with strict inequality. From this, it is easy to see that initially:

$$D_i^l \leq \frac{1}{R_k}\Big[B^l(w_k^l, \vec{w}^l, \vec{R}) - B^l(w_k^l - 1, \vec{w}^l, \vec{R})\Big] \tag{C.5}$$

$$D_j^l \geq \frac{1}{\sum_{p \mid w_k^l < w_p^l \leq w_j^l} R_p}\Big[B^l(w_j^l, \vec{w}^l, \vec{R}) - B^l(w_k^l, \vec{w}^l, \vec{R})\Big] \tag{C.6}$$

where equality is achieved in the first equation if $k = i$ and if v.c. i is of lowest priority as compared to the v.c.'s in position $1, \ldots, w_i^l - 1$, and where equality is achieved in the second equation if $D_j^l = D_p^l$ for all p, $w_k^l < w_p^l \leq w_j^l$, and if v.c. j has full priority over the v.c.'s in position $w_j^l + 1, \ldots, V^l$.

It follows immediately from equations (C.5) and (C.6) that:

$$D_j^l - D_i^l \geq \frac{\mu^l \sum_{p \mid w_k^l \leq w_p^l \leq w_j^l} R_p}{\left(\mu^l - \sum_{p \mid w_p^l \leq w_k^l} R_p\right)\left(\mu^l - \sum_{p \mid w_p^l < w_k^l} R_p\right)\left(\mu^l - \sum_{p \mid w_p^l \leq w_j^l} R_p\right)} \tag{C.7}$$

so that, using assumption (A.4.1.1), we obtain:

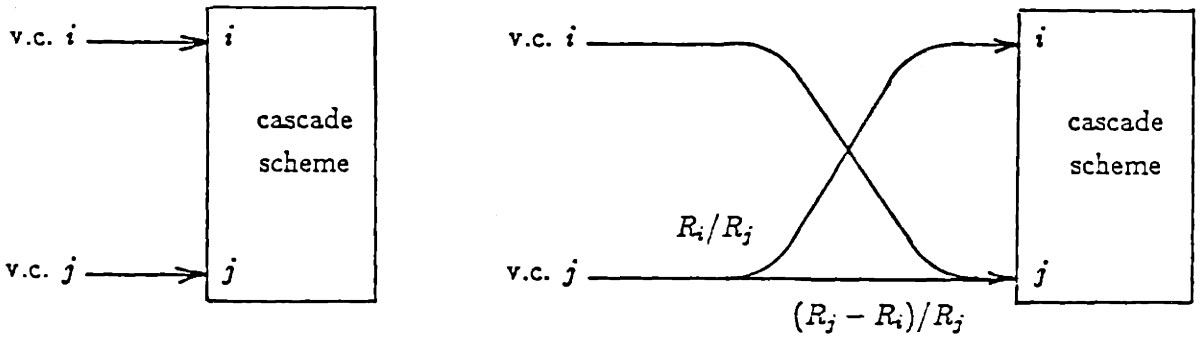
$$D_j^l - D_i^l \geq \frac{\mu^l K_1}{(K_2)^3} \tag{C.8}$$

Now, assume that $R_j \geq R_i$ (this assumption is not restrictive, as we will soon see) and consider a cascade scheme in which all the packets of v.c. i are routed to the entry port of v.c. j, and in which a proportion $\frac{R_i}{R_j}$ of the packets of v.c. j are routed to the input port of v.c. i. This situation is depicted in Figure C.1.

It is easy to see that the rates at the input ports of v.c.'s i and j are still respectively R_i and R_j, which implies that the delays of the v.c.'s $k \neq i$, j are identical in the two systems. Denoting by $\bar{D}_i^l$ and $\bar{D}_j^l$ the new delays of v.c. i and j, it is easy to see that:

$$\bar{D}_i^l = D_j^l \tag{C.9}$$

$$\bar{D}_j^l = \frac{R_i}{R_j} D_i^l + \Big(\frac{R_j - R_i}{R_j}\Big) D_j^l \tag{C.10}$$



a) Initially b) Modified

Figure C.1

From equations (C.8), (C.9) and (C.10), it follows that:

$$\begin{aligned} \bar{D}_i^l - D_i^l &\geq \frac{\mu^l K_1}{(K_2)^3} \\ \bar{D}_j^l - D_j^l &\leq \frac{R_i \mu^l K_1}{R_j (K_2)^3} \end{aligned} \tag{C.11}$$

Using the convexity of the set of strictly feasible delays, this implies that for all Δ satisfying:

$$\Delta \in \left[0, \frac{R_i \mu^l (K_1)^2}{R_j (K_2)^3}\right] \tag{C.12}$$

the assignment:

$$\begin{aligned} \bar{D}_i &= D_i + \frac{\Delta}{R_i} \\ \bar{D}_j &= D_j - \frac{\Delta}{R_j} \end{aligned} \tag{C.13}$$

is strictly feasible. As $R_j \leq \mu^l$ for all $j \in \mathcal{V}^l$, it immediately follows that the proposition holds for $K_4 = \left(K_1/K_2\right)^3$.

Using a completely analogous argument, it is easy to see that the preceeding result also holds when it is assumed that $R_i \geq R_j$.

Q.E.D.

C.3 Proof of Theorem 4.3.

The overall idea of the proof is to modify slightly the cost functions so as to make the assignment $\vec{\hat{D}}$ optimal.

Assume that the v.c.'s are labelled in order of decreasing marginal delay cost per unit rate. That is the v.c. labelled 1 is such that $\frac{1}{R_1}C_1'(\hat{D}_1)$ is not less than $\frac{1}{R_i}C_i'(\hat{D}_i)$ for any other v.c. i, the v.c. labelled 2 is such that $\frac{1}{R_2}C_2'(\hat{D}_2)$ not less than $\frac{1}{R_i}C_i'(\hat{D}_i)$ for any other v.c. $i \neq 1$, etc.

Consider the set of v.c.'s $F_1 = \{1, \ldots, q_1\}$ defined as follows. V.c. 1 is always in F_1. In general, v.c. i is in F_1 if there exists a v.c. $\tilde{i} \in F_1$, $\tilde{i} \neq i$ such that:

$$\frac{1}{R_i}C_i'(\hat{D}_i) \geq \frac{1}{R_{\tilde{i}}}C_{\tilde{i}}'(\hat{D}_{\tilde{i}}) - \gamma \tag{C.14}$$

It is easy to see that if v.c.'s i and k are in F_1, then all the v.c.'s j, $i \leq j \leq k$, are in F_1. By construction if $i \in F_1$ and $\tilde{i} \notin F_1$, then:

$$\frac{1}{R_i}C_i'(\hat{D}_i) - \frac{1}{R_{\tilde{i}}}C_{\tilde{i}}'(\hat{D}_{\tilde{i}}) > \gamma \tag{C.15}$$

This means from condition (3) of the theorem that all the v.c.'s in F_1 have full priority over the v.c.'s not in F_1. For $i \in F_1$ define a new cost function $\tilde{C}_i(\cdot)$ as follows:

$$\tilde{C}_i(D_i) = C_i(D_i) + \varepsilon_i D_i \tag{C.16}$$

where:

$$\varepsilon_i = R_i\left[\frac{1}{R_1}C_1'(\hat{D}_1) - \frac{1}{R_i}C_i'(\hat{D}_i)\right] \tag{C.17}$$

Note that by construction $\varepsilon_i \geq 0$ so that the cost function defined in equation (C.16) satisfies assumptions (A.3.1). Also:

$$\begin{aligned}\varepsilon_i =& R_i \sum_{k=1}^{i-1}\left[\frac{1}{R_k}C_k'(\hat{D}_k) - \frac{1}{R_{k+1}}C_{k+1}'(\hat{D}_{k+1})\right] \\ \leq& R_i(i-1)\gamma \\ \leq& (\max_l \mu^l)V\gamma \end{aligned} \tag{C.18}$$

Now define $F_2 = \{q_1+1, \ldots, q_2\}$ in a completely analogous manner as F_1. That is v.c. q_1+1 is always in F_2. V.c. i is in F_2 if for some other v.c. $\tilde{i} \in F_2$:

$$\frac{1}{R_i} C_i'(\hat{D}_i) \geq \frac{1}{R_{\tilde{i}}} C_{\tilde{i}}'(\hat{D}_{\tilde{i}}) - \gamma \qquad (C.19)$$

For $i \in F_2$ define the cost functions:

$$\tilde{C}_i(D_i) = C_i(D_i) + \epsilon_i D_i \qquad (C.20)$$

where:

$$\epsilon_i = R_i \left[\frac{1}{R_{q_1+1}} C_{q_1+1}'(\hat{D}_{q_1+1}) - \frac{1}{R_i} C_i'(\hat{D}_i) \right] \qquad (C.21)$$

Similarly as before we find for all $i \in F_2$:

$$0 \leq \epsilon_i \leq (\max_l \mu^l) V \gamma \qquad (C.22)$$

and that the v.c.'s in F_2 have full priority over the v.c.'s not in F_1 or F_2.

Repeating this procedure, a sequence of sets $F_1, \ldots, F_m$ will be constructed. The v.c.'s in the k^{th} set $F_k = \{q_{k-1}+1, \ldots, q_k\}$ have full priority over the v.c.'s in the sets $F_{k+1}, \ldots F_m$ and are of lowest priority as compared to the v.c.'s in the sets $F_1, \ldots F_{k-1}$. Each v.c. $i \in F_k$ has a new cost function $\tilde{C}_i(\cdot)$ defined in a completely analogous manner as in equations (C.20) and (C.21). Accordingly, it follows that equation (C.22) holds for all i.

By construction we have for all F_k, $k = 1, \ldots, m$;

$$\frac{1}{R_i} \tilde{C}_i'(\hat{D}_i) = \frac{1}{R_{q_{k-1}+1}} \tilde{C}_{q_{k-1}+1}'(\hat{D}_{q_{k-1}+1}) \quad \text{for all } i \in F_k \qquad (C.23)$$

It is easy to see in view of equation (C.23), the strict feasibility of $\vec{\bar{D}}$, and the construction of the F_k, $k = 1, \ldots, m$, that $\vec{\hat{D}}$ is an optimal solution of the (NP_s) problem with cost functions $\tilde{C}_i(\cdot)$, $i = 1, \ldots, V$. Thus:

$$\begin{aligned} \sum_{i=1}^{V} C_i(\hat{D}_i) + \epsilon_i \hat{D}_i &= \min_{\vec{D} \text{ feasible}} \sum_{i=1}^{V} C_i(D_i) + \epsilon_i D_i \\ &\leq \sum_{i=1}^{V} C_i(D_i^*) + \epsilon_i D_i^* \end{aligned} \qquad (C.24)$$

where $\vec{D}^*$ is any optimal solution to the initial (NP_s) problem. It follows from this equation and equation (C.22) that:

$$S(\vec{\hat{D}}) - S^* \leq \sum_{i=1}^{V} \varepsilon_i (D_i^* - \hat{D}_i)$$
$$\leq V^2 (\max_l \mu^l)(\max_i |D_i^* - \hat{D}_i|)\gamma \qquad (C.25)$$

From the condition (2) of the theorem, we know that $\vec{\hat{D}}$ and $\vec{D}^*$ lie in a bounded set. Let D be an upper bound for this set. Then, it follows from the preceeding equation that:

$$S(\vec{\hat{D}}) - S^* \leq 2V^2 (\max_l \mu^l) D\gamma \qquad (C.26)$$

Thus, setting $K_6 = 2V^2(\max_l \mu^l)D$ proves the theorem.

Q.E.D.

C.4 Generalization of Alg_NP_s_a in the case where $\gamma_d = 0$ is allowed.

In this section we show that the convergence properties of Alg_NP_s_a established in section 4.3.2 are maintained when $\gamma_d = 0$. The only things that need to be modified to account for the case $\gamma_d = 0$ are the proof of theorem 4.2 and the discussion of pp. 98–99. on the convergence of the version of Alg_NP_s_a presented in section 4.3.3. It is not difficult to see that if theorem 4.2 can be extended to the case $\gamma_d = 0$, it will also be possible to modify the discussion of pp. 98–99 so as to hold when $\gamma_d = 0$. For this reason we concentrate in this section on a new proof for theorem 4.2. We leave the generalization of the discussion of pp. 98–99 to the reader.

In the proof of theorem 4.2 all the arguments except the argument used to show the convergence of the delays hold when $\gamma_d = 0$. Thus, to generalize theorem 4.2, we only need a new argument for showing the convergence of the delays which holds when $\gamma_d = 0$. The aim of this section in to present such an argument. The basic idea consists of showing first the convergence of the delay of one v.c. Then, using this result, it will be possible to show the convergence of the delay of a second v.c. Generalizing the procedure, we will be able to show that all the delays converge.

We now prove that all the D_i^l, $i = 1, \ldots, V$, $l \in \mathcal{L}_i$, converge. Apart from the possibility that $\gamma_d = 0$ we make the same assumptions as in theorem 4.2. Also, as in theorem 4.2, we prove the result for the version of Alg_NP_s_a defined in section 4.3.1.

Define:

$$B_1(k) = \max_j \frac{1}{R_j} C_j'(D_j(k)) \qquad (C.27)$$

It is easy to see that $B_1(k)$ is a non–increasing function of k.

Let i be a v.c. such that at time $\underline{k} \geq 0$:

$$\frac{1}{R_i} C_i'(D_i(\underline{k})) = B_1(\underline{k}) \qquad (C.28)$$

Assume that there exists a finite $\bar{k} > \underline{k}$ such that:

$$D_i(\bar{k}) \leq D_i(\underline{k}) - \frac{\varepsilon_d}{L} \qquad (C.29)$$

Assume also that $\bar{k}$ is the smallest time after $\underline{k}$ at which equation (C.29) is satisfied.

We show that when $\bar{k}$ exists then:

a) All the updates involving v.c. i in the interval $[\underline{k}, \bar{k}]$ result in D_i being reduced.

b) For all $k \geq \bar{k}$, $D_i(k) \leq D_i(\underline{k}) - \varepsilon_d/L$.

Proposition (a) holds by definition at times $\underline{k}$ and $\bar{k}$. To see that it holds for all $k \in]\underline{k}, \bar{k}[$, assume by contradiction that it is false. Accordingly, there must be an update in the interval $]\underline{k}, \bar{k}[$ resulting in and increase of D_i. Assume w.l.o.g. that the update occurs with v.c. j on link l at time k. Since D_i increases, we must have:

$$\frac{1}{R_j} C_j'\left(d_j^{[l]}(k^-) - \varepsilon_d - \frac{\bar{\Delta}_n}{R_j}\right) - \frac{1}{R_i} C_i'\left(d_i^{[l]}(k^-) + \varepsilon_d + \frac{\bar{\Delta}_n}{R_i}\right) > \gamma_d \qquad (C.30)$$

Using equation (4.7) and the facts that $\gamma_d \geq 0$ and that the functions $C_i'(\cdot)$ and $C_j'(\cdot)$ are non–decreasing, this implies:

$$\frac{1}{R_j} C_j'\left(D_j(k^-)\right) > \frac{1}{R_i} C_i'\left(D_i(k^-) + \frac{\varepsilon_d}{L}\right) \qquad (C.31)$$

Since $k^- \in]\underline{k}, \bar{k}[$, we have $D_i(\underline{k}) < D_i(k^-) + \varepsilon_d/L$, so that we get from equation (C.31):

$$\begin{aligned} \frac{1}{R_j} C_j'\left(D_j(k^-)\right) &> \frac{1}{R_i} C_i'\left(D_i(\underline{k})\right) \\ &> B_1(\underline{k}) \end{aligned} \qquad (C.32)$$

This implies that $B_1(k^-) > B_1(\underline{k})$, contradicting the fact that $B_1(\cdot)$ is non–increasing. Thus, proposition (a) is proven.

