ENFORCING COOPERATION AND PROVIDING QUALITY OF SERVICE IN WIRELESS NETWORKS

BY

JUAN JOSE JARAMILLO JIMENEZ

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2010

Urbana, Illinois

Doctoral Committee:

Professor R. Srikant, Chair
Professor P. R. Kumar
Professor Nitin H. Vaidya
Professor Venugopal V. Veeravalli

ABSTRACT

The purpose of this dissertation is to design algorithms that provide quality of service and enforce cooperation in wireless ad hoc networks. Using a simple network model, we first study the performance of some previously proposed cooperation-enforcing strategies and then present a new mechanism. We prove that our mechanism is robust to imperfect measurements, is collusion-resistant, and can achieve full cooperation among nodes. Assuming cooperation is being enforced, we then study the problem of optimal routing and admission control for flows which require a pre-specified bandwidth from the network. We develop an algorithm whose performance is close to that of an omniscient off-line algorithm that has complete *a priori* knowledge of the entire sequence of flow arrivals and their bandwidth requests, including the future. We then study the problem of congestion control and scheduling in wireless ad hoc networks that have to support a mixture of best-effort and real-time traffic. We propose a model for incorporating the requirements of packets with deadlines in an optimization framework. The solution to the problem results in a joint congestion control and scheduling algorithm which fairly allocates resources to meet the fairness objectives of both elastic and inelastic flows, and the per-packet delay requirements of inelastic flows.

To my parents and sister

ACKNOWLEDGMENTS

First, I would like to express my sincere gratitude to my adviser, Professor R. Srikant, for his advice and continuous support during my Ph.D. program. Without his encouragement and insightful guidance I could not have accomplished my degree. He has been both a friend and a mentor to me. Working with him is an honor and a privilege for which I am grateful.

I want to thank Professor P. R. Kumar, Professor Nitin H. Vaidya, and Professor Venugopal V. Veeravalli for serving on my Ph.D. committee and providing invaluable comments on my research.

Finally, I would like to thank my parents and sister, whose support during these years has been invaluable to help me reach for my dreams.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Wireless ad hoc networks promise to provide ubiquitous connectivity without the need for any infrastructure. Such networks are expected to support services that require quality of service (QoS) guarantees, such as voice, video, and remote sensing. To efficiently provide such services, these networks rely on the assumption that intermediate nodes are willing to relay packets to distant destinations when single-hop source-destination communications are not possible or desirable. In this dissertation we first study how to enforce cooperation in relay nodes, and assuming cooperation is achieved, we then design algorithms that can guarantee minimum-bandwidth and maximum per-packet delay requirements. More specifically, we have studied the following problems:

1. A Reputation Mechanism to Incentivize Cooperation in Wireless Ad Hoc Networks

   In wireless ad hoc networks one way to incentivize nodes to forward other nodes' packets is through the use of reputation mechanisms, where cooperation is induced by the threat of partial or total network disconnection if a node acts selfishly. The problem is that packet collisions and interference may make cooperative nodes appear selfish sometimes, generating unnecessary and unwanted punishments.

   The contributions are the following:

   - We use a simple game-theoretic network model to study the robustness of some previously proposed reputation-based strategies. We show that some strategies are not self-enforcing, meaning that there is an incentive to deviate from the expected behavior, while others punish selfish behavior at the expense of the throughput of cooperative nodes, potentially leading to complete network disconnection due to retaliation.

- We propose a new strategy called DARWIN (Distributed and Adaptive Reputation Mechanism for Wireless ad hoc Networks) that effectively detects and punishes selfish behavior. We derive conditions under which no node can gain from deviating from our strategy, prove that full cooperation can emerge among nodes, and that our scheme is collusion resistant.
- Simulations are presented to complement the theoretical contributions. Our results show that the throughput achieved with DARWIN is better than any of the other strategies studied, and that DARWIN can be implemented with low overhead.

