ABSTRACT

Title of dissertation: MULTIPATH ROUTING ALGORITHMS FOR COMMUNICATION NETWORKS: ANT ROUTING AND OPTIMIZATION BASED APPROACHES

Punyaslok Purkayastha
Doctor of Philosophy, 2009

Dissertation directed by: Professor John S. Baras
Department of Electrical and Computer Engineering

In this dissertation, we study two algorithms that accomplish multipath routing in communication networks. The first algorithm that we consider belongs to the class of Ant-Based Routing Algorithms (ARA) that have been inspired by experimental observations of ant colonies. It was found that ant colonies are able to 'discover' the shorter of two paths to a food source by laying and following 'pheromone' trails. ARA algorithms proposed for communication networks employ probe packets called ant packets (analogues of ants) to collect measurements of various quantities (related to routing performance) like path delays. Using these measurements, analogues of pheromone trails are created, which then influence the routing tables.

We study an ARA algorithm, proposed earlier by Bean and Costa, consisting of a delay estimation scheme and a routing probability update scheme, that updates routing probabilities based on the delay estimates. We first consider a simple sce-

nario where data traffic entering a source node has to be routed to a destination node, with N available parallel paths between them. An ant stream also arrives at the source and samples path delays *en route* to the destination. We consider a stochastic model for the arrival processes and packet lengths of the streams, and a queueing model for the link delays. Using stochastic approximation methods, we show that the evolution of the link delay estimates can be closely tracked by a deterministic ODE (Ordinary Differential Equation) system. A study of the equilibrium points of the ODE enables us to obtain the equilibrium routing probabilities and the path delays. We then consider a network case, where multiple input traffic streams arriving at various sources have to be routed to a single destination. For both the N parallel paths network as well as for the general network, the vector of equilibrium routing probabilities satisfies a fixed point equation. We present various supporting simulation results.

The second routing algorithm that we consider is based on an optimization approach to the routing problem. We consider a problem where multiple traffic streams entering at various source nodes have to be routed to their destinations via a network of links. We cast the problem in a multicommodity network flow optimization framework. Our cost function, which is a function of the individual link delays, is a measure of congestion in the network. Our approach is to consider the dual optimization problem, and using dual decomposition techniques we provide primal-dual algorithms that converge to the optimal routing solution. A classical interpretation of the Lagrange multipliers (drawing an analogy with electrical networks) is as 'potential differences' across the links. The link potential difference can

be then thought of as 'driving the flow through the link'. Using the relationships between the link potential differences and the flows, we show that our algorithm converges to a loop-free routing solution. We then incorporate in our framework a rate control problem and address a joint rate control and routing problem.

MULTIPATH ROUTING ALGORITHMS FOR COMMUNICATION NETWORKS: ANT ROUTING AND OPTIMIZATION BASED APPROACHES

by

Punyaslok Purkayastha

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2009

Advisory Committee:
Professor John S. Baras, Chair/Advisor
Professor Armand M. Makowski
Professor Richard J. La
Professor André L. Tits
Professor S. Raghavan

Acknowledgments

It is a pleasure to record here my appreciation and gratitude for my advisor, Prof. John Baras, for his support and encouragement during the entire period of my graduate studies here at the University of Maryland. He has been very patient, helpful, and encouraging, throughout the various stages of evolution of the dissertation. He has brought to my attention a wide variety of interesting research ideas and problems, and provided me freedom, advice and guidance as I tried to find my way through the dissertation. During the course of solving various problems in the dissertation, I have often benefitted from his wide-ranging interests and expertise. He has always impressed upon me the importance of discovering unifying principles among disparate research areas and issues, and has always insisted upon trying to achieve elegance and clarity in thinking and research. Above all, his enthusiasm for research and his energy, drive and passion shall always remain for me a source of inspiration.

I am also grateful to my thesis committee members - Prof. A. M. Makowski, Prof. R. J. La, Prof. A. L. Tits and Prof. S. Raghavan - for agreeing to serve in my committee. Prof. Makowski and Prof. La have provided me with very useful feedback during my Thesis Proposal Examination. I must also thank Prof. Tits for providing very detailed feedback on the Optimal Routing portion of the dissertation which improved the writeup, and which also helped uncover a couple of embarassing errors.

I am thankful to the many professors in the University of Maryland under

whom I have taken courses - Prof. R. L. Johnson, Prof. M. Freidlin, Prof. P. Smith (Mathematics and Statistics), Prof. P. S. Krishnaprasad, Prof. J. S. Baras, Prof. S. I. Marcus, Prof. A. Papamarcou, Prof. S. Ulukus, Prof. E. Abed and Prof. R. J. La (Electrical and Computer Engineering). All of these courses were beautifully organized and presented, which made the task of learning quite enjoyable. These courses also provided a good foundation for my research work. I would like to specially thank Prof. Krishnaprasad for taking such a keen interest in my development as a student, for pointing out to me many interesting sources of information on various topics, and for always encouraging me in my research pursuits. I would also like to specially thank Prof. La for being so encouraging, appreciative and supportive.

It is my pleasure to record my gratitude to Prof. Vinod Sharma, my advisor at the Indian Institute of Science, Bangalore, where I did my masters work. Prof. Sharma mentored me through my first research experience (which was quite enjoyable), and has ever since been a source of steadfast support, encouragement and wise counsel.

A number of friends and colleagues in the department and in the university have made my stay here enjoyable; I would like to especially thank Pedram, Maben, George Papageorgiou, Kiran, Senni, Vahid, Vladimir Ivanov, Huigang Chen, Tao Jiang, Amit, Ion and Svetlana. I feel very fortunate to have had many close acquaintances here at Maryland whose company has been a source of much comfort and joy over the years - Vishwa, Vikas Raykar, Arun, Kaushik Mitra, Rajeshree Varangaonkar, Jay Kumar, Amit Trehan, Bhaskar Khubchandani and Vijay Nenmeni. They have always been very supportive, and they have always wished me well

(I apologise to the many others, whose names I might have inadvertently left out).

A special note of thanks is due to Kimberley Edwards for helping me out with many official matters on innumerable occasions, and for the grace, care and patience with which she has done so. Thanks are due to Althia Kirlew who also helped me with many official matters during the early years of my stay here.

My most profound sense of gratitude is to my family consisting of my mother, my late father, and my younger brother. My parents have always encouraged me to pursue my interests, and have been always supportive throughout. They have also been very self-sacrificing, and have withstood long periods of separation, as I found my way through the dissertation. My father, in particular, would have been very happy to see me reach this point in my life. My younger brother, who is also a graduate student here, has provided me good companionship over the past few years. This thesis is dedicated to my family, and to all the love I have received from them.

My research here has been generously supported by various grants from the U. S. Army Research Laboratory under the CTA C & N Consortium Coop. Agreement ($DAAD19-01-2-0011$), a MURI grant from the U. S. Army Research Office ($DAAD19-01-1-0465$, $DAAD19-01-1-0494$, $DAAD19-02-1-0319$), and a grant from the National Aeronautics and Space Administration (NASA) Marshall Space Flight Center ($NCC8-235$). The support is acknowledged with gratitude.

Table of Contents

List of Figures

Chapter 1

Introduction

The routing problem in communication networks is concerned with the task of guiding incoming traffic from the various source nodes of the network to the destination nodes. The design of algorithms that accomplish this task is quite challenging because such algorithms have to be distributed in nature (i.e., the nodes have to implement the algorithm by exchanging information and coordinating with each other), have to be able to adapt to changing input traffic conditions, and even be able to cope with changes in the network topology. Communication networks are required to handle a wide variety of input application traffic with their own peculiar Quality of Service (QoS) requirements. This requires the design of routing algorithms that are also able to act in conjunction with algorithms at the other layers of the protocol stack in order to cater to the requirements of the input traffic to the network.

The main functions of routing algorithms are to find routes between source-destination node pairs in a network, and to then guide traffic along such routes. For wireless networks (especially for Mobile Adhoc Networks (MANETs), where the network topology is usually time-varying) an additional task is to collect and maintain information about the network topology, which is required for the smooth

execution of the above-mentioned tasks. The way the above functionalities are implemented depend on the nature of the application that the network is supposed to cater to and the approach taken to transmit packets from the source to the destination. For example, for packet-switched networks, routing decisions are taken at each node of the network as to which outgoing link to direct an incoming packet to, in its journey to a destination. Every packet can thus, in principle, follow a different sequence of nodes to the destination. In circuit-switched (or virtual circuit-switched) networks, the path that the packets of a new incoming connection must follow is decided before the packets are transmitted (the 'circuit set-up' phase), so that all packets follow the same path. The design of the corresponding routing algorithms also follows a different set of criteria, based on the different service requirements of the corresponding applications.

In this dissertation, we concern ourselves with wireline packet-switched networks and with routing algorithms that cater to applications involving bulk-data transfer ('elastic' traffic) from various source nodes of the network to the corresponding destination nodes. Routing algorithms for such applications essentially involve the construction of routing tables at the nodes of the network. The routing table at a node is used to decide, for an incoming packet bound for a particular destination node, which outgoing link to direct the packet to. This kind of hop-by-hop routing enables a packet to eventually reach its intended destination. We consider two different types of routing algorithms for such applications. The first routing algorithm that we consider belongs to a class of routing algorithms called Ant-Based Routing Algorithms. These algorithms have been inspired by observa-

tions of the foraging behavior of ants in nature. The second routing algorithm that we consider is based on an optimization approach to the routing problem. The routing objective is to establish a packet traffic flow pattern so that packets are directed to their respective destinations while, at the same time, minimizing a cost related to the congestion in the network. Both the routing algorithms can be implemented in a distributed manner and both of them have the capability to adapt to changing incoming traffic conditions [1]. Both the algorithms require that information related to congestion in the network is available at the network nodes, which utilize this information to adaptively adjust the routing tables. The congestion information that is used in both the algorithms are measurements of delays in the links and the paths of the network.

In the following two sections, Section 1.1 and Section 1.2, we provide a brief overview of the literature related to Ant-Based Routing and optimal routing algorithms, respectively. The next section Section 1.3, summarizes the contributions of the thesis, and the final section of the chapter, Section 1.4, discusses the organization of the thesis.

1.1 Literature survey on Ant-Based Routing Algorithms

"Ant algorithms" constitute a class of algorithms that have been proposed to solve a variety of problems arising in optimization and distributed control. They form a subset of the larger class of what are referred to as "Swarm Intelligence"

[1] We would typically require that the incoming traffic conditions change slowly enough so that the routing algorithms can converge.

algorithms, a topic which has received widespread attention recently, and to which entire books have been devoted (see, for example, Bonabeau, Dorigo, and Theraulaz [13]). The central idea here is that a "swarm" of relatively simple agents can interact through simple mechanisms and collectively solve complex problems. There are numerous examples in nature that illustrate this idea. Bonabeau, Dorigo, and Theraulaz [13] give examples of insect societies like those of ants, honey bees, and wasps, which accomplish fairly complex tasks of building intricate nests, finding food, responding to external threats etc., even though the individual insects themselves have limited capabilities. The abilities of ant colonies to collectively accomplish complex tasks have served as sources of inspiration for the design of "Ant algorithms".

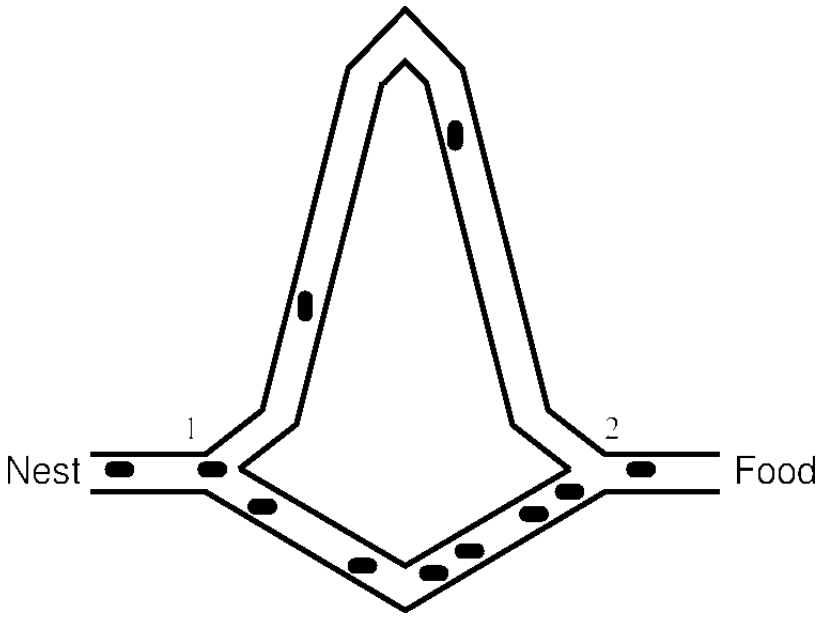


Figure 1.1: The Binary Bridge Experiment: Ants discover shortest paths.

Examples of "Ant algorithms" are the set of Ant-Based Routing algorithms

that have been proposed for communication networks. It has been observed in an experiment conducted by biologists Deneubourg, Aron, Goss, and Pasteels, called the double bridge experiment [19], that a colony of ants, when presented with two paths to a source of food, is able to collectively converge to the shorter path (see Figure 1.1). Every ant lays a trail of a chemical substance called *pheromone* as it walks along a path; subsequent ants follow and reinforce this trail. Notice that (assuming all ants move at the same speed and deposit pheromone at the same constant rate) the ants will move back and forth through the shorter path more quickly, which would result in a more rapid accumulation of pheromone in the shorter path. Because subsequent ants follow the stronger pheromone trails, this 'positive reinforcement' effect eventually leads to all ants following, and thus discovering, the shorter path. The double bridge experiment also considers the case where there are two branches of equal length. It was observed that, if due to random fluctuations a larger number of ants initially choose a branch, due to reinforcement effects all ants converge to that branch eventually. The following model was considered to explain the observation: Let a number A_n out of n ants choose a branch, say A. Then the next, i.e., $(n+1)$st ant, chooses path A with probability proportional to the ν-th power of $K + A_n$, where $K \geq 0$ is a constant. For certain values of $K \geq 0$ and $\nu \geq 0$, the model was found to agree well with simulations. The simple intuition involved in the double bridge experiment has been nicely captured in the mathematical formulations of Makowski [38] and Das and Borkar [18]. Makowski considers the model proposed for the double bridge experiment (with equal length branches). The paper shows that the asymptotic behavior of the algorithm criti-

cally depends on the parameter ν. Using martingale and stochastic approximation techniques, it is rigorously shown that, in fact, only for $\nu > 1$, is it true that all ants eventually choose one branch. For $0 < \nu < 1$, ants choose with each branch with equal probability, whereas for $\nu = 1$, the asymptotic probability is a $[0, 1]$-valued random variable, whose distribution depends upon the initial value A_1. Das and Borkar in their paper (dubbed as the Multi-Agent Foraging Ant Colony Optimization (MAFACO) scheme) explore a scheme whereby a fixed number of agents shuttle back and forth between a source and a destination node, sampling paths between them, and 'positively reinforcing' the shorter paths. There are three algorithms involved - a pheromone update scheme, which updates the pheromone content on a path based on the number of ants traversing the path; a scheme which updates an estimate of the utility of a path based on the pheromone content of the path; and a scheme which updates the probabilities of an agent choosing the paths, based on the utility estimates of the paths. The paper views the problem as being an example of a stochastic approximation algorithm and uses ODE approximation methods to study the convergence of the algorithm. It is shown that if the initial probabilities are so chosen that the probability on the shortest path is larger than the probabilities on the alternative paths then the algorithm converges to the shortest path, i.e., asymptotically, all agents choose the shortest path with probability one ('equilibrium selection through initial bias'). In Das and Borkar, though the delays are allowed to be stochastic, they are not functions of the routing probabilities themselves, which is typically the case that would arise in a communication network.

Most of the Ant-Based Routing Algorithms proposed in the literature are

inspired by the basic idea of the double bridge experiment, viz., the creation and reinforcement of a pheromone trail on a path that serves as a measure of the quality of the path. These algorithms employ probe packets called ant packets (analogues of ants) to explore the network and measure various quantities that are indicators of network routing performance, like link and path delays. These measurements are used to create analogues of pheromone trails. In turn, these trails are used to update the routing tables at the network nodes. The update algorithms tend to reinforce those outgoing links which lie on paths with lower delays.

Schoonderwoerd *et. al.* [43] propose an Ant Routing Algorithm for circuit-switched networks. Ant packets are launched from the various nodes of the network towards the destination nodes. While traversing through the network, an ant packet picks up information related to the spare capacity available on each link (spare capacity here refers to the number of circuits (connections) that are not being used), and assigns a weight to the link that is an exponentially decreasing function of the spare capacity. These weights are then used to form pheromone estimates that influence the routing tables. They implemented the algorithm on a 30-node network of British Telecom and report much better performance - the percentage of dropped calls is lesser - compared to other algorithms (including fixed shortest path schemes; for details please refer to the paper). This generated interest in Ant Routing Algorithms for all kinds of networks - circuit- and packet-switched wireline networks, and even packet-switched wireless networks. Ant Routing Algorithms for wireless networks have been proposed in Baras and Mehta [2], Di Caro, Ducatelle, and Gambardella [21], and Gunes *et. al.* [30]. For wireline packet-switched networks Ant

Routing Algorithms have been proposed in Di Caro and Dorigo [22], Gabber and Smith [27], and Bean and Costa [4] (among many others). A discussion of these algorithms is provided in Chapter 2 of the thesis.

Though a large number of Ant Routing Algorithms have been proposed, very few analytical studies are available in the literature. Analytical studies of Ant Colony Optimization algorithms are the above-mentioned articles by Das and Borkar [18], as also the article by Gutjahr [31] and the analysis available in Dorigo and Stutzle [23] (Chapter 4). Analytical investigation of the properties and convergence of Ant Routing Algorithms have also been pursued in Subramanian, Druschel, Chen [45] and Yoo, La, and Makowski [50]. Yoo, La, and Makowski [50] consider a network consisting of two nodes connected by L parallel bidirectional links. The ant packets move back and forth between the nodes sampling the delays in the links, and are routed probabilistically at the nodes. In turn, the delays that the ants sample are used to reinforce the probabilistic routing tables at the nodes using a slight variant of what are known as Linear Reinforcement Schemes (Kaelbling, Littman, and Moore [32] and Thathachar and Sastry [46]). The paper considers both uniform routing of ants as well as routing based on the routing tables at the nodes (called regular routing). An elegant analysis then reveals that the routing tables converge in distribution (regardless of the initial condition) for the uniform ants case, and almost surely to the shortest path solution for the regular ants case. However, the results are for the case when the link delays are constants (not time-varying). The scheme that Yoo, La, and Makowski analyse is essentially (with a slight generalization) that proposed for a general network by Subramanian, Druschel, and Chen. Though

the paper by Subramanian, Druschel, and Chen announces convergence results for the two-node, L parallel links network, there are no formal proofs provided, and in fact some of the results as they stand, are incorrect (as pointed out by [50]). A paper which seeks to use reinforcement learning approaches for routing in packet-switched wireline networks is Boyan and Littman [15]. Inspired by the reinforcement learning technique of Q-learning, the scheme proposed tries to make estimates at a node, of delays along routes going through the outgoing links of the node. However no convergence results are available for the scheme.

The above set of analytical studies have mostly concentrated on networks where the delays in the links are deterministic (or stochastic, as in Das and Borkar [18]), but independent of the offered traffic loads. Most of the results in the papers show convergence of the algorithms to shortest path solutions.

1.2 Literature survey on Optimal Routing Algorithms

An early important work on optimal routing for packet-switched communication networks was Gallager [28]. The cost considered was the sum of all the average link delays in the network and is a measure of the network-wide congestion. A distributed algorithm was proposed to solve the problem. The algorithm preserved loop-freedom after each iteration, and converged to an optimal routing solution. Bertsekas, Gafni, and Gallager [10] proposed a means to improve the speed of convergence by using additional information of the second derivatives of link delays. Both the algorithms mentioned above require that related routing information like

marginal delays be passed throughout the network before embarking on the next iteration. Another cost function that has been considered in the literature is a sum of the integrals of queueing delays; see Kelly [34] and Borkar and Kumar [14]. The solution to this problem can be characterized by the so-called Wardrop equilibrium [49] - between a source-destination pair, the delays along routes being actively used are all equal, and smaller than the delays of the inactive routes. Another formulation of the optimal routing problem, called the path flow formulation (see Bertsekas [7] and Bertsekas and Gallager [9]), has been in vogue. Tsitsiklis and Bertsekas [47] considered this formulation, and used a gradient projection algorithm to solve the optimal routing problem in a virtual circuit network setting. They also proved convergence of a distributed, asynchronous implementation of the algorithm. This formulation was considered by Elwalid, Jin, Low, and Widjaja [24] to accomplish routing in IP datagram networks using the Multi-Protocol Label Switching (MPLS) architecture. The routing functionality is shifted to the edges of the network (a feature of the path flow formulation; this is similar to 'source routing'), and requires measurements of marginal delays on all paths linking a source and a destination. Because the number of such paths could scale exponentially as the network size grows, it is not clear that the solution would scale computationally. Multipath routing and flow control based on utility maximization have also been considered, notably in Kar, Sarkar, and Tassiulas [33] and in Wang, Palaniswami, and Low [48]. Kar, Sarkar and Tassiulas propose an algorithm which uses congestion indicators to effect control of flow rates on individual links, but do not explicitly consider congestion indicators based on queueing delays as we do (their congestion indicators are based

on flows on links). Their algorithm avoids the scalability issue mentioned above. On the other hand, Wang, Palaniswami, and Low consider a path flow formulation of the multipath routing and flow control problem, which as we have mentioned above has scalability problems. However, they do consider the effect of queueing delays, extracting related information by measuring the round trip times. Recently, dual decomposition techniques have been used by Eryilmaz and Srikant [25], Lin and Shroff [37], Neely, Modiano, and Rohrs [40], and Chen, Low, Chiang, and Doyle [16], to design distributed primal-dual algorithms for optimal routing and scheduling in wireless networks. Such techniques consider the dual to the primal optimization problem and exploit separable structures in the costs and/or constraints to come up with a decomposition which automatically points the way towards distributed implementations. The seminal work of Kelly, Maulloo, and Tan [35] showed how congestion control can be viewed in this way. The approach (called Network Utility Maximization (NUM)) has gained currency and has been applied to a variety of problems in wireline and wireless communication networks, cutting across all layers of the protocol stack (see [17] for an overview).

1.3 Contributions of the Thesis

Convergence Results for Ant-Based Routing Algorithms. In contrast to the analytical studies available in the literature, we consider convergence results for an Ant Routing Algorithm when the delays in the links of the network can be time-varying (and stochastic), which is interesting as well as important. The delays

are dependent on the offered traffic loads and the routing probabilities. The Ant Routing Algorithm that we consider is the one proposed by Bean and Costa [4]. The scheme of Bean and Costa retains most of the useful and interesting features of Ant Routing Algorithms. It can be implemented in a distributed manner. The routing tables are updated based on the information regarding the link and path delays collected by the ant packets, which enables the algorithm to be adaptive. There is thus a delay estimation algorithm and a routing table update algorithm, which makes use of the delay estimates. Furthermore, the scheme provides a multipath routing solution, which has the benefits of providing better utilization of network resources and good throughput performance for incoming connections.

We first consider a simple routing scenario where data traffic entering a single source node has to be routed towards a single destination node, and there are N available parallel paths between them. We model the arrival processes and packet lengths of both the ant and the data streams that arrive at the source node, and argue, using methods from the theory of adaptive algorithms and stochastic approximation, that the evolution of the link delay estimates can be closely tracked by a deterministic ODE system, when the step size of the estimation scheme is small. Then a study of the equilibrium points of the ODE gives us the equilibrium behavior of the routing algorithm; in particular, the equilibrium routing probabilities and the mean delays in the N links under equilibrium can be obtained. We also show that the fixed point equations that the equilibrium routing probabilities satisfy are actually the necessary and sufficient conditions of a convex optimization problem. This enables us to show that if there is a solution to the fixed point equations then

such a solution is unique. We then explore further properties of the equilibrium routing solution. We show that by tuning a parameter (β) in our algorithm, we can influence the equilibrium routing behavior. We provide and discuss results obtained by performing discrete event simulation of the system.

We then turn our attention to the more general case when multiple traffic streams enter a network at various source nodes and which have to be routed to a single destination node (the "single commodity" case). We formulate the problem in a similar manner as the N parallel links network, taking into account the asynchronous nature of operation of the algorithm. We then derive the appropriate deterministic ODE system that tracks the evolution of the vector of link delay estimates. We also study the equilibrium routing behavior of the algorithm.

The approach that we use is most closely related to the work of Borkar and Kumar [14], which studies an adaptive algorithm that converges to a Wardrop equilibrium. Our framework is similar to theirs - they have a delay estimation algorithm and a routing probability update algorithm which utilizes the delay estimates. Their routing probability update algorithm is designed so that the routing probabilities converge to a Wardrop equilibrium. Their probability update scheme moves on a slower time scale than their delay estimation scheme. Using two time scale stochastic approximation methods they prove convergence of their scheme for a general network, the algorithms being completely distributed and asynchronous.

An Optimal Routing Algorithm using Dual Decomposition Techniques. We consider an optimal routing problem and cast it in a multicommodity network flow optimization framework. Our cost function is related to the conges-

tion in the network, and is a function of the flows on the links of the network. The optimization is over the set of flows in the links corresponding to the various destinations of the incoming traffic. We separately address the single commodity and the multicommodity versions of the routing problem. Our approach is to consider the dual optimization problems, and using dual decomposition techniques we provide primal-dual algorithms that converge to the optimal solutions of the problems. Our algorithms, which are subgradient algorithms to solve the corresponding dual problems, can be implemented in a distributed manner by the nodes of the network. For online, adaptive implementations of our algorithms, the nodes in the network need to utilize 'locally available information' like estimates of queue lengths on the outgoing links. We show convergence to the optimal routing solutions for synchronous versions of the algorithms, with perfect (noiseless) estimates of the queueing delays (these essentially are the convergence results of the corresponding subgradient algorithms). Our optimal routing solution is not an end-to-end solution (not a path-based formulation) like many of the above-cited works [47], [24], [48]. Consequently, our algorithms would avoid the scalability issues related to such an approach. Every node of the network controls the total as well as the commodity flows on the outgoing links using the distributed algorithms. Our optimal solutions also have the attractive property of being multipath routing solutions. Furthermore, by using a parameter (β) we can tune the optimal flow pattern in the network, so that more flow can be directed towards the links with high capacities by increasing the parameter (we observe this in our numerical computations).

The Lagrange multipliers (dual variables) can be interpreted as 'potentials'

on the nodes for the single commodity case, and as 'potential differences' across the links for the multicommodity case. We can then associate with every link of the network a 'characteristic curve' (see Bertsekas [7] and Rockafellar [42]), which describes the relationship between the potential difference across the link and the link flow, the potential difference being thought of as 'driving the flow through the link'. Using the relationships between the potential differences and the flows, we then provide simple proofs showing that our algorithm converges to a loop-free optimal solution, which is a desirable property. A related piece of work which has gained some attention recently is the paper by Basu, Lin, and Ramanathan [3] which constructs a potential function on the nodes of the network. This potential function is a convex combination of the distance of the node from a destination node (the distance is computed based on some weights on the edges of the graph) and a metric based on the queue lengths of the outgoing links. The latter feature, according to the authors, enables the algorithm to be 'traffic-aware'. The routing algorithm at each node now computes the route to the destination as the direction (i.e., the next hop) in which the potential field decreases fastest (direction of the 'force field'). In our case, we show that the Lagrange multipliers can be naturally interpreted as 'potentials' (or 'potential differences'), and in fact their values decrease as one moves along any path in the network from a source node towards a destination node, being a minimum at the destination. Our techniques are related to those employed in the literature on NUM methods (though the details involved and the interpretations are different). We also show how we can incorporate in our framework the rate control problem, and consequently address a joint rate control and optimal routing problem.

1.4 Organization of the Thesis

The remainder of the dissertation is organised as follows. Chapter 2 is wholly devoted to the formulation and study of an Ant-Based Routing Algorithm. In Chapter 3 we study the optimal routing problem. Chapter 4 provides a summary and a few concluding remarks, and also outlines some directions for future research.

Chapter 2

Convergence Results for Ant Routing Algorithms via Stochastic Approximation and Optimization

In this chapter we study the convergence and equilibrium behavior of ant routing algorithms. The chapter is organised as follows. In Section 2.1 we describe the general framework and the mechanism of operation of Ant-Based Routing Algorithms. We also describe in brief the various routing schemes that have been proposed in the literature. In Section 2.2 we describe a scheme proposed by Bean and Costa [4] which is adaptive and which admits of a distributed implementation. For these reasons this scheme can be a suitable candidate for deployment in communication networks, and in the next two sections we devote ourselves to an analysis of the algorithm. In Section 2.3 we provide convergence results for the algorithm and discuss its equilibrium behavior for the simple case when data has to be transported from a source node to a destination node through N parallel paths. Section 2.4 considers the routing problem when there are multiple sources of traffic that attempt to transfer data through a network to a single destination node (the "single commodity" case).

2.1 Ant-Based Routing: General Framework and Routing Schemes

We provide in this section a brief formal description of the general framework of ant routing for a wireline communication network. Such a network can be represented by a directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{L})$, where $\mathcal{N}$ denotes the set of nodes of the network, and $\mathcal{L}$ the set of directed links. The framework that we follow for a formal description, is the one described in Di Caro and Dorigo [22], [23], which is general enough and adequate for our purposes.