Proposition (b) holds by definition at time $\bar{k}$. We show that it also holds for $k > \bar{k}$ by contradiction. Indeed, suppose that the proposition is false. Then, there must occur an update after time $\bar{k}$, say at time k, which causes the delay of v.c. i to satisfy:

$$D_i(k) > D_i(\underline{k}) - \frac{\varepsilon_d}{L} \qquad (C.33)$$

Assume w.l.o.g. that the update is made with v.c. j on link l.

It is clear that the update causes D_i^l to increase. Accordingly equation (C.30) holds. Also, from the update rule (4.9), we know that:

$$\begin{aligned} D_i(k) - D_i(k^-) &\leq \frac{\bar{\Delta}_n}{R_i} \\ D_j(k^-) - D_j(k) &\leq \frac{\bar{\Delta}_n}{R_j} \end{aligned} \qquad (C.34)$$

Starting with equation (C.30) and using successively equations (4.7), (C.34) and (C.33), we get:

$$\begin{aligned}\frac{1}{R_j}C_j'(D_j(k)) &> \frac{1}{R_i}C_i'(D_i(\underline{k})) \\ &> B_1(\underline{k}) \end{aligned} \qquad (C.35)$$

which again contradicts the fact that $B_1(\cdot)$ is non–increasing. Thus proposition (b) also holds.

It follows from propositions (a) and (b) that the link delays of at least one v.c. must converge. Indeed, suppose v.c. i satisfies equation (C.28). If there is no $\bar{k}$ such that $D_i(\bar{k}) \leq D_i(\underline{k}) - \varepsilon_d/L$, proposition (a) guarantees that all the updates involving v.c i after time $\underline{k}$ reduce D_i. As the total decrease of each D_i^l is bounded below by ε_d/L (otherwise $\bar{k}$ would exist), this implies that all the D_i^l, $l \in \mathcal{L}_i$, converge.

On the other hand if there exists a finite $\bar{k}$, then proposition (b) guarantees that for all $k \geq \bar{k}$:

$$D_i(k) \leq D_i(\underline{k}) - \varepsilon_d/L \qquad (C.36)$$

Now we can repeat this argument for $k \geq \bar{k}$ and, in fact, we can continue as long as finite $\bar{k}$ can be found. If there is always a finite time $\bar{k}$ such that equation (C.36) holds for $k \geq \bar{k}$, then, as the number of v.c.'s is finite, there must be at least one v.c. whose delay converges to $-\infty$. Clearly this is impossible as we know that strict feasibility is always maintained. Thus, we must eventually be unable to find a finite $\bar{k}$. At this time, the argument of the preceding paragraph will apply, which guarantees that there is at least one v.c., say i, whose D_i^l, $l \in \mathcal{L}_i$, converge.

Let i be the v.c. whose link delays converge, and let k_0 be a time such that for all $l \in \mathcal{L}_i$ and $k \geq k_0$:

$$\left|D_i^l(k) - D_i^l(k_0)\right| \leq \frac{\varepsilon_d K_1}{L^2 R_i} \qquad (C.37)$$

Assume also that k_0 is such that all the updates involving v.c. i after k_0 result in a decrease of D_i. It is easy to see from the preceeding discussion that a k_0 satisfying these conditions can always be found.

Define:

$$B_2(k) = \max_{j \neq i} \frac{1}{R_j} C_j'\big(D_j(k)\big) \tag{C.38}$$

We now show that for any $k_1 \geq k_0$ and $j \neq i$:

$$B_2(k_1) \geq \frac{1}{R_j} C_j'\Big[D_j(k_2) + \frac{R_i}{R_j}\big(D_i(k_2) - D_i(k_1)\big)\Big] \quad \text{for all } k_2 \geq k_1 \tag{C.39}$$

We use induction on the sequence of updates occuring after time k_1. Let τ be the time of the first update occuring after time k_1. Then by definition equation (C.39) holds for all $k_2 \in [k_1, \tau[$. Now we show that it also holds for $k_2 = \tau$; i.e., that the equation is maintained after the first update. We distinguish two cases depending if v.c. i is involved in the update or not.

First assume that v.c. i is involved in the update. More specificaly assume w.l.o.g. that the update is between v.c.'s i and j and occurs on link l. Because strict feasibility is maintained, we have:

$$R_i\big(D_i^l(\tau) - D_i^l(\tau^-)\big) = -R_j\big(D_j^l(\tau) - D_j^l(\tau^-)\big) \tag{C.40}$$

Using this equation it is easy to see that:

$$D_j(\tau) + \frac{R_i}{R_j}\big(D_i(\tau) - D_i(k_1)\big) = D_j(\tau^-) + \frac{R_i}{R_j}\big(D_i(\tau^-) - D_i(k_1)\big) \tag{C.41}$$

which implies that equation (C.39) is still satisfied at time τ.

Now assume that the update occuring at time τ does not involve v.c. i. Assume w.l.o.g. that the v.c.'s involved are $j \neq i$ and $p \neq i$, that the update occurs on link l, and that D_j^l decreases. Since D_j^l decreases equation (C.39) stays obviously satisfied for j. Concerning p it is known that:

$$\frac{1}{R_j} C_j'\big(d_j^{[l]}(\tau^-) - \varepsilon_d - \frac{\bar{\Delta}_n}{R_j}\big) - \frac{1}{R_p} C_p'\big(d_p^{[l]}(\tau^-) + \varepsilon_d + \frac{\bar{\Delta}_n}{R_p}\big) > \gamma_d \tag{C.42}$$

Also, we have from equation (C.37):

$$\frac{R_i}{R_p}\big|D_i(\tau) - D_i(k_1)\big| \leq \frac{\varepsilon_d}{L} \tag{C.43}$$

Using this equation we get:

$$D_p(\tau) + \frac{R_i}{R_p}\left(D_i(\tau) - D_i(k_1)\right) \leq d_p^{[l]}(\tau^-) + \varepsilon_d + \frac{\bar{\Delta}_n}{R_p} \qquad (C.44)$$

Similarly it is easy to see that:

$$d_j^{[l]}(\tau^-) - \varepsilon_d - \frac{\bar{\Delta}_n}{R_j} \leq D_j(\tau^-) + \frac{R_i}{R_j}\left(D_i(\tau^-) - D_i(k_1)\right) \qquad (C.45)$$

Starting with equation (C.42) and using equations (C.44) and (C.45), we get:

$$\begin{aligned}\frac{1}{R_p}C_p'\Big[D_p(\tau) + \frac{R_i}{R_p}\left(D_i(\tau) - D_i(k_1)\right)\Big] &\leq \frac{1}{R_j}C_j'\Big[D_j(\tau^-) + \frac{R_i}{R_j}\left(D_i(\tau^-) - D_i(k_1)\right)\Big] \\ &\leq B_2(k_1) \end{aligned} \qquad (C.46)$$

where the last inequality follows from the fact that equation (C.39) holds at time τ^-. The last equation shows that p satisfies equation (C.39) at time τ. Thus, in all cases, equation (C.39) is maintained after the first update. This argument can straightforwardly be generalized into an induction argument. We leave this task to the reader.

Let j be a v.c. such that at time $\underline{k} \geq k_0$ we have:

$$\frac{1}{R_j}C_j'\left(D_j(\underline{k})\right) = B_2(\underline{k}) \qquad (C.47)$$

Assume that there exists a finite time $\bar{k} > \underline{k}$ such that:

$$D_j(\bar{k}) \leq D_j(\underline{k}) - \frac{\varepsilon_d}{L} \qquad (C.48)$$

Assume also that $\bar{k}$ is the smallest time after $\underline{k}$ at which the preceeding equation holds.

We show that if $\bar{k}$ exists, then:

c) In the interval $[\underline{k}, \bar{k}]$ any update between v.c. j and a v.c. $p \neq i$ results in D_j being reduced.

d) For all $k \geq \bar{k}$, $D_j(k) \leq D_j(\underline{k}) - \varepsilon_d/L$.

Note that there is an obvious similarity between these propositions and the propositions (a) and (b) discussed above.

Proposition (c) holds by definition at time $k = \underline{k}$ and $\bar{k}$. To see that it holds for all times in the interval $]\underline{k}, \bar{k}[$ assume by contradiction that this is not the case. Accordingly,

assume w.l.o.g. that an update between v.c.'s j and $p \neq i$ occurs on link l at time $k \in]\underline{k}, \bar{k}[$, and that the update causes D_j^l to increase. If this is the case we must have:

$$\frac{1}{R_p}C_p'\left(d_p^{[l]}(k^-) - \varepsilon_d - \frac{\bar{\Delta}_n}{R_p}\right) - \frac{1}{R_j}C_j'\left(d_j^{[l]}(k^-) + \varepsilon_d + \frac{\bar{\Delta}_n}{R_j}\right) > \gamma_d \qquad (C.49)$$

From this equation, we easily obtain:

$$\frac{1}{R_p}C_p'\left(D_p(k^-) - \frac{\varepsilon_d}{L} - \frac{\bar{\Delta}_n}{R_p}\right) > \frac{1}{R_j}C_j'\left(D_j(k^-) + \frac{\varepsilon_d}{L} + \frac{\bar{\Delta}_n}{R_j}\right) \qquad (C.50)$$

Also, since $\underline{k} \geq k_0$, we have using equation (C.37):

$$D_p(k^-) - \frac{\varepsilon_d}{L} - \frac{\bar{\Delta}_n}{R_p} \leq D_p(k^-) + \frac{R_i}{R_p}\left(D_i(k) - D_i(\underline{k})\right) \qquad (C.51)$$

and, since $k^- \in]\underline{k}, \bar{k}[$;

$$D_j(k^-) > D_j(\underline{k}) - \frac{\varepsilon_d}{L} \qquad (C.52)$$

Starting with equation (C.50) and using equations (C.51), (C.52) and (C.47), we get:

$$\frac{1}{R_p}C_p'\left[D_p(k^-) + \frac{R_i}{R_p}\left(D_i(k^-) - D_i(\underline{k})\right)\right] > B_2(\underline{k}) \qquad (C.53)$$

But this contradicts equation (C.39). Thus proposition (c) is proven.

Proposition (d) holds by definition at time $k = \bar{k}$. To show that the proposition holds for all $k > \bar{k}$ we assume by contradiction that the proposition is false. Accordingly, assume w.l.o.g. that an update occurs at time $k > \bar{k}$ between v.c.'s j and $p \neq i$ on link l and that, as a result, we get:

$$D_j(k) > D_j(\underline{k}) - \frac{\varepsilon_d}{L} \qquad (C.54)$$

For the same reason as before equations (C.50) and (C.51) still hold. Accordingly, starting with equation (C.50) and using equations (C.51) and (C.54), we get:

$$\begin{aligned}\frac{1}{R_p}C_p'\left[D_p(k^-) + \frac{R_i}{R_p}\left(D_i(k^-) - D_i(\underline{k})\right)\right] &> \frac{1}{R_j}C_j'\left[D_j(k) + \frac{\varepsilon_d}{L}\right] \\ &> \frac{1}{R_j}C_j'\left(D_j(\underline{k})\right) \\ &> B_2(\underline{k}) \qquad (C.55)\end{aligned}$$

contradicting equation (C.39). Thus proposition (d) is also proven.

Now, similarly as in the case of v.c. i, we can show using propositions (c) and (d) that there exists a v.c. $j \neq i$ such that all the D_j^l, $l \in \mathcal{L}_j$, converge. Indeed, suppose that equation (C.47) holds. If there is no finite $\bar{k}$ such that equation (C.48) is satisfied, then proposition (c) insures that, apart from the updates with v.c. i, all the updates involving v.c. j after time $\underline{k}$ result in D_j being reduced. However, as the D_i^l, $l \in \mathcal{L}_i$ are non–increasing and converge, we can make the overall impact of the updates with i as small as we wish. Thus, in this case, all the D_j^l, $l \in \mathcal{L}_j$, converge.

If there is a finite $\bar{k}$ such that equation (C.48) holds for $k \geq \bar{k}$ we can use the exact same argument as in the case of v.c. i to show that the argument of the preceeding paragraph must eventually apply.

We have now shown that there exists two distinct v.c.'s i and j whose delays D_i^l, $l \in \mathcal{L}_i$, and D_j^l, $l \in \mathcal{L}_j$, converge. Although the argument used for showing the convergence of the D_i^l, $l \in \mathcal{L}_i$, is not general, the argument used for j is general, and can straighforwardly be generalized in an induction argument. This is left to the reader.

C.5 Proof of theorem 4.4.

Our proof is very similar to a proof first proposed by Gallager [Gal77]. The overall plan of the proof is the following. By upper bounding the magnitude of the first and second derivatives of the objective function in the direction in which the update is made, we first show that each iteration of Alg_NP_s_e can only reduce the value of the objective function. Following this, we show that whenever the assignment at the beginning of an iteration is not optimal, the iteration results in a strictly better assignment. Next we show that whenever an assignment, say $\vec{D}$, is sufficiently close to a given non–optimal assignment, the assignment produced by Alg_NP_s_e from $\vec{D}$ is strictly better than the given non–optimal assignment. This result is essentially a continuity condition guaranteeing that any converging sequence of delays converges toward an optimal assignment. Combining these results, we then prove theorem 4.4.

We establish the results mentioned above through a sequence of several lemmas. Basically each lemma proves one result. First we introduce some notation and state some trivial facts.

Let H^* be the set of optimal solutions to (NP_s). Also, let $A^p(\vec{D})$ denote the assignment produced by by executing p iterations of Alg_NP_s_e, starting with $\vec{D}$. For convenience we also sometimes denote the assignment $A(\vec{D})$ by $\vec{\tilde{D}}$.

Define:

$$H = \{\vec{D} \mid \vec{D} \geq 0,\ S(\vec{D}) \leq S(\vec{D}(0))\} \tag{C.56}$$

In view of assumption (A.4.1) and of the form of the cost functions, it is clear that H is compact. It follows from this fact that there must exist strictly positive constants K_7, K_8 and K_9 depending only on $\vec{D}(0)$ such that for all $\vec{D} \in H$:

$$D_i \leq K_7 \tag{C.57}$$

$$C_i'(D_i) \leq K_8 \tag{C.58}$$

$$C_i''(D_i) \leq K_9 \tag{C.59}$$

Define:

$$\bar{H} = \{\vec{D} \mid \vec{D} \geq 0 \text{ and for some } \vec{\bar{D}} \in H,\ ||\vec{\bar{D}} - \vec{D}|| \leq K_{10}\} \tag{C.60}$$

where $K_{10} > 0$ is some constant. Clearly, $\bar{H}$ is also compact. Accordingly, it follows that there must exist a constant $K_{11} > 0$ such that for all $\vec{D} \in \bar{H}$:

$$C_i''(D_i) \leq K_{11} \tag{C.61}$$

Let $\vec{D}$ be an arbitrary strictly feasible assignment in H. We define U as the set of updates resulting from one iteration of Alg_NP_s_e starting from $\vec{D}$. Each element of U is a 5–tuple $(u, l_u, \tau_u, i_u, j_u)$ which defines one particular update. u is a label identifying the update. l_u is the link on which the update occurs. τ_u, i_u and j_u are respectively the "τ_{ij}^l", "i" and "j" of step 2) of Alg_NP_s_e for the update.