2. Admission Control and Routing in Multi-Hop Wireless Networks

   We consider the problem of optimal routing and admission control for flows which require a pre-specified bandwidth from the network. This problem has been extensively studied for wireline networks, but until now, there has been no systematic study on the role of load balancing in obtaining good wireless network performance. We study the problem using competitive analysis, where a given on-line (causal) algorithm's performance is compared against the performance of an off-line algorithm which also has access to future traffic patterns.

   Our contributions are summarized as follows:

   - We developed a model for admission control and routing with minimum-bandwidth constraints in multi-hop wireless networks, which allows us to derive an algorithm with provable performance guarantees using competitive analysis. Our algorithm makes no statistical assumptions on the flow arrival pattern or other parameters of the arriving requests.
   - We showed that no other algorithm performs better in an asymptotic sense to be described later. The proof of this result is more involved than the corresponding result for wireline networks and relies on the unique characteristics of wireless networks.
   - The optimal algorithm is not in a form that is amenable to distributed implementation. Thus, we converted the algorithm into a form that allows the use of standard shortest-path algorithms.

3. Optimal Scheduling for Fair Resource Allocation in Ad Hoc Networks with Elastic and Inelastic Traffic

   We study the problem of congestion control and scheduling in peer to peer ad hoc wireless networks that have to support a mixture of best-effort and real-time traffic. Optimization and stochastic network theory have been successful in designing architectures for fair resource allocation to meet long-term throughput demands. However, to the best of our knowledge, strict per-packet delay deadlines were not considered in this framework previously.

   The main contributions are as follows:

   - We present an optimization framework for resource allocation in a wireless network consisting of both best-effort flows and flows that generate traffic with per-packet delay constraints. The framework allows for very general interference, channel and arrival models.
   - Using a dual decomposition approach, we derive an optimal scheduling and congestion control algorithm that fairly allocates resources and ensures that a required fraction of each inelastic flow's packets are delivered on time by appealing to connections between Lagrange multipliers, queues, and service deficits. The scheduling algorithm seamlessly integrates inelastic and elastic traffic into a unified max-weight scheduling framework, extending the classical results in [1] and the recent results of [2, 3].
   - The convergence of the above algorithm in an appropriate stochastic sense is proved and it is also shown that the network is stable.

CHAPTER 2

A REPUTATION MECHANISM TO INCENTIVIZE COOPERATION IN WIRELESS AD HOC NETWORKS

Wireless ad hoc networks consist of a set of self-configuring nodes that do not rely on any infrastructure to communicate among each other. To achieve this goal, a source communicates with a distant destination through intermediate nodes that act as relays. It is usually assumed that in such networks, nodes are willing to cooperate by forwarding packets, but this assumption is not necessarily true in the case where all nodes are not under the control of a single authority. In these cases, there can be selfish nodes that want to maximize their own welfare without regard to social welfare, where we define a node's welfare as the benefit of its actions minus the cost of its actions. In such scenarios, cooperation cannot be taken for granted, and it is therefore necessary to develop mechanisms that allow cooperation to emerge even in the presence of selfish users.

Incentive mechanisms can be broadly divided in two categories: credit-exchange systems and reputation-based systems. In credit-exchange systems [4–10], cooperation is induced by means of payments received every time a node acts as a relay and forwards a packet, and such credit can later be used by these nodes to encourage others to cooperate. To guarantee that nodes do not counterfeit payments, some strategies rely on the use of tamper-proof hardware to store credit and guarantee the checks and balances, but this strategy may hinder their ability to find widespread acceptance; other strategies rely on the presence of an off-line central trusted authority which may be hard to guarantee in some scenarios. In reputation-based strategies [11–19], a node's behavior is measured by other nodes in the network. Selfish behavior is then discouraged by the threat of partial or total network disconnection. The problem is that due to interference and collisions it is not always possible to perfectly estimate how a node behaves, so sometimes cooperative nodes are perceived as being selfish and punished accordingly; such scenarios can lead to retaliation situations that may potentially decrease

the throughput of cooperative nodes.