Every node i in the network maintains two key data structures - a matrix of routing probabilities, the routing table $\mathcal{R}(i)$, and a matrix of various kinds of statistics used by the routing algorithm, called the network information table $\mathcal{I}(i)$. For a particular node i, let $N(i,k)$ denote the set of neighbors of i through which node i routes packets towards destination k. For the communication network consisting of $|\mathcal{N}|$ nodes, the matrix of routing probabilities $\mathcal{R}(i)$, has $|\mathcal{N}| - 1$ columns, corresponding to the $|\mathcal{N}| - 1$ destinations towards which node i could route packets, and $|\mathcal{N}| - 1$ rows, corresponding to the maximum number of neighbor nodes through which node i could route packets to a particular destination. The entries of $\mathcal{R}(i)$ are the probabilities ϕ_j^{ik}. ϕ_j^{ik} denotes the probability of routing an incoming packet at node i and bound for destination k via the neighbor $j \in N(i,k)$. The matrix $\mathcal{I}(i)$ has the same dimensions as $\mathcal{R}(i)$, and its (j,k)-th entry contains various statistics pertaining to the route from i to k that goes via j (denoted henceforth by $i \to j \to \cdots \to k$). Examples of such statistics could be mean delay and delay variance estimates of the route $i \to j \to \cdots \to k$. These statistics are maintained

and updated based on the information the ant packets collect about the route. The matrix $\mathcal{I}(i)$ represents the characteristics of the network that are learned by the nodes through the ant packets. Based on the information collected in $\mathcal{I}(i)$, "local decision-making" - the update of the routing table $\mathcal{R}(i)$ - is done. The iterative algorithms that are used to update $\mathcal{I}(i)$ and $\mathcal{R}(i)$ will be referred to as the *learning algorithms*.

We now describe the mechanism of operation of Ant-Based Routing Algorithms. For ease of exposition, we restrict attention to a particular fixed destination node, and consider the problem of routing from every other node to this node, which we label as D (see Figure 2.1). The network information table $\mathcal{I}(i)$ contains only statistics related to estimates of the mean delay.

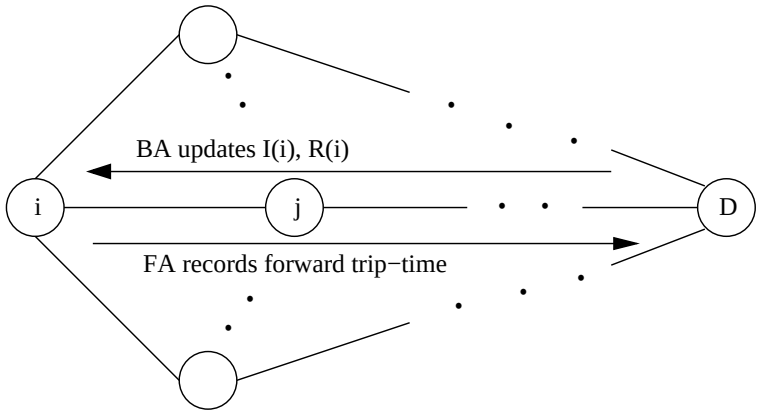


Figure 2.1: Forward Ant and Backward Ant packets

Forward ant generation and routing. At certain intervals, forward ant (FA) packets are launched from a node i towards the destination node D to discover low delay paths to it. The FA packets sample walks on the graph representing the communication network, based on the current routing probabilities at the nodes.

FA packets share the same queues as data packets and so experience similar delay characteristics as data packets. Every FA packet maintains a stack of data structures containing the IDs of nodes in its path and the per hop delays encountered. The per hop delay measurements are obtained through time stamping of the packets as they pass through the various nodes. Depending on the nature of the application, which determines the statistics being reinforced at the nodes, other relevant information may be collected by the FA packets. For example, for secure routing applications, FA packets could obtain measurements reflecting the 'security level' of the links, and update an appropriate statistic on the nodes.

Backward ant generation and routing. Upon arrival of an FA at the destination node D, a backward ant (BA) packet is generated. The FA packet transfers its stack to the BA. The BA packet then retraces back to the source the path traversed by the FA packet. BA packets travel back in high priority queues, so as to quickly get back to the source and minimize the effects of outdated or stale measurements. At each node that the BA packet traverses through, it transfers the information that was gathered by the corresponding FA packet. This information is used to update the matrices $\mathcal{I}$ and $\mathcal{R}$ at the respective nodes. Thus the arrival of the BA packet at the nodes triggers the iterative learning algorithms.

Various learning algorithms have been proposed in the literature. In the following subsections, we briefly describe some of the algorithms.

2.1.1 Algorithm A

Di Caro and Dorigo [22], [23], suggest the following scheme. Suppose that an FA packet measures the delay Δ_j^{iD} associated with a walk $i \to j \to \cdots \to D$. When the corresponding BA packet arrives back at node i, this delay information is used to update estimates of the mean delay X_j^{iD} and the delay variance Y_j^{iD} using the algorithms (these are simply the exponential estimators)

$$X_j^{iD} := X_j^{iD} + \epsilon \left(\Delta_j^{iD} - X_j^{iD}\right), \tag{1}$$

$$Y_j^{iD} := Y_j^{iD} + \epsilon \left(\left(\Delta_j^{iD} - X_j^{iD}\right)^2 - Y_j^{iD}\right), \tag{2}$$

where $\epsilon \in (0, 1)$ is a small constant. Similar updates of the mean delay and the delay variance estimates take place on all the nodes along the route that the BA retraces back to the source (and which the corresponding FA packet had earlier traversed).

Simultaneously, the routing probability ϕ_j^{iD} (also called *pheromone* by Di Caro and Dorigo) is updated using the algorithm

$$\phi_j^{iD} := \phi_j^{iD} + r \left(1 - \phi_j^{iD}\right), \tag{3}$$

and for the other neighbor nodes $k \in N(i, D)$ the probabilities are proportionally decreased so that they sum to unity,

$$\phi_k^{iD} := \phi_k^{iD} - r\ \phi_k^{iD}. \tag{4}$$

r is the reinforcement parameter which depends on a window of observed values of the delays Δ_j^{iD}. In fact r can also be kept constant, but the authors argue that there can be benefits when it is allowed to depend on the delay values.

The form that the dependence should take has also been described by Di Caro and Dorigo. They found this form to give empirically the best system routing performance [1]. It is given by the following formula:

$$r = c_1 . \frac{W_{best}}{\Delta_j^{iD}} + c_2 . \frac{I_{\sup} - I_{\inf}}{\left(I_{\sup} - I_{\inf}\right) + \left(\Delta_j^{iD} - I_{\inf}\right)}, \tag{5}$$

where W_{best} is the best trip time recorded over a window of past observations, and $I_{\sup}$ and $I_{\inf}$ denote the upper and lower limits of an approximate confidence interval constructed for the estimated mean delay X_j^{iD} [2].

Baras and Mehta [2] suggest a very similar scheme where the mean delay, the delay variance, and the routing probability estimates are calculated as above but they suggest a different rule for the reinforcement parameter:

$$r = \frac{k}{f(X_j^{iD})}, \tag{6}$$

where k is a positive constant, and f is an increasing function of the delay estimate.

2.1.2 Algorithm *B*

As in Algorithm *A*, a BA packet arrives back at node i with a measurement of the delay Δ_j^{iD}. However, instead of updating estimates of the average delay to the destination, Di Caro, Ducatelle, and Gambardella [21] suggest that the average of the inverse delay be estimated, and they call this the pheromone content τ_j^{iD} associated

[1] The authors don't define very precisely though what exactly is the system routing performance metric.

[2] In fact, the actual reinforcement r that is used is a *squashed* function of the above quantity. Refer to [22], [23] for the details.

with the link (i, j). Consequently, the update equation for the pheromone content is given by

$$\tau_j^{iD} := \tau_j^{iD} + \epsilon \left(\frac{1}{\Delta_j^{iD}} - \tau_j^{iD} \right), \tag{7}$$

where, as usual $\epsilon \in (0, 1)$.

The routing probabilities are then updated by

$$\phi_j^{iD} = \frac{(\tau_j^{iD})^\beta}{\sum_{k \in N(i,D)} (\tau_k^{iD})^\beta}, \quad j \in N(i, D), \tag{8}$$

where β is a positive constant.

A variety of schemes, other than the above, have been proposed in the literature on Ant-Based Routing Algorithms. Most of these are heuristics that have been found experimentally to give good results in a few cases. In fact even the schemes Algorithm A and Algorithm B described above have been studied only through simulations. The scheme for the update of the routing probabilities (pheromones) in Algorithm A is actually the Linear Reinforcement Scheme considered in studies of learning stochastic automata; see Kaelbling, Littman, and Moore [32], and Thathachar and Sastry [46]. A Linear Reinforcement Scheme is also used by the Trail Blazer ant routing algorithm of Gabber and Smith [27] to update their routing probabilities. Yoo, La, and Makowski [50] consider a version of Algorithm A where the link delays are constant. They thus, have only a routing probability update scheme. They show that the probability update scheme converges almost surely to the shortest path solution for the case of a network consisting of N parallel links between a pair of source-destination nodes.

We would like to consider the more general and interesting case where the

link delays are stochastically varying with time and there is a learning algorithm which "learns" the delays, and this information is fed back to the routing update algorithm. This enables the routing algorithm to be adaptive, i.e., reponsive to changing topology and traffic conditions. Di Caro and Dorigo do provide such a framework in Algorithm A above, though it seems that the routing algorithm they consider (the Linear Reinforcement Scheme) is designed to (and can) work only when the delays are constant (not stochastically varying). We do not consider Algorithm B because we are not able to find a good rationale behind it.

2.2 The Routing Scheme of Bean and Costa

We now consider the routing scheme proposed by Bean and Costa [4]. Bean and Costa suggest the following scheme for the learning algorithms. As in Algorithms A and B, suppose a BA packet (corresponding to some FA packet) arrives at node i with the delay information Δ_j^{iD}. This information is used to update the estimate of the mean delay X_j^{iD} using the simple exponential estimator

$$X_j^{iD} := X_j^{iD} + \epsilon \, (\Delta_j^{iD} - X_j^{iD}), \tag{9}$$

where $0 < \epsilon < 1$ is a small constant. The mean delay estimates X_m^{iD}, corresponding to the other neighbors m of node i, are left unchanged

$$X_m^{iD} := X_m^{iD}. \tag{10}$$

Simultaneously, the routing probabilities at the nodes are updated using the

scheme:

$$\phi_j^{iD} = \frac{\left(\frac{1}{X_j^{iD}}\right)^{\beta}}{\sum_{k \in N(i,D)} \left(\frac{1}{X_k^{iD}}\right)^{\beta}}, \quad j \in N(i,D), \tag{11}$$

where β is a constant positive integer. β influences the extent to which outgoing links with lower delay estimates are favored compared to the ones with higher delay estimates.

We can interpret the quantity $\frac{1}{X_j^{iD}}$ as analogous to a pheromone deposit on the outgoing link (i, j). This deposit gets dynamically updated by the ant packets. The pheromone content influences the routing tables through the relation (11). Equation (11) shows that the outgoing link (i, j) is more desirable when X_j^{iD}, the delay in routing through j, is smaller (i.e., when the pheromone content is higher) relative to the other routes.

This scheme has been studied in some detail by Bean and Costa [4] using a combination of simulation and analysis. The scheme is a multipath routing scheme and the link delays can be stochastically varying. The scheme tries to form estimates of the means of the delays and these are used to update the routing probabilities. The authors employ 'a time-scale separation approximation' whereby the delay estimates are computed 'before' the routing probabilities are updated.An analytical model that just consists of the equations (9) and (11) is considered (as also a variant of it, see [4]), and it is found that results from numerical iterations of the model and those from simulations agree well. However, the exact nature of the 'time-scale separation' is not clear nor is any formal proof of convergence provided. Bean and Costa, however, do recognize the need for a formal proof of convergence of ant rout-

ing algorithms in general (see the section Conclusions in their paper [4]) and suggest that the theory of stochastic approximation algorithms can be used to demonstrate convergence.

In the following two sections, Section 2.3 and Section 2.4, we study the convergence and the equilibrium routing behavior of the routing scheme of Bean and Costa analytically. We also study other aspects of the routing scheme - for instance, the effect of the parameter β that appears in equation (11), and the relation of the equilibrium routing behavior to the capacities of the links.

2.3 The N Parallel Paths Case

The first model that we consider pertains to the simple routing scenario where arriving traffic at a single source node S has to be routed to a single destination node D. There are N available parallel (disjoint) paths between the source and the destination node through which the traffic could be routed. The network and its equivalent queueing theoretic model are shown in Figures 2.2 and 2.3, respectively. The queues represent the output buffers (which we assume to be infinite) at the source and are associated with the N outgoing links. We assume in our model that the queueing delays dominate the propagation and the packet processing delays in the N branches. These additional (usually deterministic) delay components can be incorporated into our model with no additional complexity, but to keep the discussion simple, we assume they are negligible. Two traffic streams, an ant and a data stream, arrive at the source node S. At node S, every packet of the com-

bined stream is routed with probabilities $\phi_1, \ldots, \phi_N$ (the *current* values) towards the queues $Q_1, \ldots, Q_N$, respectively. These probabilities are updated dynamically based on running estimates of the means of the delays (waiting times) in the N queues. Samples of the delays in the N queues are collected by the ant packets (these are forward ant packets) as they traverse through the queues. These samples are then used to construct the running estimates of the means of the delays in the N queues. We now describe our model in detail.

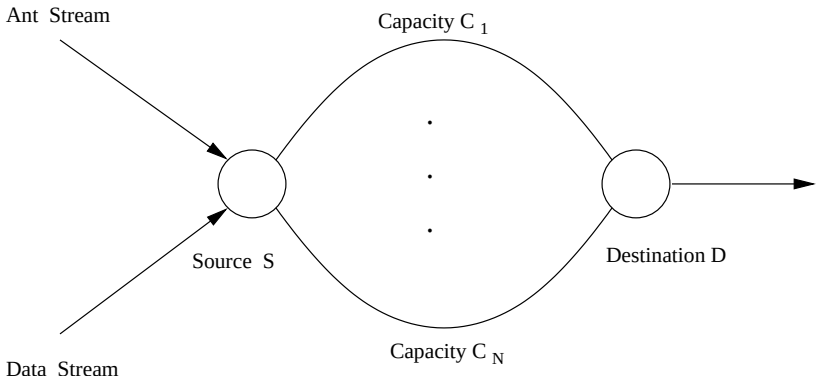


Figure 2.2: The network with N parallel paths

We model the arrival processes of ant and data stream packets at the source node S as independent Poisson processes of rates λ_A and λ_D packets/sec, respectively. The lengths of the packets of the combined stream constitute an i.i.d. sequence, which is also statistically independent of the packet arrival processes. The capacity of link i is C_i bits/sec ($i = 1, \ldots, N$). We assume that the length of an ant packet is generally distributed with mean L_A bits, and that the length of a data packet is generally distributed with mean L_D bits. If we denote the service times

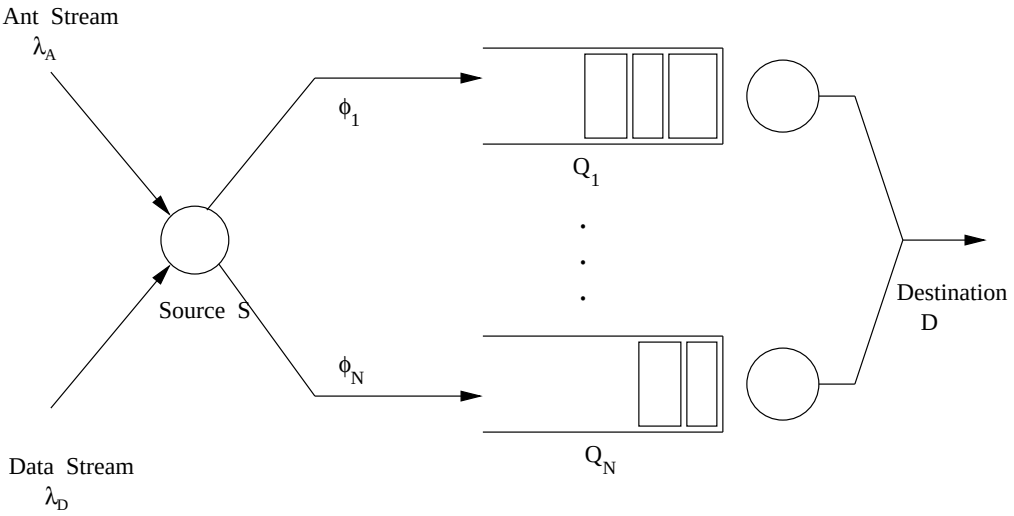


Figure 2.3: N parallel paths : The queueing theoretic model

of an ant and a data packet in queue Q_i by the generic random variables S_i^A and S_i^D, then S_i^A and S_i^D are generally distributed (according to some c.d.f.'s, say G_i^A and G_i^D) with means $E[S_i^A] = \frac{L_A}{C_i}$ and $E[S_i^D] = \frac{L_D}{C_i}$, respectively. The ant stream essentially acts as a probing stream in our system collecting samples of delays while traversing through the queues along with the data packets. Thus the packets of this stream would in general be much smaller in size compared to the data packets.

Let $\{\Delta_i(m)\}$ denote the sequence of delays experienced by successive ant packets traversing Q_i. Here delay refers to the total waiting time in the system Q_i (waiting time in the queue plus packet service time). Let $\{t(m)\}$ denote the sequence of arrival times of packets at the source S (the packets could be either ant or data packets). Also, let $\{\delta(n)\}$ denote the sequence of successive arrival times of ant packets at the destination node D. Then the n-th arrival of an ant packet at D occurs at $\delta(n)$. Suppose that this ant packet has arrived via Q_i. We denote the

decision variable for routing by $R(n)$; that is, for $i = 1, \ldots, N$, we say that the event $\{R(n) = i\}$ has occurred if the n-th ant packet that arrives at D has been routed via Q_i. $\psi_i(n) = \sum_{k=1}^{n} I_{\{R(k)=i\}}$, thus, gives the number of ant packets that have been routed via Q_i among a total of n ant arrivals at destination D (I_A is the indicator random variable of event A). Once the ant packet arrives, the estimate X_i of the mean of the delay through queue Q_i is immediately updated using a simple exponential averaging estimator

$$X_i(n) = X_i(n-1) + \epsilon \, (\Delta_i(\psi_i(n)) - X_i(n-1)), \tag{12}$$

$0 < \epsilon < 1$ being a small constant.

The delay estimates for the other queues are left unchanged, i.e.,

$$X_j(n) = X_j(n-1), \quad j \in \{1, \ldots, N\}, \; j \neq i. \tag{13}$$

In general thus, the evolution of the delay estimates in the N queues can be described by the following set of stochastic iterative equations

$$X_i^{\epsilon}(n) = X_i^{\epsilon}(n-1) + \epsilon \, I_{\{R^{\epsilon}(n)=i\}} \left(\Delta_i^{\epsilon}(\psi_i^{\epsilon}(n)) - X_i^{\epsilon}(n-1) \right), \quad i = 1, \ldots, N, \quad n \geq 1, \tag{14}$$

along with a set of initial conditions $X_1^{\epsilon}(0) = x_1, \ldots, X_N^{\epsilon}(0) = x_N$. The ϵ's in the superscript, in the equation (14) above, recognize the dependence of the evolution of the quantities involved (for example, the delay estimates X_i) on ϵ. (This notation is used in the section 2.4.2, where a convergence result for the evolution of the delay estimates to an approximating ODE, as $\epsilon \downarrow 0$, is provided.)

At time $\delta(n)$, besides the delay estimates, the routing probabilities $\phi_i(n), i =$

$1, \ldots, N$, are also updated simultaneously according to the equations

$$\phi_i^\epsilon(n) = \frac{(X_i^\epsilon(n))^{-\beta}}{\sum_{j=1}^N (X_j^\epsilon(n))^{-\beta}}, \quad i = 1, \ldots, N, \tag{15}$$

β being a constant positive integer. The initial values of the probabilities are $\phi_i^\epsilon(0) = \frac{(x_i)^{-\beta}}{\sum_{j=1}^N (x_j)^{-\beta}}, i = 1, \ldots, N$. We note that $\phi_1^\epsilon(n) + \cdots + \phi_N^\epsilon(n) = 1$, for all n. Again, the ϵ's in the superscript, in the equation (15) above, recognize the dependence of the evolution of the quantities involved on ϵ.

The vector of delay estimates $X = (X_1, \ldots, X_N)$ and the vector of routing probabilities $\phi = (\phi_1, \ldots, \phi_N)$ thus get updated at the times $\delta(n), n = 1, 2, 3, \ldots$. Let us also consider the continuous time processes, $\{x(t), t \geq 0\}$ and $\{f(t), t \geq 0\}$, defined by the equations

$$x(t) = X(n), \quad \text{for } \delta(n) \leq t < \delta(n+1), \tag{16}$$

$$f(t) = \phi(n), \quad \text{for } \delta(n) \leq t < \delta(n+1). \tag{17}$$

In the above model we consider only forward ants and do not incorporate the effects of backward ants. We assume that the estimates X_i of the means of the delays and the probabilities ϕ_i are updated as soon as the (forward) ant packets arrive at the destination D, and this information is available instantaneously thereafter at the source node S. We thus assume, in effect, that there is negligible delay as the backward ant packets travel back carrying the delay information to the source. Because backward ants are expected to travel back to the source through priority queues, the delay may not be very significant, except for large-sized networks. On the other hand, incorporating the effect of delays in our model introduces additional asynchrony, making the problem harder.

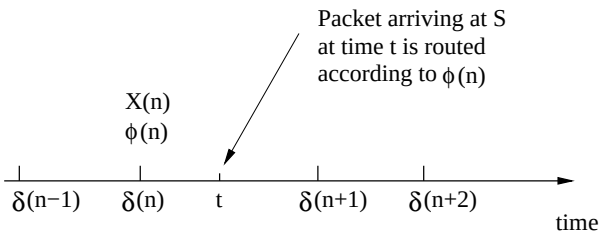


Figure 2.4: Routing of arriving packets at source S. Sequence $\{\delta(n)\}$ represents the times at which algorithm updates take place.

As mentioned earlier, the arriving packets at source S (ant or data packets) are routed according to the prevalent routing probabilities at S. Thus, in the context of the discussion above, a packet that arrives at the point of time t, is routed according to the routing probability vector $f(t^-)$, the value just before the arrival time t (see Figure 2.4).

Another important point to note is that our delay estimation scheme (14) is a constant step size scheme. As is well known in the literature on adaptive algorithms (see, for example [5]), this enables the scheme to adapt to (track) long term changes in statistics of the delay processes. This is important for communication networks, because the statistics of arrival processes at the nodes as well as the network characteristics typically change with time.

2.3.1 Analysis of the Algorithm

We view the routing algorithm, consisting of equations (14) and (15), as a set of discrete stochastic iterations of the type usually considered in the literature on

stochastic approximation methods [5], [36]. We provide below the main convergence result which states that, when ϵ is small enough, the discrete iterations are closely tracked by a system of Ordinary Differential Equations (ODEs). In Appendix A, we provide a heuristic analysis which enables us to arrive at the appropriate ODE. This section is organised as follows. In subsection 2.3.1.1 we discuss the ODE approximation. In subsection 2.3.1.2 we study the equilibrium behavior of the routing algorithm, and in subsection 2.3.1.3 we provide some simulation results.

2.3.1.1 The ODE Approximation

An analysis of the dynamics of the system, as given by equations (14) and (15), is fairly complicated. However, when $\epsilon > 0$ is small, a time-scale decomposition simplifies matters considerably. The key observation is that, when ϵ is small, the delay estimates X_i evolve much more slowly compared to the waiting time (delay) processes Δ_i. Also, because the probabilities ϕ_i are (memoryless) functions of the delay estimates X_i, they too evolve at the same time-scale as the delay estimates. Consequently, with the vector $(X_1(n), \ldots, X_N(n))$ fixed at $(z_1, ..., z_N)$ (equivalently, $\phi_i(n), i = 1, \ldots, N$, fixed at $\phi_i = \frac{(z_i)^{-\beta}}{\sum_{j=1}^{N}(z_j)^{-\beta}}, i = 1, \ \ldots, \ N$), the delay processes $\Delta_i(.), i = 1, \ \ldots, \ N$, can be considered as converged to a stationary distribution, which depends on $(z_1, \ldots, z_N)$. Also, when ϵ is small, the evolution of the delay estimates can be tracked by a system of ODEs. A heuristic analysis of the algorithm,

as provided in Appendix A, shows that the ODE system for our case is given by

$$\begin{aligned}
\frac{dz_1(t)}{dt} &= \frac{(z_1(t))^{-\beta}\Big(D_1(z_1(t),\ldots,z_N(t)) - z_1(t)\Big)}{\sum_{k=1}^{N}(z_k(t))^{-\beta}}, \\
&\vdots \qquad \vdots \\
\frac{dz_N(t)}{dt} &= \frac{(z_N(t))^{-\beta}\Big(D_N(z_1(t),\ldots,z_N(t)) - z_N(t)\Big)}{\sum_{k=1}^{N}(z_k(t))^{-\beta}},
\end{aligned} \tag{18}$$

with the set of initial conditions $z_1(0) = x_1, \ldots, z_N(0) = x_N$. $D_i(z_1, \ldots, z_N), i = 1, \ldots, N$, are the mean waiting times in the queues under stationarity (as seen by arriving ant packets) with the delay estimates considered fixed at $z_1, \ldots, z_N$.

Formally, the ODE approximation result can be stated as follows (see Benveniste, Metivier, and Priouret [5]). For any fixed $\epsilon > 0$ and for $i = 1, \ldots, N$, consider the piecewise constant interpolation of $X_i(n)$ given by the equations : $z_i^{\epsilon}(t) = X_i(n)$ for $t \in [n\epsilon, (n+1)\epsilon\ ),\ n = 0, 1, 2, \ldots$, with the initial value $z_i^{\epsilon}(0) = X_i(0)$. Then the processes $\{z_i^{\epsilon}(t), t \geq 0\}, i = 1, \ldots, N$, converge to the solution of the ODE system (7) in the following sense : as $\epsilon \downarrow 0$, for any $0 \leq T < \infty$,

$$\sup_{0 \leq t \leq T} |z_i^{\epsilon}(t) - z_i(t)| \xrightarrow{P} 0, \tag{19}$$

where $\xrightarrow{P}$ denotes convergence in probability.

In order to obtain the evolution of the ODE, we need to compute the quantities $D_i(z_1, \ldots, z_N)$, for our queueing system. We recall that $D_i(z_1, \ldots, z_N), i = 1, \ldots, N$, refer to the means of the waiting times as seen by ant packet arrivals to the queues when the delay estimates are considered fixed at $z_1, \ldots, z_N$. Then the routing probabilities to the N queues are $\phi_i = \frac{(z_i)^{-\beta}}{\sum_{j=1}^{N}(z_j)^{-\beta}}, i = 1, \ldots, N$. We now discuss how to

compute the quantities $D_i(z_1, \ldots, z_N)$ given our assumptions on the statistics of the arrival processes and on the packet lengths of the arrival streams.

Under such conditions, every incoming arrival at source S from either of the Poisson streams (the ant or the data stream) is routed (independent of other arrivals) with probability ϕ_i towards queue Q_i. Thus the incoming arrival process in queue Q_i (for each i) is a superposition of two independent Poisson processes with rates $\lambda_A\phi_i$ and $\lambda_D\phi_i$. Consequently, every incoming packet into Q_i is, with probability $\frac{\lambda_A}{\lambda_A+\lambda_D}$, an ant packet, and with probability $\frac{\lambda_D}{\lambda_A+\lambda_D}$, a data packet. Also, under our assumptions on the statistics of the packet lengths of the arrival streams and on the arrival processes, all of the queues evolve as $M/G/1$ queues. The cumulative incoming stream into Q_i is Poisson with rate $(\lambda_A + \lambda_D)\phi_i$, and every incoming packet's service time is distributed according to the c.d.f G_i^A with probability $\frac{\lambda_A}{\lambda_A+\lambda_D}$ and according to the c.d.f. G_i^D with probability $\frac{\lambda_D}{\lambda_A+\lambda_D}$. We further assume that the queues are within the stability region of operation given by the inequalities : $(\lambda_A + \lambda_D)\phi_i E[S_i] < 1$, $i = 1, \ldots, N$, where $E[S_i]$, the mean packet service time in Q_i, is given by $E[S_i] = \frac{\lambda_A E[S_i^A]+\lambda_D E[S_i^D]}{\lambda_A+\lambda_D}$. We note that the average waiting time in the system as experienced by successive ant arrivals to queue Q_i, is the same as the average waiting time in Q_i by the PASTA (Poisson Arrivals See Time Averages) property. Thus, using the Pollaczek-Khinchin formula for the average waiting time and assuming that the queues are stable, we finally obtain the expression for $D_i(z_1, \ldots, z_N)$ $(i = 1, \ldots, N)$:

$$D_i(z_1, \ldots, z_N) = E[S_i] + \frac{(\lambda_A + \lambda_D)\phi_i E[S_i^2]}{2(1 - (\lambda_A + \lambda_D)\phi_i E[S_i])}, \tag{20}$$

where $E[S_i]$ and $E[S_i^2]$ are given respectively by $E[S_i] = \frac{\lambda_A E[S_i^A] + \lambda_D E[S_i^D]}{\lambda_A + \lambda_D}$ and $E[S_i^2] = \frac{\lambda_A E[(S_i^A)^2] + \lambda_D E[(S_i^D)^2]}{\lambda_A + \lambda_D}$, and $\phi_i = \frac{(z_i)^{-\beta}}{\sum_{j=1}^{N} (z_j)^{-\beta}}$.

Once the expressions for $D_i(z_1, \ldots, z_N)$ are available, we can numerically solve the ODE system (18), starting with the initial conditions $z_1(0), \ldots, z_N(0)$. We observe in our simulations that if we start the system with initial conditions such that we are inside the stability region, the system stays within the stability region thereafter.