For a given assignment $\vec{D} \in H$, define:

$$s(\lambda) = S\big(\vec{D} + \lambda(A(\vec{D}) - \vec{D})\big) \tag{C.62}$$

Using a Taylor series expansion we easily obtain:

$$S\big(A(\vec{D})\big) - S(\vec{D}) = \left.\frac{ds(\lambda)}{d\lambda}\right|_{\lambda=0} + \frac{1}{2}\left.\frac{d^2 s(\lambda)}{d\lambda^2}\right|_{\lambda=\lambda^*} \tag{C.63}$$

where λ^* is some number in the interval $[0, 1]$.

We use equation (C.63) to show that Alg_NP_s_e cannot increase the objective function. The idea is to show that by choosing $\bar{\tau}$ small enough we can ensure that the first term in the right hand side is sufficiently negative to completely dominate the second term. The next two lemmas respectively provide a suitable bound on each of these terms. Then, in the third lemma, we combine these results to show that ALg_NP_s_e cannot increase the objective function.

Lemma C.1:

$$\left.\frac{ds(\lambda)}{d\lambda}\right|_{\lambda=0} = -\sum_{u \in U} \tau_u \left(\Delta C_{i_u j_u}\right)^2$$

Proof: Direct differentiation of $s(\lambda)$ gives:

$$\begin{aligned}\left.\frac{ds(\lambda)}{d\lambda}\right|_{\lambda=0} &= \sum_{i=1}^{V} C_i'(D_i)(\hat{D}_i - D_i) \\ &= \sum_{i=1}^{V} C_i'(D_i)\left[-\sum_{u \in U | i = i_u} \frac{\tau_u}{R_i} \Delta C_{i j_u} + \sum_{u \in U | i = j_u} \frac{\tau_u}{R_i} \Delta C_{i_u i}\right]\end{aligned} \tag{C.64}$$

Collecting, for each update, the two terms common to the update, we get:

$$\left.\frac{ds(\lambda)}{d\lambda}\right|_{\lambda=0} = \sum_{u\in U}\left[-C'_{i_u}(D_{i_u})\frac{\tau_u}{R_{i_u}}\Delta C_{i_u j_u} + C'_{j_u}(D_{j_u})\frac{\tau_u}{R_{j_u}}\Delta C_{i_u j_u}\right]$$
$$= -\sum_{u\in U}\tau_u(\Delta C_{i_u j_u})^2 \qquad (C.65)$$

Q.E.D.

Lemma C.2: Assume that:

$$\bar{\tau} \leq \frac{K_1 K_{10}}{VLK_8}$$

Then for all $\lambda \in [0,1]$;

$$\left|\frac{d^2 s(\lambda)}{d\lambda^2}\right| \leq \frac{VLK_{11}}{(K_1)^2}\sum_{u\in U}(\tau_u \Delta C_{i_u j_u})^2$$

Proof: Direct differentiation of $s(\lambda)$ gives:

$$\frac{d^2 s(\lambda)}{d\lambda^2} = \sum_{i=1}^{V} C_i''\big(D_i + \lambda(\hat{D}_i - D_i)\big)(\hat{D}_i - D_i)^2 \qquad (C.66)$$

It follows from the construction of $\bar{H}$ and from the condition imposed on $\bar{\tau}$ in the lemma that the assignments $\lambda\vec{D} + (1-\lambda)\vec{\tilde{D}}$, $\lambda \in [0,1]$, are in $\bar{H}$. This implies using equation (C.61) that for all $\lambda \in [0,1]$:

$$\left|\frac{d^2 s(\lambda)}{d\lambda^2}\right| \leq K_{11}\sum_{i=1}^{V}(\hat{D}_i - D_i)^2 \qquad (C.67)$$

Now, for all i, we have:

$$|\hat{D}_i - D_i| = \left|\sum_{u\in U|i_u=i}\frac{-\tau_u}{R_i}\Delta C_{ij_u} + \sum_{u\in U|j_u=i}\frac{\tau_u}{R_i}\Delta C_{i_u i}\right|$$
$$\leq \frac{1}{K_1}\sum_{u\in U}|\tau_u \Delta C_{i_u j_u}| \qquad (C.68)$$

Since there is at most one update per link the sum in the last equation contains at most L terms. Using this fact and Cauchy's inequality, we get:

$$(\hat{D}_i - D_i)^2 \leq \frac{L}{(K_1)^2}\sum_{u\in U}(\tau_u \Delta C_{i_u j_u})^2 \qquad (C.69)$$

Using this equation in (C.67), we get:

$$\left|\frac{d^2 s(\lambda)}{d\lambda^2}\right| \leq \frac{VLK_{11}}{(K_1)^2} \sum_{u \in U} (\tau_u \Delta C_{i_u j_u})^2 \qquad (C.70)$$

Q.E.D.

Lemma C.3: Assume that:

$$\bar{\tau} \leq \frac{(K_1)^2}{VLK_{11}}$$

Then:

$$S\big(A(\vec{D})\big) - S(D) \leq -\frac{1}{2} \sum_{u \in U} \tau_u \big(\Delta C_{i_u j_u}\big)^2$$

Proof: From equation (C.63) and lemmas C.1 and C.2, we get:

$$\begin{aligned} S\big(A(\vec{D})\big) - S(D) &\leq -\sum_{u \in U} \tau_u (\Delta C_{i_u j_u})^2 + \frac{VLK_{11}}{2(K_1)^2} \sum_{u \in U} (\tau_u \Delta C_{i_u j_u})^2 \\ &\leq -\sum_{u \in U} \left[1 - \frac{VLK_{11}}{2(K_1)^2} \tau_u\right] \tau_u (\Delta C_{i_u j_u})^2 \\ &\leq -\sum_{u \in U} \left[1 - \frac{VLK_{11}}{2(K_1)^2} \bar{\tau}\right] \tau_u (\Delta C_{i_u j_u})^2 \\ &\leq -\frac{1}{2} \sum_{u \in U} \tau_u (\Delta C_{i_u j_u})^2 \qquad (C.71) \end{aligned}$$

where the last step follows directly from the condition imposed on $\bar{\tau}$ in the lemma.

Q.E.D.

The last result shows that, whatever the current assignment $\vec{D}$ may be, the assignment $A(\vec{D})$ cannot be worse. However, this result does not guarantee that when $\vec{D}$ is not optimal $A(\vec{D})$ actually improves the situation. This, however, is the case, as the next result shows.

Lemma C.4: Let $\vec{D} \in H - H^*$ be a strictly feasible assignment. Then one iteration of Alg_NP_s_e with $\vec{D}$ results in at least one update.

Proof: We argue by contradiction. If there is no update it must be that whenever a v.c. i does not have full priority over another v.c. j, the condition:

$$\frac{1}{R_i} C_i'(D_i) = \frac{1}{R_j} C_j'(D_j) \qquad (C.72)$$

is satified. But, as $\vec{D}$ is strictly feasible, this implies that $\vec{D}$ is optimal, contradicting the fact that $\vec{D} \in H - H^*$.

Q.E.D.

The next lemma is a trivial technical result.

Lemma C.5: Let $\vec{D} \in H$ be a strictly feasible assignment. Let $\vec{\tilde{D}} \in H$ be any strictly feasible assignment satisfying:

$$||\vec{D} - \vec{\tilde{D}}|| \leq \varepsilon$$

Then, for $\varepsilon > 0$ sufficiently small, the following conditions hold. For all i, j:

1) $$D_i^l < D_j^l \Rightarrow \tilde{D}_i^l < \tilde{D}_j^l$$

2) $$\frac{1}{R_i}C_i'(D_i) < \frac{1}{R_j}C_j'(D_j) \Rightarrow \frac{1}{R_i}C_i'(\tilde{D}_i) < \frac{1}{R_j}C_j'(\tilde{D}_j)$$

3) If for some ΔN_{ij} the assignment:

$$D_i^l \leftarrow D_i^l - \frac{\Delta N_{ij}}{R_i}$$
$$D_j^l \leftarrow D_j^l + \frac{\Delta N_{ij}}{R_j}$$

is strictly feasible, then the assignment:

$$\tilde{D}_i^l \leftarrow \tilde{D}_i^l - \frac{\Delta N_{ij}}{2R_i}$$
$$\tilde{D}_j^l \leftarrow \tilde{D}_j^l + \frac{\Delta N_{ij}}{2R_j}$$

is also strictly feasible.

The proof of this result is left to the reader.

The first two conditions guarantee that in a sufficiently small neighbourhood of an assignment the constraints imposed on the ordering of the delays and on the ordering of the marginal costs by the assignment prevail everywhere in the neighbourhood. The third condition guarantees that if an assignment $\vec{D}$ can be updated in a certain way, then the assignments in the neighbourhood of $\vec{D}$ can also, to a substantial extent, be updated in the same way.

Our next objective is to establish a continuity condition which will essentially guarantees that any converging sequence generated by the repeated application of Alg_NP_s_e converges toward an optimal solution.

Lemma C.6: Let $\vec{D} \in H - H^*$ be a strictly feasible assignment. Also, let $\vec{\tilde{D}} \in H$ be any strictly feasible assignment satisfying:

$$||\vec{D} - \vec{\tilde{D}}|| \leq \varepsilon \tag{C.73}$$

Then, for $\varepsilon > 0$ sufficiently small, we have:

$$S\left(A(\vec{\tilde{D}})\right) < S(\vec{D})$$

Proof: First note from equations (C.59) and (C.73) that for all i, j:

$$\left|\Delta C_{ij}(\text{in } \vec{D}) - \Delta C_{ij}(\text{in } \vec{\tilde{D}})\right| \leq K_{12}\varepsilon \tag{C.74}$$

where $K_{12} > 0$ is a constant depending only on K_9 and on the rate assignment. (We explicitly indicate here the dependency of ΔC_{ij} on $\vec{D}$ because our shorthand notation would otherwise be confusing.)

Since $\vec{D}$ is not optimal, lemma C.4 guarantees that one iteration of Alg_NP_s_e with $\vec{D}$ results in at least one update. Accordingly, let u be one of these updates.

Assume that ε is sufficiently small to insure that the propositions of lemma C.5 hold and consider the update:

$$\begin{aligned} \tilde{D}^{l_u}_{i_u} &\leftarrow \tilde{D}^{l_u}_{i_u} - \frac{\tau}{R_{i_u}}\Delta C_{i_u j_u}(\text{in } \vec{\tilde{D}}) \\ \tilde{D}^{l_u}_{j_u} &\leftarrow \tilde{D}^{l_u}_{j_u} + \frac{\tau}{R_{j_u}}\Delta C_{i_u j_u}(\text{in } \vec{\tilde{D}}) \end{aligned} \tag{C.75}$$

If $\Delta C_{i_u j_u}(\text{in } \vec{D})$ and $\Delta C_{i_u j_u}(\text{in } \vec{\tilde{D}})$ were strictly equal, it would follow immediately from proposition 3 of lemma C.5 that the assignment produced by the update (C.75) would be strictly feasible for $\tau = \frac{1}{2}\tau_u(\text{in } \vec{D})$ (again, we explicitly indicate the dependency of τ_u on $\vec{D}$ because our shorthand notation would otherwise be confusing). We do not have strict equality between $\Delta C_{i_u j_u}(\text{in } \vec{D})$ and $\Delta C_{i_u j_u}(\text{in } \vec{\tilde{D}})$ but, however, equation (C.74) guarantees

that by choosing ε small enough, the difference between the two terms can be made as small as desired. In these conditions it is clear that we can choose ε small enough to guarantee that:

$$\tau_u(\text{in } \vec{\bar{D}}) \geq \frac{1}{3}\tau_u(\text{in } \vec{D}) \tag{C.76}$$

From equations (C.74) and (C.76), we get:

$$\begin{aligned}\tau_u(\text{in } \vec{\bar{D}})\left(\Delta C_{i_u j_u}(\text{in } \vec{\bar{D}})\right)^2 &\geq \frac{1}{3}\tau_u(\text{in } \vec{D})\left(\Delta C_{i_u j_u}(\text{in } \vec{D}) - K_{12}\varepsilon\right)^2 \\ &\geq \frac{1}{3}\tau_u(\text{in } \vec{D})\left(\Delta C_{i_u j_u}(\text{in } \vec{D})\right)^2 + K_{13}\varepsilon + K_{14}\varepsilon^2 \end{aligned} \tag{C.77}$$

where K_{13} and K_{14} are some strictly positive constants independent of ε.

We also have:

$$\left|S(\vec{D}) - S(\vec{\bar{D}})\right| \leq K_{15}\varepsilon \tag{C.78}$$

where $K_{15} > 0$ depends only on K_8.

Now from lemma C.3 and the update rule we have using equations (C.77) and (C.78):

$$\begin{aligned} S\left(A(\vec{\bar{D}})\right) &\leq -\frac{1}{2}\tau_u(\text{in } \vec{\bar{D}})\left(\Delta C_{i_u j_u}(\text{in } \vec{\bar{D}})\right)^2 + S(\vec{\bar{D}}) \\ &\leq -\frac{1}{6}\tau_u(\text{in } \vec{D})\left(\Delta C_{i_u j_u}(\text{in } \vec{D})\right)^2 + \left(\frac{K_{13}}{2} + K_{15}\right)\varepsilon + \frac{K_{14}}{2}\varepsilon^2 + S(\vec{D}) \end{aligned} \tag{C.79}$$

The first term in the right hand side is independent of ε and, by assumption, it is strictly negative. Thus, for ε small enough, we obtain:

$$S\left(A(\vec{\bar{D}})\right) < S(\vec{D}) \tag{C.80}$$

Q.E.D.

Now, we are ready to prove theorem 4.4. The proof is essentially identical to the proof of lemma 7 of [Gal77]. We provide it here for completeness. Of course we assume that $\bar{\tau}$ satisfies the conditions of lemmas C.2 and C.3.

The first proposition of theorem 4.4 follows immediately from assumptions (A.4.1) and from the form of the update rule. Now, concerning the second proposition, it follows from lemma C.3 that $S\left(A^k(\vec{D}(0))\right)$ is non–increasing in k, which immediately imply that the whole sequence $\left\{(A^k(\vec{D}(0))\right\}_{k=0}^{\infty}$ is contained in H. Since H is compact, it follows that

the sequence must have at least one converging subsequence. Let $\left\{\left(A^k(\vec{D}(0))\right\}_{k\in\Theta}\right.$ be one such subsequence and let $\vec{D}^* \in H$ be the point to which the subsequence converges.