The contributions of this chapter are twofold: first, we use a simple game-theoretic network model to study the robustness of some previously proposed reputation-based strategies. We show that some strategies are not self-enforcing, meaning that there is an incentive to deviate from the expected behavior, while others punish selfish behavior at the expense of the throughput of cooperative nodes, potentially leading to complete network disconnection due to retaliation. Second, we propose a new strategy called DARWIN (Distributed and Adaptive Reputation Mechanism for Wireless ad hoc Networks) that effectively detects and punishes selfish behavior. We derive conditions under which no node can gain from deviating from our strategy, prove that full cooperation can emerge among nodes, and prove that our scheme is collusion-resistant.

Simulations are also presented to complement the theoretical contributions. Our results show that the throughput achieved with DARWIN is better than any of the other strategies studied, and that DARWIN can be implemented with low overhead.

The rest of the chapter is organized as follows. Section 2.1 introduces some concepts from game theory that are used in this chapter. In Section 2.2 we define the network model which will be used in Section 2.3 to analyze some of the previously proposed strategies. We introduce our strategy in Section 2.4, analyze the conditions under which cooperation can emerge, study its performance, and show that it is relatively insensitive to parameter choices. The impact of collusion among nodes is also studied there. Section 2.5 presents the results of a simulation-based study of DARWIN and how it compares to other reputation-based strategies. Section 2.6 presents an overview of related work.

2.1 Basic Game Theory Concepts

Here we introduce the concepts from game theory [20] that are used in this chapter. As an illustration, we use a well-known game between two players known as The Prisoner's Dilemma. Both players have two possible *pure strategies*, Cooperate (C) or Defect (D), and the payoffs they receive for their actions are given in Table 2.1. Then player i's *strategy space* S_i is defined

Table 2.1: Payoff Matrix of the Prisoner's Dilemma Game

		Player 2			
		Cooperate		Defect	
Player 1	Cooperate	1	1	−1	2
	Defect	2	− 1	0	0

to be the set of pure strategies available to it. In this case $S_i = \{C, D\}$ for $i = \{1, 2\}$. A *strategy profile* is defined to be an element of the product-space of strategy spaces of each player. An example is for player 1 to play D and player 2 to play C.

Definition 1. *A* Nash equilibrium *is a strategy profile having the property that no player can benefit by unilaterally deviating from its strategy.* ◇

Such a strategy profile is considered to be *self-enforcing*. In this example, the Nash equilibrium would be the strategy profile $s = (D, D)$. Assume now that this game is repeated infinitely many times, and for each k, the outcomes of the $k-1$ preceding plays are observed before the kth stage begins. In this case, the total payoff of the game for player i is the discounted sum of the stage payoffs. Denoting the stage payoffs by $u_i^{(k)}$, the total payoff is given by

$$U_i = \sum_{k=0}^{\infty} w^k u_i^{(k)} \ ,$$

where $w \in (0, 1)$ is the *discount factor*. The infinitely repeated game can also be interpreted as a repeated game that ends after a random number of repetitions. Under this interpretation, the length of the game is a geometric random variable with mean $1/(1 - w)$.

In this game a player's strategy specifies the action it will take at each stage, for each possible history of play through previous stages. In our example a strategy for player 1 could be to cooperate until player 2 defects, and then defect forever. Since both players know the previous history, we can view the game starting at stage k with a given history h^k as a new game; this is called a *subgame* of the original game.

Definition 2. *For a given set of strategies that are in Nash equilibrium, his-*

tory h^k is on the equilibrium path *if it can be reached with positive probability if the game is played according to the equilibrium strategies, and is* off the equilibrium path *otherwise.* ◇

Definition 3. *A Nash equilibrium is* subgame perfect *if the players' strategies constitute a Nash equilibrium in every subgame.* ◇

Subgame perfection is a stronger concept that eliminates *noncredible* equilibria, since it analyzes the case when a game is on or off the equilibrium path. This will later help us analyze whether a given reputation scheme is robust enough to handle the case when, due to inaccurate measurements, nodes appear to be out of their predicted behavior.