2.3.1.2 Equilibrium behavior of the routing algorithm

We now obtain the equilibrium points of the ODE system (18) which would, in turn, enable us to obtain the equilibrium routing behavior of the system. In particular, we can obtain the equilibrium routing probabilities and the mean delays in the system under steady state operation of the network. For ϵ small, the steady state values of the estimates of the average waiting times (delays) in the N queues are approximately given by the components of the equilibrium points z^* of the ODE system (18). The equilibrium points of the ODE, z^*, must satisfy the set of equations given by

$$\begin{aligned}
\frac{(z_1^*)^{-\beta}}{\sum_{j=1}^{N} (z_j^*)^{-\beta}} \cdot \Big[D_1(z_1^*, \ldots, z_N^*) - z_1^* \Big] &= 0, \\
&\vdots \quad \vdots \\
\frac{(z_N^*)^{-\beta}}{\sum_{j=1}^{N} (z_j^*)^{-\beta}} \cdot \Big[D_N(z_1^*, \ldots, z_N^*) - z_N^* \Big] &= 0. \qquad (21)
\end{aligned}$$

The steady state routing probabilities, $\phi_1^*, \ldots, \phi_N^*$, are related to the average delay estimates, $z_1^*, \ldots, z_N^*$, through the equations, $\phi_i^* = \frac{(z_i^*)^{-\beta}}{\sum_{j=1}^N (z_j^*)^{-\beta}}$, $i = 1, \ldots, N$. Because we have assumed that our queues are in the stable region of operation, the steady state estimates of average delays must be finite, and so z_i^* must be finite for every $i = 1, \ldots, N$. Then the steady state routing probabilities, $\phi_i^* = \frac{(z_i^*)^{-\beta}}{\sum_{j=1}^N (z_j^*)^{-\beta}}$, $i = 1, \ldots, N$, are all strictly positive. Equations (21) then reduce to : $z_i^* = D_i(z_1^*, \ldots, z_N^*), i = 1, \ldots, N$. We also notice, from equation (20), that for each i, $D_i(z_1^*, \ldots, z_N^*)$ is a function solely of ϕ_i^*, and so, with a slight abuse of notation, we denote it by $D_i(\phi_i^*)$. Then, utilizing the fact that $\phi_i^* = \frac{(z_i^*)^{-\beta}}{\sum_{j=1}^N (z_j^*)^{-\beta}}$, we find that the equilibrium routing probabilities, $\phi_1^*, \ldots, \phi_N^*$, must satisfy the following fixed-point system of equations

$$\begin{aligned} \phi_1^* &= \frac{(D_1(\phi_1^*))^{-\beta}}{\sum_{j=1}^N (D_j(\phi_j^*))^{-\beta}}, \\ &\vdots \quad\quad \vdots \\ \phi_N^* &= \frac{(D_N(\phi_N^*))^{-\beta}}{\sum_{j=1}^N (D_j(\phi_j^*))^{-\beta}}. \end{aligned} \tag{22}$$

Notice that $\phi_1^* + \cdots + \phi_N^* = 1$.

A point to note is that the steady state probabilities, $\phi_1^*, \ldots, \phi_N^*$, must not only satisfy the above system of equations, but must all be strictly positive and satisfy the following stability conditions for the system: $(\lambda_A + \lambda_D)\phi_i^* E[S_i] < 1, i = 1, \ldots, N$. We now show that the system of equations (22) are actually the necessary and sufficient optimality conditions for an optimization problem involving the minimization of a convex objective function of $(\phi_1, \ldots, \phi_N)$ subject to the above mentioned constraints.

We show as a consequence that, if there exists a solution to the set of equations (22) that also satisfies the above mentioned constraints, then such a solution is unique.

Consider the optimization problem

$$\text{Minimize } F(\phi_1, \ldots, \phi_N) = \sum_{i=1}^{N} \int_0^{\phi_i} x[D_i(x)]^\beta dx,$$

$$\text{subject to } \phi_1 + \cdots + \phi_N = 1,$$

$$0 < \phi_1 < a_1,$$

$$\vdots$$

$$0 < \phi_N < a_N,$$

where $a_i = \frac{1}{(\lambda_A + \lambda_D)E[S_i]}, i = 1, \ldots, N$.

Let us denote by C the set defined by the constraints of the above optimization problem – the feasible set. It is easy to see that C is a convex subset of $\mathbb{R}^N$. It is possible that the set C is empty (for a given set of values of λ_A, λ_D, and $E[S_i], i = 1, \ldots, N$), which means that there are no feasible solutions to the above optimization problem in such a case. We assume, in what follows, that there exists at least one feasible solution to the above optimization problem, i.e., C is non-empty.

Before we attempt to solve the optimization problem, we make certain natural assumptions on the delay functions $D_i(x), i = 1, \ldots, N$. We assume that the functions $D_i(x)$ are positive real-valued, differentiable and monotonically increasing on their domains of definition. This holds true in most cases of interest, because when the routing probability for an outgoing link increases, the amount of traffic flow into that link also increases, resulting in an increase of the delay. The following proposition describes the optimal solutions ϕ^* of the above optimization problem.

Proposition 1. *Given the above assumptions on the delay functions* $D_i(x), i = 1, \ldots, N$, *a probability vector* ϕ^* *is a local minimum of* F *over* C *if and only if* ϕ^* *satisfies the set of fixed-point equations* (22). ϕ^* *is then also the unique global minimum of* F *over* C.

Proof: The Hessian of F is a diagonal matrix given by

$$\nabla^2 F(\phi_1, \ldots, \phi_N) = \text{diag}\Big([D_i(\phi_i)]^{\beta-1} \{ D_i(\phi_i) + \beta \phi_i D_i'(\phi_i) \} \Big), \tag{23}$$

where $D_i'(.)$ denotes the derivative of $D_i(.)$. Under the above assumptions on the $D_i(x)$'s, $\nabla^2 F$ is positive definite over C, and so F is a strictly convex function on C. Consequently, any local minimum of F is also a global minimum of F over C; furthermore, there is atmost one such global minimum [6].

If $\phi^* = (\phi_1^*, \ldots, \phi_N^*)$ is a local minimum of F over C, we must have (Proposition 2.1.2 of Bertsekas [6]),

$$\sum_{i=1}^{N} \frac{\partial F}{\partial \phi_i}(\phi^*)(\phi_i - \phi_i^*) \geq 0, \forall \phi \in C. \tag{24}$$

Let us fix a pair of indices $i, j, i \neq j$. Then choose $\phi_i = \phi_i^* + \delta$ and $\phi_j = \phi_j^* - \delta$, and let $\phi_k = \phi_k^*, \forall k \neq i, j$. Now, choosing $\delta > 0$ small enough that the vector $\phi = (\phi_1, \ldots, \phi_N)$ is also in C, the above condition becomes

$$\Big(\frac{\partial F}{\partial \phi_i}(\phi^*) - \frac{\partial F}{\partial \phi_j}(\phi^*) \Big) \delta \geq 0,$$

$$\text{or}, \quad \phi_i^* [D_i(\phi_i^*)]^\beta \geq \phi_j^* [D_j(\phi_j^*)]^\beta.$$

By a similar argument, we can show that $\phi_j^* [D_j(\phi_j^*)]^\beta \geq \phi_i^* [D_i(\phi_i^*)]^\beta$. Thus, the necessary conditions for ϕ^* to be a local minimum are

$$\phi_1^* [D_1(\phi_1^*)]^\beta = \cdots = \phi_N^* [D_N(\phi_N^*)]^\beta.$$

Combining this with the normalization condition, $\phi_1^* + \cdots + \phi_N^* = 1$, gives us the system of equations (22).

The necessary conditions above can also be written in the form

$$\frac{\partial F}{\partial \phi_1}(\phi^*) = \cdots = \frac{\partial F}{\partial \phi_N}(\phi^*).$$

We check that these conditions are also sufficient for ϕ^* to be a local minimum. Suppose $\phi^* \in C$ satisfies the above conditions. Then for every other vector $\phi \in C$, we have $\sum_{i=1}^{N}(\phi_i - \phi_i^*) = 0$. So, the quantity

$$\sum_{i=1}^{N} \frac{\partial F}{\partial \phi_i}(\phi^*)(\phi_i - \phi_i^*) = \frac{\partial F}{\partial \phi_1}(\phi^*) \sum_{i=1}^{N}(\phi_i - \phi_i^*) = 0.$$

Then, because F is convex over C, by Proposition 2.1.2 of Bertsekas [6], ϕ^* is a local minimum. $\square$

For our model, it is easy to check that the functions $D_i(x)$, as given by (20), are positive real-valued, differentiable and monotonically increasing on their domains of definition. Thus, there is a unique equilibrium probability vector ϕ^* which satisfies the fixed-point equations (22).

2.3.1.3 Simulation Results and Discussion

In this subsection we consider a few illustrative examples. The queueing system as described at the beginning of Section 2.3 has been implemented using a discrete event simulator. We present here results for the case when the number of parallel paths is $N = 3$. The step size ϵ was set at the value 0.002 in the examples described below. In the first example β is set at 1 and in the second example β is set at 2.

For both the examples, the ant and the data traffic arrival processes are Poisson with rates $\lambda_A = 1$ and $\lambda_D = 1$, respectively. For the ant packets, the service times in the three queues are exponential with means $E[S_1^A] = 1/3.0$, $E[S_2^A] = 1/4.0$ and $E[S_3^A] = 1/5.0$, respectively. For the data packets also, the service times in the three queues are exponential with means $E[S_1^D] = 1/3.0$, $E[S_2^D] = 1/4.0$ and $E[S_3^D] = 1/5.0$, respectively. The initial values of the delay estimates in the three queues were set at $X_1(0) = 0.8$, $X_2(0) = 2.8$, and $X_3(0) = 5.6$.

We now discuss the results for the case when $\beta = 1$. With the values of the initial delay estimates set as above, the initial routing probabilities are $\phi_1(0) = 0.7$, $\phi_2(0) = 0.2$, and $\phi_3(0) = 0.1$, which ensures that, initially, we are inside the stability region of the queueing sytem. We observed in our simulations that as the queueing system evolved over time, never did the system become unstable.

Let us denote by $\mu_i = \frac{1}{E[S_i]}$ the service rate of packets in the queue $Q_i, i = 1, 2, 3$. Because the ant and the data packets both have the same average service times in each queue Q_i, and because the packets are exponentially distributed, the delays $D_i(\phi_i^*)$ (computed using the Pollaczek-Khinchin formula (20)) in the fixed point equations (22) are given by

$$D_i(\phi_i^*) = \frac{1}{\mu_i - \lambda\phi_i^*}, \tag{25}$$

where $\lambda = \lambda_A + \lambda_D$ (the familiar $M/M/1$ relations). Consequently, the fixed point equations (22) reduce to

$$\phi_i^* = \frac{\mu_i - \lambda\phi_i^*}{\sum_{j=1}^{3}\big(\mu_j - \lambda\phi_j^*\big)}, \quad i = 1, 2, 3,$$

which on simplification gives us

$$\phi_i^* = \frac{\mu_i}{\sum_{j=1}^{3} \mu_j}, \quad i = 1, 2, 3.$$

Thus in this very special case, the equilibrium routing probabilities are directly proportional to the service rates of packets in the queues. The equilibrium delays from (25) are then

$$D_i(\phi_i^*) = \frac{\sum_{j=1}^{3} \mu_i}{\mu_i\big(\sum_{j=1}^{3} \mu_j - \lambda\big)}, \quad i = 1, 2, 3.$$

Figures 2.5(a), 2.5(b) and 2.5(c) provide plots of the interpolated delay estimates $z_i^{\epsilon}(t), i = 1, 2, 3$, in the three queues versus the ODE approximations $z_1(t), z_2(t), z_3(t)$, obtained by numerically solving (18). We see that the theoretical ODEs track the simulated delay estimates fairly well. Figures 2.6(a), 2.6(b) and 2.6(c) provide plots of routing probabilities $\phi_1(n), \phi_2(n)$, and $\phi_3(n)$. The routing probabilities converge to the equilibrium values $\phi_1^* = 3/12, \phi_2^* = 4/12, \phi_3^* = 5/12$ (note that $\mu_1 = 3, \mu_2 = 4, \mu_3 = 5$).

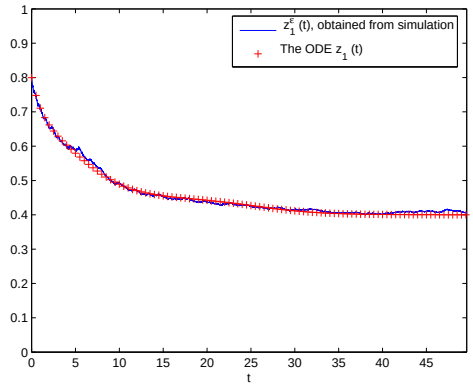
We now consider the case when $\beta = 2$. The fixed point equations (22) are now

$$\phi_i^* = \frac{\left(\mu_i - \lambda\phi_i^*\right)^2}{\sum_{j=1}^{3} \left(\mu_j - \lambda\phi_j^*\right)^2}, \quad i = 1, 2, 3.$$

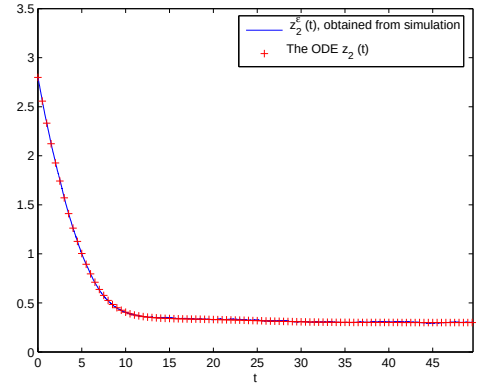
We numerically solve the fixed point system for the above set of values of arrival rates and service rates, and obtain the solution $\phi_1^* = 0.197, \phi_2^* = 0.326, \phi_3^* = 0.477$. We note that with the larger value of β more of the incoming flow is directed towards the path with the highest capacity (or service rate). Further discussion of this is provided in the next subsection 2.3.1.4 below. Once the routing probabilities are

obtained, the equilibrium delays can be immediately calculated from (25) above; they are found to be $D_1(\phi_1^*) = 0.384, D_2(\phi_2^*) = 0.300, D_3(\phi_3^*) = 0.247$.

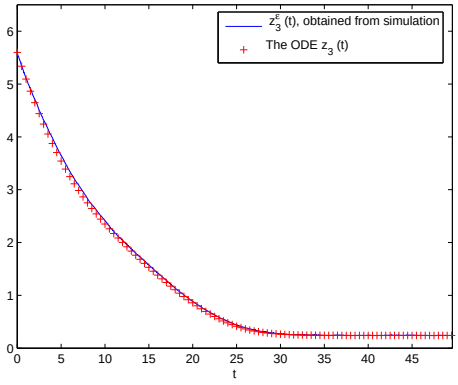
Figures 2.7(a), 2.7(b) and 2.7(c) provide plots of the interpolated delay estimates $z_i^\epsilon(t), i = 1, 2, 3$, in the three queues versus the ODE approximations $z_1(t), z_2(t), z_3(t)$, obtained by numerically solving (18). Figures 2.8(a), 2.8(b) and 2.8(c) provide plots of routing probabilities $\phi_1(n), \phi_2(n)$, and $\phi_3(n)$.



(a) The ODE approximation for $X_1(n)$

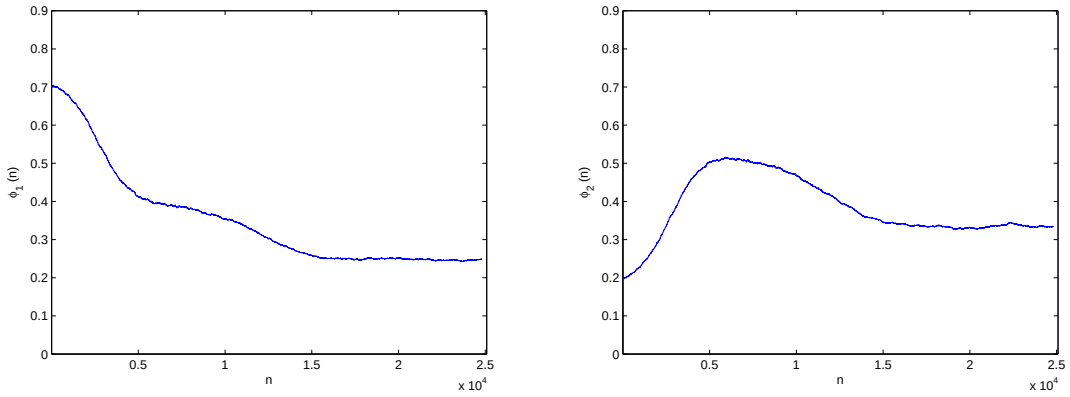


(b) The ODE approximation for $X_2(n)$



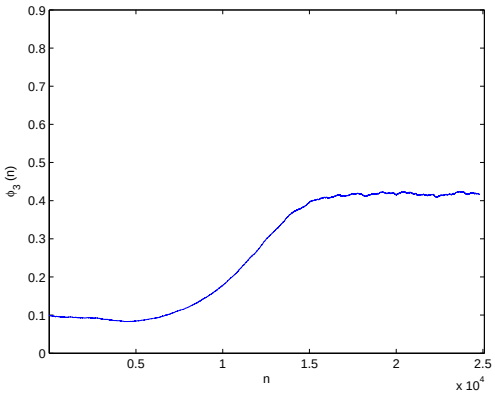
(c) The ODE approximation for $X_3(n)$

Figure 2.5: The ODE approximations. Parameters: $\lambda_A = 1, \lambda_D = 1, \beta = 1, E[S_1^A] = E[S_1^D] = 1/3.0, E[S_2^A] = E[S_2^D] = 1/4.0, E[S_3^A] = E[S_3^D] = 1/5.0$.



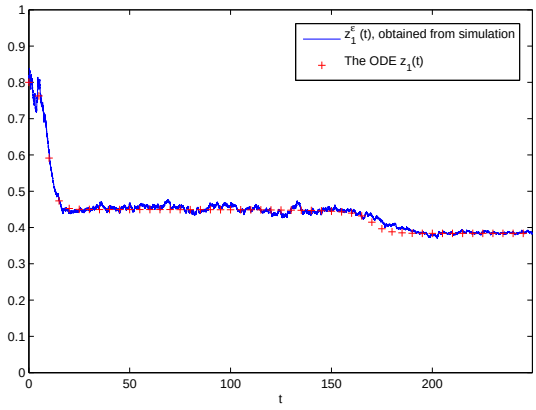
(a) The plot for $\phi_1(n)$

(b) The plot for $\phi_2(n)$

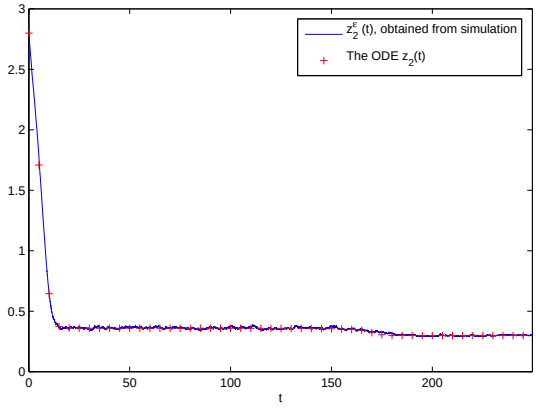


(c) The plot for $\phi_3(n)$

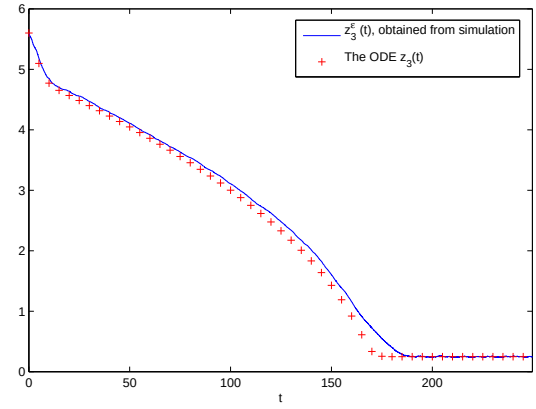
Figure 2.6: Plots for the routing probabilities. Parameters: $\lambda_A = 1, \lambda_D = 1, \beta = 1, E[S_1^A] = E[S_1^D] = 1/3.0, E[S_2^A] = E[S_2^D] = 1/4.0, E[S_3^A] = E[S_3^D] = 1/5.0$.



(a) The ODE approximation for $X_1(n)$

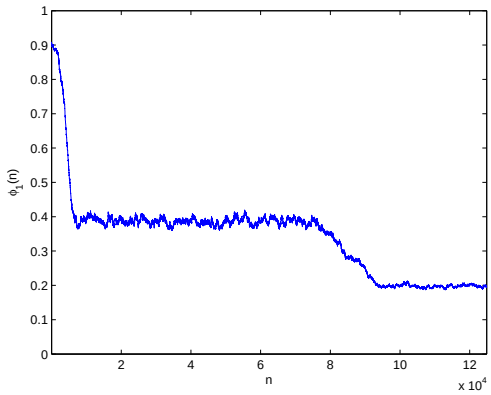


(b) The ODE approximation for $X_2(n)$

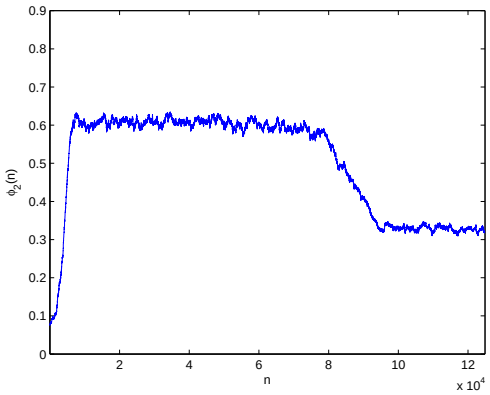


(c) The ODE approximation for $X_3(n)$

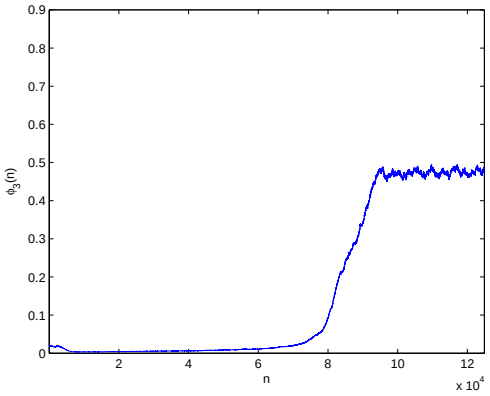
Figure 2.7: The ODE approximations. Parameters: $\lambda_A = 1, \lambda_D = 1, \beta = 2, E[S_1^A] = E[S_1^D] = 1/3.0, E[S_2^A] = E[S_2^D] = 1/4.0, E[S_3^A] = E[S_3^D] = 1/5.0$.



(a) The plot for $\phi_1(n)$



(b) The plot for $\phi_2(n)$



(c) The plot for $\phi_3(n)$

Figure 2.8: Plots for the routing probabilities. Parameters: $\lambda_A = 1, \lambda_D = 1, \beta = 2, E[S_1^A] = E[S_1^D] = 1/3.0, E[S_2^A] = E[S_2^D] = 1/4.0, E[S_3^A] = E[S_3^D] = 1/5.0$.

2.3.1.4 Equilibrium routing behavior and the parameter β

In this section, we study the effect of the parameter β on the equilibrium routing behavior. In Section 2.3.1.2 we noted that the equilibrium routing probability vector ϕ^* was such that (for a given β) its components satisfied the following set of equations (as also the queue stability conditions)

$$\phi_1^*[D_1(\phi_1^*)]^\beta = \cdots = \phi_N^*[D_N(\phi_N^*)]^\beta,$$

$$\phi_1^* + \phi_2^* + \cdots + \phi_N^* = 1.$$

We denote the (unique) solution by $\phi^*(\beta)$, to emphasize its dependence on β. In this section, we show that at equilibrium, as β increases, more flow is diverted towards the outgoing link with the most capacity. To show this concretely, we consider in this section the case when the delays D_i are functions only of the first moments $E[S_i]$ of the service times, and hence only of the capacities C_i. (This would be true, for example, when the lengths of both the ant and the data packets are exponentially distributed.) Consequently, the delay function of the i-th queue can be written as $D_i(\phi_i) = D(\phi_i, C_i)$. We assume that this function has the following properties: it is positive, and it is a strictly increasing function of ϕ_i when C_i is held fixed, and a strictly decreasing function of C_i when ϕ_i is held fixed.

Suppose $C_1 > C_2 = \cdots = C_N$. Then using the relations

$$\phi_1^*(\beta)[D(\phi_1^*(\beta), C_1)]^\beta = \cdots = \phi_N^*(\beta)[D(\phi_N^*(\beta), C_N)]^\beta,$$

it is not difficult to check that

$$\phi_1^*(\beta) > \phi_2^*(\beta) = \cdots = \phi_N^*(\beta)$$

and consequently that

$$D(\phi_1^*(\beta), C_1) < D(\phi_2^*(\beta), C_2) \ \big(= \cdots = D(\phi_N^*(\beta), C_N)\big). \tag{26}$$

In this subsection, lets assume that β is a nonnegative real number (instead of being a positive integer). To arrive at a contradiction, lets suppose, for some small positive $\delta\beta$ that $\phi_1^*(\beta + \delta\beta) < \phi_1^*(\beta)$; then we also have $\phi_2^*(\beta + \delta\beta) > \phi_2^*(\beta)$. This implies that

$$\frac{\phi_1^*(\beta + \delta\beta)}{\phi_2^*(\beta + \delta\beta)} < \frac{\phi_1^*(\beta)}{\phi_2^*(\beta)}. \tag{27}$$

Using the relationships with the delays, we then have

$$\Big[\frac{D(\phi_2^*(\beta + \delta\beta), C_2)}{D(\phi_1^*(\beta + \delta\beta), C_1)}\Big]^{\beta+\delta\beta} < \Big[\frac{D(\phi_2^*(\beta), C_2)}{D(\phi_1^*(\beta), C_1)}\Big]^{\beta},$$

$$\text{or, } \Big[\frac{D(\phi_2^*(\beta + \delta\beta), C_2)}{D(\phi_2^*(\beta), C_2)} . \frac{D(\phi_1^*(\beta), C_1)}{D(\phi_1^*(\beta + \delta\beta), C_1)}\Big]^{\beta+\delta\beta} < \Big[\frac{D(\phi_1^*(\beta), C_1)}{D(\phi_2^*(\beta), C_2)}\Big]^{\delta\beta}.$$

Using the hypothesis and the monotonicity property of the delay function with respect to the routing probability, it is easy to see that the left hand side of the above inequality is greater than one, which implies that

$$D(\phi_1^*(\beta), C_1) > D(\phi_2^*(\beta), C_2),$$

which contradicts the relation (26).

With some additional effort, we can in fact show that $\frac{d\phi_1^*(\beta)}{d\beta} > 0$, when $C_1 > C_i, i = 2, \ldots, N$ (see Appendix B). However, if additionally $C_2 > C_i, i = 3, \ldots, N$, it is not true that $\frac{d\phi_2^*(\beta)}{d\beta} > 0$. The example in Subsection 2.3.1.3 shows this fact (the value of ϕ_2^* is smaller for $\beta = 2$ than for $\beta = 1$).

We now consider a couple of examples studying what happens when $\beta \to \infty$. As in Subsection 2.3.1.3 let us denote by $\mu_i = \frac{1}{E[S_i]}$ the service rate of packets in

the queues Q_i, and assume that the lengths of the ant and the data packets are exponentially distributed and that the average service times of both ant and data packets in a particular queue are the same ($E[S_i^A] = E[S_i^D]$). The number of parallel paths $N = 3$. The delays in the queues are then given by the formula $D_i(\phi_i^*) = \frac{1}{\mu_i - \lambda\phi_i^*}$. For the first example, we assume that $\lambda_A = 1, \lambda_D = 1, \mu_1 = 4, \mu_2 = 3, \mu_3 = 3$. The fixed point equations that the equilibrium routing probabilities must satisfy are given by (note that $\phi_2^* = \phi_3^*$)

$$\begin{aligned} \phi_1^* &= \frac{\left(4 - 2\phi_1^*\right)^\beta}{\left(4 - 2\phi_1^*\right)^\beta + 2\left(3 - 2\phi_2^*\right)^\beta}, \\ \phi_2^* &= \frac{1 - \phi_1^*}{2}. \end{aligned}$$

We solve the above fixed point system of equations in Mathematica for increasing values of β. The equilibrium routing probabilities are equal to $\phi_1^* = 0.664, \phi_2^* = 0.168, \phi_3^* = 0.168$ for high values of β [3]. It may be noted however that once $\phi_1^* \geq \frac{2}{3}$, $D_1(\phi_1^*) = \frac{1}{4-2\phi_1^*} \geq D_2(\phi_2^*) = \frac{1}{3-2\phi_2^*}$, the delay in the higher capacity path is more than or equal to the delay in the lower capacity path, which is impossible. Thus it may be surmised in this case that when β increases to ∞, ϕ_1^* increases to 2/3 but never attains that value.

We now consider the same case as above but increase the service rate in queue Q_1 to $\mu_1 = 6$. In this case, the equilibrium routing probabilities get arbitrarily close to $\phi_1^* = 1.0, \phi_2^* = 0, \phi_3^* = 0$ with increasing β; all the incoming traffic is routed through Q_1 [4]. It may be noted in this case, that for no $\phi_1^* \in [0, 1]$ (with $\phi_2^* = \frac{1-\phi_1^*}{2}$),

[3] These are the values when $\beta = 516$. Mathematica reports errors due to machine precision problems beyond this value of β.

[4] Mathematica gives these values when $\beta = 40$.

is it possible that $D_1(\phi_1^*) = \frac{1}{6-2\phi_1^*} \geq D_2(\phi_2^*) = \frac{1}{3-2\phi_2^*}$. We can thus have all the traffic routed through a particular path, for large β, provided that the capacity of the path is quite large compared to the other paths.

Thus β acts like a tuning parameter that can be used to modulate the fraction of flow on the outgoing links under equilibrium. Higher values of β make the flows more concentrated on the outgoing links with more capacity (in the limiting case of $\beta \to \infty$, as the example above shows we can even have all the incoming flow routed to the highest capacity path) whereas lower values of beta make the flows more evenly distributed on the outgoing links (in the limiting case of $\beta = 0$, the flows are evenly distributed: $\phi_i^* = \frac{1}{N}, i = 1, 2, \ldots, N$).