Since $S(\cdot)$ is continuous:

$$S(\vec{D}^*) = \lim_{k\to\infty,\ k\in\Theta} S\left(A^k(\vec{D}(0))\right) \tag{C.81}$$

which, in view of the fact that $S\left(A^k(\cdot)\right)$ in non–increasing in k, imply:

$$S(\vec{D}^*) = \lim_{k\to\infty} S\left(A^k(\vec{D}(0))\right) \tag{C.82}$$

Namely, the whole sequence converges to $S(\vec{D}^*)$. To prove the second proposition it is sufficient to show that $\vec{D}^*$ is an optimal solution. We argue by contradiction. Suppose $\vec{D}^*$ is not optimal. Take $k \in \Theta$ such that:

$$\left|\left|A^k(\vec{D}(0)) - \vec{D}^*\right|\right| \leq \varepsilon \tag{C.83}$$

where ε is sufficiently small to insure that lemma C.6 holds. Since the subsequence converges to $\vec{D}^*$, it is clear that a k such that equation (C.83) holds can always be found. Accordingly, lemma C.6 insures that:

$$S\left(A^{k+1}(\vec{D}(0))\right) < S(\vec{D}^*) \tag{C.84}$$

since $S\left(A^k(\cdot)\right)$ is non–increasing in k this implies that equation (C.82) is false, which is a contradiction.

Q.E.D.

Appendix D: Complement to Chapter 5.

D.1 Proof of lemma 5.1.

Let $(\vec{\tilde{R}}, \vec{\tilde{N}})$ and $(\vec{\hat{R}}, \vec{\hat{N}})$ be two weakly feasible assignments on link l and consider the assignment:

$$(\vec{R}, \vec{N}) = \alpha(\vec{\tilde{R}}, \vec{\tilde{N}}) + (1-\alpha)(\vec{\hat{R}}, \vec{\hat{N}}) \tag{D.1}$$

We must show that $(\vec{R}, \vec{N})$ is weakly feasible on link l for any $\alpha \in [0,1]$. Clearly, for all $i \in \mathcal{V}^l$:

$$\begin{aligned} R_i &= \alpha\tilde{R}_i + (1-\alpha)\hat{R}_i \\ &\geq 0 \end{aligned} \tag{D.2}$$

and:

$$\begin{aligned} \sum_{i\in\mathcal{V}^l} R_i &= \alpha \sum_{i\in\mathcal{V}^l} \tilde{R}_i + (1-\alpha) \sum_{i\in\mathcal{V}^l} \hat{R}_i \\ &\leq \alpha\mu^l + (1-\alpha)\mu^l \\ &\leq \mu^l \end{aligned} \tag{D.3}$$

To complete the proof it remains to show that $(\vec{R}, \vec{N})$ satisfies equation (5.4). Let g be any non–empty subset of $\mathcal{V}^l$. Since $(\vec{\tilde{R}}, \vec{\tilde{N}})$ and $(\vec{\hat{R}}, \vec{\hat{N}})$ are weakly feasible on link l, we have from theorem 2.2;

$$\sum_{i\in g} \tilde{N}_i^l \geq \frac{\sum_{i\in g} \tilde{R}_i}{\mu^l - \sum_{i\in g} \tilde{R}_i} \tag{D.4}$$

$$\sum_{i\in g} \hat{N}_i^l \geq \frac{\sum_{i\in g} \hat{R}_i}{\mu^l - \sum_{i\in g} \hat{R}_i} \tag{D.5}$$

Using these equations and the convexity of the function $f(x) = x/(\mu - x)$, it follows that for all $\alpha \in [0,1]$:

$$\begin{aligned} \sum_{i\in g} N_i^l &= \alpha \sum_{i\in g} \tilde{N}_i^l + (1-\alpha) \sum_{i\in g} \hat{N}_i^l \\ &\geq \alpha \frac{\sum_{i\in g} \tilde{R}_i}{\mu^l - \sum_{i\in g} \tilde{R}_i} + (1-\alpha) \frac{\sum_{i\in g} \hat{R}_i}{\mu^l - \sum_{i\in g} \hat{R}_i} \end{aligned}$$

$$\geq \frac{\sum_{i\in g}\left[\alpha\tilde{R}_i + (1-\alpha)\hat{R}_i\right]}{\mu^l - \sum_{i\in g}\left[\alpha\tilde{R}_i + (1-\alpha)\hat{R}_i\right]}$$

$$\geq \frac{\sum_{i\in g} R_i}{\mu^l - \sum_{i\in g} R_i} \qquad (D.6)$$

As g is arbitrary it follows that equation (5.4) holds for any ordering $\vec{w}^l$ valid for the assignment $(\vec{R}, \vec{N})$.

Q.E.D.

D.2 Proof of theorem 5.1.

In view of the form of the cost functions it is not difficult to see that (FC_s) can be written as:

$$\begin{gathered}\min S(\vec{R},\vec{N}) \\ \{R_i,\ N_i^l,\ i\in\mathcal{V}^l,\ \text{weakly feasible}\}\quad l=1,\ldots,L\end{gathered} \tag{D.7}$$

Indeed this follows immediately from the fact that any optimal solution to this problem must in fact be strictly feasible. By definition of weak feasibility and using theorem 2.2 this means that (FC_s) can be written as:

$$\begin{aligned}&\min S(\vec{R},\vec{N})\\ &R_i\geq 0\quad i=1,\ldots,V\\ &\sum_{i\in\mathcal{V}^l}R_i\leq\mu^l\quad l=1,\ldots,L\\ &\sum_{i\in g^l}N_i^l\geq\frac{\sum_{i\in g^l}R_i}{\mu^l-\sum_{i\in g^l}R_i}\quad\text{for all } g^l\subseteq\mathcal{V}^l,\quad l=1,\ldots,L\end{aligned} \tag{D.8}$$

It is easy to see that the set of feasible assignments in (D.8) is convex, and that it contains a strict interior point. Moreover in view of the form of the cost functions it is also easy to see from (D.8) that (FC_s) has at least one optimal solution.

Define the dual functional:

$$\begin{aligned}q(\vec{\varsigma})=\min\Bigg\{&\sum_{i=1}^{V}G_i(R_i,N_i)+\sum_{l=1}^{L}\sum_{g^l\subseteq\mathcal{V}^l}\varsigma_{g^l}^l\Big(\frac{\sum_{i\in g^l}R_i}{\mu^l-\sum_{i\in g^l}R_i}-\sum_{i\in g^l}N_i^l\Big)\Bigg\}\\ &R_i\geq 0\quad i=1,\ldots,V\\ &\sum_{i\in\mathcal{V}^l}R_i\leq\mu^l\quad l=1,\ldots,L\end{aligned} \tag{D.9}$$

$\varsigma_{g^l}^l$ is the dual variable associated on link l with the constraint defined by the set g^l. Because the set of feasible assignments in (D.8) contains a strict interior point theorem 3.1 guarantees the existence of a Lagrange multiplier vector $\vec{\varsigma^*}$. Furthermore we obtain from theorem 3.2 that $(\vec{R^*},\vec{N^*})$ and $\vec{\varsigma^*}$ are respectively an optimal solution to (FC_s) and a Lagrange multiplier vector if and only if the following conditions hold:

C.D.1.1) $(\vec{R^*},\vec{N^*})$ is strictly feasible.

C.D.1.2) $\vec{\varsigma^*} \geq 0$.

C.D.1.3) For all $l = 1, \ldots, L$, and $g^l \subseteq \mathcal{V}^l$;

$$\varsigma_{g^l}^{*\,l}\Big(\frac{\sum_{i\in g^l} R_i^*}{\mu^l - \sum_{i\in g^l} R_i^*} - \sum_{i\in g^l} N_i^{*\,l}\Big) = 0$$

C.D.1.4) $(\vec{R^*}, \vec{N^*})$ achieves the minimum of $q(\vec{\varsigma})$ for $\vec{\varsigma} = \vec{\varsigma^*}$.

We are going to show that these conditions are equivalent to the conditions (1)–(3) of theorem 5.1. First we show that if $(\vec{R^*}, \vec{N^*})$ satisfies the conditions of theorem 5.1, then it also satisfies the conditions given above. Following this the reverse implication is shown.

1) $(\vec{R^*}, \vec{N^*})$ satisfies theorem 5.1 $\Rightarrow$ $(\vec{R^*}, \vec{N^*})$ satisfies conditions (C.D.1.1)–(C.D.1.4).

Condition (C.D.1.1) is true because it is identical to the condition (1) of theorem 5.1.

For $l = 1, \ldots, L$, let $\vec{w}^l$ be an ordering on link l valid for the assignment $(\vec{R^*}, \vec{N^*})$. Take a particular link l, $l = 1, \ldots, L$, and consider the (P_s^l) problem that can be defined on link l using the assignment $(\vec{R^*}, \vec{N^*})$. Namely this problem is:

$$\begin{aligned} &\min \sum_{i\in\mathcal{V}^l} C_i^l(D_i^l) \\ &D_i^l,\ i \in \mathcal{V}^l,\ \text{strictly feasible} \end{aligned} \tag{D.10}$$

where for all $i \in \mathcal{V}^l$:

$$C_i^l(D_i^l) = G_i\big[R_i^*, R_i^*\big(\sum_{\bar{l}\in\mathcal{L}_i,\ \bar{l}\neq l} D_i^{*\,\bar{l}} + D_i^l\big)\big] \tag{D.11}$$

By assumption $D_i^{*\,l}$, $i \in \mathcal{V}^l$, is the optimal solution to this problem. Accordingly it follows from theorem 3.3 that there exist $\lambda_k^{*\,l}$, $k = 1, \ldots, V^l$, satisfying:

$$\lambda_k^{*\,l} \geq 0 \quad k = 1, \ldots, V^l \tag{D.12}$$

$$\lambda_k^{*\,l}\big(B^l(k, \vec{w}^l, \vec{R^*}) - \sum_{p|w_p^l\leq k} N_p^{*\,l}\big) = 0 \quad k = 1, \ldots, V^l \tag{D.13}$$

For $l = 1, \ldots, L$, consider the assignment;

$$\varsigma_{g^l}^{*\,l} = \begin{cases} \lambda_k^{*\,l}, & \text{if } g^l = \{p \mid w_p^l \leq k\}, \quad k = 1, \ldots, V^l \\ 0, & \text{otherwise} \end{cases} \tag{D.14}$$

From equations (D.12) and (D.14) we get condition (C.D.1.2). Also from equations (D.13) and (D.14) we get condition (C.D.1.3).

For the dual variable assignment given by equation (D.14) the dual functional can be written as:

$$q(\vec{\varsigma}^*) = \min\left\{\sum_{i=1}^{V} G_i(R_i, N_i) + \sum_{l=1}^{L}\sum_{k=1}^{V^l} \lambda_k^{*\,l}(B^l(k, \vec{w}^l, \vec{R}^*) - \sum_{p|w_p^l \le k} N_p^{*\,l})\right\} \qquad (D.15)$$

$$R_i \ge 0 \quad i = 1, \ldots, V$$

$$\sum_{i \in \mathcal{V}^l} R_i \le \mu^l \quad l = 1, \ldots, L \qquad (D.16)$$

Clearly, $(\vec{R}^*, \vec{N}^*)$ satisfies equations (D.16). Indeed this follows immediately from the fact that $(\vec{R}^*, \vec{N}^*)$ is strictly feasible.

To complete the proof we must show that $(\vec{R}^*, \vec{N}^*)$ minimizes the expression in the right hand side of (D.15). As this function is convex over the set defined by equations (D.16) a sufficient condition for proving that $(\vec{R}^*, \vec{N}^*)$ minimizes it is that all the partial derivatives vanish. Evaluating the partial derivative with respect to R_i, $i = 1, \ldots, V$, we get the equation:

$$\frac{\partial}{\partial R} G_i(R_i^*, N_i^*) + \sum_{l \in \mathcal{L}_i} \sum_{k=w_i^l}^{V^l} B^{l\prime}(k, \vec{w}^l, \vec{R}^*)\lambda_k^{*\,l} = 0 \qquad (D.17)$$

As $\vec{N}^*$ is is optimal for $\vec{R}^*$ it follows that corollary 4.2 applies. Accordingly for any v.c. i, $i = 1, \ldots, V$, we have for all $l \in \mathcal{L}_i$:

$$\frac{\partial}{\partial N} G_i(R_i^*, N_i^*) = \sum_{k=w_i^l}^{V^l} \lambda_k^{*\,l} \qquad (D.18)$$

Using this equation to eliminate the $\lambda_k^{*\,l}$ in equation (D.17) we obtain:

$$\frac{\partial}{\partial R} G_i(R_i^*, N_i^*) = -\sum_{l \in \mathcal{L}_i} \left\{ B^{l\prime}(w_i^l, \vec{w}^l, \vec{R}^*) \frac{\partial}{\partial N} G_i(R_i^*, N_i^*) + \sum_{j|w_j^l > w_i^l} [B^{l\prime}(w_j^l, \vec{w}^l, \vec{R}^*) - B^{l\prime}(w_j^l - 1, \vec{w}^l, \vec{R}^*)] \frac{\partial}{\partial N} G_j(R_j^*, N_j^*) \right\} \qquad (D.19)$$

This condition is precisely the condition (3) of theorem 5.1. Thus, by assumption, it holds for all i, $i = 1, \ldots, V$.

Similarly, evaluating the partial derivative of the expression to be minimized in the right hand side of equation (D.15) with respect to N_i^l, $i = 1, \ldots, L$, $l \in \mathcal{L}_i$, we obtain the equation:

$$\frac{\partial}{\partial N} G_i(R_i^*, N_i^*) - \sum_{k=w_i^l}^{V^l} \lambda_k^{*\,l} = 0 \qquad (D.20)$$

Clearly equation (D.18) guarantees that this equation holds for all i, $i = 1, \ldots, V$, and $l \in \mathcal{L}_i$. This shows that condition (C.D.1.4) also holds, which completes the proof of the first implication.

2) $(\vec{R}^*, \vec{N}^*)$ satisfy conditions (C.D.1.1)–(C.D.1.4) $\Rightarrow$ $(\vec{R}^*, \vec{N}^*)$ satisfy theorem 5.1.

Condition (1) of theorem 5.1 holds because it is identical to condition (C.D.1.1). Condition (2) of theorem 5.1 follows from the fact that if $(\vec{R}^*, \vec{N}^*)$ is an optimal solution to (FC_s), then it is not possible to reduce the objective function value by adjusting $\vec{N}$ for the optimal $\vec{R}^*$, so that $\vec{N}^*$ must be an optimal solution of the (NP_s) problem obtained from (FC_s) by setting $\vec{R} = \vec{R}^*$. Also condition (C.D.1.4) implies that $\vec{R}^*$ satisfies equation (D.19), which is condition (3) of theorem 5.1.

Q.E.D.

D.3 Proof of lemma 5.2.

Before proving lemma 5.2 we first introduce a simple technical result. The result states that when in a strictly feasible assignment two v.c.'s have the same delay on a link, the left hand side of the feasibility constraint containing only one of the two v.c.'s is larger than the right hand side by a positive amount depending only on the rate assignment. Note that if two v.c.'s have the same delay on a link the feasibility constraint on the link containing only one of the two v.c.'s must be satisfied with strict inequality. Indeed if this was not the case it would follow that one v.c. has full priority over the other, so that their delays would not be equal. The result of the lemma tightens this fact by lower bounding in terms of the rate assignment the difference between the left hand side and the right hand side.