Definition 4. *A game is* continuous at infinity *if for each player i the payoff U_i satisfies:*

$$\sup_{h,\tilde{h} \ s.t. \ h^k=\tilde{h}^k} \left| U_i(h) - U_i(\tilde{h}) \right| \to 0 \ as \ k \to \infty.$$

◇

Under this definition, events in the distant future are relatively unimportant. This holds true if the total payoff of the game is the discounted sum of the per-period payoffs $u_i^{(k)}$, and the per-period payoffs are uniformly bounded. In our example this holds true since $u_i^{(k)} \leq 2$ for all k.

Lemma 1 (One-Stage Deviation Principle)**.** *In an infinite-horizon multistage game with observed actions that is continuous at infinity, strategy profile s is subgame perfect if and only if there is no player i and strategy $\hat{s}_i$ that agrees with s_i except at a single stage k and h^k, and such that $\hat{s}_i$ gives a better payoff than s_i conditional on history h^k being reached.* ◇

For a proof see [20]. We say that s satisfies the One-Stage Deviation Principle if no player can gain by deviating from s, either on or off the equilibrium path, in a single stage.

In the rest of this chapter we will develop a prisoner's dilemma model for wireless networks. Such an exercise has been carried out before in other papers, but our approach and solution are quite different.

Table 2.2: Payoff Matrix of the Packet Forwarding Game

		Node 2			
		Forward		Drop	
Node 1	Forward	$\alpha - 1$	$\alpha - 1$	$-\alpha - 1$	α
	Drop	α	$-\alpha - 1$	$-\alpha$	$-\alpha$

2.2 Network Model

We assume that nodes are selfish but not malicious. A selfish node is a rational user that wants to maximize its own welfare, defined as the benefit minus the cost of its actions. Links are assumed to be bidirectional. Wireless links are often bidirectional, and many MAC layers require bidirectional packet exchanges to avoid collisions, as is the case in IEEE 802.11. Finally, nodes are assumed to operate in promiscuous mode, so they are able to listen to all packets transmitted by their neighbors.

Forwarding a packet consumes resources. We define the normalized relaying cost to be 1. The reward a node receives if its packet is relayed is α, where we assume $\alpha \geq 1$ since the value of a packet should be at least equal to the cost of the resources used to send it. We assume that the interaction among nodes is reciprocal, i.e., any two neighbors have uniform network traffic demands and need each other to forward packets. Thus, we can isolate any pair of nodes and study the interaction between them as a two-player game. Later in Section 2.5 we simulate a random network with asymmetric and spatially non-uniform traffic without this assumption and test whether our conclusions still hold.

In the two-player game, one way to model the nodes is to assume that they send packets to each other and then simultaneously decide whether to drop or forward their respective packets, and repeat this game iteratively. In this scenario the stage payoffs matrix is given in Table 2.2. Without loss of generality, we do an affine transformation to the payoff matrix as shown in Table 2.3 using the following formula: let x be any entry in Table 2.2, and let y be the respective entry in Table 2.3, then $y = (x + \alpha)/(2\alpha - 1)$. Using standard game theory notation, we will denote by $i \in \{1, 2\}$ a generic node and by $-i$ its neighbor.

Table 2.3: Affine Transformation to the Payoff Matrix of the Packet Forwarding Game

		Node 2			
		Forward		Drop	
Node 1	Forward	1	1	$\frac{-1}{2\alpha-1}$	$\frac{2\alpha}{2\alpha-1}$
	Drop	$\frac{2\alpha}{2\alpha-1}$	$\frac{-1}{2\alpha-1}$	0	0

Since the interaction among nodes is asynchronous in nature, we refine the game assuming that time is divided into slots and that slots last long enough to allow each node to send a sufficiently large number of packets. At the end of the slot each node finds the ratio of packets dropped by its neighbor; if the number of packets exchanged is sufficiently large, then this ratio is a good estimate of the probability of dropping a packet. This assumption is implicitly used in other papers on reputation mechanisms as well [15, 16].