2.4 The General Network Model: The "Single Commodity" Case

We consider the Ant Routing Algorithm for a general network in this section. The set of nodes in the network is denoted by $\mathcal{N}$, and the set of links by $\mathcal{L}$. We consider the problem of routing from the various nodes i of the network to a single destination node D (the "single commodity" problem). Let $N(i)$ denote the set of neighbor nodes of i, and let $(i, j), j \in N(i)$, denote the set of outgoing links from the node i. At every node i, there exists queues associated with the outgoing links (i, j); we assume these queues to be of infinite size. As in the case for the N parallel links network, we assume that the queuing delays dominate the processing and propagation delays in the links (though again, this can be accommodated with slight modifications). We recall that every node i in the network maintains the

following data structures: the routing table $\mathcal{R}(i)$ and the network information table $\mathcal{I}(i)$. The entries of $\mathcal{R}(i)$ for our case are the routing probabilities ϕ_j^{iD} of routing an incoming packet at node i bound for D, through the outgoing link (i, j). The entries of $\mathcal{I}(i)$ are the mean delay estimates X_j^{iD} related to the routes from i to D that go via the neighbor nodes j.

The general algorithm of Bean and Costa is asynchronous and distributed. This is because the nodes launch the Forward Ants (FAs) towards the destination in an uncoordinated way. Moreover, each Forward Ant - Backward Ant (BA) pair experiences an arbitrary (random) delay while traversing the network, and the algorithms at the nodes (involving updates of $\mathcal{R}(i)$ and $\mathcal{I}(i)$) are triggered at arbitrary points of time. We consider a more simplified view of operation of the algorithm, which is still asynchronous and distributed and retains the main characteristics of the algorithm, but is easier to analyse.

We assume that ant packets (FA packets) are generated according to some point process of positive rate λ_A at each of the nodes. These packets are routed towards the destination node D by the intermediate nodes of the network based on the *prevalent* routing probabilities at the nodes. Initially, at every node i the mean delay estimate for the route $i \to j \to \cdots \to D$ is given by $X_j^{iD}(0) = x_j^{iD}$, and the outgoing routing probabilities are given by $\phi_j^{iD}(0) = \frac{(x_j^{iD})^{-\beta}}{\sum_{k \in N(i)} (x_k^{iD})^{-\beta}}$. Whenever a BA packet (corresponding to an FA packet) arrives at a node, the mean delay estimates and the routing probabilities at that node are updated. We assume that the BA packets arrive back at the source nodes (from which the corresponding FA packets are launched) from the destination node D instantaneously; that is, there

is no delay associated with travel of the BA packet from D to the source node. We consider the (asynchronous) general operation of the algorithm as described earlier (Sections 2.1 and 2.2) with this modification. Furthermore, we note that in the general operation, the BA packet updates the delay estimates at every node that it traverses on its way back to the source, besides the source itself. In what follows, we shall consider the more simplified operation, where only the delay estimates and the routing probabilities at the source node (from where the corresponding FA packet was launched) are updated. Let $\{\delta(n)\}_{n=1}^{\infty}$ denote the sequence of times at which successive BA packets arrive back at the nodes of the network. These are also the sequence of times at which the algorithm updates are triggered at the various nodes of the network. At time $\delta(n)$, let $X(n)$ and $\phi(n)$ denote the vector of mean delay estimates and the vector of outgoing routing probabilities at the network nodes, respectively. The components of $X(n)$ and $\phi(n)$ are $X_j^{iD}(n), j \in N(i), i \in \mathcal{N}$, and $\phi_j^{iD}(n), j \in N(i), i \in \mathcal{N}$, respectively.

By time $\delta(n)$, n BA packets will have arrived at the network nodes. Let $T(n)$ be the $\mathcal{N}$-valued random variable that indicates which node the n-th BA packet arrives back to. Then $\xi^i(n) = \sum_{k=1}^{n} I_{\{T(k)=i\}}$ gives the number of BA packets that have arrived at node i by time $\delta(n)$. Let $R^i(.)$ denote the routing decision variable for ant packets launched from node i; that is, we say that the event $\{R^i(k) = j\}$ has occurred if the k-th FA packet that arrives at D and that has been launched from i, has been routed via the outgoing link (i, j) (this is also the k-th BA packet that arrives back at i through link (i, j), by the zero delay assumption on the BA packets). Let $\psi_j^i(n) = \sum_{k=1}^{\xi^i(n)} I_{\{R^i(k)=j\}}$; $\psi_j^i(n)$ gives the number of BA packets that

arrive at node i by time $\delta(n)$ and which have arrived via the outgoing link (i,j) (the corresponding FA packets have been routed via the outgoing link (i,j)). Also, let $\{\Delta_j^{iD}(m)\}$ denote the sequence of delay measurements obtained by successive FA packets arriving at D, having traversed the route $i \to j \to \cdots \to D$. This is also the sequence of delay measurements about the route $i \to j \to \cdots \to D$ made available to the source i by the BA packets.

Lets suppose that at time $\delta(n)$ a BA packet arrives back at the node i. Furthermore, suppose that the corresponding FA packet was routed via the outgoing link (i,j). When this BA packet arrives back at node i, the delay estimate X_j^{iD} is updated using an exponential estimator ($\epsilon \in (0,1)$)

$$X_j^{iD}(n) = X_j^{iD}(n-1) + \epsilon\Big(\Delta_j^{iD}(\psi_j^i(n)) - X_j^{iD}(n-1)\Big). \tag{28}$$

The delay estimates X_k^{iD} for the other outgoing routes $i \to k \to \cdots \to D \quad (k \in N(i), k \neq j)$ are left unchanged

$$X_k^{iD}(n) = X_k^{iD}(n-1); \tag{29}$$

also the delay estimates for the other nodes are not changed

$$X_p^{lD}(n) = X_p^{lD}(n-1), \quad \forall p \in N(l), \forall l \neq i. \tag{30}$$

Also as soon as the delay estimates are updated at node i, the outgoing routing probabilities are also updated

$$\phi_j^{iD}(n) = \frac{\left(X_j^{iD}(n)\right)^{-\beta}}{\sum_{k\in N(i)}\left(X_k^{iD}(n)\right)^{-\beta}}, \quad j \in N(i). \tag{31}$$

The routing probabilities at the other nodes are left unchanged.

In general thus the evolution of the delay estimates at the various nodes of the network can be described by the following set of stochastic iterative equations

$$\begin{aligned} X_j^{iD(\epsilon)}(n) &= X_j^{iD(\epsilon)}(n-1) + \epsilon\, I_{\{T^{(\epsilon)}(n)=i, R^{i(\epsilon)}(\xi^{i(\epsilon)}(n))=j\}} \times \\ &\quad \left(\Delta_j^{iD(\epsilon)}(\psi_j^{i(\epsilon)}(n)) - X_j^{iD(\epsilon)}(n-1)\right), \forall j \in N(i), \forall i \in \mathcal{N}, n \geq 1, \end{aligned} \tag{32}$$

starting with the initial conditions $X_j^{iD(\epsilon)}(0) = x_j^{iD}, \forall j \in N(i), \forall i \in \mathcal{N}$. As in the N parallel links case, the ϵ's in the superscripts indicate the dependence of the evolution of the quantities involved on ϵ.

The routing probabilities are updated in the usual way

$$\phi_j^{iD(\epsilon)}(n) = \frac{\left(X_j^{iD(\epsilon)}(n)\right)^{-\beta}}{\sum_{k \in N(i)} \left(X_k^{iD(\epsilon)}(n)\right)^{-\beta}}, \quad \forall j \in N(i), \forall i \in \mathcal{N}, n \geq 1, \tag{33}$$

starting with the initial conditions $\phi_j^{iD(\epsilon)}(0) = \frac{(x_j^{iD})^{-\beta}}{\sum_{k \in N(i)} (x_k^{iD})^{-\beta}}, \forall j \in N(i), \forall i \in \mathcal{N}$ [5].

2.4.1 Analysis of the Algorithm

As in the case for the N parallel links network, we view the routing algorithm, consisting of equations (32) and (33), as a set of discrete stochastic iterations of the type considered in the literature on stochastic approximation. In the following subsection 2.4.1.1, we discuss an ODE approximation result, and in subsection 2.4.1.2 we consider the equilibrium behavior of the routing algorithm.

[5] There are no records of delay estimates and routing probabilities at the destination D, because all packets are destined for it. This is assumed understood in the above equations, though not explicitly mentioned.

2.4.1.1 The ODE Approximation

In this subsection we discuss the ODE approximation result. The key observation again is that there is a time-scale decomposition when $\epsilon > 0$ is small enough; the delay estimates X_j^{iD} then evolve much more slowly compared to the delay processes Δ_j^{iD}. Also then the probabilities ϕ_j^{iD} evolve at the "same time-scale" as the delay estimates. We assume that, with the vector $X(n)$ fixed at z (equivalently $\phi(n)$ fixed at ϕ, the components of ϕ being $\phi_j^{iD} = \frac{(z_j^{iD})^{-\beta}}{\sum_{k\in N(i)}(z_k^{iD})^{-\beta}}$), the delay processes $\{\Delta_j^{iD}(.)\}$ converge to a proper stationary distribution, which is dependent on z. This would be true when the network queues are stable; that is, the average input rate of traffic into every queue is smaller than the average service rate of packets in the queue. We assume that this is true in what follows, and we then denote the mean of the delay processes under stationarity by $D_j^{iD}(z)$.

As in the N parallel paths case, the evolution of the vector $X(n)$ can again be tracked by a deterministic ODE, when the step-size ϵ of the estimation scheme is small. A heuristic development of the ODE approximation can be attempted following exactly the same procedure as described in Appendix A for the N parallel paths case. We do not describe the steps here; instead later in Section 2.4.2, we outline a proof of convergence to the ODE. We have the following ODE system for our case

$$\frac{dz_j^{iD}(t)}{dt} = \frac{\zeta_i(z(t))\ (z_j^{iD}(t))^{-\beta}\Big(D_j^{iD}(z(t)) - z_j^{iD}(t)\Big)}{\sum_{k\in N(i)}(z_k^{iD}(t))^{-\beta}}, \quad \forall j \in N(i), \forall i \in \mathcal{N}, t > 0, \tag{34}$$

with the initial conditions given by $z_j^{iD}(0) = x_j^{iD}, \forall j \in N(i), \forall i \in \mathcal{N}$. We note again

that $D_j^{iD}(z)$ is the average delay (under stationarity) experienced by FA packets traveling to the destination D via the outgoing link (i, j), when the delay estimate vector is considered fixed at z. Also, $\zeta_i(z)$ is a function of z and assumes values from the set $(0, 1)$. It is the long term fraction of BA packets arriving back at node i, and hence also the fraction of times the update algorithms are triggered at node i, when the delay estimate vector is considered fixed at z. The right hand side of the ODE (34) is denoted by the function F_{ij}, $F_{ij}(z(t)) = \dfrac{\zeta_i(z(t))\ (z_j^{iD}(t))^{-\beta}\Big(D_j^{iD}(z(t)) - z_j^{iD}(t)\Big)}{\sum_{k\in N(i)}\left(z_k^{iD}(t)\right)^{-\beta}}$.

Numerical computation of the ODE requires the computation of the means $D_j^{iD}(z)$ (for given z), as well as the fractions $\zeta_i(z) \in (0, 1)$. The computation of the means depends upon the particular network (a system of inter-connected queues) under consideration. For general input arrival processes (of ant and data packets) at the source nodes and for general packet lengths of ant and data packets, and for an arbitrary network topology, it would be impossible to compute these means (no known procedure exists to solve such queuing networks analytically). Also, the fractions $\zeta_i(z)$ would generally be difficult to compute. For only some special network topologies, and making certain specific approximations and assumptions on the statistics of the arrival processes as well as the service times of packets in the network queues, can one actually compute the ODE.

2.4.1.2 Equilibrium behavior of the Routing Algorithm

The equilibrium behavior of the routing algorithm can be obtained by equating the right hand sides of the ODE approximation (34) to zero. For this section, we

adopt the following notation for convenience. We drop the superscripts D in all the quantities, and write z_j^{iD} as z_{ij}, $D_j^{iD}(z)$ as $D_{ij}(z)$, and ϕ_j^{iD} as ϕ_{ij}. Then we have because the $\zeta_i(z(.))$ are all positive [6], and denoting the values under equilibrium by a $*$ in the superscript, we have

$$\frac{(z_{ij}^*)^{-\beta}}{\sum_{k\in N(i)}(z_{ik}^*)^{-\beta}} \cdot \Big(D_{ij}(z^*) - z_{ij}^*\Big) = 0, \quad \forall j \in N(i), \forall i \in \mathcal{N}. \tag{35}$$

Now, the steady state routing probabilities are related to the steady state delay estimates through the relations $\phi_{ij}^* = \frac{(z_{ij}^*)^{-\beta}}{\sum_{k\in N(i)}(z_{ik}^*)^{-\beta}}, \forall j \in N(i), \forall i \in \mathcal{N}$. Under our stability assumptions, the steady state delay estimates are finite, and so the steady state routing probabilities must be all positive. Consequently, $D_{ij}(z^*) = z_{ij}^*, \forall j \in N(i), \forall i \in \mathcal{N}$. Now, denoting the functional dependence of the mean stationary delays on the routing probabilities also by $D_{ij}(\phi^*)$ (a slight abuse of notation), and noting that $\phi_{ij}^* = \frac{(z_{ij}^*)^{-\beta}}{\sum_{k\in N(i)}(z_{ik}^*)^{-\beta}}$, we finally find that the equilibrium routing probabilities must satisfy the following fixed-point system of equations

$$\phi_{ij}^* = \frac{(D_{ij}(\phi^*))^{-\beta}}{\sum_{k\in N(i)}(D_{ik}(\phi^*))^{-\beta}}, \quad \forall j \in N(i), \forall i \in \mathcal{N}. \tag{36}$$

Now $D_{ij}(\phi^*)$ can be written as

$$D_{ij}(\phi^*) = w_{ij}(\phi^*) + J^j(\phi^*), \tag{37}$$

where $J^j(\phi^*)$ is the expected delay from node j to the destination and $w_{ij}(\phi^*)$ is the expected delay along the link (i,j), both experienced by a FA packet when the routing probability vector is ϕ^*. The quantities ('costs-to-go') $J^j(\phi^*), j \in \mathcal{N}$, satisfy

[6] This is because of our assumption that ant packets are generated at a positive rate at each of the nodes.

the following set of equations

$$\begin{aligned} J^i(\phi^*) &= \sum_{j \in N(i)} \phi^*_{ij} \Big(w_{ij}(\phi^*) + J^j(\phi^*) \Big), \quad \forall i \in \mathcal{N}, i \neq D, \\ J^D(\phi^*) &= 0. \end{aligned} \tag{38}$$

Because our steady state routing probabilities ϕ^*_{ij} are all positive, clearly there exists a positive probability path from every node i to the destination node D. Consequently, there exists a unique vector solution $J(\phi^*)$ (see Bertsekas, Tsitsiklis [12]) to the set of equations above (a ϕ^* as ours is analogous to a proper policy for a stochastic shortest path problem with D as the termination state). Thus for every ϕ^*, there is a unique vector $J(\phi^*)$. This then implies that, for every ϕ^*, there is a unique vector of $D_{ij}(\phi^*)$'s. [7]

2.4.2 Proof of Convergence of the Ant Routing Algorithm

As discussed in Section 2.4, the evolution of the delay estimates and the routing probabilities is given by the equations

$$\begin{aligned} X^{iD}_j(n) &= X^{iD}_j(n-1) + \epsilon \, I_{\{T(n)=i, R^i(\xi^i(n))=j\}} \Big(\Delta^{iD}_j(\psi^i_j(n)) - X^{iD}_j(n-1) \Big), \\ & \qquad \forall j \in N(i), \forall i \in \mathcal{N}, n \geq 1, \end{aligned} \tag{39}$$

and the equations

$$\phi^{iD}_j(n) = \frac{(X^{iD}_j(n))^{-\beta}}{\sum_{k \in N(i)} (X^{iD}_k(n))^{-\beta}}, \quad \forall j \in N(i), \forall i \in \mathcal{N}, n \geq 1, \tag{40}$$

respectively, with the appropriate initial conditions.

[7] We do not have a uniqueness and existence result for the vector of equilibrium routing probabilities satisfying the fixed-point system (36), though.

We adopt the following notation for convenience. We drop the destination D from the superscript, and write X_j^{iD} as X_{ij}, $R^i(\xi^i(n))$ as $R_i(\xi_i(n))$, Δ_j^{iD} as Δ_{ij}, ψ_j^i as ψ_{ij}, ϕ_j^{iD} as ϕ_{ij}, and D_j^{iD} as D_{ij}. Also, recognizing the dependence on ϵ $(0 < \epsilon < 1)$ of the evolution of the delay estimates and the routing probabilities, we rewrite the above equations as

$$X_{ij}^{\epsilon}(n) = X_{ij}^{\epsilon}(n-1) + \epsilon\, I_{\{T^{\epsilon}(n)=i, R_i^{\epsilon}(\xi_i^{\epsilon}(n))=j\}} \Big(\Delta_{ij}^{\epsilon}(\psi_{ij}^{\epsilon}(n)) - X_{ij}^{\epsilon}(n-1)\Big), \quad \forall j \in N(i), \forall i \in \mathcal{N}, \tag{41}$$

$$\phi_{ij}^{\epsilon}(n) = \frac{(X_{ij}^{\epsilon}(n))^{-\beta}}{\sum_{k\in N(i)} (X_{ik}^{\epsilon}(n))^{-\beta}}, \quad \forall j \in N(i), \forall i \in \mathcal{N}. \tag{42}$$

We now consider the piecewise constant interpolation of $\{X_{ij}^{\epsilon}(n)\}$ given by the equations

$$z_{ij}^{\epsilon}(t) = X_{ij}^{\epsilon}(n), \quad \text{for } t \in [n\epsilon, (n+1)\epsilon), \tag{43}$$

$(n = 0, 1, 2, \ldots)$ with the initial value $z_{ij}^{\epsilon}(0) = X_{ij}^{\epsilon}(0) = x_{ij}$. Define the vector-valued piecewise constant process $z^{\epsilon}(t) = (z_{ij}^{\epsilon}(t))_{(i\in\mathcal{N}, j\in N(i))}$ for $t \geq 0$. The process $\{z^{\epsilon}(t)\}$ evolves on the path space $D^{|\mathcal{L}|}[0,\infty)$, consisting of right-continuous $\mathbb{R}^{|\mathcal{L}|}$-valued functions possessing left hand limits.

The stochastic iterations (41) (considered along with (42)) are an example of constant step-size stochastic approximation algorithms. For the proof of the ODE approximation for the algorithm, we follow the approach as given in the standard textbook of Kushner and Yin [36]. A brief outline of the proof is as follows:

- We first show that the family of processes $\{z^{\epsilon}(t)\}, \epsilon \in (0,1)$, is tight. Then there exists a subsequence $\epsilon(k) \to 0$ as $k \to \infty$ and a process $\{z(t)\}$ with

Lipschitz continuous paths such that $\{z^{\epsilon(k)}(t)\}$ converges weakly to $\{z(t)\}$; this is denoted by

$$z^{\epsilon(k)} \Rightarrow z \tag{44}$$

- The limit process $\{z(t)\}$ will be then shown to have the following property ($z(t)$ has components $z_{ij}(t), i \in \mathcal{N}, j \in N(i)$). Let $t, \tau > 0$ be arbitrary numbers, and let $0 \leq s_1, s_2, \ldots, s_p \leq t$ also be a set of arbitrary numbers. Then, for a bounded continuous function h, we have

$$E[h(z(s_1), z(s_2), \ldots, z(s_p))\Big(z_{ij}(t+\tau) - z_{ij}(t) - \int_t^{t+\tau} F_{ij}(z(u))du\Big)] = 0, \tag{45}$$

for each $i \in \mathcal{N}, j \in N(i)$. This fact then implies that $z_{ij}(t) - z_{ij}(0) - \int_0^t F_{ij}(z(u))du, t \geq 0$, is a martingale with respect to the filtration generated by the process $\{z(t)\}$. This martingale process, because it has Lipschitz continuous paths, can be shown to have zero quadratic variation. It is hence a constant. Because the martingale is zero at $t = 0$, it is identically zero with probability one. We shall thus have the result.

The fact that (45) holds will be shown by showing that

$$E[h(z^{\epsilon}(s_1), z^{\epsilon}(s_2), \ldots, z^{\epsilon}(s_p))\Big(z^{\epsilon}_{ij}(t+\tau) - z^{\epsilon}_{ij}(t) - \int_t^{t+\tau} F_{ij}(z^{\epsilon}(u))du\Big)] = 0, \tag{46}$$

and using the fact that $\{z^{\epsilon}(t)\}$ converges weakly to $\{z(t)\}$ (we are actually going through the subsequence $\epsilon(k)$).

Let us consider the increasing sequence of σ-fields $\{\mathcal{F}^{\epsilon}(n)\}$, where $\mathcal{F}^{\epsilon}(n)$ encapsulates the entire history of the algorithm for the time $t < \delta(n+1)$. In particular, it contains the σ-field generated by the random variables $X^{\epsilon}(0), X^{\epsilon}(1), \ldots, X^{\epsilon}(n)$. We

now make the following set of assumptions, under which we obtain our convergence results.

Assumptions:

$(A1)$ We assume that, for every i, j, and for every $\epsilon \in (0,1)$, the sequence $\{\Delta_{ij}^{\epsilon}(m)\}$ is uniformly integrable.

$(A2)$ We assume that, if the sequence $X(n)$ is fixed at a value x (the sequence $\phi(n)$ is then fixed at a value ϕ; ϕ has components $\phi_{ij} = \frac{(x_{ij})^{-\beta}}{\sum_{k \in N(i)} (x_{ik})^{-\beta}}$) then, for every $l \geq 0$, and for every $j \in N(i), i \in \mathcal{N}$,

$$\lim_{r \to \infty} \frac{1}{r} \sum_{m=l+1}^{l+r} E[I_{\{T(m)=i, R_i(\xi_i(m))=j\}} \ \Delta_{ij}(\psi_{ij}(m))/\mathcal{F}(m-1)] - \zeta_i(x) \ \phi_{ij} \ D_{ij}(x) \ = \ 0, \tag{47}$$

$$\text{and} \qquad \lim_{r \to \infty} \frac{1}{r} \sum_{m=l+1}^{l+r} E[I_{\{T(m)=i, R_i(\xi_i(m))=j\}}/\mathcal{F}(m-1)] - \zeta_i(x) \ \phi_{ij} \ = \ 0. \tag{48}$$

$(A3)$ We assume that the quantities $\zeta_i(x)\phi_{ij}D_{ij}(x)$ and $\zeta_i(x)\phi_{ij}$ are continuous functions of x.

We now embark on the proof. We first show the tightness of the family $\{z^{\epsilon}(t)\}, \epsilon \in (0,1)$, using the uniform integrability assumption.

From equation (41) we can write

$$\begin{aligned} X_{ij}^{\epsilon}(n+1) \ &= \ \Big(1 - \epsilon I_{\{T^{\epsilon}(n+1)=i, R_i^{\epsilon}(\xi_i^{\epsilon}(n+1))=j\}}\Big) X_{ij}^{\epsilon}(n) \\ &\qquad + \epsilon I_{\{T^{\epsilon}(n+1)=i, R_i^{\epsilon}(\xi_i^{\epsilon}(n+1))=j\}} \Delta_{ij}^{\epsilon}(\psi_{ij}^{\epsilon}(n+1)), \\ X_{ij}^{\epsilon}(n+1) \ &\leq \ X_{ij}^{\epsilon}(n) + \epsilon I_{\{T^{\epsilon}(n+1)=i, R_i^{\epsilon}(\xi_i^{\epsilon}(n+1))=j\}} \Delta_{ij}^{\epsilon}(\psi_{ij}^{\epsilon}(n+1)). \end{aligned}$$

Iterating we can see then that for every positive integer m,

$$X_{ij}^{\epsilon}(n+m) \leq X_{ij}^{\epsilon}(n) + \epsilon\Big(\sum_{k=n+1}^{n+m} I_{\{T^{\epsilon}(k)=i, R_i^{\epsilon}(\xi_i^{\epsilon}(k))=j\}} \Delta_{ij}^{\epsilon}(\psi_{ij}^{\epsilon}(k))\Big).$$

Consequently, for any $L > 0$, we have

$$\begin{aligned} E[|X_{ij}^{\epsilon}(n+m) - X_{ij}^{\epsilon}(n)|] &\leq \epsilon \sum_{k=n+1}^{n+m} E[\Delta_{ij}^{\epsilon}(\psi_{ij}^{\epsilon}(k))], \\ &= \epsilon \sum_{k=n+1}^{n+m} E[\Delta_{ij}^{\epsilon}(\psi_{ij}^{\epsilon}(k)) I_{\{\Delta_{ij}^{\epsilon}(\psi_{ij}^{\epsilon}(k)) \geq L\}} \\ &\qquad + \Delta_{ij}^{\epsilon}(\psi_{ij}^{\epsilon}(k)) I_{\{\Delta_{ij}^{\epsilon}(\psi_{ij}^{\epsilon}(k)) < L\}}]. \end{aligned}$$

Thus, for any $n, n+m \in \{0, 1, 2, \ldots, \lfloor \frac{T}{\epsilon} \rfloor\}$ (for some fixed $0 < T < \infty$), we have

$$E[|X_{ij}^{\epsilon}(n+m) - X_{ij}^{\epsilon}(n)|] \leq T \sup_{k \geq 1} E[\Delta_{ij}^{\epsilon}(\psi_{ij}^{\epsilon}(k)) I_{\{\Delta_{ij}^{\epsilon}(\psi_{ij}^{\epsilon}(k)) \geq L\}}] + Lm\epsilon.$$

If we now let $t = n\epsilon$ and $\tau = m\epsilon$, and noting that $z_{ij}^{\epsilon}(t) = X_{ij}^{\epsilon}(\lfloor \frac{t}{\epsilon} \rfloor)$, we have

$$\sup_{0 \leq t \leq t+\tau \leq T} E[|z_{ij}^{\epsilon}(t+\tau) - z_{ij}^{\epsilon}(t)|] \leq T \sup_{k \geq 1} E[\Delta_{ij}^{\epsilon}(\psi_{ij}^{\epsilon}(k)) I_{\{\Delta_{ij}^{\epsilon}(\psi_{ij}^{\epsilon}(k)) \geq L\}}] + L\tau. \quad (49)$$

With T fixed, the uniform integrability of the sequence $\{\Delta_{ij}^{\epsilon}(m)\}$ allows us to choose L large enough that the first term on the right hand side can be made as small as we like. Once L is so chosen, we can choose τ small enough that the second term can be made as small as we like. We then have the following result. For any $0 < T < \infty$,

$$\lim_{\tau \downarrow 0} \lim_{\epsilon \downarrow 0} \sup_{0 \leq t \leq t+\tau \leq T} E[|z_{ij}^{\epsilon}(t+\tau) - z_{ij}^{\epsilon}(t)|] = 0. \quad (50)$$

(Simple modifications would enable us to argue the result for arbitrary $t, t+\tau \in [0, T]$, which are not integral multiples of ϵ.) Now, because

$$E[\| z^{\epsilon}(t+\tau) - z^{\epsilon}(t) \|] \leq \sum_{j \in N(i), i \in \mathcal{N}} E[|z_{ij}^{\epsilon}(t+\tau) - z_{ij}^{\epsilon}(t)|],$$

we have

$$\lim_{\tau\downarrow 0}\lim_{\epsilon\downarrow 0}\sup_{0\le t\le t+\tau\le T} E[\| z^\epsilon(t+\tau)-z^\epsilon(t)\|]=0. \tag{51}$$

The fact that this holds for every $0\le T<\infty$ is sufficient for the family $\{z^\epsilon(t),\epsilon\in(0,1)\}$ to be tight.

We now show the validity of (46). We have the following expression for $X^\epsilon_{ij}(n)$

$$\begin{aligned}
X^\epsilon_{ij}(n) &= X^\epsilon_{ij}(0)+\epsilon\sum_{m=1}^{n}\Big(I_{\{T^\epsilon(m)=i,R^\epsilon_i(\xi^\epsilon_i(m))=j\}}\Delta^\epsilon_{ij}(\psi^\epsilon_{ij}(m)) \\
&\qquad -E[I_{\{T^\epsilon(m)=i,R^\epsilon_i(\xi^\epsilon_i(m))=j\}}\Delta^\epsilon_{ij}(\psi^\epsilon_{ij}(m))/\mathcal{F}^\epsilon(m-1)]\Big) \\
&\quad +\epsilon\sum_{m=1}^{n}\Big(E[I_{\{T^\epsilon(m)=i,R^\epsilon_i(\xi^\epsilon_i(m))=j\}}\ \Delta^\epsilon_{ij}(\psi^\epsilon_{ij}(m))/\mathcal{F}^\epsilon(m-1)] \\
&\qquad -\zeta_i(X^\epsilon(m-1))\phi^\epsilon_{ij}(m-1)D_{ij}(X^\epsilon(m-1))\Big) \\
&\quad +\epsilon\sum_{m=1}^{n}\Big(\zeta_i(X^\epsilon(m-1))\phi^\epsilon_{ij}(m-1)D_{ij}(X^\epsilon(m-1))\Big) \\
&\quad -\epsilon\sum_{m=1}^{n}\Big(I_{\{T^\epsilon(m)=i,R^\epsilon_i(\xi^\epsilon_i(m))=j\}}\ X^\epsilon_{ij}(m-1) \\
&\qquad -E[I_{\{T^\epsilon(m)=i,R^\epsilon_i(\xi^\epsilon_i(m))=j\}}\ X^\epsilon_{ij}(m-1)/\mathcal{F}^\epsilon(m-1)]\Big) \\
&\quad -\epsilon\sum_{m=1}^{n}\Big(E[I_{\{T^\epsilon(m)=i,R^\epsilon_i(\xi^\epsilon_i(m))=j\}}\ X^\epsilon_{ij}(m-1)/\mathcal{F}^\epsilon(m-1)] \\
&\qquad -\zeta_i(X^\epsilon(m-1))\phi^\epsilon_{ij}(m-1)X^\epsilon_{ij}(m-1)\Big) \\
&\quad -\epsilon\sum_{m=1}^{n}\Big(\zeta_i(X^\epsilon(m-1))\phi^\epsilon_{ij}(m-1)X^\epsilon_{ij}(m-1)\Big).
\end{aligned} \tag{52}$$

We can then write, using the fact that $z_{ij}^{\epsilon}(t) = X_{ij}^{\epsilon}(\lfloor \frac{t}{\epsilon} \rfloor)$ for all $t \geq 0$,

$$\begin{aligned} z_{ij}^{\epsilon}(t) &= z_{ij}^{\epsilon}(0) + \epsilon \sum_{m=1}^{\lfloor \frac{t}{\epsilon} \rfloor} M_{ij}^{\epsilon}(m) + \epsilon \sum_{m=1}^{\lfloor \frac{t}{\epsilon} \rfloor} N_{ij}^{\epsilon}(m) \\ &\quad + \epsilon \sum_{m=1}^{\lfloor \frac{t}{\epsilon} \rfloor} \Big(\zeta_i(X^{\epsilon}(m-1)) \phi_{ij}^{\epsilon}(m-1) D_{ij}(X^{\epsilon}(m-1)) \Big) \\ &\quad - \epsilon \sum_{m=1}^{\lfloor \frac{t}{\epsilon} \rfloor} P_{ij}^{\epsilon}(m) - \epsilon \sum_{m=1}^{\lfloor \frac{t}{\epsilon} \rfloor} Q_{ij}^{\epsilon}(m) \\ &\quad - \epsilon \sum_{m=1}^{\lfloor \frac{t}{\epsilon} \rfloor} \Big(\zeta_i(X^{\epsilon}(m-1)) \phi_{ij}^{\epsilon}(m-1) X_{ij}^{\epsilon}(m-1) \Big), \end{aligned}$$

where $M_{ij}^{\epsilon}(m), N_{ij}^{\epsilon}(m), P_{ij}^{\epsilon}(m)$, and $Q_{ij}^{\epsilon}(m)$ refer to the corresponding quantities in the equation (52) above.