Lemma D.1: Let $(\vec{R}, \vec{N})$ be a strictly feasible assignment and let $\vec{w}^l$ be an ordering on link l valid for this assignment. Assume that there exist $K_1 > 0$ and $K_2 > 0$ such that:

$$R_i \geq K_1 \quad \text{for all } i \in \mathcal{V}^l$$

$$\mu^l - \sum_{p \in \mathcal{V}^l} R_i \geq K_2$$

Assume that $D_i^l = D_j^l$ for some $i,\ j \in \mathcal{V}^l,\ i \neq j$ and also assume w.l.o.g. that $w_i^l < w_j^l$. Then:

$$\sum_{p | w_p^l \leq w_i^l} N_p^l \geq B^l(w_i^l, \vec{w}^l, \vec{R}) + \left(\frac{K_1}{K_2}\right)^2$$

Proof: First suppose that i is the v.c. with the highest position in the ordering (i.e., suppose that $w_i^l = 1$). Then we have:

$$\sum_{p | w_p^l \leq w_j^l} N_p^l = D_i^l \sum_{p | w_p^l \leq w_j^l} R_p \qquad (D.21)$$

Also since the assignment is strictly feasible:

$$\sum_{p | w_p^l \leq w_j^l} N_p^l \geq B^l(w_j^l, \vec{w}^l, \vec{R}) \qquad (D.22)$$

From equations (D.21) and (D.22) it follows that;

$$D_i^l \geq \frac{1}{\mu^l - \sum_{p | w_p^l \leq w_j^l} R_j} \qquad (D.23)$$

Multiplying both sides of this equation by R_i and rearranging the right hand side we get:

$$N_i^l \geq \frac{R_i}{\mu^l - R_i} + \frac{R_i \sum_{p|w_i^l < w_p^l \leq w_j^l} R_p}{(\mu^l - R_i)(\mu^l - \sum_{p|w_p^l \leq w_j^l} R_j)}$$
$$\geq \frac{R_i}{\mu^l - R_i} + \left(\frac{K_1}{K_2}\right)^2 \qquad (D.24)$$

where the last step follows immediately from the conditions imposed on the rate assignment in the lemma. This proves the result when i is assumed to have the highest position in the ordering. As a consequence we can assume for the remaining of the proof that i is not the v.c. with the highest position in the ordering.

Consider the problem:

$$\min \sum_{p|w_p^l \leq w_i^l} \tilde{N}_p^l$$
$$\tilde{D}_i^l = \tilde{D}_j^l$$
$$(\vec{R}, \vec{\tilde{N}}) \text{ strictly feasible}$$
$$\vec{w}^l \text{ is valid for } (\vec{R}, \vec{\tilde{N}})$$

We first show that in any optimal solution to this problem the v.c.'s in position $1, \ldots, w_i^l - 1$ have full priority over v.c. i. We use a contradiction argument.

Let $(\vec{R}, \vec{N}^*)$ be an optimal solution to the problem and assume that it is such that some v.c. q satisfying $w_q^l < w_i^l$ has not full priority over v.c. i. W.l.o.g., assume that q satisfies:

$$D_p^{*\,l} < D_q^{*\,l} \text{ for all } p \text{ such that } w_p^l < w_q^l$$
$$D_p^{*\,l} = D_q^{*\,l} \text{ for all } p \text{ such that } w_q^l < w_p^l < w_i^l \qquad (D.25)$$

A typical example illustrating the choice of q satisfying these constraints is depicted in Figure D.1.

Also let r be the v.c. satisfying:

$$D_r^{*\,l} = D_i^{*\,l}$$
$$D_p^{*l} > D_r^{*\,l} \text{ for all } p \text{ such that } w_p^l > w_r^l \qquad (D.26)$$

It is easy to see that the constraints of the problem guarantee that r exists, and that $w_r^l \geq w_j^l$.

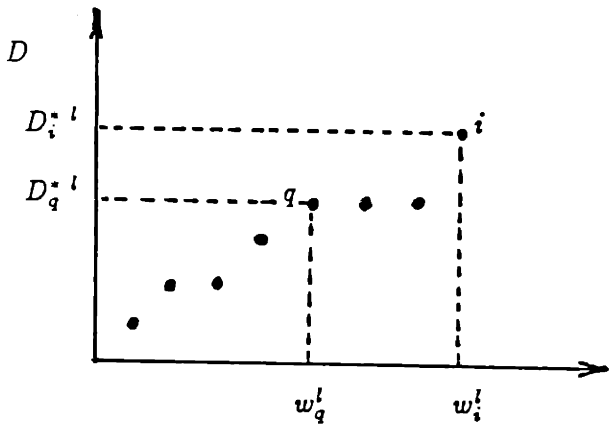


Figure D.1

Consider the update:

$$\begin{aligned} D_q^l &\leftarrow D_q^{*\,l} - \Delta_1 \\ D_p^l &\leftarrow D_p^{*\,l} + \Delta_2 \text{ for all } p \text{ such that } w_i^l \leq w_p^l \leq w_r^l \end{aligned} \tag{D.27}$$

where Δ_1 and Δ_2 satisfy:

$$\Delta_1 R_q = \Delta_2 \sum_{p|w_i^l \leq w_p^l \leq w_r^l} R_p \tag{D.28}$$

It follows immediately from the choice of q and r that there exist $\Delta_1 > 0$ and $\Delta_2 > 0$ satisfying equation (D.28) such that the assignment resulting from the update (D.27) is strictly feasible and satisfies the ordering $\vec{w}^l$.

It is easy to see that the variation of $\sum_{p|w_p^l \leq w_i^l} N_p$ resulting from the update (D.27) is:

$$\begin{aligned} & -\Delta_1 R_q + \Delta_2 R_i \\ &= -\Delta_2 \sum_{p|w_i^l < w_p^l \leq w_r^l} R_p \\ &\leq -\Delta_2 R_j \end{aligned} \tag{D.29}$$

where the second step follows from equation (D.28), and the last step from the fact that by construction of the problem the sum contains at least the v.c. j.

It follows from equation (D.29) that the assignment $(\vec{R}, \vec{N}^*)$ is not optimal, contradicting our assumption. Thus we may conclude that in any optimal assignment the v.c.'s in

position $1, \ldots, w_i^l - 1$ have full priority over v.c. i. Namely any optimal assignment $(\vec{R}, \vec{N}^*)$ satisfies:

$$\sum_{p|w_p^l < w_i^l} N_p^{*\,l} = B^l(w_i^l - 1, \vec{w}^l, \vec{R}) \qquad (D.30)$$

Now since $D_i^{*\,l} = D_j^{*\,l}$ and since the ordering $\vec{w}^l$ is valid for the assignment $(\vec{R}, \vec{N}^*)$ we have:

$$\sum_{p|w_i^l \le w_p^l \le w_j^l} N_p^{*\,l} = D_i^{*\,l} \sum_{p|w_i^l \le w_p^l \le w_j^l} R_p \qquad (D.31)$$

Also as strict feasibility is maintained;

$$\sum_{p|w_p^l \le w_j^l} N_p^{*\,l} \ge B^l(w_j^l, \vec{w}^l, \vec{R}) \qquad (D.32)$$

It follows from equations (D.30), (D.31) and (D.32) that:

$$D_i^{*\,l} \ge \frac{\mu^l}{(\mu^l - \sum_{p|w_p^l \le w_j^l} R_p)(\mu^l - \sum_{p|w_p^l < w_i^l} R_p)} \qquad (D.33)$$

using equations (D.30) and (D.33) to lower bound $\sum_{p|w_p^l \le w_i^l} N_p^{*\,l}$, we obtain:

$$\sum_{p|w_p^l \le w_i^l} N_p^{*\,l} \ge B^l(w_i^l - 1, \vec{w}^l, \vec{R}) + \frac{\mu^l R_i}{(\mu^l - \sum_{p|w_p^l \le w_j^l} R_p)(\mu^l - \sum_{p|w_p^l < w_i^l} R_p)} \qquad (D.34)$$

rearranging the right hand side the last expression can be written as:

$$\begin{aligned}\sum_{p|w_p^l \le w_i^l} N_p^{*\,l} \ge & B^l(w_i^l, \vec{w}^l, \vec{R}) \\ & + \frac{\mu^l R_i \sum_{p|w_i^l < w_p^l \le w_j^l} R_p}{(\mu^l - \sum_{p|w_p^l \le w_j^l} R_p)(\mu^l - \sum_{p|w_p^l \le w_i^l} R_p)(\mu^l - \sum_{p|w_p^l < w_i^l} R_p)} \\ \ge & B^l(w_i^l, \vec{w}^l, \vec{R}) + \left(\frac{K_1}{K_2}\right)^2 \qquad (D.35)\end{aligned}$$

As $\sum_{p|w_p^l \le w_i^l} N_p^{*\,l}$ lower bounds $\sum_{p|w_p^l \le w_i^l} N_p^l$, this completes the proof.

Q.E.D.

Now we can prove lemma 5.2. We prove the lemma only in the case of the first assignment; i.e., when the rate of a v.c. is increased. The proof in the case of the second assignment is very similar and is left to the reader.

The plan of the proof is as follows. We first show that the feasibility constraints $w^l_{\underline{i}(f)}$, where f is an arbitrary delay group on link l, remain satisfied when the assignment is updated as in lemma 5.2 in the case in which the rate of a v.c. is increased. Following this we show that for all $l \in \mathcal{L}_i$ the feasibility constraint V^l remains satisfied with equality. Finally, combining these results, we prove lemma 5.2.

Let $(\vec{R}, \vec{N})$ be a strictly feasible assignment satisfying the conditions of lemma 5.2. Let i be the v.c. whose rate is increased and denote by ΔR_i the variation of i's rate. Also, let $(\vec{\hat{R}}, \vec{\hat{N}})$ be the assignment resulting from the update when the update is as specified in lemma 5.2 in the case in which the rate of a v.c. is increased.

Let f be an arbitrary delay group on some link $l \in \mathcal{L}_i$. We first show that there exists $K_4 > 0$ depending only on K_1, K_2 and ν such that for all ΔR_i, $0 \leq \Delta R_i \leq K_4$, the feasibility constraint $w^l_{\underline{i}(f)}$ in the assignment $(\vec{R}, \vec{N})$ remains the same feasibility constraint in the assignment $(\vec{\hat{R}}, \vec{\hat{N}})$, and that this constraint is still satisfied.

By definition of the delay groups we have:

$$\begin{aligned} D^l_j &> D^l_k + \nu \quad \text{for all } j \in f,\ k \in \tilde{f},\ f > \tilde{f},\ f,\ \tilde{f} \in F^l \\ D^l_j &< D^l_k - \nu \quad \text{for all } j \in f,\ k \in \tilde{f},\ f < \tilde{f},\ f,\ \tilde{f} \in F^l \end{aligned} \tag{D.36}$$

It follows from these conditions, from the conditions imposed on the rate assignment in the lemma and from the form of the update rule used to construct the assignment $(\vec{\hat{R}}, \vec{\hat{N}})$ that there must exist $K_4 > 0$ depending only on K_1, K_2 and ν such that for all ΔR_i, $0 \leq \Delta R_i \leq K_4$, we have:

$$\begin{aligned} \hat{D}^l_j &> \hat{D}^l_k \quad \text{for all } j \in f,\ k \in \tilde{f},\ f > \tilde{f},\ f,\ \tilde{f} \in F^l \\ \hat{D}^l_j &< \hat{D}^l_k \quad \text{for all } j \in f,\ k \in \tilde{f},\ f < \tilde{f},\ f,\ \tilde{f} \in F^l \end{aligned} \tag{D.37}$$

where f, $\tilde{f}$ and F^l are still defined based on the assignment $(\vec{R}, \vec{N})$. It is easy to see that the preceeding equation implies that the feasibility constraint $w^l_{\underline{i}(f)}$ in the assignment $(\vec{\hat{R}}, \vec{\hat{N}})$ is the same as in the assignment $(\vec{R}, \vec{N})$.

If f is such that $w^l_{\underline{i}(f)} < w^l_i$ it is clear that the constraint $w^l_{\underline{i}(f)}$ is still satisfied in the assignment $(\vec{\hat{R}}, \vec{\hat{N}})$ simply because the a.n.o.p. and rate assignment of the v.c.'s in positions

$1, \ldots, w^l_{\underline{i}(f^l_i - 1)}$ do not change, and because by assumption the initial assignment is strictly feasible. If $w^l_{\underline{i}(f)} \geq w^l_i$ the feasibility constraint $w^l_{\underline{i}(f)}$ in the assignment $(\vec{\hat{R}}, \vec{\hat{N}})$ is:

$$\sum_{p | w^l_p \leq w^l_{\underline{i}(f)}} \hat{N}^l_p \geq B^l(w^l_{\underline{i}(f)}, \vec{w}^l, \vec{R} + \overrightarrow{\Delta R_i}) \qquad (D.38)$$

Note that by construction of the assignment $(\vec{\hat{R}}, \vec{\hat{N}})$ we have:

$$\begin{aligned} \sum_{p | w^l_p \leq w^l_{\underline{i}(f)}} \hat{N}^l_p = & \sum_{p | w^l_p \leq w^l_{\underline{i}(f)}} N^l_p + B^l(w^l_{\underline{i}(f^l_i)}, \vec{w}^l, \vec{R} + \overrightarrow{\Delta R_i}) - B^l(w^l_{\underline{i}(f^l_i)}, \vec{w}^l, \vec{R}) \\ & + \sum_{\tilde{f} \in F^l,\ f^l_i < \tilde{f} \leq f} \Big\{ B^l(w^l_{\underline{i}(\tilde{f})}, \vec{w}^l, \vec{R} + \overrightarrow{\Delta R_i}) - B^l(w^l_{\underline{i}(\tilde{f})}, \vec{w}^l, \vec{R}) \\ & - B^l(w^l_{\underline{i}(\tilde{f}-1)}, \vec{w}^l, \vec{R} + \overrightarrow{\Delta R_i}) + B^l(w^l_{\underline{i}(\tilde{f}-1)}, \vec{w}^l, \vec{R}) \Big\} \end{aligned} \qquad (D.39)$$

cancelling the common terms in the right hand side we get:

$$\sum_{p | w^l_p \leq w^l_{\underline{i}(f)}} \hat{N}^l_p = \sum_{p | w^l_p \leq w^l_{\underline{i}(f)}} N^l_p + B^l(w^l_{\underline{i}(f)}, \vec{w}^l, \vec{R} + \overrightarrow{\Delta R_i}) - B^l(w^l_{\underline{i}(f)}, \vec{w}^l, \vec{R}) \qquad (D.40)$$

Now since the assignment $(\vec{R}, \vec{N})$ is strictly feasible we have:

$$\sum_{p | w^l_p \leq w^l_{\underline{i}(f)}} N^l_p \geq B^l(w^l_{\underline{i}(f)}, \vec{w}^l, \vec{R}) \qquad (D.41)$$

Using equation (D.41) in equation (D.40) it follows immediately that equation (D.38) holds. This completes the proof that there exists $K_4 > 0$ depending only on K_1, K_2 and ν such that for all $l = 1, \ldots, L$, $f \in F^l$, and ΔR_i, $0 \leq \Delta R_i \leq K_4$, the feasibility constraint $w^l_{\underline{i}(f)}$ in the assignment $(\vec{R}, \vec{N})$ remains the same in the assignment $(\vec{\hat{R}}, \vec{\hat{N}})$ and that this constraint remains satisfied.

Next we show that on all links $l \in \mathcal{L}_i$, the feasibility constraint V^l remains satisfied with equality in the assignment $(\vec{\hat{R}}, \vec{\hat{N}})$. Consider a particular link $l \in \mathcal{L}_i$ and let $f_{\max}$ be the largest delay group on this link. Namely $f_{\max}$ satisfies $f_{\max} \geq f$ for all $f \in F^l$. It is easy to see that the v.c. $\underline{i}(f_{\max})$ is the v.c. in position V^l in the ordering. It follows from this fact and equation (D.40) that:

$$\sum_{p \in \mathcal{V}^l} \hat{N}^l_p = \sum_{p \in \mathcal{V}^l} N^l_p + B^l(V^l, \vec{w}^l, \vec{R} + \overrightarrow{\Delta R_i}) - B^l(V^l, \vec{w}^l, \vec{R}) \qquad (D.42)$$

Also since the assignment $(\vec{R}, \vec{N})$ is strictly feasible we have:

$$\sum_{p \in \mathcal{V}^l} N_p^l = B^l(V^l, \vec{w}^l, \vec{R}) \qquad (D.43)$$

From equations (D.42) and (D.43) we get:

$$\sum_{p \in \mathcal{V}^l} \hat{N}_p^l = B^l(V^l, \vec{w}^l, \vec{R} + \overrightarrow{\Delta R_i}) \qquad (D.44)$$

which shows that the feasibility constraint V^l on link l is still satisfied with equality.

Now we use the preceeding results to prove the lemma. Consider a particular link $l \in \mathcal{L}_i$. It follows from equations (D.37) that for all ΔR_i, $0 \leq \Delta R_i \leq K_4$, the delays of the v.c.'s in positions $1, \ldots, w^l_{\underline{i}(f_i^l - 1)}$ remain smaller than the delays of the v.c.'s in positions $w^l_{\underline{i}(f_i^l - 1)} + 1, \ldots, V^l$. As the rate and a.n.o.p. assignment of the v.c.'s in position $1, \ldots, w^l_{\underline{i}(f_i^l - 1)}$ do not change it follows that the feasibility constraints $1, \ldots, w^l_{\underline{i}(f_i^l - 1)}$ in the assignment $(\vec{R}, \vec{N})$ remains the same in the assignment $(\vec{\hat{R}}, \vec{\hat{N}})$ and that they are still satisfied.

Now consider the next $w^l_{\underline{i}(f_i^l)} - w^l_{\underline{i}(f_i^l)}$ feasibility constraints, namely the feasibility constraints $w^l_{\underline{i}(f_i^l)}, \ldots, w^l_{\underline{i}(f_i^l)}$. Of course if f_i^l contains only one v.c.; namely i, equations (D.37) guarantees that for all ΔR_i, $0 \leq \Delta R_i \leq K_4$, the feasibility constraints $w^l_{\underline{i}(f_i^l)}, \ldots, w^l_{\underline{i}(f_i^l)}$, which in this case reduces to the constraint w_i^l, are still maintained. Thus we can assume that f_i^l contains more than one v.c. In this case it follows from equations (D.37) that for all ΔR_i, $0 \leq \Delta R_i \leq K_4$, the v.c. appearing in each of the feasibility constraint $w^l_{\underline{i}(f_i^l)}, \ldots, w^l_{\underline{i}(f_i^l)}$ is a v.c. in f_i^l. We distinguish two cases depending if $\bar{i}(f_i^l) = i$ or not.

First assume that $\bar{i}(f_i^l) \neq i$. In this case among the v.c.'s in f_i^l, only the delay of v.c. $\bar{i}(f_i^l)$ and v.c. i vary as ΔR_i increases. The delay of v.c. $\bar{i}(f_i^l)$ increases because its a.n.o.p. increases while its rate is constant and the delay of v.c. i decreases because its a.n.o.p. is constant while its rate increases. If the ordering of the v.c.'s in f_i^l does not change for all ΔR_i, $0 \leq \Delta R_i \leq K_4$, it is easy to see that the constraints $w^l_{\underline{i}(f_i^l)}, \ldots, w^l_{\underline{i}(f_i^l)}$ remains satisfied. Indeed in this case the v.c. $\bar{i}(f_i^l)$ remains in all these constraints for all ΔR_i, $0 \leq \Delta R_i \leq K_4$, so that the left hand side of these constraints increases by:

$$B^l(w^l_{\underline{i}(f_i^l)}, \vec{w}^l, \vec{R} + \overrightarrow{\Delta R_i}) - B^l(w^l_{\underline{i}(f_i^l)}, \vec{w}^l, \vec{R}) \qquad (D.45)$$

As the right hand side of these constraints do not increase by more than the preceeding quantity and as the assignment is initially strictly feasible it follows that the constraints remain satisfied.

Now if the ordering changes, it must be that either the delay of v.c. $\bar{i}(f_i^l)$ becomes larger than the delay of some v.c. $j \in f_i^l$ initially satisfying $w_j^l > w_{\bar{i}(f_i^l)}^l$ or that the delay of v.c. i becomes smaller than the delay of some v.c. $j \in f_i^l$ initially satisfying $w_i^l > w_j^l$. Consider the first case. Immediately before the delay of v.c. $\bar{i}(f_i^l)$ becomes equal to the delay of v.c. j, one of the feasibility constraints must be such that it contains v.c. $\bar{i}(f_i^l)$ but not v.c. j. However when the delays become equal it is easy to see that this feasibility constraint can be replaced by the feasibility constraint which contains v.c. j instead of v.c. $\bar{i}(f_i^l)$ but which is otherwise identical to the preceeding constraint. In fact when the delays become equal either constraint can be used. Note that at this point lemma D.1 guarantees that in both constraints the left hand side is greater than the right hand side by a strictly positive quantity depending only on the rate assignment. When D_j^l becomes smaller than $D_{\bar{i}(f_i^l)}$, the feasibility constraint containing j but not $\bar{i}(f_i^l)$ replaces the feasibility constraint containing $\bar{i}(f_i^l)$ but not j, and the feasibility constraint containing $\bar{i}(f_i^l)$ but not j becomes irrelevant. In view of the conditions imposed on the rate assignment in lemma 5.2 it follows that there exists $K_3 > 0$ depending only on K_1, K_2 and ν such that for all ΔR_i, $0 \leq \Delta R_i \leq K_3$, the new feasibility constraint containing j but not $\bar{i}(f_i^l)$ remains satisfied. In the second case, namely when the delay of v.c. i becomes smaller that the delay of some v.c. j satisfying initially $w_j^l > w_i^l$, we can use the exact same argument to show that there must exist a constant $K_3 > 0$ depending only on K_1, K_2 and ν such that for all ΔR_i, $0 \leq \Delta R_i \leq K_3$, the new feasibility constraint containing i but not j remains satisfied. Clearly this argument can be generalized to the case where many changes of ordering occur. Thus when $\bar{i}(f_i^l) \neq i$ we may conclude that there exists a constant $K_3 > 0$ such that for all ΔR_i, $0 \leq \Delta R_i \leq K_3$, the feasibility constraints $w_{\bar{i}(f_i^l)}^l, \ldots, w_{\bar{i}(f_i^l)}^l$ remain satisfied.

Now suppose $\bar{i}(f_i^l) = i$. Then the only v.c. in f_i^l whose delay varies is i. If D_i^l decreases as R_i increases in the interval $[0, K_4]$ the ordering of the v.c.'s in f_i^l does not change so that the feasibility constraints $w_{\bar{i}(f_i^l)}^l, \ldots, w_{\bar{i}(f_i^l)}^l$ remain satisfied. If D_i^l increases it is easy to see

using an argument identical to the argument made in the preceeding paragraph that there must exists a constant $K_3 > 0$ depending only on K_1, K_2 and ν such that for all ΔR_i, $0 \leq \Delta R_i \leq K_3$, the feasibility constraints $w^l_{\underline{i}(f^l_i)}, \cdots, w^l_{\underline{i}(f^l_i)}$ remain satisfied.

This proves that there exists $K_3 > 0$ depending only on K_1, K_2 and ν such that in all case the feasibility constraints $w^l_{\underline{i}(f^l_i)}, \cdots, w^l_{\underline{i}(f^l_i)}$ remain satisfied for all ΔR_i, $0 \leq \Delta R_i \leq K_3$. Now using a similar argument it is not difficult to see that this result still hold for the following $w^l_{\underline{i}(f^l_i+1)} - w^l_{\underline{i}(f^l_i+1)}$ feasibility constraints (namely the feasibility constraints associated with the v.c.'s in the delay group $f^l_i + 1$) and, by induction, that the result holds for all the following feasibility constraints.

Q.E.D.

D.4 Proof of theorem 5.2.

The proof is similar to that of theorem 4.4. We first prove that A_n and A_r maintain strict feasibility. Following this we prove that A_n and A_r are descent algorithms. Then we prove a similar continuity condition as that of lemma C.6. Finally, combining these results, we prove theorem 5.2.

D.4.1 Strict feasibility.

Define:

$$H = \left\{(\vec{R},\vec{N}) \mid (\vec{R},\vec{N}) \text{ is strictly feasible and } S(\vec{R},\vec{N}) \leq S\big(\vec{R}(0),\vec{N}(0)\big)\right\} \qquad (D.46)$$

Also define H^* as the set of optimal solutions to (FC_s).

By definition of strict feasibility, and in view of the form of the cost functions, it is clear that H is compact. Also it is easy to see that there exist strictly positive constants $K_1,\ldots,K_{10}$ such that for any assignment $(\vec{R},\vec{N}) \in H$;

$$\begin{aligned}
R_i &\geq K_1 \\
N_i^l &\leq K_2 \\
\frac{\partial}{\partial N} G_i(R_i,N_i) &\leq K_3 \\
\frac{\partial^2}{\partial N^2} G_i(R_i,N_i) &\leq K_4 \\
\mu^l - \sum_{i\in\mathcal{V}^l} R_i &\geq K_5 \\
-\frac{\partial}{\partial R} G_i(R_i,N_i) &\leq K_6 \\
\frac{\partial^2}{\partial R^2} G_i(R_i,N_i) &\leq K_7 \\
z_i^l &\leq K_8 \\
\underline{z}_i^l &\leq K_9 \\
\overline{z}_i^l &\leq K_{10}
\end{aligned} \qquad (D.47)$$

Let $A_n(\vec{R},\vec{N})$ and $A_r(\vec{R},\vec{N})$ denote respectively the assignment produced by one iteration of A_n and A_r starting from the assignment $(\vec{R},\vec{N})$. We have the following result.

Lemma D.2: Let $(\vec{R}, \vec{N})$ be any assignment in H. Then $A_n(\vec{R}, \vec{N})$ is strictly feasible. Moreover there exists $K_{11} > 0$ depending only on K_1, K_5 and K_6 such that for all σ, $0 \leq \sigma \leq K_{11}$, $A_r(\vec{R}, \vec{N})$ is strictly feasible.

Proof: By construction A_n clearly maintains strict feasibility. Concerning A_r the result follows immediately from lemma 5.2 (refer to the discussion of A_r in section 5.3.1).

Q.E.D.

D.4.2 Descent properties of A_n.

Define:

$$\bar{H}_n = \left\{(\vec{R}, \vec{N}) \mid ||\vec{N} - \vec{\tilde{N}}|| \leq \bar{r}_1 \text{ for some } (\vec{R}, \vec{\tilde{N}}) \in H\right\} \qquad (D.48)$$

where $\bar{r}_1$ is some strictly positive constant.

It is easy to see that $\bar{H}_n$ is compact for any $\bar{r}_1$. Moreover, for each $\bar{r}_1$, there exists $K_{12} > 0$ such that for any $(\vec{R}, \vec{N}) \in \bar{H}_n$:

$$\frac{\partial^2}{\partial N^2} G_i(R_i, N_i) \leq K_{12} \quad \text{for all } i,\ i = 1 \ldots, V, \qquad (D.49)$$

In this section we temporarily use for convenience the notation $(\vec{R}, \vec{\hat{N}}) = A_n(\vec{R}, \vec{N})$. Note that, as must be the case, the assignments $(\vec{R}, \vec{\hat{N}})$ and $(\vec{R}, \vec{N})$ have the same rate assignment.

Let:

$$\bar{r} \leq \frac{\bar{r}_1}{2LK_3} \qquad (D.50)$$

Then it follows immediately from the form of the update rule in A_n and from the definition of H and $\bar{H}_n$ that:

$$(\vec{R}, \vec{N}) \in H \ \Rightarrow\ A_n(\vec{R}, \vec{N}) \in \bar{H}_n \qquad (D.51)$$

.

Let $(\vec{R}, \vec{N})$ be an arbitrary assignment in H. Similarly as in the proof of convergence of Alg_NP_s_e we define $U = \{(u, l_u, r_u, i_u, j_u)\}$ as the set of updates resulting from one iteration of A_n starting from the assignment $(\vec{R}, \vec{N})$. Each 5–tuple (u, l_u, r_u, i_u, j_u) represents one particular update. u is a label identifying the update. l_u is the link on which the

update occurs. τ_u, i_u and j_u are respectively the "τ_{ij}^l", "i" and "j" of step 2 of A_n for the update.

The following results summarize the descent properties of A_n.

Lemma D.3: Assume that in addition to satisfying equation (D.50) $\bar{\tau}$ also satisfies:

$$\bar{\tau} \leq (K_{12} V L)^{-1}$$

Then for all $(\vec{R}, \vec{N}) \in H$:

$$S\big(A_n(\vec{R}, \vec{N})\big) - S(\vec{R}, \vec{N}) \leq -\frac{1}{2} \sum_{u \in U} \tau_u \big(\Delta_n G_{i_u j_u}\big)^2$$

Proof: The proof uses essentially the same arguments as the proofs of lemmas C.1–C.3. For this reason the development made here is relatively concise.