We introduce the following quantities: $G_1^{\epsilon}(t) = \epsilon \sum_{m=1}^{\lfloor \frac{t}{\epsilon} \rfloor} M_{ij}^{\epsilon}(m), G_2^{\epsilon}(t) = \epsilon \sum_{m=1}^{\lfloor \frac{t}{\epsilon} \rfloor} N_{ij}^{\epsilon}(m), G_3^{\epsilon}(t) = \epsilon \sum_{m=1}^{\lfloor \frac{t}{\epsilon} \rfloor} P_{ij}^{\epsilon}(m)$, and $G_4^{\epsilon}(t) = \epsilon \sum_{m=1}^{\lfloor \frac{t}{\epsilon} \rfloor} Q_{ij}^{\epsilon}(m)$. Now the term

$$\epsilon \sum_{m=1}^{\lfloor \frac{t}{\epsilon} \rfloor} \zeta_i(X^{\epsilon}(m-1)) \phi_{ij}^{\epsilon}(m-1) \Big(D_{ij}(X^{\epsilon}(m-1)) - X_{ij}^{\epsilon}(m-1) \Big) = \int_0^t F_{ij}(z^{\epsilon}(u)) du,$$

when t is an integral multiple $n\epsilon$ of ϵ, and is an approximation otherwise, the approximation error vanishing when $\epsilon \to 0$.

Hence, in order to show (46), from equation (52) and the remark above, we can see that all that we need to show is that

$$\begin{aligned} &\lim_{\epsilon \to 0} E[h(z^{\epsilon}(s_1), z^{\epsilon}(s_2), \ldots, z^{\epsilon}(s_p)) \Big((G_1^{\epsilon}(t+\tau) - G_1^{\epsilon}(t)) \\ &\quad + (G_2^{\epsilon}(t+\tau) - G_2^{\epsilon}(t)) - (G_3^{\epsilon}(t+\tau) - G_3^{\epsilon}(t)) - (G_4^{\epsilon}(t+\tau) - G_4^{\epsilon}(t)) \Big)] = 0. \end{aligned}$$

We show that each of the summands in the expectation above tend to zero as $\epsilon \to 0$, i.e., for $i = 1, 2, 3, 4, \lim_{\epsilon \to 0} E[h(z^{\epsilon}(s_1), z^{\epsilon}(s_2), \ldots, z^{\epsilon}(s_p)) \Big(G_i^{\epsilon}(t+\tau) - G_i^{\epsilon}(t) \Big)] = 0$.

We start with showing that $\lim_{\epsilon\to 0} E[h(z^\epsilon(s_1), z^\epsilon(s_2), \ldots, z^\epsilon(s_p))\Big(G_1^\epsilon(t+\tau) - G_1^\epsilon(t)\Big)] = 0$. Now,

$$E[h(z^\epsilon(s_1), z^\epsilon(s_2), \ldots, z^\epsilon(s_p))\Big(G_1^\epsilon(t+\tau) - G_1^\epsilon(t)\Big)] =$$

$$E[h(z^\epsilon(s_1), z^\epsilon(s_2), \ldots, z^\epsilon(s_p))\Big(\epsilon \sum_{m=\lfloor \frac{t}{\epsilon} \rfloor + 1}^{\lfloor \frac{t+\tau}{\epsilon} \rfloor} M_{ij}^\epsilon(m)\Big)].$$

It can be checked that the sequence $\{M_{ij}^\epsilon(m)\}$ is a martingale difference sequence with respect to $\{\mathcal{F}^\epsilon(m-1)\}$ (that is, the sequence $T_{ij}^\epsilon(n) = \sum_{m=1}^n M_{ij}^\epsilon(m)$ is a martingale with respect to the same filtration). It then follows that

$$E[h(z^\epsilon(s_1), z^\epsilon(s_2), \ldots, z^\epsilon(s_p))\Big(G_1^\epsilon(t+\tau) - G_1^\epsilon(t)\Big)] = 0;$$

the result then also holds true when $\epsilon \to 0$.

In a similar manner, we can show that $\lim_{\epsilon\to 0} E[h(z^\epsilon(s_1), \ldots, z^\epsilon(s_p))\Big(G_3^\epsilon(t+\tau) - G_3^\epsilon(t)\Big)] = 0$.

The arguments for showing that the relations, $\lim_{\epsilon\to 0} E[h(z^\epsilon(s_1), \ldots, z^\epsilon(s_p))\Big(G_2^\epsilon(t+\tau) - G_2^\epsilon(t)\Big)] = 0$ and $\lim_{\epsilon\to 0} E[h(z^\epsilon(s_1), \ldots, z^\epsilon(s_p))\Big(G_4^\epsilon(t+\tau) - G_4^\epsilon(t)\Big)] = 0$, hold, are similar in nature. Consequently, we shall discuss in detail the steps for only one of them.

Lets consider the term $\lim_{\epsilon\to 0} E[h(z^\epsilon(s_1), \ldots, z^\epsilon(s_p))\Big(G_2^\epsilon(t+\tau) - G_2^\epsilon(t)\Big)] = 0$. We recall that

$$\begin{aligned} G_2^\epsilon(t+\tau) - G_2^\epsilon(t) &= \epsilon \sum_{m=\lfloor \frac{t}{\epsilon} \rfloor + 1}^{\lfloor \frac{t+\tau}{\epsilon} \rfloor} \Big(E[I_{\{T^\epsilon(m)=i, R_i^\epsilon(\xi_i^\epsilon(m))=j\}}\ \Delta_{ij}^\epsilon(\psi_{ij}^\epsilon(m))/\mathcal{F}^\epsilon(m-1)] \\ &\quad - \zeta_i(X^\epsilon(m-1))\phi_{ij}^\epsilon(m-1)D_{ij}(X^\epsilon(m-1))\Big). \end{aligned} \tag{53}$$

Now, for a scalar $\eta > \epsilon > 0$ (with $\eta < \tau$), the expression on the right hand side of (53) is approximately equal to the following (complicated looking) expression

$$\sum_{j=0}^{\lfloor \frac{\tau}{\eta} \rfloor} \eta \left[\frac{\epsilon}{\eta} \sum_{m=\lfloor \frac{t+j\eta}{\epsilon} \rfloor + 1}^{\lfloor \frac{t+(j+1)\eta}{\epsilon} \rfloor} \Big(E[I_{\{T^\epsilon(m)=i, R_i^\epsilon(\xi_i^\epsilon(m))=j\}} \ \Delta_{ij}^\epsilon(\psi_{ij}^\epsilon(m))/\mathcal{F}^\epsilon(m-1)] \right.$$

$$\left. -\zeta_i(X^\epsilon(m-1))\phi_{ij}^\epsilon(m-1)D_{ij}(X^\epsilon(m-1)) \Big) \right].$$

In the interval $\{\lfloor \frac{t+j\eta}{\epsilon} \rfloor + 1, \ldots, \lfloor \frac{t+(j+1)\eta}{\epsilon} \rfloor\}$, each of the summands above can be written as the sum of the terms

$$E[I_{\{T^\epsilon(m)=i, R_i^\epsilon(\xi_i^\epsilon(m))=j\}} \ \Delta_{ij}^\epsilon(\psi_{ij}^\epsilon(m))/\mathcal{F}^\epsilon(m-1)]$$

$$-\zeta_i(X^\epsilon(\lfloor \frac{t+j\eta}{\epsilon} \rfloor))\phi_{ij}^\epsilon(\lfloor \frac{t+j\eta}{\epsilon} \rfloor)D_{ij}(X^\epsilon(\lfloor \frac{t+j\eta}{\epsilon} \rfloor))$$

and

$$\zeta_i(X^\epsilon(\lfloor \frac{t+j\eta}{\epsilon} \rfloor))\phi_{ij}^\epsilon(\lfloor \frac{t+j\eta}{\epsilon} \rfloor)D_{ij}(X^\epsilon(\lfloor \frac{t+j\eta}{\epsilon} \rfloor))$$

$$-\zeta_i(X^\epsilon(m-1))\phi_{ij}^\epsilon(m-1)D_{ij}(X^\epsilon(m-1)).$$

Choosing an $\eta > 0$ small enough, and noting that $\zeta_i(x)\phi_{ij}D_{ij}(x)$ is assumed to be a continuous function of x (Assumption $(A3)$), and the fact that $\{z^\epsilon(t)\}$ is tight, shows that the latter terms tend to zero in the mean as $\epsilon \to 0$. For the former terms, we just notice that for small $\eta > 0$, the expression

$$\frac{\epsilon}{\eta} \sum_{m=\lfloor \frac{t+j\eta}{\epsilon} \rfloor + 1}^{\lfloor \frac{t+(j+1)\eta}{\epsilon} \rfloor} \Big(E[I_{\{T^\epsilon(m)=i, R_i^\epsilon(\xi_i^\epsilon(m))=j\}} \ \Delta_{ij}^\epsilon(\psi_{ij}^\epsilon(m))/\mathcal{F}^\epsilon(m-1)]$$

$$-\zeta_i(X^\epsilon(\lfloor \frac{t+j\eta}{\epsilon} \rfloor))\phi_{ij}^\epsilon(\lfloor \frac{t+j\eta}{\epsilon} \rfloor)D_{ij}(X^\epsilon(\lfloor \frac{t+j\eta}{\epsilon} \rfloor)) \Big) = 0,$$

tends to zero as ϵ tends to zero, by Assumption $(A2)$.

2.5 Appendix A: ODE approximation for N Parallel Paths Case

The heuristic analysis provided below largely follows Benveniste, Metivier, and Priouret (Section 2.2, Chapter 2) [5]. Let us consider the mean delay estimation scheme given by equation (14). We can write, for a positive integer M,

$$\begin{aligned} X_1(n+M) &= X_1(n) + \epsilon \sum_{k=1}^{M} I_{\{R(n+k)=1\}} \Big(\Delta_1(\psi_1(n+k)) - X_1(n+k-1) \Big), \\ &\vdots \qquad \vdots \\ X_N(n+M) &= X_N(n) + \epsilon \sum_{k=1}^{M} I_{\{R(n+k)=N\}} \Big(\Delta_N(\psi_N(n+k)) - X_N(n+k-1) \Big). \end{aligned} \tag{54}$$

If $\epsilon > 0$ is small enough, the vector $(X_1(n), \ldots, X_N(n))$ can be assumed to have not changed much in the discrete interval $\{n, n+1, \ldots, n+M\}$, and we can write the following approximate equations

$$\begin{aligned} X_1(n+M) &\approx X_1(n) + M\epsilon \Big(P_1(n,M) - Q_1(n,M) X_1(n) \Big), \\ &\vdots \qquad \vdots \\ X_N(n+M) &\approx X_N(n) + M\epsilon \Big(P_N(n,M) - Q_N(n,M) X_N(n) \Big). \end{aligned} \tag{55}$$

Now we try to find approximations to the quantities $Q_i(n,M) = \frac{\sum_{k=1}^{M} I_{\{R(n+k)=i\}}}{M}$ and $P_i(n,M) = \frac{\sum_{k=1}^{M} I_{\{R(n+k)=i\}} \Delta_i(\psi_i(n+k))}{M}$ for $i = 1, \ldots, N$, when the values of the mean delay estimates $X_1(.), \ldots, X_N(.)$ are considered fixed at $X_1(n), \ldots, X_N(n)$, and M is large. Then the routing probability vector $(\phi_1(.), \ldots, \phi_N(.))$, too, can be regarded as essentially constant in the interval $\{n, \ldots, n+M\}$, because the probabilities are continuous functions of the mean delay estimates. The routing

probabilities are then approximately equal to $\phi_i(n) = \frac{(X_i(n))^{-\beta}}{\sum_{j=1}^{N}(X_j(n))^{-\beta}}, i = 1, \ \ldots, \ N$. Now, assuming that M is large enough that a law of large numbers effect takes over, the average $\frac{\sum_{k=1}^{M} I_{\{R(n+k)=i\}}}{M}$, which is the fraction of ant packets that have arrived at destination via Q_i when the routing probabilities are $\phi_i(n)$, can be approximated by $\phi_i(n)$. With the routing probabilities fixed, the delay processes $\Delta_i(.)$, can converge to a stationary distribution, the mean under stationarity being denoted by $D_i(X_1(n), \ldots, X_N(n))$. The quantities $\frac{\sum_{k=1}^{M} I_{\{R(n+k)=i\}} \Delta_i(\psi_i(n+k))}{M}$ can then be approximated by $\phi_i(n).D_i(X_1(n), \ldots, X_N(n))$. Note that $D_i(X_1(n), \ldots, X_N(n))$ are the mean waiting times as seen by ant packets. Employing the approximations as described above, we notice from (15) that the evolution of the vector $(X_1(n), \ldots, X_N(n))$ resembles that of a discrete-time approximation to the following ODE system when ϵ is small enough,

$$\begin{aligned}
\frac{dz_1(t)}{dt} &= \frac{(z_1(t))^{-\beta}\Big(D_1(z_1(t), \ldots, z_N(t)) - z_1(t)\Big)}{\sum_{j=1}^{N}(z_k(t))^{-\beta}}, \\
&\vdots \qquad \vdots \\
\frac{dz_N(t)}{dt} &= \frac{(z_N(t))^{-\beta}\Big(D_N(z_1(t), \ldots, z_N(t)) - z_N(t)\Big)}{\sum_{j=1}^{N}(z_k(t))^{-\beta}},
\end{aligned} \tag{56}$$

with the set of initial conditions $z_1(0) = x_1, \ldots, z_N(0) = x_N$.

2.6 Appendix B

We show in this appendix that for the N parallel paths case, $\frac{d\phi_1^*(\beta)}{d\beta} > 0$, when $C_1 > C_i, i = 2, \ldots, N$. Notice that then we have (as in Section 2.3.1.4)

$$\phi_1^*(\beta) > \phi_i^*(\beta), \quad i = 2, \ldots, N,$$

and consequently that

$$D(\phi_1^*(\beta), C_1) < D(\phi_i^*(\beta), C_i), \quad i = 2, \ldots, N. \tag{57}$$

From the equation $\phi_1^*(\beta) + \phi_2^*(\beta) + \cdots + \phi_N^*(\beta) = 1$, we have

$$\frac{d\phi_1^*(\beta)}{d\beta} = -\Big(\frac{d\phi_2^*(\beta)}{d\beta} + \cdots + \frac{d\phi_N^*(\beta)}{d\beta}\Big). \tag{58}$$

Differentiating with respect to β the equation $\phi_1^*(\beta)[D(\phi_1^*(\beta), C_1)]^\beta = \phi_2^*(\beta)[D(\phi_2^*(\beta), C_2)]^\beta$, we have

$$\begin{aligned}
&\frac{d\phi_1^*(\beta)}{d\beta}\left([D(\phi_1^*(\beta), C_1)]^\beta + \beta\ .\phi_1^*(\beta)\ .[D(\phi_1^*(\beta), C_1)]^\beta\ .\frac{\frac{dD(\phi_1^*, C_1)}{d\phi_1^*}}{D(\phi_1^*(\beta), C_1)}\right) \\
&\qquad +\phi_1^*(\beta)\ .[D(\phi_1^*(\beta), C_1)]^\beta\ .\log D(\phi_1^*(\beta), C_1) \quad = \\
&\frac{d\phi_2^*(\beta)}{d\beta}\left([D(\phi_2^*(\beta), C_2)]^\beta + \beta\ .\phi_2^*(\beta)\ .[D(\phi_2^*(\beta), C_2)]^\beta\ .\frac{\frac{dD(\phi_2^*, C_2)}{d\phi_2^*}}{D(\phi_2^*(\beta), C_2)}\right) \\
&\qquad +\phi_2^*(\beta)\ .[D(\phi_2^*(\beta), C_2)]^\beta\ .\log D(\phi_2^*(\beta), C_2).
\end{aligned} \tag{59}$$

Differentiating each of the equations $\phi_1^*(\beta)[D(\phi_1^*(\beta), C_1)]^\beta = \phi_i^*(\beta)[D(\phi_i^*(\beta), C_i)]^\beta, i = 3, \ldots, N$, we obtain $N - 2$ similar equations as equation (59) above. Adding all the equations we obtain the following relation

$$\begin{aligned}
&(N-1)\frac{d\phi_1^*(\beta)}{d\beta}\left([D(\phi_1^*(\beta), C_1)]^\beta + \beta\ .\phi_1^*(\beta)\ .[D(\phi_1^*(\beta), C_1)]^\beta\ .\frac{\frac{dD(\phi_1^*, C_1)}{d\phi_1^*}}{D(\phi_1^*(\beta), C_1)}\right) \\
&\qquad +(N-1)\phi_1^*(\beta)\ .[D(\phi_1^*(\beta), C_1)]^\beta\ .\log D(\phi_1^*(\beta), C_1) \quad = \\
&\sum_{i=2}^{N}\left(\frac{d\phi_i^*(\beta)}{d\beta}\Big([D(\phi_i^*(\beta), C_i)]^\beta + \beta\ .\phi_i^*(\beta)\ .[D(\phi_i^*(\beta), C_i)]^\beta\ .\frac{\frac{dD(\phi_i^*, C_i)}{d\phi_i^*}}{D(\phi_i^*(\beta), C_i)}\Big)\right. \\
&\qquad \left. +\phi_i^*(\beta)\ .[D(\phi_i^*(\beta), C_i)]^\beta\ .\log D(\phi_i^*(\beta), C_i)\right).
\end{aligned} \tag{60}$$

Now let

$$\Gamma = \min_{i=2,\ldots,N}\left([D(\phi_i^*(\beta), C_i)]^\beta + \beta\ .\phi_i^*(\beta)\ .[D(\phi_i^*(\beta), C_i)]^\beta\ .\frac{\frac{dD(\phi_i^*, C_i)}{d\phi_i^*}}{D(\phi_i^*(\beta), C_i)}\right).$$

Γ is positive, because each of the terms in the minimum above is positive under our assumptions. Consequently, we can write

$$(N-1)\frac{d\phi_1^*(\beta)}{d\beta}\left([D(\phi_1^*(\beta),C_1)]^\beta+\beta\ .\phi_1^*(\beta)\ .[D(\phi_1^*(\beta),C_1)]^\beta\ .\frac{\frac{dD(\phi_1^*,C_1)}{d\phi_1^*}}{D(\phi_1^*(\beta),C_1)}\right)$$
$$+(N-1)\phi_1^*(\beta)\ .[D(\phi_1^*(\beta),C_1)]^\beta\ .\log D(\phi_1^*(\beta),C_1)\ \geq$$
$$\Gamma\left(\frac{d\phi_2^*(\beta)}{d\beta}+\cdots+\frac{d\phi_N^*(\beta)}{d\beta}\right)+$$
$$\sum_{i=2}^{N}\phi_i^*(\beta)\ .[D(\phi_i^*(\beta),C_i)]^\beta\ .\log D(\phi_i^*(\beta),C_i). \quad (61)$$

Using equation (58) and transposing terms we obtain

$$\frac{d\phi_1^*(\beta)}{d\beta}\left((N-1)[D(\phi_1^*(\beta),C_1)]^\beta+\beta\ .(N-1)\ .\phi_1^*(\beta)\ .[D(\phi_1^*(\beta),C_1)]^\beta\ .\frac{\frac{dD(\phi_1^*,C_1)}{d\phi_1^*}}{D(\phi_1^*(\beta),C_1)}\right.$$
$$\left.+\ \Gamma\right)\geq\sum_{i=2}^{N}\phi_i^*(\beta)\ .[D(\phi_i^*(\beta),C_i)]^\beta\ .\log\frac{D(\phi_i^*(\beta),C_i)}{D(\phi_1^*(\beta),C_1)}.$$

We observe now that the coefficient of $\frac{d\phi_1^*(\beta)}{d\beta}$ on the left hand side of the inequality is positive. Also, using the fact that $\phi_1^*(\beta)[D(\phi_1^*(\beta),C_1)]^\beta=\cdots=\phi_N^*(\beta)[D(\phi_N^*(\beta),C_N)]^\beta$ the expression on the right hand side can be written as

$$\phi_1^*(\beta)\ .[D(\phi_1^*(\beta),C_1)]^\beta\ .\sum_{i=2}^{N}\log\frac{D(\phi_i^*(\beta),C_i)}{D(\phi_1^*(\beta),C_1)},$$

and then using the relation (57), we have the desired result, $\frac{d\phi_1^*(\beta)}{d\beta}>0$.

Chapter 3

An Optimal Distributed Routing Algorithm using Dual Decomposition Techniques

In this chapter we consider an optimization based approach to the routing problem. In Section 3.1 we provide a formulation of the routing problem as an optimization problem. The cost of the optimization problem is a measure of congestion in the network and the constraints of the problem are the flow balance relations at the network nodes. We consider separately the single commodity and the multi-commodity versions of the problem, in Sections 3.2 and 3.3, respectively. For both the versions, we provide distributed algorithms that converge to the optimal routing solutions, and we also discuss the various properties of the optimal routing solutions.

3.1 General Formulation of the Routing Problem

We now describe in brief our formulation. Let $r_i^{(k)} \geq 0$ denote the rate of input traffic entering the network at node i and destined for node k [1]. The flow on link (i,j) corresponding to packets bound for destination k is denoted by $f_{ij}^{(k)}$. The total flow on link (i,j) is denoted by F_{ij} and is given by $F_{ij} = \sum_{k \in \mathcal{N}} f_{ij}^{(k)}$.

[1] The arrival process is usually modeled as a stationary stochastic process, and $r_i^{(k)}$ then refers to the time average rate of the process.

All packet flows having the same node as destination are said to belong to one commodity, irrespective of where they originate. Let C_{ij} denote the total capacity of link (i,j). At node i, for every outgoing link (i,j), there is an associated queue which is assumed to be of infinite size. Let $D_{ij}(F_{ij})$ denote the average packet delay in the queue when the total traffic flow through (i,j) is F_{ij}, with F_{ij} satisfying the constraint $0 \leq F_{ij} < C_{ij}$. (Quantities $F_{ij}, r_i^{(k)}, f_{ij}^{(k)}$ and C_{ij} are all expressed in the same units of packets/sec.)

Let $\mathbf{f}$ denote the (column) vector of commodity link flows $f_{ij}^{(k)}, (i,j) \in \mathcal{L}, k \in \mathcal{N}$, in the network. We consider the following optimal routing problem:

Problem (A) : Minimize the (separable) cost function

$$G(\mathbf{f}) = \sum_{(i,j)\in\mathcal{L}} G_{ij}(F_{ij}) = \sum_{(i,j)\in\mathcal{L}} \int_0^{F_{ij}} u[D_{ij}(u)]^\beta du,$$

subject to

$$\sum_{j:(i,j)\in\mathcal{L}} f_{ij}^{(k)} = r_i^{(k)} + \sum_{j:(j,i)\in\mathcal{L}} f_{ji}^{(k)}, \ \forall i, k \neq i, \tag{1}$$

$$f_{ij}^{(k)} \geq 0, \ \ \forall (i,j) \in \mathcal{L}, \ k \neq i, \tag{2}$$

$$f_{ij}^{(i)} = 0, \ \ \forall (i,j) \in \mathcal{L}, \tag{3}$$

$$F_{ij} = \sum_k f_{ij}^{(k)}, \ \ \forall (i,j) \in \mathcal{L}, \tag{4}$$

$$\text{with} \quad 0 \leq F_{ij} < C_{ij}, \quad \forall (i,j) \in \mathcal{L}.$$

In the work on convergence of Ant-Based Routing Algorithms (Chapter 2 of the thesis), we showed, for a simple network involving N parallel links between a source-destination pair of nodes, that the equilibrium routing flows were such that they solved an optimization problem with a similar cost function and with

similar capacity constraints as above. The scheme also yielded a multipath routing solution. It is natural to look for a generalization for the network case that has similar attractive properties. We shall see, using dual decomposition techniques, that the solution to our (optimization) Problem (A) is also a multipath routing solution, which can be implemented in a distributed manner by the nodes in the network. Our cost function is related to the network-wide congestion as measured by the link delays, and is small if the link delays are small. (Other cost functions, related to network-wide congestion, have been used in the literature: in Gallager [28] and Bertsekas, Gallager and Gafni [10] it is of the form (in our notation) $D(\mathbf{f}) = \sum_{(i,j)\in\mathcal{L}} D_{ij}(F_{ij})$; and in the Wardrop routing formulation (see Kelly [34]) it is of the form $W(\mathbf{f}) = \sum_{(i,j)\in\mathcal{L}} \int_0^{F_{ij}} D_{ij}(u)du$.) The parameter β in our cost is a constant positive integer that can be used to change the overall optimal flow pattern in the network. Roughly speaking, a low value of β results in the flows being more 'uniformly distributed' on the paths, whereas a high value of β tends to make the flows more concentrated on links lying on higher capacity paths.

Constraints (1) above are the per-commodity flow balance equations at the network nodes (flow out of the node = flow into the node), and constraints (3) express the fact that once a packet reaches its intended destination it is not routed back into the network. The optimization is over the set of link flow vectors $\mathbf{f}$.

3.2 The Single Commodity Problem : Formulation and Analysis

We consider in this section the single commodity problem, which involves routing of flows to a common destination node, which we label as D. We restate the problem for this special case in the following manner:

$$\textbf{Problem } (B) : \text{Minimize } G(\mathbf{F}) = \sum_{(i,j)\in\mathcal{L}} G_{ij}(F_{ij}) = \sum_{(i,j)\in\mathcal{L}} \int_0^{F_{ij}} u[D_{ij}(u)]^{\beta} du,$$

subject to

$$\sum_{j:(i,j)\in\mathcal{L}} F_{ij} = r_i + \sum_{j:(j,i)\in\mathcal{L}} F_{ji}, \ \forall i \in \mathcal{N}, \tag{5}$$

$$F_{Dj} = 0, \quad \text{for } (D,j) \in \mathcal{L}, \tag{6}$$

$$\text{with} \quad 0 \le F_{ij} < C_{ij}, \quad \forall (i,j) \in \mathcal{L}.$$

r_i is the incoming rate for traffic arriving at node i, and destined for D. The optimization is over the set of link flow vectors $\mathbf{F}$, whose components are the individual link flows $F_{ij}, (i,j) \in \mathcal{L}$. As usual, equations (5) give the flow balance equations at every node and equations (6) refer to the fact that once a packet reaches D, it is not re-routed back into the network.

We use a dual decomposition technique of Bertsekas [7] to develop a distributed primal-dual algorithm that solves the above-stated optimal routing problem. We carry out our analysis under the following fairly natural assumptions. These assumptions are also used, almost verbatim, for the multicommodity version of the problem in Section 3.3.

Assumptions:

$(A1)$ $D_{ij}(u)$ is a nondecreasing, continuously differentiable, positive real-valued function of u, defined over the interval $[0, C_{ij})$.

$(A2)$ $\lim_{u\uparrow C_{ij}} D_{ij}(u) = +\infty$. Also, $\lim_{F\uparrow C_{ij}} \int_0^F u[D_{ij}(u)]^\beta du = +\infty$.

$(A3)$ There exists at least one feasible solution of the primal problem (B).

Assumption $(A1)$ is a reasonable one, because when the flow u through a link increases, the average queueing delay (which is a function of the flow u) increases too. The first part of Assumption $(A2)$ is satisfied for most queueing delay models of interest. We also require the second part to hold to ensure existence of an optimal solution (see Lemma 3 in the Appendix to the chapter). This holds, for example, when the growth rate of the delay D_{ij}, as a function of the flow, is "fast enough" when the flow is close to the capacity C_{ij}. It is not difficult to check, by straightforward integration, that for the delay function of the $M/M/1$ queue $D_{ij}(u) = \frac{1}{C_{ij}-u}$, the condition holds ($\beta$ being a positive integer). Assumption $(A3)$ implies that there exists a link flow pattern in the network such that the incoming traffic can be accommodated without the flow exceeding the capacity in any link. Then we can check that the function $G_{ij}(F_{ij})$ is convex on $[0, C_{ij})$, and so the objective function of our optimization problem is a convex function.