Define:

$$s(\lambda) = S\Big[(\vec{R}, \vec{N}) + \lambda\big(A_n(\vec{R}, \vec{N}) - (\vec{R}, \vec{N})\big)\Big] \tag{D.52}$$

Differentiating $s(\cdot)$ and expressing the variations of the a.n.o.p. assignment as a function of the update parameters we get:

$$\begin{aligned}
\left.\frac{ds(\lambda)}{d\lambda}\right|_{\lambda=0} &= \sum_{i=1}^{V} \frac{\partial}{\partial N} G_i(R_i, N_i)(\hat{N}_i - N_i) \\
&= \sum_{i=1}^{V} \frac{\partial}{\partial N} G_i(R_i, N_i)\Big[\sum_{u \in U | i = i_u} -\tau_u \Delta_n G_{i_u j_u} + \sum_{u \in U | i = j_u} \tau_u \Delta_n G_{i_u j_u}\Big] \\
&= \sum_{u \in U} \Big[-\frac{\partial}{\partial N} G_{i_u}(R_{i_u}, N_{i_u}) \tau_u \Delta_n G_{i_u j_u} + \frac{\partial}{\partial N} G_{j_u}(R_{j_u}, N_{j_u}) \tau_u \Delta_n G_{i_u j_u}\Big] \\
&= -\sum_{u \in U} \tau_u (\Delta_n G_{i_u j_u})^2
\end{aligned} \tag{D.53}$$

Note that for all i, $i = 1, \ldots, V$;

$$\begin{aligned}
|\hat{N}_i - N_i| &= \Big|\sum_{u \in U | i = i_u} -\tau_u \Delta_n G_{i_u j_u} + \sum_{u \in U | i = j_u} \tau_u \Delta_n G_{i_u j_u}\Big| \\
&\leq \sum_{u \in U} \tau_u |\Delta_n G_{i_u j_u}|
\end{aligned} \tag{D.54}$$

Using Cauchy's inequality it follows that for all i, $i = 1, \ldots, V$:

$$(\hat{N}_i - N_i)^2 \leq L \sum_{u \in U} (\tau_u \Delta_n G_{i_u j_u})^2 \qquad (D.55)$$

The conditions imposed on $\bar{\tau}$ in the lemma guarantee that the assignment $A_n(\vec{R}, \vec{N})$ is in $\bar{H}_n$. Since the assignment $(\vec{R}, \vec{N})$ is clearly also in $\bar{H}_n$ it follows that for any $\lambda \in [0,1]$ the assignment $(\vec{R}, \vec{N}) + \lambda[A_n(\vec{R}, \vec{N}) - (\vec{R}, \vec{N})]$ is in $\bar{H}_n$. Using this fact and the preceeding equation we get that for all $\lambda \in [0,1]$:

$$\begin{aligned}
\left| \frac{d^2 s(\lambda)}{d\lambda^2} \right| &= \left| \sum_{i=1}^{V} \frac{\partial^2}{\partial N^2} G_i \Big[(R_i, R_i) + \lambda \big((\hat{R}_i, \hat{N}_i) - (R, N) \big) \Big] (\hat{N}_i - N_i)^2 \right| \\
&\leq K_{12} \sum_{i=1}^{V} (\hat{N}_i - N_i)^2 \\
&\leq V L K_{12} \sum_{u \in U} (\tau_u \Delta_n G_{i_u j_u})^2 \qquad (D.56)
\end{aligned}$$

Now expanding $s(\cdot)$ in Taylor series we can write:

$$S\big(A_n(\vec{R}, \vec{N})\big) - S(\vec{R}, \vec{N}) = \left. \frac{ds(\lambda)}{d\lambda} \right|_{\lambda=0} + \frac{1}{2} \left. \frac{d^2 s(\lambda)}{d\lambda^2} \right|_{\lambda=\lambda^*} \qquad (D.57)$$

where λ^* is some number in the interval $[0,1]$.

Using equations (D.53), (D.56) and the conditions imposed on $\bar{\tau}$ in the lemma we obtain:

$$\begin{aligned}
S\big(A_n(\vec{R}, \vec{N})\big) - S(\vec{R}, \vec{N}) &\leq -\sum_{u \in U} \tau_u (\Delta_n G_{i_u j_u})^2 + \frac{1}{2} V L K_{12} \sum_{u \in U} (\tau_u \Delta_n G_{i_u j_u})^2 \\
&\leq -\sum_{u \in U} \Big[1 - \frac{V L K_{12} \bar{\tau}}{2} \Big] \tau_u (\Delta_n G_{i_u j_u})^2 \\
&\leq -\frac{1}{2} \sum_{u \in U} \tau_u (\Delta_n G_{i_u j_u})^2 \qquad (D.58)
\end{aligned}$$

Q.E.D.

Lemma D.4: Let $(\vec{R}, \vec{N}) \in H$ and assume $\vec{N}$ is not optimal for $\vec{R}$. Then one iteration of A_n with the assignment $(\vec{R}, \vec{N})$ results in at least one update.

The proof of this lemma is identical to that of lemma C.4.

D.4.3 Descent properties of A_r.

In the case of A_r there are two possibilities to consider depending if the iteration causes the rate of a v.c. to increase or to decrease. However, as the arguments are nearly identical in both cases, we only prove the results in the case in which a rate increase occurs.

Let $\bar{H}_r$ be the set containing the assignments $(\vec{R}, \vec{N})$ satisfying for some assignment $(\vec{\tilde{R}}, \vec{\tilde{N}}) \in H$:

$$|\tilde{R}_i - R_i| \leq \frac{1}{2}\min(K_1, K_5) \quad \text{for all } i,\ i = 1, \ldots, V \tag{D.59}$$

$$|\tilde{N}_i^l - N_i^l| \leq \frac{2\mu^l}{K_5} \quad \text{for all } i,\ i = 1, \ldots, V,\ l \in \mathcal{L}_i \tag{D.60}$$

It is easy to see that $\bar{H}_r$ is compact and that there exist strictly positive constants $K_{13}, \ldots, K_{19}$ such that for any assignment $(\vec{R}, \vec{N}) \in \bar{H}_r$:

$$\begin{aligned} R_i &\geq K_{13} \\ N_i^l &\leq K_{14} \\ \frac{\partial}{\partial N} G_i(R_i, N_i) &\leq K_{15} \\ \frac{\partial^2}{\partial N^2} G_i(R_i, N_i) &\leq K_{16} \\ -\frac{\partial}{\partial R} G_i(R_i, N_i) &\leq K_{17} \\ \frac{\partial^2}{\partial R^2} G_i(R_i, N_i) &\leq K_{18} \\ 1 - \frac{1}{\mu^l} \sum_{i \in \nu^l} R_i &\geq K_{19} \end{aligned} \tag{D.61}$$

In this section we temporarily use for convenience the notation $(\vec{\hat{R}}, \vec{\hat{N}}) = A_r(\vec{R}, \vec{N})$.

Let:

$$\sigma \leq \frac{\min(K_1, K_5)}{2L(K_6 + K_9 + K_{10})} \tag{D.62}$$

Then it follows from the form of the update rule in A_r and from the definition of H and $\bar{H}_r$ that:

$$(\vec{R}, \vec{N}) \in H \ \Rightarrow\ A_r(\vec{R}, \vec{N}) \in \bar{H}_r \tag{D.63}$$

Indeed suppose that $A_r(\vec{R}, \vec{N})$ results in R_i being increased. Namely:

$$\hat{R}_i = R_i + \sigma\Big[-\frac{\partial}{\partial R} G_i(R_i, N_i) - \bar{z}_i\Big] \tag{D.64}$$

using equations (D.47) and (D.62) it follows that:

$$|\hat{R}_i - R_i| \leq \frac{\min(K_1, K_5)(K_6 + LK_{10})}{2L(K_6 + K_9 + K_{10})}$$
$$\leq \frac{1}{2}\min(K_1, K_5) \qquad (D.65)$$

which proves that equation (D.59) holds.

Also let N_j^l be any a.n.o.p. increased as the result of i's rate increase. It is easy to see that:

$$|\hat{N}_j^l - N_j^l| \leq B^{l\prime}(V^l, \vec{w}^l, \vec{R} + \overrightarrow{\Delta R_i})\Delta R_i \qquad (D.66)$$

where $\Delta R_i = \sigma \Delta_r G_i$. Using equations (D.62) and (D.47) this implies:

$$|\hat{N}_j^l - N_j^l| \leq \frac{\mu^l \min(K_1, K_5)}{2\left(\mu^l - \sum_{p \in \nu^l} R_p - \frac{1}{2}\min(K_1, K_5)\right)^2}$$
$$\leq \frac{2\mu^l \min(K_1, K_5)}{(K_5)^2}$$
$$\leq \frac{2\mu^l}{K_5} \qquad (D.67)$$

which proves that equation (D.60) also holds. Thus we may conclude that whenever equation (D.62) holds, equation (D.63) also holds.

Let $(\vec{R}, \vec{N}) \in H$ be a given assignment. Assuming that $A_r(\vec{R}, \vec{N})$ results in a rate update we denote by i the v.c. whose rate is updated. Also, for $l \in \mathcal{L}_i$, we denote by U^l the set of v.c.'s whose a.n.o.p. is updated on link l as a result of i's rate update.

Define:

$$s(\lambda) = S\Big[(\vec{R}, \vec{N}) + \lambda\big(A_r(\vec{R}, \vec{N}) - (\vec{R}, \vec{N})\big)\Big] \qquad (D.68)$$

We have the following results.

Lemma D.5: Assume that σ satisfies equation (D.62). Then for all $(\vec{R}, \vec{N}) \in H$;

$$\left.\frac{ds(\lambda)}{d\lambda}\right|_{\lambda=0} \leq -\sigma(\Delta_r G_i)^2 + \frac{8VLK_3}{(K_5)^2}(\sigma\Delta_r G_i)^2$$

Proof: We assume that R_i increases. Differentiating $s(\cdot)$ we obtain:

$$\begin{aligned}\left.\frac{ds(\lambda)}{d\lambda}\right|_{\lambda=0} &= \frac{\partial}{\partial R}G_i(R_i,N_i)(\hat{R}_i - R_i) + \sum_{j=1}^{V}\frac{\partial}{\partial N}G_j(R_j,N_j)(\hat{N}_j - N_j) \\ &= \frac{\partial}{\partial R}G_i(R_i,N_i)\sigma\Delta_r G_i + \sum_{l\in\mathcal{L}_i}\sum_{j\in U^l}\frac{\partial}{\partial N}G_j(R_j,N_j)(\hat{N}_j^l - N_j^l) \qquad (D.69)\end{aligned}$$

where the last step follows directly from the form of the update rule.

It is easy to see that the functions:

$$\begin{aligned}f(x) &= \frac{a+x}{\mu-a-x} - \frac{a}{\mu-a} \\ g(x) &= \left[\frac{a+b+x}{\mu-a-b-x} - \frac{a+b}{\mu-a-b}\right] - \left[\frac{a+x}{\mu-a-x} - \frac{a}{\mu-a}\right]\end{aligned} \qquad (D.70)$$

satisfy for all $a \geq 0$, $b \geq 0$, $x \geq 0$, $\mu \geq 0$ such that $a+b+x < \mu$:

$$\begin{aligned}f(x) &\leq \frac{\mu x}{(\mu-a)^2} + \frac{\mu x^2}{(\mu-a-x)^3} \\ g(x) &\leq \left[\frac{\mu}{(\mu-a-b)^2} - \frac{\mu}{(\mu-a)^2}\right]x + \frac{\mu x^2}{(\mu-a-b-x)^3}\end{aligned} \qquad (D.71)$$

Also, assuming that R_i increases, we obtain from equations (D.47) and (D.62):

$$\begin{aligned}\Delta R_i &= \sigma\left[-\frac{\partial}{\partial R}G_i(R_i,N_i) - \bar{z}_i\right] \\ &\leq \frac{\min(K_1,K_5)(K_6+LK_{10})}{2L(K_6+K_9+K_{10})} \\ &\leq \frac{1}{2}\min(K_1,K_5)\end{aligned} \qquad (D.72)$$

Now using the update rule, equations (D.71), (D.72), and finally equation (D.47) we have for all $l \in \mathcal{L}_i$:

$$\begin{aligned}&\sum_{j\in U^l}\frac{\partial}{\partial N}G_i(R_j,N_j)(\hat{N}_j^l - N_j^l) \\ &= \left(B^l(w^l_{\underline{i}(f_i^l)},\vec{w}^l,\vec{R}+\overrightarrow{\Delta R_i}) - B^l(w^l_{\underline{i}(f_i^l)},\vec{w}^l,\vec{R})\right)\frac{\partial}{\partial N}G_{\overline{i}(f_i^l)}(R_{\overline{i}(f_i^l)},N_{\overline{i}(f_i^l)}) \\ &\quad + \sum_{f\in F^l,\ f>f_i^l}\Big\{\left(B^l(w^l_{\underline{i}(f)},\vec{w}^l,\vec{R}+\overrightarrow{\Delta R_i}) - B^l(w^l_{\underline{i}(f)},\vec{w}^l,\vec{R})\right) \\ &\quad - \left(B^l(w^l_{\underline{i}(f-1)},\vec{w}^l,\vec{R}+\overrightarrow{\Delta R_i}) - B^l(w^l_{\underline{i}(f-1)},\vec{w}^l,\vec{R})\right)\Big\}\frac{\partial}{\partial N}G_{\overline{i}(f)}(R_{\overline{i}(f)},N_{\overline{i}(f)}) \\ &\leq \left(B^{ll}(w^l_{\underline{i}(f_i^l)},\vec{w}^l,\vec{R})\sigma\Delta_r G_i + \frac{\mu^l(\sigma\Delta_r G_i)^2}{\left(\mu^l - \sum_{p|w_p^l\leq w^l_{\underline{i}(f_i^l)}}R_p - \sigma\Delta_r G_i\right)^3}\right)\frac{\partial}{\partial N}G_{\overline{i}(f_i^l)}(R_{\overline{i}(f_i^l)},N_{\overline{i}(f_i^l)})\end{aligned}$$

$$+\sum_{f\in F^l,\ f>f_i^l}\Bigg\{\Big(B^{l\prime}(w^l_{\bar{i}(f)},\vec{w}^l,\vec{R})-B^{l\prime}(w^l_{\bar{i}(f-1)},\vec{w}^l,\vec{R})\Big)\sigma\Delta_r G_i$$

$$+\frac{\mu^l(\sigma\Delta_r G_i)^2}{\left(\mu^l-\sum_{p|w^l_p\le w^l_{\bar{i}(f)}}R_p-\sigma\Delta_r G_i\right)^3}\Bigg\}\frac{\partial}{\partial N}G_{\bar{i}(f)}(R_{\bar{i}(f)},N_{\bar{i}(f)})$$

$$\le\sigma\overline{z}^l_i\Delta_r G_i$$

$$+\frac{K_3\mu^l(\sigma\Delta_r G_i)^2}{\left(\mu^l-\sum_{p|w^l_p\le w^l_{\bar{i}(f^l_i)}}R_p-K_5/2\right)^3}+\sum_{f\in F^l,\ f>f_i^l}\frac{K_3\mu^l(\sigma\Delta_r G_i)^2}{\left(\mu^l-\sum_{p|w^l_p\le w^l_{\bar{i}(f)}}R_p-K_5/2\right)^3}$$

$$\le\sigma\overline{z}^l_i\Delta_r G_i+\frac{8VK_3(\sigma\Delta_r G_i)^2}{(K_5)^2}\qquad(D.73)$$

substituting in equation (D.69) we get:

$$\left.\frac{ds(\lambda)}{d\lambda}\right|_{\lambda=0}=\frac{\partial}{\partial R}G_i(R_i,N_i)\sigma\Delta_r G_i+\sum_{l\in\mathcal{L}_i}\left[\overline{z}^l_i\sigma\Delta_r G_i+\frac{8VK_3(\sigma\Delta_r G_i)^2}{(K_5)^2}\right]$$

$$\le\left[\frac{\partial}{\partial R}G_i(R_i,N_i)+\overline{z}_i\right]\sigma\Delta_r G_i+\frac{8VLK_3(\sigma\Delta_r G_i)^2}{(K_5)^2}$$

$$\le-\sigma(\Delta_r G_i)^2+\frac{8VLK_3(\sigma\Delta_r G_i)^2}{(K_5)^2}\qquad(D.74)$$

Q.E.D.