We start the analysis by attaching prices (Lagrange multipliers) $p_i \in \mathbb{R}$, to the flow balance equations (5) and form the Lagrangian function $L(\mathbf{F}, \mathbf{p})$

$$L(\mathbf{F}, \mathbf{p}) = \sum_{(i,j)\in\mathcal{L}} G_{ij}(F_{ij}) + \sum_{i\in\mathcal{N}} p_i \Big(\sum_{j:(j,i)\in\mathcal{L}} F_{ji} + r_i - \sum_{j:(i,j)\in\mathcal{L}} F_{ij} \Big),$$

a function of the (column) price vector $\mathbf{p}$ and the link flow vector $\mathbf{F}$. We can rearrange the Lagrangian to obtain the following convenient form

$$L(\mathbf{F}, \mathbf{p}) = \sum_{(i,j)\in\mathcal{L}} \Big(G_{ij}(F_{ij}) - (p_i - p_j)F_{ij} \Big) + \sum_{i\in\mathcal{N}} p_i r_i. \tag{7}$$

Using the Lagrangian, the dual function $Q(\mathbf{p})$ can be found by

$$Q(\mathbf{p}) = \inf L(\mathbf{F}, \mathbf{p}),$$

where the infimum is taken over all vectors $\mathbf{F}$, such that the components F_{ij} satisfy $0 \leq F_{ij} < C_{ij}$.

From the form (7) for the Lagrangian function, we can immediately see that

$$\begin{aligned} Q(\mathbf{p}) &= \sum_{(i,j)\in\mathcal{L}} \inf_{\{F_{ij}:0\leq F_{ij}<C_{ij}\}} \Big(G_{ij}(F_{ij}) - (p_i - p_j)F_{ij} \Big) + \sum_{i\in\mathcal{N}} p_i r_i, \\ &= \sum_{(i,j)\in\mathcal{L}} Q_{ij}(p_i - p_j) + \sum_{i\in\mathcal{N}} p_i r_i, \end{aligned} \tag{8}$$

where the function $Q_{ij} : \mathbb{R} \to \mathbb{R}$ is given by $Q_{ij}(p_i - p_j) = \inf_{\{F_{ij}:0\leq F_{ij}<C_{ij}\}} \Big(G_{ij}(F_{ij}) - (p_i - p_j)F_{ij} \Big)$. We can extend the definition of the function G_{ij} to the whole of $\mathbb{R}$, by simply setting it to be $+\infty$ outside $[0, C_{ij})$. Then the function $-Q_{ij}(p_i - p_j) = \sup_{\{F_{ij}\in\mathbb{R}\}} \Big((p_i - p_j)F_{ij} - G_{ij}(F_{ij}) \Big)$ is just the conjugate or the Legendre transform of the function G_{ij}.

The dual optimization problem is

$$\text{Maximize } Q(\mathbf{p})$$

$$\text{subject to no constraints on } \mathbf{p} \text{ (i.e., } \mathbf{p} \in \mathbb{R}^{|\mathcal{N}|}).$$

The dual function is a concave function and the dual optimization problem is a convex optimization problem. According to our Assumption $(A3)$ and from the

fact that $G_{ij}(F_{ij})$ is differentiable for every F_{ij} in $[0, C_{ij})$, with the derivative being $G'_{ij}(F_{ij}) = F_{ij}[D_{ij}(F_{ij})]^{\beta}$, there exists a regular [2] primal feasible solution to our primal problem, Problem (B). Then, by Proposition 9.3 of Bertsekas [7], if $\mathbf{F}^*$ is an optimal solution of the primal problem, there exists an optimal solution $\mathbf{p}^*$ of the dual problem that satisfies the following Complementary Slackness (CS) conditions together with $\mathbf{F}^*$

$$G_{ij}(F^*_{ij}) - (p^*_i - p^*_j)F^*_{ij} = \inf_{\{F_{ij}:0\leq F_{ij}<C_{ij}\}} \Big(G_{ij}(F_{ij}) - (p^*_i - p^*_j)F_{ij} \Big), \quad \forall (i,j) \in \mathcal{L}. \tag{9}$$

Also, by Proposition 9.4 of Bertsekas [7], the optimal primal and dual costs are equal - that is, the duality gap is zero [3]. Consider the minimization problem in the CS-condition (9) (for each link (i,j))

$$\text{Minimize } G_{ij}(F_{ij}) - (p_i - p_j)F_{ij} = \textstyle\int_0^{F_{ij}} u[D_{ij}(u)]^{\beta} du - (p_i - p_j)F_{ij}$$
$$\text{subject to } 0 \leq F_{ij} < C_{ij}.$$

The second derivative of G_{ij} is $G''_{ij}(F_{ij}) = [D_{ij}(F_{ij})]^{\beta} + \beta F_{ij}[D_{ij}(F_{ij})]^{\beta-1} D'_{ij}(F_{ij})$. Under our Assumption $(A1)$, $G_{ij}(F_{ij})$ is twice continuously differentiable and strictly convex on the interval $[0, C_{ij})$, so that the minimization problems above are all convex optimization problems on convex sets. We can show that for any price vector $\mathbf{p}$ (in particular, for an optimal dual vector $\mathbf{p}^*$), there exists a unique $F_{ij} \in [0, C_{ij})$ (for every (i,j)) which attains the minimum in the above optimization problem (Lemma 3, Appendix).

[2] A flow vector is called regular if for every link (i,j), the left derivative $G^-_{ij}(F_{ij}) < \infty$, and the right derivative $G^+_{ij}(F_{ij}) > -\infty$ [7].

[3] This fact is nontrivial and a proof requires the techniques of monotropic programming [42], [7].

Conditions equivalent to (9) that an optimal primal-dual pair $(\mathbf{F}^*, \mathbf{p}^*)$ must satisfy are given by (for each $(i,j) \in \mathcal{L}$)

$$F_{ij}^*[D_{ij}(F_{ij}^*)]^\beta \;\geq\; p_i^* - p_j^*, \text{ if } F_{ij}^* = 0, \tag{10}$$

$$F_{ij}^*[D_{ij}(F_{ij}^*)]^\beta \;=\; p_i^* - p_j^*, \text{ if } F_{ij}^* > 0. \tag{11}$$

We also make the following observation. Suppose $p_i^* - p_j^* \leq 0$; then because for any $F_{ij} > 0$, $G_{ij}(F_{ij}) - (p_i^* - p_j^*)F_{ij} = \int_0^{F_{ij}} u[D_{ij}(u)]^\beta du - (p_i^* - p_j^*)F_{ij} > G_{ij}(0) - (p_i^* - p_j^*).0 = 0$, $F_{ij}^* = 0$ must then be the unique global minimum. Now, consider the contrapositive of (10) which reads: if $p_i^* - p_j^* > 0$ then $F_{ij}^* > 0$. Thus, if $p_i^* - p_j^* > 0$ then F_{ij}^* is positive, and is given by the solution to the nonlinear equation

$$F_{ij}^*[D_{ij}(F_{ij}^*)]^\beta = p_i^* - p_j^*.$$

Because D_{ij} is a nondecreasing and continuously differentiable function, the above equation has a unique solution for F_{ij}^*.

To summarize, an optimal primal-dual pair $(\mathbf{F}^*, \mathbf{p}^*)$ is such that the following relationships are satisfied for each link (i,j),

$$F_{ij}^* \;=\; 0, \quad \text{if } \; p_i^* - p_j^* \leq 0, \tag{12}$$

$$F_{ij}^*[D_{ij}(F_{ij}^*)]^\beta \;=\; p_i^* - p_j^*, \quad \text{if } \; p_i^* - p_j^* > 0, \tag{13}$$

and in this case $F_{ij}^* > 0$. In analogy with electrical networks, the relations above can be interpreted as providing the 'terminal characteristics' of the 'branch' (i,j). The Lagrange multipliers p_i^* can be thought of as 'potentials' on the nodes, and the flows F_{ij}^* as 'currents' on the links. The branch can be thought of as consisting of an ideal diode in series with a nonlinear current-dependent resistance. The difference

of the 'potentials' or 'voltage' $p_i^* - p_j^*$, when positive, drives the 'current' or flow F_{ij}^* through a nonlinear flow-dependent resistance according to the law defined by (13) [4].

3.2.1 Distributed Solution of the Dual Optimization Problem

We now focus on solving the dual problem using a distributed primal-dual algorithm. We first make a quick remark on the differentiability properties of the dual function $Q(\mathbf{p})$. It can be verified that, for each (i,j) and (j,i), the partial derivatives $\frac{\partial Q_{ij}(p_i - p_j)}{\partial p_i}$ and $\frac{\partial Q_{ji}(p_j - p_i)}{\partial p_i}$ exist for all $p_i \in \mathbb{R}$. Then, at any point $\mathbf{p}$, the partial derivatives $\frac{\partial Q(\mathbf{p})}{\partial p_i}$ all exist and can be easily seen to be given by

$$\frac{\partial Q(\mathbf{p})}{\partial p_i} = \sum_{j:(i,j)\in\mathcal{L}} \frac{\partial Q_{ij}(p_i - p_j)}{\partial p_i} + \sum_{j:(j,i)\in\mathcal{L}} \frac{\partial Q_{ji}(p_j - p_i)}{\partial p_i} + r_i, \quad i \in \mathcal{N}. \tag{14}$$

The gradient vector $\nabla Q(\mathbf{p})$ can thus be evaluated at each point $\mathbf{p}$.

The dual optimization problem can now be solved by the following simple

[4]This analogy with electrical circuit theory helps in developing intuition. It was known to Maxwell (see Bertsekas [7], Rockafellar [42]) for the case of a quadratic cost function, who showed that minimizing the power in the network (which is the sum of the powers in the individual links) subject to Kirchoff's laws for conservation of flow (current) at every node, leads us to an Ohm's law description of the 'terminal characteristics' of each branch. It was exploited by Dennis [20], who suggested that flow optimization problems with separable convex costs can be solved by setting up a network with arcs having terminal characteristics that can be derived in the same way as for our case. Once the network reaches equilibrium (starting from some initial condition), the currents and potentials can be simply 'read off' and are the optimal solutions to the primal and dual optimization problems, respectively. This amounts to solving the flow optimization problem using analog computation.

gradient algorithm starting from an arbitrary initial price vector $\mathbf{p}^0$

$$\mathbf{p}^{n+1} = \mathbf{p}^n + \alpha_n \nabla Q(\mathbf{p}^n), \quad n \geq 0, \tag{15}$$

where $\{\alpha_n\}$ is a suitably chosen step-size sequence that ensures convergence of the gradient algorithm to an optimal dual vector $\mathbf{p}^*$. We now try to simplify the expression (14), and get it into a form that is suitable for computational purposes.

We had shown earlier that the minimum in the expression

$$\inf_{\{F_{ij}:0\leq F_{ij}<C_{ij}\}} \Big(G_{ij}(F_{ij}) - (p_i - p_j)F_{ij} \Big) \equiv Q_{ij}(p_i - p_j)$$

is uniquely attained for each scalar $p_i - p_j$ by the flow F_{ij} which satisfies relations (12) and (13). Let us denote such a flow using the notation $F_{ij}(p_i - p_j)$, emphasizing its functional dependence on the price difference $p_i - p_j$. Then

$$Q_{ij}(p_i - p_j) = G_{ij}(F_{ij}(p_i - p_j)) - (p_i - p_j)F_{ij}(p_i - p_j).$$

Notice that $\frac{\partial Q_{ij}(p_i - p_j)}{\partial p_i} = -F_{ij}(p_i - p_j)$, and that $\frac{\partial Q_{ji}(p_j - p_i)}{\partial p_i} = F_{ji}(p_j - p_i)$, so that

$$\frac{\partial Q(\mathbf{p})}{\partial p_i} = \sum_{j:(j,i)\in\mathcal{L}} F_{ji}(p_j - p_i) - \sum_{j:(i,j)\in\mathcal{L}} F_{ij}(p_i - p_j) + r_i, \quad i \in \mathcal{N}. \tag{16}$$

The right hand side in the above equation can be interpreted to be the surplus flow at node i. The gradient algorithm (15) can now be written down in terms of iterations on the individual components of the price vector $\mathbf{p}$ as follows

$$p_i^{n+1} = p_i^n + \alpha_n \Big(\sum_{j:(j,i)\in\mathcal{L}} F_{ji}(p_j^n - p_i^n) - \sum_{j:(i,j)\in\mathcal{L}} F_{ij}(p_i^n - p_j^n) + r_i \Big), \tag{17}$$

where for each (i, j),

$$F_{ij}(p_i^n - p_j^n) = 0, \quad \text{if } p_i^n - p_j^n \leq 0, \tag{18}$$

$$F_{ij}(p_i^n - p_j^n) > 0, \quad \text{if } p_i^n - p_j^n > 0, \tag{19}$$

and can be determined by solving the equation $F_{ij}[D_{ij}(F_{ij})]^{\beta} = p_i^n - p_j^n$. The relations (17), (18), and (19), can be used as a basis for a distributed algorithm that converges to a solution of the dual optimization problem. We now describe a general, online version of such an algorithm. The algorithm can be initialized with an arbitrary price vector $\mathbf{p}^0$. (Each node can choose a real number as the initial value of its price variable.) Let's suppose that at the start of a typical iteration, the dual vector is $\mathbf{p}$, with each node i having available information of its own price p_i as well as the prices p_j of its neighbor nodes j such that $(i,j) \in \mathcal{L}$. Each node i makes an estimate of the average queueing delay on each of its outgoing links (i,j). This estimate can be made by taking measurements of the packet delays over a time window and taking an average, or by using a 'running' estimator like an exponential averaging estimator. The flows $F_{ij}(p_i - p_j)$ are then determined by using relations (18) and (19), and node i adjusts the flows on its outgoing links to these values. Each node i then broadcasts the updated flow values to its neighbors j. In this way, every node i has information regarding the flows $F_{ij}(p_i - p_j)$ and $F_{ji}(p_j - p_i)$ on the links (i,j) and (j,i), respectively. Node i can then use (17) to update its own dual variable p_i, and broadcasts to all the neighbor nodes this updated value. A fresh iteration can now commence with these updated dual variables. The general algorithm, as described, is adaptive and the updates of the dual variables and the flow variables in general take place asynchronously. As in [41], the outgoing flow on a link depends on the inverse of the estimated queueing delay on that link.

Although we have described a general, asynchronous, distributed algorithm for the dual problem, we shall discuss convergence only for the special case of the

synchronous version as given by the equations (17), (18), and (19), with 'perfect' ('noiseless') measurements of the delays. We can view the gradient algorithm (15) as a special case of a subgradient algorithm, and employ the results of convergence analysis (see, for example, Bertsekas, Nedic, and Ozdaglar [11]) for the latter algorithm. For simplicity, we confine ourselves to a discussion of the constant step-size case - $\alpha_n = \alpha$, for some small, positive α. The central result is that, if the gradient vector $\nabla Q(\mathbf{p})$ has a bounded norm (that is, $||\nabla Q(\mathbf{p})|| \leq G$, for some constant G, and for all $\mathbf{p}$), then for a small positive number h,

$$Q(\mathbf{p}^*) - \lim_{n\to\infty} Q^n < h, \tag{20}$$

where Q^n is the 'best' value found till the n-th iteration, i.e., $Q^n = \max(Q(\mathbf{p}^0), Q(\mathbf{p}^1), \ldots, Q(\mathbf{p}^n))$. The number h is a function of the step-size α, and decreases with it, and in fact decreases to zero as α decreases to zero. In our case, for any $\mathbf{p}$, the partial derivatives

$$\Big|\frac{\partial Q(\mathbf{p})}{\partial p_i}\Big| = \Big|\sum_{j:(j,i)\in\mathcal{L}} F_{ji}(p_j - p_i) - \sum_{j:(i,j)\in\mathcal{L}} F_{ij}(p_i - p_j) + r_i\Big| \leq \sum_{j:(j,i)\in\mathcal{L}} C_{ji} + \sum_{j:(i,j)\in\mathcal{L}} C_{ij} + r_i$$

are bounded, and so, the gradient vector $\nabla Q(\mathbf{p})$ is also (uniformly) upper bounded. The convergence result therefore holds in our case (with a constant step-size). Upon convergence, the algorithm yields simultaneously, the optimal dual vector $\mathbf{p}^*$, as well as the optimal flows $F_{ij}(\mathbf{p}^*)$.

3.2.2 Loop Freedom of the Algorithm

Loop freedom is a desirable feature for any routing algorithm because communication resources are wasted if packets are routed in loops through the network.

We now show, for the single commodity case under discussion, that the primal-dual algorithm converges to a set of optimal link flows, $F^*_{ij}, (i,j) \in \mathcal{L}$, which are loop free.

Lemma 1. *An optimal link flow vector* $\mathbf{F}^*$ *is loop free.*

Proof. Suppose that an optimal link flow vector $\mathbf{F}^*$ is such that it forms a loop in the network. Then for some sequence of links $(i_1, i_2), (i_2, i_3), \ldots, (i_n, i_1)$ that form a cycle, there is a positive flow on each of the links : $F^*_{i_1 i_2} > 0, F^*_{i_2 i_3} > 0, \ldots, F^*_{i_n i_1} > 0$. This implies, by relation (11), that

$$p^*_{i_1} - p^*_{i_2} > 0, p^*_{i_2} - p^*_{i_3} > 0, \ldots, p^*_{i_n} - p^*_{i_1} > 0,$$

which is impossible. (None of the nodes $i_1, i_2, \ldots, i_n$ above can be the destination node D, because of the condition (equation (6)) that flows are not re-routed back to the network once they reach the destination.) □

3.2.3 An Example

We consider a simple example network in this subsection. We illustrate the computations and show how the allocation of flows to the links by the routing algorithm changes as the link capacities are changed, and that the optimal flow allocations avoid forming loops in the network.

The network that we consider is shown in Figure 3.1. There are incoming traffic flows at nodes 1 and 2, and the destination node is 4. The incoming traffic rates at nodes 1 and 2 are $r_1 = 6$ and $r_2 = 4$. C_{ij} denotes the capacities of the links

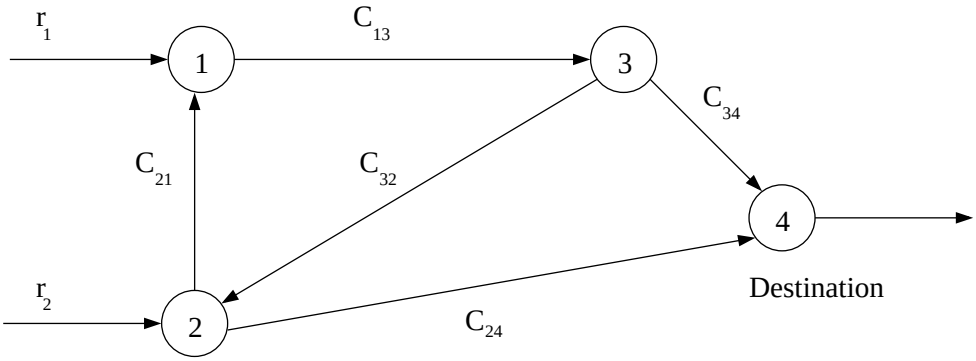


Figure 3.1: The network topology and the traffic inputs : A single commodity example

(i, j). In what follows, we assume that $\beta = 1$ and that the delay functions are of the form $D_{ij}(F_{ij}) = \frac{1}{C_{ij}-F_{ij}}$. (This is the commonly made $M/M/1$ approximation, also referred to as Kleinrock's independence assumption [9].) This delay function satisfies our assumptions $(A1)$ and $(A2)$. We now set up the gradient algorithm.

For this case, the relations (18) and (19) reduce to

$$\begin{aligned} F_{ij}(p_i^n - p_j^n) &= 0, \quad \text{if } p_i^n - p_j^n \leq 0, \\ F_{ij}(p_i^n - p_j^n) &= \frac{(p_i^n - p_j^n)C_{ij}}{1 + p_i^n - p_j^n}, \quad \text{if } p_i^n - p_j^n > 0. \end{aligned}$$

We set up the gradient algorithm with a small constant step-size, i.e., $\alpha_n = \alpha$, for some small positive α, and start with an arbitrary initial dual vector $\mathbf{p}^0$. Each dual vector component is updated using the equation

$$p_i^{n+1} = p_i^n + \alpha\Big(\sum_{j:(j,i)\in\mathcal{L}} F_{ji}(p_j^n - p_i^n) - \sum_{j:(i,j)\in\mathcal{L}} F_{ij}(p_i^n - p_j^n) + r_i \Big),$$

where $F_{ij}(p_i^n - p_j^n)$ and $F_{ji}(p_j^n - p_i^n)$ are computed using the above equations.

The capacities of the links are $C_{13} = 10, C_{21} = 4, C_{32} = 4, C_{34} = 14, C_{24} = 4$, and α is set equal to 0.05. The optimal flows and potentials with this setting are

Table 3.1: Optimal flows and 'potentials' ($C_{13} = 10, C_{21} = 4, C_{32} = 4, C_{34} = 14, C_{24} = 4$)

Optimal link flows	Optimal node potentials
$F^*_{13} = 6.89$	$p^*_1 = 3.19$
$F^*_{21} = 0.89$	$p^*_2 = 3.48$
$F^*_{24} = 3.11$	$p^*_3 = 0.97$
$F^*_{34} = 6.89$	$p^*_4 = 0.00$
$F^*_{32} = 0.00$	

Table 3.2: Optimal flows and 'potentials' ($C_{13} = 10, C_{21} = 4, C_{32} = 4, C_{34} = 14, C_{24} = 8$)

Optimal link flows	Optimal node potentials
$F^*_{13} = 6.00$	$p^*_1 = 2.25$
$F^*_{21} = 0.00$	$p^*_2 = 1.00$
$F^*_{24} = 4.00$	$p^*_3 = 0.75$
$F^*_{34} = 6.00$	$p^*_4 = 0.00$
$F^*_{32} = 0.00$	

tabulated in Table 3.1.

Keeping the capacities of the other links fixed, we can increase the capacity C_{24} until the entire flow that arrives at node 2 goes through the link $(2, 4)$ and no fraction traverses $(2, 1)$. This happens when C_{24} is increased to 8. The optimal flows and potentials that result are tabulated in Table 3.2.

If we now further increase the capacity C_{24}, because the flow coming in at node 3 now sees an additional available path that goes through node 2 to node 4 and has high capacity, a fraction of the flow arriving at 3 goes through this path. The optimal flows and potentials when C_{24} is set at 16 are tabulated in Table 3.3.

Table 3.3: Optimal flows and 'potentials' ($C_{13} = 10, C_{21} = 4, C_{32} = 4, C_{34} = 14, C_{24} = 16$)

Optimal link flows	Optimal node potentials
$F^*_{13} = 6.00$	$p^*_1 = 2.11$
$F^*_{21} = 0.00$	$p^*_2 = 0.41$
$F^*_{24} = 4.67$	$p^*_3 = 0.61$
$F^*_{34} = 5.33$	$p^*_4 = 0.00$
$F^*_{32} = 0.67$	

The optimal flow allocations on links thus vary as the capacities vary relative to each other. In fact the routing algorithm can be seen as accomplishing a form of resource allocation (the resources being the link capacities) that helps relieve congestion in the network. Also the optimal flow allocations are always loop free.

3.2.4 Effect of the parameter β

We had noted in Section 3.1.4 of Chapter 2, for the N parallel links case, that increasing the parameter β enables more of the incoming flow at the source node to be diverted towards the outgoing link with the highest capacity. We also observed, roughly speaking, that lower the value of β, the more 'spread out' is the distribution of flows on the outgoing links. For the N parallel links network, our optimal routing solution is the same as the equilibrium routing solution as considered in Section 2.3.1.4, and so the same considerations and results apply. For the general network single commodity case, we have carried out numerous numerical computations for various examples, and the same general observation seems to be true, viz., that as β increases more and more of the outgoing flow at every node is directed towards

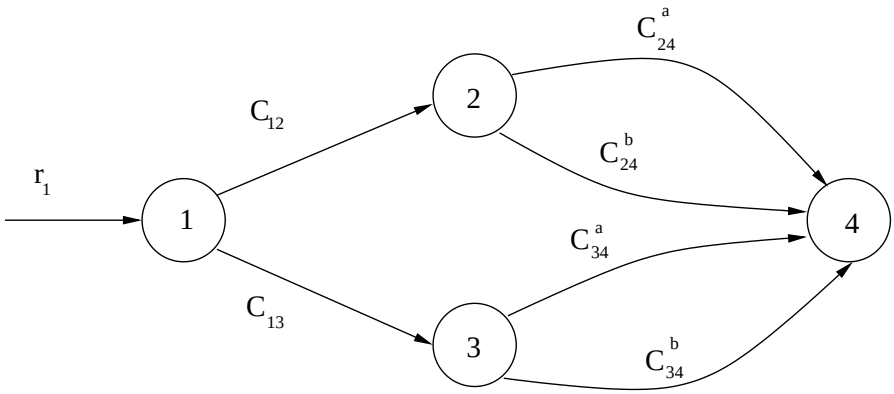


Figure 3.2: Network for illustrating effect of the parameter β

the links that lie on high capacity paths. Unfortunately, we haven't been able to prove this conjecture. In this section we document the results of our computations for various simple networks.

We consider the network shown in Figure 3.2. This network is slightly more complex than the network with parallel links. Routing decisions need to be made at the source node 1 and also at the downstream nodes 2 and 3. There is only one incoming flow r_1 ($r_1 = 7$ units) into the network, which needs to be routed to the destination node 4. We first consider the case when the capacities are given by $C_{12} = 10, C_{13} = 8, C^a_{24} = 8, C^b_{24} = 8, C^a_{34} = 8, C^b_{34} = 8$. Table 3.4 enlists the values of the flows in the various links as β increases. We notice that the flow along the path consisting of links with capacities C_{12} and C^a_{24} increases as the value of β increases. We notice the same trend for the case (Table 3.5) when the capacities of the links are given by $C_{12} = 10, C_{13} = 8, C^a_{24} = 10, C^b_{24} = 10, C^a_{34} = 8, C^b_{34} = 8$, and with the incoming flow given by $r_1 = 7$.

Table 3.4: Optimal flows as β changes ($C_{12} = 10, C_{13} = 8, C_{24}^a = 8, C_{24}^b = 8, C_{34}^a = 8, C_{34}^b = 8$)

β	Optimal link flows
1	$F_{12}^* = 3.8$, $F_{13}^* = 3.2$, $F_{24}^{a\,*} = 1.9$, $F_{24}^{b\,*} = 1.9$, $F_{34}^{a\,*} = 1.6$, $F_{34}^{b\,*} = 1.6$
2	$F_{12}^* = 3.94$, $F_{13}^* = 3.06$, $F_{24}^{a\,*} = 1.97$, $F_{24}^{b\,*} = 1.97$, $F_{34}^{a\,*} = 1.53$, $F_{34}^{b\,*} = 1.53$
3	$F_{12}^* = 4.04$, $F_{13}^* = 2.96$, $F_{24}^{a\,*} = 2.02$, $F_{24}^{b\,*} = 2.02$, $F_{34}^{a\,*} = 1.48$, $F_{34}^{b\,*} = 1.48$
4	$F_{12}^* = 4.10$, $F_{13}^* = 2.90$, $F_{24}^{a\,*} = 2.05$, $F_{24}^{b\,*} = 2.05$, $F_{34}^{a\,*} = 1.45$, $F_{34}^{b\,*} = 1.45$

Table 3.5: Optimal flows as β changes ($C_{12} = 10, C_{13} = 8, C_{24}^a = 10, C_{24}^b = 10, C_{34}^a = 8, C_{34}^b = 8$)

β	Optimal link flows
1	$F_{12}^* = 3.88$, $F_{13}^* = 3.12$, $F_{24}^{a\,*} = 1.94$, $F_{24}^{b\,*} = 1.94$, $F_{34}^{a\,*} = 1.56$, $F_{34}^{b\,*} = 1.56$
2	$F_{12}^* = 4.08$, $F_{13}^* = 2.92$, $F_{24}^{a\,*} = 2.04$, $F_{24}^{b\,*} = 2.04$, $F_{34}^{a\,*} = 1.46$, $F_{34}^{b\,*} = 1.46$
3	$F_{12}^* = 4.18$, $F_{13}^* = 2.82$, $F_{24}^{a\,*} = 2.09$, $F_{24}^{b\,*} = 2.09$, $F_{34}^{a\,*} = 1.41$, $F_{34}^{b\,*} = 1.41$
4	$F_{12}^* = 4.24$, $F_{13}^* = 2.76$, $F_{24}^{a\,*} = 2.12$, $F_{24}^{b\,*} = 2.12$, $F_{34}^{a\,*} = 1.38$, $F_{34}^{b\,*} = 1.38$

3.3 Analysis of the Optimal Routing Problem : The Multicommodity Case

Our approach takes as a starting point a decomposition technique that can be found, for example, in Rockafellar [42]. This technique decomposes a multicommodity flow optimization problem into a set of per link simple nonlinear convex flow problems and a set of per commodity linear network flow problems. We propose a primal-dual approach to solve our optimal routing problem, Problem (A), utilizing this decomposition, aiming in particular to provide a solution that can be implemented in a completely distributed manner by the nodes themselves.

We carry out our analysis under the same assumptions $(A1)$, $(A2)$ and $(A3)$ of Section 3.2, with a slight modification of Assumption $(A3)$, whereby we now require that the primal problem (A) has at least one feasible solution (we still refer to this assumption as Assumption $(A3)$).

We start by attaching Lagrange multipliers $z_{ij} \in \mathbb{R}$ to each of the constraints (4), and construct the Lagrangian function

$$L(\mathbf{f}, \mathbf{z}) = \sum_{(i,j)\in\mathcal{L}} G_{ij}(F_{ij}) + \sum_{(i,j)\in\mathcal{L}} z_{ij}\Big(-F_{ij} + \sum_{k} f_{ij}^{(k)}\Big),$$

where $\mathbf{z}$ is the (column) vector of dual variables $z_{ij}, (i,j) \in \mathcal{L}$. The above equation can be rewritten in the form

$$L(\mathbf{f}, \mathbf{z}) = \sum_{(i,j)\in\mathcal{L}} \Big(G_{ij}(F_{ij}) - z_{ij}F_{ij}\Big) + \sum_{k} \sum_{(i,j)\in\mathcal{L}} z_{ij} f_{ij}^{(k)}. \tag{21}$$

The dual function $Q(\mathbf{z})$, which is a concave function, can then be written down as

$$Q(\mathbf{z}) = \inf L(\mathbf{f}, \mathbf{z}), \tag{22}$$

where the minimization is over all vectors $\mathbf{f}$ satisfying the constraints (1), (2), and (3), and the capacity constraints $0 \leq F_{ij} < C_{ij}, (i,j) \in \mathcal{L}$. The separable form (21) of the Lagrangian function simplifies the computation of $Q(\mathbf{z})$, and we can obtain the following form for $Q(\mathbf{z})$

$$Q(\mathbf{z}) = Q_N(\mathbf{z}) + \sum_k Q_L^{(k)}(\mathbf{z}). \tag{23}$$

$Q_N(\mathbf{z})$ involves the solution of a set (one per link) of simple one-dimensional non-linear optimization problems and is given by

$$Q_N(\mathbf{z}) = \sum_{(i,j)\in\mathcal{L}} \min_{0\leq F_{ij}<C_{ij}} \Big(G_{ij}(F_{ij}) - z_{ij}F_{ij} \Big), \tag{24}$$

and for each commodity k, $Q_L^{(k)}(\mathbf{z})$ involves the solution of a linear network flow optimization problem with the costs associated with links (i,j) being the Lagrange multipliers z_{ij},

$$Q_L^{(k)}(\mathbf{z}) = \min_{\substack{f_{ij}^{(k)}\geq 0,(i,j)\in\mathcal{L}, \\ \sum_j f_{ij}^{(k)} = r_i^{(k)} + \sum_j f_{ji}^{(k)}, i\in\mathcal{N}, \\ f_{ij}^{(i)}=0,(i,j)\in\mathcal{L}}} \sum_{(i,j)\in\mathcal{L}} z_{ij} f_{ij}^{(k)}, , \tag{25}$$

the constraints being the commodity flow balance equations. Note that (25) is a linear program. An interesting interpretation of the above decomposition in terms of marginal costs and the notion of Wardrop equilibrium is provided in Rockafellar [42].

Once the dual function is available, the dual optimization problem can be cast as

$$\text{Maximize } Q(\mathbf{z})$$

$$\text{subject to no constraint on } \mathbf{z} \text{ (i.e., } \mathbf{z} \in \mathbb{R}^{|\mathcal{L}|}).$$

Under our assumptions $(A1)$ and $(A3)$, a regular primal feasible (see [42]) solution to the optimization problem, Problem (A), exists. (This is because, as in Section 3.2, the function $G_{ij}(F_{ij})$ is differentiable, and the derivative $G'_{ij}(F_{ij}) = F_{ij}[D_{ij}(F_{ij})]^{\beta}$ is finite for F_{ij} in $[0, C_{ij})$.) Then it can be shown [42] that strong duality holds - the optimal primal and dual costs are equal [5]. Suppose further that $\mathbf{z}^*$ is an optimal solution to the dual optimization problem, and $\mathbf{f}^*$ is an optimal solution to the primal optimization problem. Then $(\mathbf{f}^*, \mathbf{z}^*)$ solves the set of commodity linear optimization problems (25) (with the z_{ij} being set to z^*_{ij}). Also, for each $(i,j) \in \mathcal{L}$, the optimal total flow $F^*_{ij} = \sum_k f^{(k)*}_{ij}$, and satisfies along with z^*_{ij} the relation (equation (24))

$$G_{ij}(F^*_{ij}) - z^*_{ij}F^*_{ij} = \min_{0 \le F_{ij} < C_{ij}} \Big(G_{ij}(F_{ij}) - z^*_{ij}F_{ij} \Big). \tag{26}$$

Now, for a given dual vector $\mathbf{z}$, let $\mathbf{F}(\mathbf{z})$ and $\mathbf{f}(\mathbf{z})$ be a pair of vectors that attain the minimum in (24) and (25), respectively. The components of $\mathbf{F}(\mathbf{z})$ are the flows $F_{ij}(\mathbf{z})$, and the components of $\mathbf{f}(\mathbf{z})$ are the flows $f^{(k)}_{ij}(\mathbf{z})$. We discuss in the following subsection how to compute $\mathbf{F}(\mathbf{z})$ and $\mathbf{f}(\mathbf{z})$ in a completely distributed manner, given a dual vector $\mathbf{z}$. We shall use this in Section 3.3.2 to develop a distributed primal-dual algorithm that solves the dual optimization problem. Because of strong duality and the comments in the preceding paragraph, we shall have also obtained alongside the optimal flows $\mathbf{f}^*$.

[5] The proof of this fact also can be accomplished by using the techniques of monotropic programming [42].

3.3.1 Flow Vector Computations

For a given dual vector $\mathbf{z}$, we first turn our attention towards the problem of obtaining $\mathbf{F}(\mathbf{z})$. It is clear from the form of the expression in the right hand side of (24) that the computation can be arranged in a distributed manner, with each node i computing the flows $F_{ij}(\mathbf{z})$ on its outgoing links (i,j) by solving the problem

$$\text{Minimize } G_{ij}(F_{ij}) - z_{ij}F_{ij} = \int_0^{F_{ij}} u[D_{ij}(u)]^\beta du - z_{ij}F_{ij},$$

$$\text{subject to } 0 \leq F_{ij} < C_{ij}.$$

Under assumption $(A1)$ this problem is a minimization problem of a strictly convex function over a convex set (arguments as in Section 3.2). Also, Lemma 3 of the Appendix shows that for every $\mathbf{z}$, there exists a unique minimum $F_{ij}(\mathbf{z})$ for the problem. Equivalent (necessary and sufficient) set of conditions that $F_{ij}(\mathbf{z})$ must satisfy are :

$$F_{ij}(\mathbf{z})[D_{ij}(F_{ij}(\mathbf{z}))]^\beta \;\geq\; z_{ij}, \text{ if } F_{ij}(\mathbf{z}) = 0, \tag{27}$$

$$F_{ij}(\mathbf{z})[D_{ij}(F_{ij}(\mathbf{z}))]^\beta \;=\; z_{ij}, \text{ if } F_{ij}(\mathbf{z}) > 0. \tag{28}$$

The relations (27) and (28) imply that

$$F_{ij}(\mathbf{z}) \;=\; 0, \text{ if } z_{ij} \leq 0, \tag{29}$$

$$F_{ij}(\mathbf{z})[D_{ij}(F_{ij}(\mathbf{z}))]^\beta \;=\; z_{ij}, \text{ if } z_{ij} > 0, \tag{30}$$

and in this case $F_{ij}(\mathbf{z}) > 0$ (arguments are similar to those in Section 3.2).

The relations (27) and (28) hold also for an optimal total flow and dual vector pair $(\mathbf{F}(\mathbf{z}^*), \mathbf{z}^*)$. At every node i the optimal total flows on its outgoing links could

be positive or zero, depending on the capacities of the links. We can thus, in general, have a multipath routing solution to our optimal routing problem. The outgoing *total* flow $F_{ij}(\mathbf{z}^*)$, when positive, depends on the inverse of the average link delay.

For a given vector $\mathbf{z}$, we now focus on solving the commodity linear flow optimization problems (25), a linear program. For each commodity k, solving the optimization problem gives the flows $f_{ij}^{(k)}(\mathbf{z}), (i,j) \in \mathcal{L}$. We use the ϵ-relaxation method (Bertsekas and Tsitsiklis [12], Bertsekas and Eckstein [8]), because it can be implemented in a purely distributed manner by the nodes in the network. The method is an algorithmic procedure to solve the dual to the primal linear flow optimization problem and is based on the notion of ϵ-complementary slackness. ϵ-complementary slackness involves a modification of the usual complementary slackness relations of the linear optimization problem by a small amount ϵ. At every iteration, the algorithm changes the dual prices and the incoming and outgoing link flows at every node i, while maintaining ϵ-complementary slackness and improving the value of the dual cost at the same time. We briefly provide an overview in the following paragraphs (for details see, for example, Bertsekas and Tsitsiklis [12]).

Consider the linear network flow problem for commodity k : Minimize the cost $\sum_{(i,j)\in\mathcal{L}} z_{ij} f_{ij}^{(k)}$, subject to the flow balance constraints $\sum_j f_{ij}^{(k)} = r_i^{(k)} + \sum_j f_{ji}^{(k)}$, for each node i, and the constraints $f_{ij}^{(k)} \geq 0$, for each link (i,j). We also add the constraint $f_{ij}^{(k)} \leq C_{ij}$ (which must be satisfied at optimality), which enables us to apply the method without making any modifications. The dual problem is formulated by first attaching Lagrange multipliers (prices) $p_i \in \mathbb{R}$ to the balance

equations at each node i, and forming the Lagrangian $M = \sum_{(i,j)\in\mathcal{L}}\Big(z_{ij}f_{ij}^{(k)} - (p_i - p_j)f_{ij}^{(k)}\Big) + \sum_{i\in\mathcal{N}} r_i^{(k)}p_i$. For a price vector $\mathbf{p}$ and given $\epsilon > 0$, a set of flows and prices satisfies ϵ-complementary slackness conditions if the flows satisfy the capacity constraints, and that

$$f_{ij}^{(k)} < C_{ij} \quad \Longrightarrow \quad p_i - p_j \leq z_{ij} + \epsilon,$$
$$f_{ij}^{(k)} > 0 \quad \Longrightarrow \quad p_i - p_j \geq z_{ij} - \epsilon.$$

The ϵ-relaxation method uses a fixed ϵ, and tries to solve the dual optimization problem using distributed computation. The procedure starts by considering an arbitrary initial price vector $\mathbf{p}^0$, and finds a set of flows on the links such that the flow-price pair satisfies the ϵ-complementary slackness conditions. At each iteration, the surplus at nodes i, $g_i = \sum_j f_{ji}^{(k)} + r_i^{(k)} - \sum_j f_{ij}^{(k)}$, are computed. A node i with positive surplus is chosen. (If all nodes have zero surplus, then the algorithm terminates, because the ϵ-complementary slackness conditions are satisfied and the flow balance conditions, too, are satisfied. The corresponding flow vector $\mathbf{f}$ is optimal.) The surplus g_i is driven to zero at the iterative step, and another flow-price pair satisfying ϵ-complementary slackness is produced. At the same time the dual function value is increased, by changing the i-th price coordinate p_i. At the iteration, except possibly for price p_i at node i, the prices of the other nodes are left unchanged. It can be shown (even for z_{ij} and C_{ij} that are not necessarily integers, which is the case of interest to us; see [12]) that, if ϵ is chosen small enough, the algorithm converges to an optimal flow-price pair [6].

[6]It can be shown [12] that ϵ should chosen to be smaller than the minimum, over all negative-length cycles Y, the ratio $-\frac{\text{Length of cycle Y}}{\text{Number of arcs of Y}}$. The length of the cycle is computed based on

Besides the ϵ-relaxation method, there exist other distributed algorithms, like the auction algorithm, that solve linear flow optimization problems. Any such algorithm can be used to solve the linear flow optimization problems at hand.

3.3.2 Distributed solution of the Dual Optimization Problem

We now focus on solving the dual problem using a distributed primal-dual algorithm. To that end, we first note that the dual function $Q(\mathbf{z})$ is a non-differentiable function of $\mathbf{z}$. This suggests that we can use a subgradient based iterative algorithm to compute an optimal dual vector $\mathbf{z}^*$. We shall see that the computations can be made completely distributed.

We first compute a subgradient for the concave function $Q(\mathbf{z})$ at a point $\mathbf{z}$ (in $\mathbb{R}^{|\mathcal{L}|}$). Recall that a vector $\delta(\mathbf{z})$ is a subgradient of a concave function Q at $\mathbf{z}$ if

$$Q(\mathbf{w}) \leq Q(\mathbf{z}) + \delta(\mathbf{z})^T(\mathbf{w} - \mathbf{z}), \quad \forall \mathbf{w} \in \mathbb{R}^{|\mathcal{L}|}. \tag{31}$$

Recall that for a given vector $\mathbf{z}$, $\mathbf{F}(\mathbf{z})$ and $\mathbf{f}(\mathbf{z})$ denoted a pair of vectors of total flows and commodity flows that attain the minimum in (24) and in (25), respectively. For vectors $\mathbf{z}, \mathbf{w}$, we have

$$\begin{aligned} Q(\mathbf{w}) - Q(\mathbf{z}) &= \sum_{(i,j)\in\mathcal{L}} \Big(G_{ij}(F_{ij}(\mathbf{w})) - w_{ij}F_{ij}(\mathbf{w}) \Big) + \sum_k \sum_{(i,j)\in\mathcal{L}} w_{ij} f_{ij}^{(k)}(\mathbf{w}) \\ &\quad - \sum_{(i,j)\in\mathcal{L}} \Big(G_{ij}(F_{ij}(\mathbf{z})) - z_{ij}F_{ij}(\mathbf{z}) \Big) - \sum_k \sum_{(i,j)\in\mathcal{L}} z_{ij} f_{ij}^{(k)}(\mathbf{z}). \end{aligned}$$

Because $\sum_{(i,j)\in\mathcal{L}} \Big(G_{ij}(F_{ij}(\mathbf{w})) - w_{ij}F_{ij}(\mathbf{w}) \Big) + \sum_k \sum_{(i,j)\in\mathcal{L}} w_{ij} f_{ij}^{(k)}(\mathbf{w}) \leq$

the costs z_{ij} on the edges of the cycle.

$\sum_{(i,j)\in\mathcal{L}}\Big(G_{ij}(F_{ij}(\mathbf{z})) - w_{ij}F_{ij}(\mathbf{z})\Big) + \sum_k \sum_{(i,j)\in\mathcal{L}} w_{ij} f_{ij}^{(k)}(\mathbf{z})$, we have

$$\begin{aligned} Q(\mathbf{w}) - Q(\mathbf{z}) \;\leq\; & \sum_{(i,j)\in\mathcal{L}}\Big(G_{ij}(F_{ij}(\mathbf{z})) - w_{ij}F_{ij}(\mathbf{z})\Big) + \sum_k \sum_{(i,j)\in\mathcal{L}} w_{ij} f_{ij}^{(k)}(\mathbf{z}) \\ & - \sum_{(i,j)\in\mathcal{L}}\Big(G_{ij}(F_{ij}(\mathbf{z})) - z_{ij}F_{ij}(\mathbf{z})\Big) - \sum_k \sum_{(i,j)\in\mathcal{L}} z_{ij} f_{ij}^{(k)}(\mathbf{z}). \end{aligned}$$

and so

$$Q(\mathbf{w}) - Q(\mathbf{z}) \leq \sum_{(i,j)\in\mathcal{L}} \big(w_{ij} - z_{ij}\big)\big(\sum_k f_{ij}^{(k)}(\mathbf{z}) - F_{ij}(\mathbf{z})\big),$$

which shows that a subgradient of Q at $\mathbf{z}$ is the $|\mathcal{L}|$-vector $\delta(\mathbf{z})$ with components $\sum_k f_{ij}^{(k)}(\mathbf{z}) - F_{ij}(\mathbf{z})$.

Consequently, in order to solve the dual optimization problem, we can set up the following subgradient iterative procedure, starting with an arbitrary initial vector of Lagrange multipliers $\mathbf{z}^0$,

$$\mathbf{z}^{n+1} = \mathbf{z}^n + \gamma_n\, \delta(\mathbf{z}^n), \quad n \geq 0, \tag{32}$$

where $\{\gamma_n\}$ is a suitably chosen step-size sequence that ensures convergence of the above subgradient iterations to an optimal dual vector $\mathbf{z}^*$. The vector $\delta(\mathbf{z}^n)$, with components $\sum_k f_{ij}^{(k)}(\mathbf{z}^n) - F_{ij}(\mathbf{z}^n)$, is a subgradient of Q at $\mathbf{z}^n$. In terms of the individual components, the subgradient algorithm (32) can be written as (for each $(i,j) \in \mathcal{L}$)

$$z_{ij}^{n+1} = z_{ij}^n + \gamma_n\Big(\sum_k f_{ij}^{(k)}(\mathbf{z}^n) - F_{ij}(\mathbf{z}^n)\Big). \tag{33}$$

Equation (33) above shows that the subgradient iterative procedure can be implemented in a distributed manner at the various nodes of the network. At each node i, to update the dual variables z_{ij} for the outgoing links (i,j), the quantities that are required are the optimal commodity flows $f_{ij}^{(k)}(\mathbf{z})$ and total flows $F_{ij}(\mathbf{z})$. In

Section 3.3.1 we showed how the above quantities can be computed in a completely distributed manner by every node, given $\mathbf{z}$. $F_{ij}(\mathbf{z}^n)$ can be computed (exactly as in Section 3.2.1) using estimates of average queueing delays on the outgoing links and using (29) and (30). Computation of the flows $F_{ij}(\mathbf{z}^n)$ and $f_{ij}(\mathbf{z}^n)$ require message exchange with neighbor nodes, and local information like estimates of outgoing links' queue lengths. The updated dual variables z_{ij} are broadcast to the neighbor nodes, which utilise this information in the execution of their iterations. In general, the updates of the dual variables and the flows take place asynchronously, so that the algorithm is asynchronous and adaptive.

We now briefly discuss the convergence behavior of the subgradient algorithm (32). As for the single commodity case, we restrict our attention to the synchronous version of the algorithm as given by equation (32), and we consider only the constant step size case here - $\gamma_n = \gamma$, for all n and some small, positive γ. If the subgradient vector is bounded in norm, then the subgradient algorithm converges arbitrarily closely to the optimal point. As in Section 3.2.1, the sense in which convergence takes place is the following : for a small positive number h (which decreases with γ, and in fact, decreases to zero as γ decreases to zero), we have

$$Q(\mathbf{z}^*) - \lim_{n\to\infty} Q^n < h,$$

where Q^n is the 'best' value found till the n-th iteration, i.e., $Q^n = \max(Q(\mathbf{z}^0), \ldots, Q(\mathbf{z}^n))$. In our case, because the commodity flows $f_{ij}^{(k)}(\mathbf{z})$ and total flows $F_{ij}(\mathbf{z})$ are always bounded (because of the capacity constraints), the subgradients $\delta(\mathbf{z})$ are bounded in norm, and the subgradient algorithm converges. Bertsekas, Nedic, and

Ozdaglar [11] contains other attractive (albeit more involved) step-size rules, including diminishing step-size rules, which have more attractive convergence properties. This is an avenue for future exploration. Upon convergence, the algorithm yields simultaneously, the optimal dual vector $\mathbf{z}^*$, as well as the optimal flow vectors $\mathbf{F}(\mathbf{z}^*)$ and $\mathbf{f}(\mathbf{z}^*)$.

3.3.3 Loop Freedom of the Algorithm

In this subsection we show that an optimal flow vector $\mathbf{f}(\mathbf{z}^*)$ is loop free.

Lemma 2. *An optimal flow vector* $\mathbf{f}(\mathbf{z}^*)$ *is loop free.*

Proof. Suppose that an optimal flow vector $\mathbf{f}(\mathbf{z}^*)$ is not loop free. Then for some commodity k, and for some sequence of links $(i_1, i_2), (i_2, i_3), \ldots, (i_n, i_1)$ that form a cycle, there is a positive flow on each of the links : $f^{(k)}_{i_1 i_2}(\mathbf{z}^*) > 0, f^{(k)}_{i_2 i_3}(\mathbf{z}^*) > 0, \ldots, f^{(k)}_{i_n i_1}(\mathbf{z}^*) > 0$. Consequently, for the total flows we have $F_{i_1 i_2}(\mathbf{z}^*) > 0, F_{i_2 i_3}(\mathbf{z}^*) > 0, \ldots, F_{i_n i_1}(\mathbf{z}^*) > 0$. This implies, by relation (28), that

$$z^*_{i_1 i_2} > 0, z^*_{i_2 i_3} > 0, \ldots, z^*_{i_n i_1} > 0.$$

On the other hand, the optimal flows $f^{(k)}_{ij}(\mathbf{z}^*), (i,j) \in \mathcal{L}$, constitute a solution to the linear programming problem: Minimize the cost $\sum_{(i,j)\in\mathcal{L}} z^*_{ij} f^{(k)}_{ij}$, subject to the constraints $\sum_j f^{(k)}_{ij} = r^{(k)}_i + \sum_j f^{(k)}_{ji}, i \in \mathcal{N}$, and the constraints $0 \leq f^{(k)}_{ij} < C_{ij}, (i,j) \in \mathcal{L}$. Attach Lagrange multipliers $p_i \in \mathbb{R}$ to the balance equations at each node i, and form the Lagrangian $N = \sum_{(i,j)\in\mathcal{L}} \Big(z^*_{ij} f^{(k)}_{ij} - (p_i - p_j) f^{(k)}_{ij} \Big) + \sum_{i\in\mathcal{N}} r^{(k)}_i p_i$. An optimal primal-dual vector pair $(\mathbf{f}(\mathbf{z}^*), \mathbf{p}^*)$ satisfies the following Complementary

Slackness conditions (the derivation is similar to the derivation for equations (27) and (28)) for each link (i, j),

$$\begin{aligned} z_{ij}^* &\geq p_i^* - p_j^*, \quad \text{if } f_{ij}^{(k)}(\mathbf{z}^*) = 0 \\ \text{and } z_{ij}^* &= p_i^* - p_j^*, \quad \text{if } f_{ij}^{(k)}(\mathbf{z}^*) > 0. \end{aligned}$$

From the foregoing it is clear that we must have

$$p_{i_1}^* - p_{i_2}^* > 0, p_{i_2}^* - p_{i_3}^* > 0, \ldots, p_{i_n}^* - p_{i_1}^* > 0,$$

which is a contradiction. □

3.3.4 An Illustrative Example

We consider an example network in this section and illustrate the computations. The network consists of eight nodes interconnected by multiple directed links. Figure 3.3 shows the network topology. The numbers beside the links are the capacities of the links. There are multiple sources and multiple sinks of traffic. The rates of input traffic at the sources are given by $r_1^{(6)} = 6, r_1^{(8)} = 8, r_2^{(6)} = 8, r_2^{(8)} = 6, r_2^{(7)} = 10, r_3^{(7)} = 10$. There are three commodities in the network corresponding to the three destinations for the traffic flows in the network. The capacities are such that the network is able to accommodate the incoming traffic to the network.

As in the single commodity example we assume that the delay functions $D_{ij}(F_{ij})$ are explicitly given by the formula $D_{ij}(F_{ij}) = \frac{1}{C_{ij} - F_{ij}}$. We carry out the numerical computations for the case when $\beta = 1$.

We set up the subgradient iterative algorithm (33) starting from an arbitrary

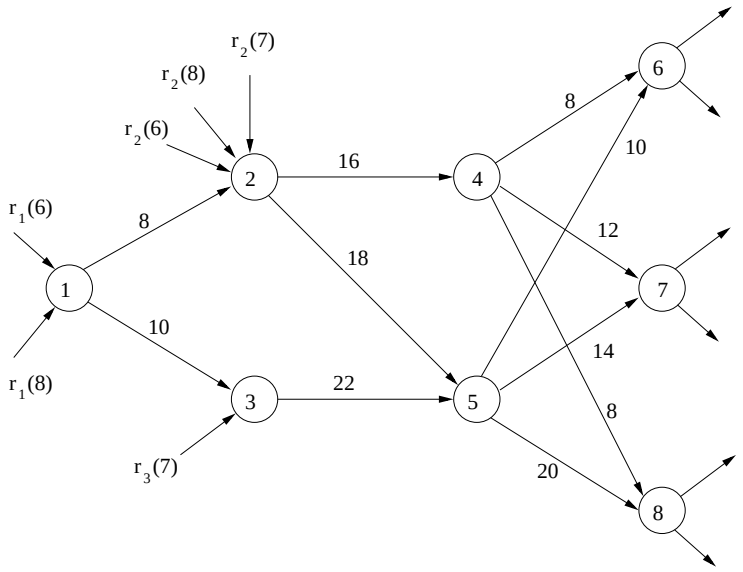


Figure 3.3: The network topology and the traffic inputs : A Multicommodity Example

initial vector $\mathbf{z}^0$

$$z_{ij}^{n+1} = z_{ij}^n + \gamma_n \Big(\sum_k f_{ij}^{(k)}(\mathbf{z}^n) - F_{ij}(\mathbf{z}^n) \Big), \quad (i,j) \in \mathcal{L},$$

where the flow vectors $\mathbf{F}(\mathbf{z}^n)$ and $\mathbf{f}(\mathbf{z}^n)$ are computed as outlined in Section 3.3.1. As we had noted in that section, computing $\mathbf{F}(\mathbf{z}^n)$ translates to satisfying the relations (29) and (30), which in our example are the equations

$$\begin{aligned} F_{ij}(\mathbf{z}^n) &= 0, \text{ if } z_{ij}^n \leq 0, \\ \frac{F_{ij}(\mathbf{z}^n)}{C_{ij} - F_{ij}(\mathbf{z}^n)} &= z_{ij}^n, \text{ if } z_{ij}^n > 0. \end{aligned}$$

The latter equation gives $F_{ij}(\mathbf{z}^n) = \frac{z_{ij}^n C_{ij}}{1+z_{ij}^n}$, a simple expression, showing that the flow is proportional to the capacity.

We use a constant step-size algorithm ($\gamma_n = \gamma$, for all n) with the step-size $\gamma = 0.01$. (This choice of small γ slows down the convergence of the algorithm.

As pointed out in Section 3.3.2, other choices of step-size sequences can potentially improve the speed of convergence.) The ϵ chosen for the ϵ-relaxation method is 0.01.

The subgradient algorithm converges and the optimal flows (upon convergence) are tabulated in Table 4. As in Section 3.2 we note that the optimal routing solution allocates a higher fraction of total incoming flows at every node to outgoing links that lie on paths consisting of higher capacity links.

The optimal routing solution splits the total incoming flow at each node among the outgoing links. The solution also describes how the total flow on each link is split among the commodity flows. We also note that our routing solution is a multipath routing solution. It is well-known [9] that multipath routing solutions improve the overall network performance by avoiding routing oscillations (shortest-path routing solutions, for instance, are known to lead to routing oscillations), and by providing better throughput for incoming connections, while at the same time reducing the average network delay.

Our routing solution is not an end-to-end routing solution, as for example [24]. The control is not effected from the end hosts, but every node i in the network controls both the total as well as the commodity flows on its outgoing links (i, j), using the distributed algorithm that we have described.

3.3.5 Joint Optimal Routing and Rate (Flow) Control

A popular way to treat rate control problems has been to cast them in a utility maximization framework where one maximizes a utility that is a function

Table 3.6: Optimal flows in links

Link (i,j)	Optimal total flow F_{ij}^*	Optimal commodity flows
$(1,2)$	5.77	$f_{12}^{(6)*} = 0$, $f_{12}^{(8)*} = 5.77$
$(1,3)$	8.23	$f_{13}^{(6)*} = 6.00$, $f_{13}^{(8)*} = 2.23$
$(2,4)$	14.31	$f_{24}^{(6)*} = 5.86$, $f_{24}^{(7)*} = 8.19$, $f_{24}^{(8)*} = 0$
$(2,5)$	15.77	$f_{25}^{(6)*} = 2.14$, $f_{25}^{(7)*} = 1.82$, $f_{25}^{(8)*} = 11.77$
$(3,5)$	18.23	$f_{35}^{(6)*} = 6.00$, $f_{35}^{(7)*} = 10.00$, $f_{35}^{(8)*} = 2.23$
$(4,6)$	5.86	$f_{46}^{(6)*} = 5.86$
$(4,7)$	8.19	$f_{47}^{(7)*} = 8.19$
$(4,8)$	0	$f_{48}^{(8)*} = 0$
$(5,6)$	8.23	$f_{56}^{(6)*} = 8.14$
$(5,7)$	11.84	$f_{57}^{(7)*} = 11.82$
$(5,8)$	14.00	$f_{58}^{(8)*} = 14.00$

of the source rates (usually the function is of separable form). The constraints to the problem are formed by considering a routing matrix that represents the interconnections between the nodes of the network (the network topology), and by noting that the sum of the flows along a link cannot exceed the link capacity. We can naturally include a rate control problem in our optimal routing framework in the following manner. We provide a brief outline in this section of the approach, showing that the additions needed are minimal. Using the utility maximization approach alluded to above, we say that a source that succeeds at transmitting at a rate $r_i^{(k)}$ towards a destination k derives a utility of $U_i(r_i^{(k)})$, where U_i is an increasing, twice continuously differentiable strictly concave function. The concavity models a 'law of diminishing returns' - the additional utility (or 'satisfaction') derived by sending an additional unit of traffic decreases with the traffic $r_i^{(k)}$, that is, $\frac{d^2U_i(r_i^{(k)}}{dr_i^{(k)^2}} < 0$. An example of a utility function that satisfies the requirements is $U_i(x) = \log x$.