Lemma D.6: Assume that σ satisfies equation (D.62). Then for all $(\vec{R},\vec{N})\in H$ and $\lambda\in[0,1]$ we have:

$$\left|\frac{d^2s(\lambda)}{d\lambda^2}\right|\le K_{20}(\sigma\Delta_r G_i)^2$$

where:

$$K_{20}=K_{18}+\frac{16L^2K_{16}}{(K_5)^2}$$

Proof: We assume that R_i increases. Differentiating $s(\cdot)$ we obtain:

$$\frac{d^2s(\lambda)}{d\lambda^2}=\frac{\partial^2}{\partial R^2}G_i\Big[(R_i,N_i)+\lambda\big((\hat{R}_i,\hat{N}_i)-(R_i,N_i)\big)\Big](\hat{R}_i-R_i)^2$$

$$+\sum_{j=1}^{V}\frac{\partial^2}{\partial N^2}G_j\Big[(R_j,N_j)+\lambda\big((\hat{R}_j,\hat{N}_j)-(R_j,N_j)\big)\Big](\hat{N}_j-N_j)^2\qquad(D.75)$$

The condition imposed on σ in the lemma guarantees that the assignment $(\vec{\hat{R}},\vec{\hat{N}})$ is in $\bar{H}_r$. As the assignment $(\vec{R},\vec{N})$ is clearly also in in $\bar{H}_r$ it follows that for all $\lambda\in[0,1]$ the

assignment $(R_j, N_j) + \lambda\big[(\hat{R}_j, \hat{N}_j) - (R_j, N_j)\big]$ is in $\bar{H}_r$. Accordingly we get using equations (D.61):

$$\left|\frac{d^2 s(\lambda)}{d\lambda^2}\right| = K_{18}(\hat{R}_i - R_i)^2 + K_{16}\sum_{j=1}^{V}(\hat{N}_j - N_j)^2 \qquad (D.76)$$

It is easy to see that for all j, $j = 1, \ldots, V$:

$$\begin{aligned}\hat{N}_j - N_j &= \sum_{l \in \mathcal{L}_i} \Delta N_j^l \\ &\leq \sum_{l \in \mathcal{L}_i} B^{ll}(V^l, \vec{w}^l, \vec{R} + \overrightarrow{\Delta R_i})\Delta R_i \\ &\leq \frac{4L\sigma\Delta_r G_i}{K_5} \qquad (D.77)\end{aligned}$$

Substituting in equation (D.76) we obtain for all $\lambda \in [0,1]$:

$$\begin{aligned}\left|\frac{d^2 s(\lambda)}{d\lambda^2}\right| &\leq K_{18}(\hat{R}_i - R_i)^2 + \frac{16L^2K_{16}}{(K_5)^2}(\sigma\Delta_r G_i)^2 \\ &\leq \Big(K_{18} + \frac{16L^2K_{16}}{(K_5)^2}\Big)(\sigma\Delta_r G_i)^2 \qquad (D.78)\end{aligned}$$

where the last step follows directly from the form of the update rule.

Q.E.D.

Now we can prove that A_r is a descent algorithm. This is summarized in the following result.

Lemma D.7: Assume that in addition to satisfying equation (D.62) σ also satisfies:

$$\sigma \leq \frac{1}{2}\Big(\frac{8VLK_3}{(K_5)^2} + \frac{K_{20}}{2}\Big)^{-1}$$

Then for all $(\vec{R}, \vec{N}) \in H$ we have:

$$S\big(A_r(\vec{R}, \vec{N})\big) - S(\vec{R}, \vec{N}) \leq -\frac{\sigma}{2}(\Delta_r G_i)^2$$

Proof: Using a Taylor series expansion of $s(\cdot)$ we obtain:

$$S\big(A_r(\vec{R}, \vec{N})\big) - S(\vec{R}, \vec{N}) = \frac{ds(\lambda)}{d\lambda}\bigg|_{\lambda=0} + \frac{1}{2}\frac{d^2 s(\lambda)}{d\lambda^2}\bigg|_{\lambda=\lambda^*} \qquad (D.79)$$

where λ^* is some number in the interval $[0,1]$.

Using lemmas D.5 and D.6 and the conditions imposed on σ in the lemma it follows from the preceeding equation that:

$$\begin{aligned} S\big(A_r(\vec{R},\vec{N})\big) - S(\vec{R},\vec{N}) &\leq -\sigma(\Delta_r G_i)^2 + \frac{8VLK_3}{(K_5)^2}(\sigma\Delta_r G_i)^2 + \frac{K_{20}}{2}(\sigma\Delta_r G_i)^2 \\ &\leq -\left[1 - \sigma\Big(\frac{8VLK_3}{(K_5)^2} + \frac{K_{20}}{2}\Big)\right]\sigma(\Delta_r G_i)^2 \\ &\leq -\frac{\sigma}{2}(\Delta_r G_i)^2 \end{aligned} \qquad (D.80)$$

Q.E.D.

D.4.4 Continuity condition.

In this section we present the counterpart of lemmas C.5 and C.6 in the context of Alg_FC_s.

Lemma D.8: Let $(\vec{R},\vec{N})$ and $(\tilde{\vec{R}},\tilde{\vec{N}})$ be assignment in H satisfying:

$$||(\vec{R},\vec{N}) - \tilde{\vec{R}},\tilde{\vec{N}})|| \leq \varepsilon$$

Then, for $\varepsilon > 0$ sufficiently small, the following conditions hold. For all i,j:

1) $$D_i^l < D_j^l \;\Rightarrow\; \tilde{D}_i^l < \tilde{D}_j^l$$

2) $$\frac{\partial}{\partial N}G_i(R_i,N_i) < \frac{\partial}{\partial N}G_j(R_j,N_j) \;\Rightarrow\; \frac{\partial}{\partial N}G_i(\tilde{R}_i,\tilde{N}_i) < \frac{\partial}{\partial N}G_j(\tilde{R}_j,\tilde{N}_j)$$

3) If for some ΔN_{ij} the assignment:

$$\begin{aligned} N_i^l &\leftarrow N_i^l - \Delta N_{ij} \\ N_j^l &\leftarrow N_j^l + \Delta N_{ij} \end{aligned}$$

is strictly feasible then the assignment:

$$\begin{aligned} \tilde{N}_i^l &\leftarrow \tilde{N}_i^l - \frac{1}{2}\Delta N_{ij} \\ \tilde{N}_j^l &\leftarrow \tilde{N}_j^l + \frac{1}{2}\Delta N_{ij} \end{aligned}$$

is also strictly feasible.

The proof of this result is left to the reader.

Lemma D.9: Assume that $\bar{r}$ satisfies the conditions of Lemma D.3 and that σ satisfies the conditions of lemmas D.2 and D.7. Also assume that:

$$\nu \leq \frac{K_1}{2(K_5)^2}$$

Let $(\vec{R}, \vec{N}) \in H - H^*$. Also let $(\vec{\bar{R}}, \vec{\bar{N}}) \in H$ be any assignment satisfying:

$$||(\vec{R}, \vec{N}) - \vec{\bar{R}}, \vec{\bar{N}})|| \leq \epsilon$$

Then for $\epsilon > 0$ sufficiently small:

$$S\Big[A_r\big(A_n(\vec{\bar{R}}, \vec{\bar{N}})\big)\Big] < S(\vec{R}, \vec{N})$$

where $A_r\big(A_n(\vec{R}, \vec{N})\big)$ denotes the assignment resulting from executing in sequence one iteration of A_n and A_r starting from the assignment $(\vec{R}, \vec{N})$.

Proof: First assume that the a.n.o.p. assignment is not optimal for the rate assignment in the assignment $(\vec{R}, \vec{N})$. Then, using an argument completely analogous to the argument made in the proof of lemma C.6, it is not difficult to show that for $\epsilon > 0$ sufficiently small all the assignments $(\vec{\bar{R}}, \vec{\bar{N}})$ satisfying the condition of the lemma also satisfy:

$$S\big(A_n(\vec{\bar{R}}, \vec{\bar{N}})\big) < S(\vec{R}, \vec{N}) \tag{D.81}$$

It follows from lemmas D.2 and D.3 that the assignment $A_n(\vec{R}, \vec{N})$ is in H. Accordingly we obtain from the preceeding equation and lemma D.7:

$$S\Big[A_r\big(A_n(\vec{\bar{R}}, \vec{\bar{N}})\big)\Big] < S(\vec{R}, \vec{N}) \tag{D.82}$$

which proves that the result holds in this case.

Thus we may assume that the assignment $(\vec{R}, \vec{N})$ is such that the a.n.o.p. assignment is optimal for the rate assignment. Under this assumption it follows from the fact that the assignment $(\vec{R}, \vec{N})$ is not optimal that for at least one v.c. i;

$$-\frac{\partial}{\partial R} G_i(R_i, N_i) \neq z_i \tag{D.83}$$

Let $(\vec{\hat{R}}, \vec{\hat{N}}) = A_n(\vec{\bar{R}}, \vec{\bar{N}})$. It is easy to see that if the assignment $(\vec{R}, \vec{N})$ is such that the a.n.o.p. assignment is optimal for the rate assignment we must have:

$$||(\vec{\hat{R}}, \vec{\hat{N}}) - (\vec{R}, \vec{N})|| \leq K_{21}\varepsilon \tag{D.84}$$

where $K_{21} > 0$ depends only on K_3 and $\bar{r}$.

Now consider two v.c.'s j and k on some link $l \in \mathcal{L}_j \cap \mathcal{L}_k$. Assume that:

$$\frac{\partial}{\partial N} G_j(R_j, N_j) > \frac{\partial}{\partial N} G_k(R_k, N_k) \tag{D.85}$$

As the a.n.o.p. assignment is optimal for the rate assignment in the assignment $(\vec{R}, \vec{N})$ it follows from this condition that j must have full priority over k on link l.

Let $\vec{w}^l$ be an ordering valid for the assignment $(\vec{R}, \vec{N})$ on link l. Also let p be the v.c. in j's priority group on link l with the lowest position in the ordering (namely $p = \underline{i}(e_j^l)$). Since j has full priority over k it is clear that $w_j^l \leq w_p^l < w_k^l$. It follows that j's delay on link l can be upper bounded by:

$$D_j^l \leq \frac{1}{\sum_{q|w_j^l \leq w_q^l \leq w_p^l} R_q} \left(B^l(w_p^l, \vec{w}^l, \vec{R}) - B^l(w_j^l - 1, \vec{w}^l, \vec{R})\right) \tag{D.86}$$

the equality being achieved if j is of lowest priority as compared to the v.c.'s in positions $1, \ldots, w_j^l - 1$, and if $D_j^l = D_q^l$ for all q such that $w_j^l \leq w_q^l \leq w_p^l$. Similarly D_k^l can be lower bounded by:

$$D_k^l \geq \frac{1}{\sum_{q|w_p^l < w_q^l \leq w_k^l} R_q} \left(B^l(w_k^l, \vec{w}^l, \vec{R}) - B^l(w_p^l, \vec{w}^l, \vec{R})\right) \tag{D.87}$$

the equality being achieved if v.c. k has full priority over the v.c.'s in positions $w_k^l + 1, \ldots, V^l$, and if $D_k^l = D_q^l$ for all q such that $w_p^l < w_q^l \leq w_k^l$.

Substracting equation (D.87) from equation (D.86), and using the fact that $w_j^l \leq w_p^l < w_k^l$ and equation (D.47) we obtain:

$$D_k^l - D_j^l \geq \frac{K_1}{(K_5)^2} \tag{D.88}$$

It follows from this equation, the conditions imposed on ν in the lemma and equation (D.84) that for $\varepsilon > 0$ sufficiently small whenever two v.c.'s j and k satisfy:

$$|\hat{D}_j^l - \hat{D}_k^l| \leq \nu \tag{D.89}$$

then they also satisfy:

$$\frac{\partial}{\partial N}G_j(R_j, N_j) = \frac{\partial}{\partial N}G_k(R_k, N_k) \tag{D.90}$$

In view of the definition of the delay groups this condition implies that for all j and $l \in \mathcal{L}_j$:

$$\begin{aligned} \left|\overline{z}_j^l(\text{in } (\vec{\hat{R}}, \vec{\hat{N}})) - z_j^l(\text{in } (\vec{R}, \vec{N}))\right| &\leq K_{22}\varepsilon \\ \left|\underline{z}_j^l(\text{in } (\vec{\hat{R}}, \vec{\hat{N}})) - z_j^l(\text{in } (\vec{R}, \vec{N}))\right| &\leq K_{23}\varepsilon \end{aligned} \tag{D.91}$$

where K_{22} and K_{23} depend only on K_3, K_5 and K_6.

Now it follows from equations (D.47), (D.83) and (D.91) that we can find $\varepsilon > 0$ sufficiently small so that the following equations hold.

$$\begin{aligned} \left|\frac{\partial}{\partial R}G_i(\hat{R}_i, \hat{N}_i) + \underline{z}_i(\text{in } (\vec{\hat{R}}, \vec{\hat{N}}))\right| &\geq \frac{1}{2}\left|\frac{\partial}{\partial R}G_i(R_i, N_i) + z_i(\text{in } (\vec{R}, \vec{N}))\right| \\ \left|\frac{\partial}{\partial R}G_i(\hat{R}_i, \hat{N}_i) + \overline{z}_i(\text{in } (\vec{\hat{R}}, \vec{\hat{N}}))\right| &\geq \frac{1}{2}\left|\frac{\partial}{\partial R}G_i(R_i, N_i) + z_i(\text{in } (\vec{R}, \vec{N}))\right| \end{aligned} \tag{D.92}$$

Together with lemma D.7 these equations imply that the iteration of A_r must reduce the objective function by at least:

$$-\frac{\sigma}{8}\left[\frac{\partial}{\partial R}G_i(R_i, N_i) + z_i(\text{in } (\vec{R}, \vec{N}))\right]^2 \tag{D.93}$$

Also in view of equations (D.47) and (D.84), we have:

$$S(\vec{R}, \vec{N}) - S(\vec{\hat{R}}, \vec{\hat{N}}) \leq K_{24}\varepsilon \tag{D.94}$$

where $K_{24} > 0$ depends only on K_3 and K_6.

From equations (D.93) and (D.94) we get:

$$S\left[A_r(A_n(\vec{\hat{R}}, \vec{\hat{N}}))\right] - S(\vec{R}, \vec{N}) \leq K_{24}\varepsilon - \frac{\sigma}{8}\left[\frac{\partial}{\partial R}G_i(R_i, N_i) + z_i(\text{in } (\vec{R}, \vec{N}))\right]^2 \tag{D.95}$$

The second term in the right hand side is strictly negative and independent of ε. Thus we can always choose $\varepsilon > 0$ sufficiently small to guarantee that the right hand side is strictly negative, which completes the proof.

Q.E.D.

D.4.5 Proof of theorem 5.2.

Now we can prove theorem 5.2. We assume that $\bar{r}$ satisfies the condition of lemma D.3, that σ satisfies the conditions of lemmas D.2 and D.7, and that ν satisfies the condition of lemma D.9.

Lemmas D.3–D.7 guarantee that A_n and A_r map H into H. Also lemma D.2 guarantees that any strictly feasible assignment in H is mapped by A_n and A_r into a strictly feasible assignment. Thus, as the initial assignment is in H and is strictly feasible, it follows that the assignments generated by the repeated application of A_n and A_r are in H and are strictly feasible. This proves the first proposition of theorem 5.2. The second proposition of the theorem can be established using exactly the same argument as in the proof of the second proposition of theorem 4.4. This is left to the reader.