We pose the joint optimal routing and rate control problem in the following manner. The cost given by $\sum_{(i,j)\in\mathcal{L}} \int_0^{F_{ij}} u\ [D_{ij}(u)]^\beta du - \sum_k \sum_{i\in\mathcal{N}, i\neq k} U_i(r_i^{(k)})$ is to be minimized, with respect to both the set of flows $f_{ij}^{(k)}$ and the rates $r_i^{(k)}$, with the usual constraints given by the flow balance equations, the capacity constraints, and the additional constraint that the rates $r_i^{(k)}$ lie in some given intervals $[0, M_i^{(k)}]$. The utility of sending at the vector of rates $r_i^{(k)}$ is thus of a separable form. Formally we can write the joint optimization problem in the following way

Problem (C): Minimize the (separable) cost function

$$P(\mathbf{f}, \mathbf{r}) = \sum_{(i,j)\in\mathcal{L}} G_{ij}(F_{ij}) - \sum_k \sum_{i\in\mathcal{N}, i\neq k} U_i(r_i^{(k)}) =$$

$$\sum_{(i,j)\in\mathcal{L}} \int_0^{F_{ij}} u[D_{ij}(u)]^{\beta} du - \sum_k \sum_{i\in\mathcal{N}, i\neq k} U_i(r_i^{(k)}),$$

subject to

$$\sum_{j:(i,j)\in\mathcal{L}} f_{ij}^{(k)} = r_i^{(k)} + \sum_{j:(j,i)\in\mathcal{L}} f_{ji}^{(k)}, \ \forall i, k \neq i, \tag{34}$$

$$f_{ij}^{(k)} \geq 0, \ \ \forall (i,j) \in \mathcal{L}, \ k \neq i, \tag{35}$$

$$f_{ij}^{(i)} = 0, \ \ \forall (i,j) \in \mathcal{L}, \tag{36}$$

$$F_{ij} = \sum_k f_{ij}^{(k)}, \ \ \forall (i,j) \in \mathcal{L}, \tag{37}$$

$$r_i^{(k)} \in [0, M_i^{(k)}], \ \ \forall i \in \mathcal{N}, k \neq i, \tag{38}$$

$$\text{with} \quad 0 \leq F_{ij} < C_{ij}, \quad \forall (i,j) \in \mathcal{L}.$$

The assumptions $(A1)$, $(A2)$, and $(A3)$ as stated at the beginning of Section 3.3 remain in force here. The optimization is over the set of all commodity flows $f_{ij}^{(k)}$ and the set of all rates $r_i^{(k)}$. It is a convex optimization problem over a convex set. We proceed, as usual, by attaching Lagrange multipliers $z_{ij} \in \mathbb{R}$ to each of the constraints (37) and form the Lagrangian

$$L(\mathbf{f}, \mathbf{r}, \mathbf{z}) = \sum_{(i,j)\in\mathcal{L}} G_{ij}(F_{ij}) - \sum_k \sum_{i\in\mathcal{N}, i\neq k} U_i(r_i^{(k)}) + \sum_{(i,j)\in\mathcal{L}} z_{ij}\Big(-F_{ij} + \sum_k f_{ij}^{(k)}\Big),$$

where $\mathbf{z}$ is the (column) vector of dual variables $z_{ij}, (i,j) \in \mathcal{L}$. The above equation can be rewritten in the following form

$$L(\mathbf{f}, \mathbf{r}, \mathbf{z}) = \sum_{(i,j)\in\mathcal{L}} \Big(G_{ij}(F_{ij}) - z_{ij}F_{ij}\Big) + \sum_k \sum_{(i,j)\in\mathcal{L}} z_{ij} f_{ij}^{(k)} - \sum_k \sum_{i\in\mathcal{N}, i\neq k} U_i(r_i^{(k)}).$$

The dual function $Q(\mathbf{z})$ is then given by

$$Q(\mathbf{z}) = \min L(\mathbf{f}, \mathbf{r}, \mathbf{z}),$$

where the minimization is over all vectors $\mathbf{f}$, $\mathbf{r}$ satisfying the constraints (34), (35), (36), (38), and the capacity constraints on the total flows. The function $Q(\mathbf{z})$ can be decomposed into the form

$$Q(\mathbf{z}) = Q_N(\mathbf{z}) + \sum_k Q_L^{(k)}(\mathbf{z}),$$

where $Q_N(\mathbf{z})$ can be computed by solving a set of simple nonlinear optimization problems

$$Q_N(\mathbf{z}) = \sum_{(i,j)\in\mathcal{L}} \min_{0\le F_{ij}<C_{ij}} \Big(G_{ij}(F_{ij}) - z_{ij}F_{ij} \Big), \tag{39}$$

and for each commodity k, $Q_L^{(k)}(\mathbf{z})$ can be obtained by solving a set of minimization problems

$$Q_L^{(k)}(\mathbf{z}) = \min_{\substack{f_{ij}^{(k)}\ge 0,(i,j)\in\mathcal{L},\\ \sum_j f_{ij}^{(k)} = r_i^{(k)} + \sum_j f_{ji}^{(k)}, i\in\mathcal{N},\\ r_i^{(k)}\in[0,M_i^{(k)}], i\in\mathcal{N}, k\neq i}} \sum_{(i,j)\in\mathcal{L}} z_{ij} f_{ij}^{(k)} - \sum_{i\in\mathcal{N}, i\neq k} U_i(r_i^{(k)}). \tag{40}$$

Our approach, as for the optimal routing problem, is to solve the following dual optimization problem

$$\text{Maximize } Q(\mathbf{z})$$

$$\text{subject to no constraint on } \mathbf{z} \text{ (i.e., } \mathbf{z} \in \mathbb{R}^{|\mathcal{L}|}).$$

To that end, we first discuss how to solve the minimization problems in the right hand sides of (39) and (40), for a given dual vector $\mathbf{z}$. For given $\mathbf{z}$, let $\mathbf{F}(\mathbf{z})$ be the vector which attains the minimum in (39). Also, let $(\mathbf{f}(\mathbf{z}), \mathbf{r}(\mathbf{z}))$ be the vector pair which attains the minimum in (40). $\mathbf{F}(\mathbf{z})$ can be obtained exactly as described in Section 3.3.1. Consider now the optimization problem on the right hand side of (40). Attaching Lagrange multipliers $p_i \in \mathbb{R}$ to the flow balance equations at the

nodes, we can form the Lagrangian function R which can be written in the following form

$$R = \sum_{(i,j)\in\mathcal{L}} \Big(z_{ij} - p_i + p_j\Big) f_{ij}^{(k)} - \sum_{i\in\mathcal{N}} \Big(U_i(r_i^{(k)}) - p_i r_i^{(k)}\Big). \tag{41}$$

The dual function is the minimum of R subject to the conditions on the links $f_{ij}^{(k)} \geq 0$, and the conditions $r_i^{(k)} \in [0, M_i^{(k)}]$. For a given vector $\mathbf{p}$ of Lagrange multipliers, let $r_i^{(k)}(\mathbf{p})$ be the scalar which attains the maximum in the following problem

$$\begin{aligned} \text{Maximize} \quad & U_i(r_i^{(k)}) - p_i r_i^{(k)} \\ \text{subject to} \quad & r_i^{(k)} \in [0, M_i^{(k)}]. \end{aligned}$$

Under our assumptions on the function U_i there exists a unique solution to this maximization problem.

We now briefly describe modifications to the ϵ-relaxation method of Section 3.3.1 to solve the dual optimization problem. We start with an arbitrary initial price vector $\mathbf{p}^0$, and first find the vector of rates $r_i^{(k)}(\mathbf{p}^0)$. These rates are then used as inputs to the ϵ-relaxation procedure to obtain a new flow-price vector pair. The new prices are used to determine a new vector of rates, which are again fed back as inputs to the ϵ-relaxation procedure. This iterative procedure converges to an optimal flow-price pair, as well as an optimal rate vector. We thus obtain $\mathbf{f}(\mathbf{z})$ and $\mathbf{r}(\mathbf{z})$.

The dual optimization problem can be solved using the subgradient iterative procedure exactly in the same way as has been described in Section 3.3.2. We note that the concave function $Q(\mathbf{z})$ would again have a subgradient at $\mathbf{z}$ that is given by

the $|\mathcal{L}|$-vector $\delta(\mathbf{z})$ with components $\sum_k f_{ij}^{(k)}(\mathbf{z}) - F_{ij}(\mathbf{z})$ (the computations proceed similarly as in Section 3.3.2).

We note thus, that we can incorporate in our framework and solve a joint optimal routing and flow control problem using the same approach. We note that the solution can be again implemented in a distributed manner. Furthermore, we note that the rate control algorithm essentially operates at the source nodes of the network (solving for the rate vectors $r_i^{(k)}(\mathbf{p})$) whereas all the network nodes participate in the implementation of the routing algorithm, which involves determination of the total as well as the commodity flows on the outgoing links. The sources need that the information regarding the prices be made available to them in order to implement the rate control algorithm.

3.4 Appendix

Lemma 3. *Under our Assumptions* $(A1)$ *and* $(A2)$*, there exists a unique solution to the following minimization problem (for any given* w_{ij}*),*

$$\text{Minimize } G_{ij}(F_{ij}) - w_{ij}F_{ij} = \int_0^{F_{ij}} u[D_{ij}(u)]^{\beta} du - w_{ij}F_{ij},$$

$$\text{subject to } 0 \leq F_{ij} < C_{ij}.$$

Proof. For any given w_{ij}, $H_{ij}(F_{ij}) = G_{ij}(F_{ij}) - w_{ij}F_{ij}$ increases to $+\infty$ as $F_{ij} \uparrow C_{ij}$ (Assumption $(A2)$). Consequently, there exists an $M \in [0, C_{ij})$, such that $H_{ij}(F_{ij}) > H_{ij}(0)$ whenever $F_{ij} > M$. The function $H_{ij}(F_{ij})$ restricted to the domain $[0, M]$ attains the (global) minimum at the same point as the function considered on the set $[0, C_{ij})$. The set $[0, M]$ is compact; applying Weierstrass theorem to the

continuous function $H_{ij}(F_{ij})$ on this set gives us the required existence of a minimum. Uniqueness follows from the fact that $H_{ij}(F_{ij})$ is strictly convex on $[0, C_{ij})$. $\square$

Chapter 4

Conclusions and Directions for Future Research

In this chapter we provide a few concluding remarks and discuss a few directions for future research which are related to the themes of the dissertation. In the following section we provide concluding remarks for both the problems we have considered, and in the section after that we discuss the directions for future research.

4.1 Concluding remarks

Convergence Results for Ant Routing Algorithms. In Chapter 2 of the dissertation we have studied the convergence and have discussed the equilibrium behavior of an Ant Routing Algorithm proposed by Bean and Costa. We have considered wireline packet-switched communication networks. We have considered a stochastic queuing model for the link delays, and have provided convergence results for the Bean-Costa routing scheme. We have considered two specific cases in the dissertation: one involving an N parallel link network, where data traffic arriving at a single source node has to be transported to a single destination node via the parallel link network, and the other involving a general network, where data traffic arriving at various source nodes in the network have to transported to a single destination node. For both the cases we assume that the network queues are stable,

and we carry out our analysis given that this fact is true. However, we haven't investigated analytically what happens if during the dynamical evolution of the system (which can be described as a stochastic dynamical system consisting of a set of interconnected queues whose arrival rates are being modulated by the routing probabilities) it veered into the unstable region of the queuing system. It is possible that the algorithm is 'self-stabilizing'; for example, for the N parallel links case, notice that the routing probability for an outgoing link is proportional to the inverse of the queuing delay estimate. Consequently, if the incoming traffic into the queue is more than the service rate, the queue would momentarily build up rapidly, which would in turn make the queuing delay estimate large. This would lower the routing probability for the outgoing link leading to a decrease in the incoming traffic to the queue. It would be interesting (and challenging) to investigate the specific issue of stability of the queuing system both for the N parallel links case as well as the general network case.

Routing algorithms like the Ant Routing Algorithms, which collect measurements of quantities related to network routing performance like link and path delays and feed this information back to update the routing tables, have certain advantages. It might appear that any routing algorithm must have access to certain network information, as for example, information regarding the network topology, the input traffic rates at the various source nodes, and the link capacities. For instance, consider the optimal routing approaches available in the literature. Most of these approaches require knowledge of the input traffic into the system. Approaches like the Ant Routing Algorithm do not need the information regarding the input traffic

rates at the source nodes nor do they need information about the link capacities. This approach instead uses only the information regarding the delays. On the other hand, such (adaptive) algorithms are known to converge slowly. It is possible to improve the convergence speed of such algorithms. For instance, the step-size of the delay estimation scheme can be made variable; information regarding the variance of the delays can be obtained which can then be used to adaptively change the step-size. Also, most of the literature on Ant Routing Algorithms that uses the delay information to update the routing probabilities, considers heuristic routing probability update algorithms (in fact, the Bean-Costa scheme is also a heuristic). It would be more appropriate to develop routing probability update algorithms that are based on some sound underlying principles.

An Optimal Routing Algorithm using Dual Decomposition Techniques. In Chapter 3 of the dissertation we have considered an optimal routing problem that involves the minimization of a cost function, which is a measure of the congestion in the network, subject to the flow balance conditions on the nodes and capacity constraints on the links. We considered the dual optimization problem, and using subgradient-based primal-dual algorithms we have provided a solution to the problem. Our algorithms can be implemented by the nodes of the network in a distributed manner using only 'locally available information' like estimates of delays on the outgoing links. When the algorithm converges we obtain both the optimal dual variables as well as the optimal primal variables, the link flows. Also, we can readily incorporate the source rate control problem in our formulation, and obtain a joint rate control and routing solution. Though our routing solution has

many useful properties, a primary drawback of our algorithm (as perhaps with all such Network Utilization Maximization approaches) is that the convergence of the algorithm can be very slow. Moreover, an online implementation of the algorithm is complex, requiring frequent message exchanges regarding the dual variable updates as well as estimation of delays on the outgoing links. Such an implementation would also be totally asynchronous, for which we do not have any convergence results. The slow convergence is also a problem if changes in the input parameters to the problem - the input traffic rates and the network topology - occur frequently. Then the algorithm will not be responsive to changing input conditions.

4.2 Future Directions of Research

4.2.1 Ant Algorithms

There is an ongoing interest in properties of Ant Algorithms in general, and their applications to various practical problems. We describe in brief below various directions in which research efforts can be extended.

Extensions for Wireless Networks. Various Ant Routing Algorithms have been proposed for wireless networks. Wireless networks represent a particularly challenging problem because packet transmission has to contend with interference and with channel fading. Besides, if the nodes are mobile, the topology changes frequently. There have been many Ant Routing Algorithms that have been proposed exclusively for wireless networks (see the very brief mention in the literature survey in Chapter 1). However analytical investigations have not been pursued for most

of them. There is a need for development of appropriate models for analytically studying routing in wireless networks.

There are other issues in applications to wireless networks. In wireless networks one can either have a reactive routing scheme where routing related information pertaining to a set of routes is obtained only when there is an incoming connection that wants to route traffic through those routes, or a proactive routing scheme where routing related information throughout the network is regularly updated and maintained at all times. Ant Routing Algorithms offer a hybrid alternative. We recall that ant packets are used to collect routing related information in networks. They can thus form a natural component of a reactive routing scheme. By tuning the rate at which ant packets are generated one can cover the range from proactive to reactive routing in wireless networks. This rate can be adjusted depending upon whether or not there are incoming connections arriving at the nodes of the network (this feature is already available in the AntNet algorithm of Di Caro and Dorigo [22]), and also on the rate at which the topology of the wireless network changes (at least locally, one can learn about topology changes through the periodic exchange of $HELLO$ messages). Di Caro, Ducatelle, and Gambardella [21] have proposed an Ant Routing Algorithm for wireless mobile networks called AntHocNet which tries to capitalise on the above mentioned idea, and show that their algorithm can outperform the AODV algorithm in terms of routing performance. Various enhancements are made to the basic Ant Routing idea in order to improve the overhead (which could be substantial for Ant Routing schemes), and which are adapted for wireless networks. There can be interesting analytical investigations into such issues. Finally, because

they provide multi-path routing solutions, and because they can be adaptive and distributed, ant routing schemes remain attractive for routing problems in mobile wireless networks.

Other Applications of Ant Algorithms. Ant Algorithms have been proposed for a wide variety of combinatorial search and optimization problems. Combinatorial optimization typically involves searching for an optimum in a finite but very large set of feasible solutions. Typical combinatorial optimization problems to which Ant Colony Optimization techniques have been applied are the Traveling Salesman problem, the Graph Coloring problem, the Multiple Knapsack problem, and the Set Covering problem. For all these problems, ant agents are used to emphasize the good solutions by constructing pheromone trails. These pheromone trails are then used to bias the combinatorial search (that is conducted by successive agents) towards the good solutions. This way an expensive exhaustive search procedure is avoided. For some of these algorithms convergence to the optimal solution has been shown, see Dorigo and Stutzle [23]. However, a formal study of the computational complexity of the procedures remains an open problem, as also the issue as to how it compares with other search procedures. On the other hand, various experiments have been conducted to study the issue as to how Ant Colony Optimization performs with respect to the other procedures like Simulated Annealing, Tabu Search, Lin Kernighan heuristic, etc., on a variety of combinatorial optimization problems, and the results have been mixed - in some instances, ACO took lesser iterations to converge to the optimum, and for some others, ACO needed more iterations to converge. A survey of the results is available in Dorigo and Stutzle [23].

4.2.2 Optimal Routing Algorithms.

Convergence Issues. In the numerical study of our optimal routing algorithms we observed that the subgradient based algorithm takes many iterations to converge (slow convergence). The convergence speed can be improved by considering decreasing step-size algorithms. This is one direction which requires a more thorough numerical and analytical study.

An important direction in which the present work can be extended is to study convergence of the subgradient algorithm for the general on-line, asynchronous case. In the general on-line version, the average delays $D_{ij}(F_{ij})$ are not explicitly known, and have to be estimated by using measurements of delays on the outgoing links and then employing estimators to compute the average delays. Establishing the convergence of the overall scheme is quite challenging. Results in this vein have been obtained by Tsitsiklis and Bertsekas [47] for circuit-switched networks, and by Elwalid, Jin, Low, and Widjaja [24] for the path-flow formulation of the optimal routing problem. An interesting fact that is quantitatively proved in [24] is that the speed of convergence of their routing algorithm decreases as the size of the network (measured in terms of the number of end-to-end hops of the longest path) increases and the asynchronism in the routing updates increases. This issue is certainly relevant for a path-flow formulation, where end-to-end delays have to be estimated by probe packets. It has some relevance for our formulation too, because in general the information regarding the 'potentials' and the 'potential differences' are exchanged asynchronously between neighboring nodes to perform the subgradient

algorithm, and this fact needs to be taken into account in the investigation of the convergence of the overall scheme.

Related Problems. Our Optimal Routing algorithm is just an instance of the general Network Utility Maximization-based approach to the design of protocols/algorithms for the operation of communication networks (wireline or wireless). Most such approaches, as ours, assume that the utility (or cost) function U_i associated with an agent (which could be a source of input traffic) depends only on the resource x_i allocated to that agent. Recently, Nedic and Ozdaglar [39] have come up with solutions for problems where the agent utilities are functions of the entire vector of allocated resources to the agents, but the optimization problems are unconstrained; i.e., the problems are of the type: Minimize $\sum_{i=1}^{m} U_i(x_1, x_2, \ldots, x_n)$ subject to $(x_1, \ldots, x_n) \in \mathbb{R}^n$. A distributed version of the problem is considered where each agent i $(i = 1, \ldots, m)$ has information about his/her cost function U_i, and can compute its subgradient using estimates of x_j's from the neighboring nodes. This distributed version of the problem is solved by adapting the standard subgradient methods. It would be interesting to extend the results to problems with constraints and then consider applications to NUM for wireline (wireless) networks (for example, in wireless networks this can be used to consider NUM based approaches for MAC design, where there is an inherent coupling due to shared access to the medium - the rate at which a user can transmit depends upon the rates at which its neighbors are attempting transmission on the medium).

An interesting issue to consider is the design of routing and flow control schemes taking into account the fact that a typical wireline network consists of

a set of subnetworks, each of which is controlled by a network operator (service provider). A typical source destination pair is connected by a set of links which belong to the different subnetworks. Network operators interact with each other by mechanisms whereby preferences are accorded to flows belonging to certain neighboring operators. A lot of complex issues regarding routing performance accorded to user flows, revenue as accrued by network operators, etc., arise in such situations and is a rich source of problems of both practical and theoretical interest; examples of related works along this direction are Feamster, Johari, Balakrishnan [26], Acemoglu, Johari, Ozdaglar [1], Griffin and Sobrinho [29], and Sobrinho [44].

Bibliography

[1] D. Acemoglu, R. Johari, and A. Ozdaglar, Partially optimal routing, *IEEE Journal on Selected Areas in Communications*, pp. 11481160, Vol. 25, No. 6, 2007.

[2] J. S. Baras, and H. Mehta, A Probabilistic Emergent Routing Algorithm for Mobile AdHoc Networks, *Proc. WiOpt*03*: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, Sophia-Antipolis, France, 2003.

[3] A. Basu, A. Lin, and S. Ramanathan, Routing using potentials: A Dynamic Traffic-Aware routing algorithm, *Proc. of ACM SIGCOMM*, pp. $37-48$, 2003.

[4] N. Bean, A. Costa, An Analytic Modeling Approach for Network Routing Algorithms that Use "Ant-like" Mobile Agents, *Computer Networks*, pp. $243-268$, vol. 49, 2005.

[5] A. Benveniste, M. Metivier, and P. Priouret, *Adaptive Algorithms and Stochastic Approximation*, Appl. of Mathematics, Springer, 1990.

[6] D. P. Bertsekas, *Nonlinear Programming*, Athena Scientific, Belmont, MA,1995.

[7] D. P. Bertsekas, *Network Optimization: Continuous and Discrete Models*, Athena Scientific, Belmont, MA, 1998.

[8] D. P. Bertsekas, and J. Eckstein, Dual Coordinate Step Methods for Linear Network Flow Problems, *Math. Programming*, Series B, Vol. 42, pp. $203-243$, 1988.

[9] D. P. Bertsekas, and R. G. Gallager, *Data Networks*, Second Edition, Prentice Hall, Englewood Cliffs, NJ, 1992.

[10] D. P. Bertsekas, E. Gafni, and R.G. Gallager, Second Derivative Algorithms for Minimum Delay Distributed Routing in Networks, *IEEE Trans. on Communications*, Vol. 32, pp. $911-919$, 1984.

[11] D. P. Bertsekas, A. Nedic, and A. E. Ozdaglar, *Convex Analysis and Optimization*, Athena Scientific, Belmont, MA, 2003.

[12] D. P. Bertsekas, and J. N. Tsitsiklis, *Parallel and Distributed Computation : Numerical Methods*, Prentice-Hall, Englewood Cliffs, NJ, 1989.

[13] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*, Santa Fe Institute Studies in the Sciences of Complexity, Oxford University Press, 1999.

[14] V. S. Borkar, and P. R. Kumar, Dynamic Cesaro-Wardrop Equilibration in Networks, *IEEE Trans. on Automatic Control*, Vol. 48, No. 3, pp. 382 − 396, 2003.

[15] J. A. Boyan, and M. L. Littman, Packet routing in dynamically changing networks: A reinforcement learning approach, In J. D. Cowan, G. Tesauro, and J. Alspector (eds.), *Advances in Neural Information Processing Systems(NIPS)*,Vol. 6, pp. 671−678, Morgan Kaufmann, San Francisco, CA, 1993.

[16] L. Chen, S. H. Low, M. Chiang, and J. C. Doyle, Cross-Layer Congestion Control, Routing and Scheduling Design in Ad hoc Wireless Networks, *Proc. IEEE INFOCOM*, pp. 1 − 13, 2006.

[17] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle, Layering as Optimization Decomposition : A Mathematical Theory of Network Architectures, *Proc. of the IEEE*, Vol. 95, No. 1, pp. 255 − 312, 2007.

[18] D. J. Das, and V. S. Borkar, A novel ACO scheme for emergent optimization via reinforcement of initial bias, Available in the author's website, *http://www.tcs.tifr.res.in/ borkar/*

[19] J. L. Deneubourg, S. Aron, S. Goss, and J. M. Pasteels, The self-organizing exploratory pattern of the Argentine ant, *Journal of Insect Behavior*, vol. 3, No. 2, pp. 159 − 168, 1990.

[20] J. B. Dennis, *Mathematical Programming and Electrical Networks*, Technology Press of M.I.T, Cambridge, MA, 1959.

[21] G. Di Caro, F. Ducatelle, and L. M. Gambardella, AntHocNet: An Adaptive Nature-Inspired Algorithm for Routing in Mobile Ad Hoc Networks, *European Transactions on Telecommunications, Special Issue on Self-organization in Mobile Networking*, Vol. 16, No. 5, October 2005.

[22] G. Di Caro, M. Dorigo, AntNet: Distributed Stigmergetic Control for Communication Networks, *Journal of Artificial Intelligence Research*, pp. 317 − 365, vol. 9, 1998.

[23] M. Dorigo, T. Stutzle, *Ant Colony Optimization*. The MIT Press; 2004.

[24] A. Elwalid, C. Jin, S. Low, and I. Widjaja, MATE: MPLS Adaptive Traffic Engineering, *Computer Networks*, Vol. 40, Issue 6, pp. 695 − 709, 2002.

[25] A. Eryilmaz and R. Srikant, Joint Congestion Control, Routing, and MAC for Stability and Fairness in Wireless Networks, *IEEE Jl. on Sel. Areas of Comm.* , Vol. 24, No. 8, pp. 1514 − 1524, 2006.

[26] N. Feamster, R. Johari, and H. Balakrishnan. Implications of autonomy for the expressiveness of policy routing. *IEEE/ACM Transactions on Networking*, 2007.

[27] E. Gabber and M. Smith, *Trail Blazer: A Routing Algorithm Inspired by Ants.* Proc. of the Intl. Conf. on Networking Protocols 2004 (ICNP 2004), Berlin, Germany, October 2004.

[28] R. G. Gallager, A Minimum Delay Routing Algorithm Using Distributed Computation, *IEEE Trans. on Communications*, Vol. 23, pp. 73 – 85, 1977.

[29] T. Griffin, and J. L. Sobrinho, Metarouting, *Proc. ACM SIGCOMM 2005*, pp. 1 – 12, August 2005.

[30] M. Gunes, U. Sorges, and I. Bouazizi, ARA-The Ant-colony Based Routing Algorithm for MANETs, in S. Olariu (Ed.), *Proc.* 2002 *ICPP Workshop on Ad Hoc Networks*, pp. 79 – 85, IEEE Comp. Soc. Press.

[31] W. J. Gutjahr, A Generalized Convergence Result for the Graph-based Ant System Metaheuristic, *Probability in the Engineering and Informational Sciences*, pp. 545 – 569, vol. 17, 2003.

[32] L. P. Kaelbling, M. L. Littman and A. W. Moore, Reinforcement Learning : A Survey, *Journal of Artificial Intelligence Research*, Vol. 4, pp. 237 – 285, 1996.

[33] K. Kar, S. Sarkar, and L. Tassiulas, Optimization Based Rate Control for Multipath Sessions, *Proc.* 17*-th Intl. Teletraffic Congress*, December, 2001.

[34] F. P. Kelly, Network Routing, *Phil. Trans. R. Soc. Lond. A: Physical Sciences and Engineering (Complex Stochastic Systems)*, Vol. 337, No. 1647, pp. 343 – 367, 1991.

[35] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, Rate Control in Communication Networks : Shadow Prices, Proportional Fairness and Stability, *Jl. of Oper. Res. Soc.*, Vol. 49, pp. 237 – 252, 1998.

[36] H. J. Kushner and G. G. Yin, *Stochastic Approximation Algorithms and Applications*, Applications of Mathematics Series, Springer Verlag, New York, 1997.

[37] X. Lin, and N. B. Shroff, Joint Rate Control and Scheduling in Multihop Wireless Networks, *Proc. IEEE Conf. on Dec. and Cont.*, Vol. 2, pp. 1484 – 1489, 2004.

[38] A. M. Makowski, The Binary Bridge Selection Problem: Stochastic Approximations and the Convergence of a Learning Algorithm, *Proc. ANTS, Sixth Intl. Conf. on Ant Col. Opt. and Swarm Intelligence*, Lecture Notes in Computer Science 5217, M. Dorigo *et. al.* (eds.), Springer Verlag, pp. 167 – 178, 2008.

[39] A. Nedic, and A. Ozdaglar, On the Rate of Convergence of Distributed Asynchronous Subgradient Methods for Multi-agent Optimization, *Proc. IEEE Conf. on Dec. and Control*, pp. 4711 – 4716, 2007.

[40] M. J. Neely, E. Modiano, and C. E. Rohrs, Dynamic Power Allocation and Routing for Time Varying Wireless Networks, *IEEE Jl. on Sel. Areas of Comm.*, Special Issue on Wireless Ad-Hoc Networks, Vol. 23, No. 1, pp. 89 – 103, 2005.

[41] P. Purkayastha and J. S. Baras, Convergence of Ant Routing Algorithms via Stochastic Approximation and Optimization, *Proc. IEEE Conf. on Dec. and Cont.*, pp. 340 – 345, December 2007.

[42] R. T. Rockafellar, *Network Flows and Monotropic Optimization*, Athena Scientific, Belmont, MA, 1998.

[43] R. Schoonderwoerd, O. E. Holland, J. L. Bruten, and L. J. M. Rothkrantz, Ant-Based Load Balancing in Telecommunications Networks, *Adaptive Behavior*, pp. 169 – 207, Vol. 5, No. 2, 1997.

[44] J. L. Sobrinho, Network Routing with Path Vector Protocols: Theory and Applications, *Proc. ACM SIGCOMM* 2003, pp. 49 – 60, August 2003.

[45] D. Subramanian, P. Druschel, and J. Chen, Ants and reinforcement learning: A Case Study in routing in dynamic networks, *Proc. of IJCAI* 1997*: The International Joint Conference on Artificial Intelligence*, 1997.

[46] M. A. L. Thathachar and P. S. Sastry, *Networks of Learning Automata : Techniques for Online Stochastic Optimization*, Kluwer Academic Publishers, Norwell, MA, USA; 2004.

[47] J. N. Tsitsiklis and D. P. Bertsekas, Distributed Asynchronous Optimal Routing in Data Networks, *IEEE Trans. on Automatic Control*, Vol. 31, No. 4, pp. 325 – 332, 1986.

[48] W.-H. Wang, M. Palaniswami, and S. H. Low, Optimal Flow Control and Routing in multi-path networks, *Perf. Evaluation Jl.*, Vol. 52, No. 2−3, pp. 119−132, Elsevier, 2003.

[49] J. G. Wardrop, Some Theoretical Aspects of Road Traffic Research, *Proc. Inst. Civil Engineers*, Vol. 1, pp. 325 – 378, 1952.

[50] J.-H. Yoo, R. J. La, and A.M. Makowski, Convergence Results for Ant Routing, *Proc. Conf. on Inf. Sc. and Systems*, Princeton, NJ, 2004.