Due to collisions, it is not always possible to detect whether a node forwarded a packet. We define $p_e \in (0, 1)$ to be the probability that a packet that has been forwarded was not overheard by the originating node. We also assume that p_e is the same for both nodes. (As mentioned before, in Section 2.5 we test this assumption by simulating a non-uniform network to compare with our analysis.) By listening to the channel, node i then estimates the perceived dropping probability $\hat{p}_{-i}^{(k)}$ of its neighbor at time slot $k \geq 0$. It must be noted that a packet is perceived to be dropped if either $-i$ dropped it or if it is not dropped but node i did not overhear the transmission. Thus

$$\hat{p}_{-i}^{(k)} = p_{-i}^{(k)} + (1 - p_{-i}^{(k)})p_e = p_e + (1 - p_e)p_{-i}^{(k)}, \tag{2.1}$$

where $p_{-i}^{(k)}$ is the probability that $-i$ drops a packet.

Thus, using the payoffs of Table 2.3, the average payoff at time slot k is:

$$u_i^{(k)} = (1 - p_i^{(k)})(1 - p_{-i}^{(k)}) + \frac{2\alpha}{2\alpha - 1}p_i^{(k)}(1 - p_{-i}^{(k)}) - \frac{1}{2\alpha - 1}(1 - p_i^{(k)})p_{-i}^{(k)}.$$

Table 2.4: Summary of Notation

	Meaning
α	Reward a node receives if a packet has been relayed
p_e	Probability that a packet that has been forwarded was not overheard by originating node
$p_i^{(k)}$	Dropping probability of player i at time slot k
$\hat{p}_i^{(k)}$	Perceived dropping probability of player i at time slot k
$\tilde{p}_{i\ S}^{(k)}$	Dropping probability player i should use at time slot k according to strategy S
w	Discount factor. Probability that both nodes continue to interact after each time slot
$u_i^{(k)}$	Player i's average payoff at time slot k
$U_i^{(n)}$	Discounted average payoff of player i starting from time slot n

Rearranging terms:

$$u_i^{(k)} = 1 + \frac{1}{2\alpha - 1} p_i^{(k)} - \frac{2\alpha}{2\alpha - 1} p_{-i}^{(k)}. \tag{2.2}$$

The discounted average payoff of player i starting from time slot n is then given by:

$$U_i^{(n)} = \sum_{k=n}^{\infty} w^{k-n} u_i^{(k)} \ , \tag{2.3}$$

where $w \in (0, 1)$ is the discount factor. Since node i cannot know $p_{-i}^{(k)}$ for sure, it does not know its payoff either. However, we use the actual payoff in the analysis since it tells us whether a given node can gain by deviating from a strategy.

Given this game, each player is allowed to use a strategy to decide whether to drop or forward packets based on the history. We use $\tilde{p}_{i\ S}^{(k)}$ to denote the dropping probability player i should use at time slot k according to strategy S. For convenience, the definitions used are given in Table 2.4.

2.3 Analysis of Prior Proposals

To motivate our new protocol, which we will present in the next section, in this section we present a few strategies that have been proposed in prior work and show their limitations.

2.3.1 Trigger Strategies

One idea to provide an incentive for cooperation is to develop a strategy such that the cooperation of a node is measured and if the fraction of packets it has dropped is above a threshold, it is considered selfish and is disconnected for a given amount of time. Formally, an *n-step trigger strategy* is defined as:

$$\tilde{p}_{i\ nT}^{(0)} = 0$$
$$\tilde{p}_{i\ nT}^{(k)} = \begin{cases} 0 & if\ \hat{p}_{-i}^{(j)} \leq T\ for\ all\ j \in \{k-n, \ldots, k-1\} \\ 1 & else, \end{cases}$$

where we define $\hat{p}_{-i}^{(j)} = 0$ for $j \in \mathbb{Z}_-$. From (2.1) it is easy to see that if node i cooperates then $\hat{p}_{-i}^{(k)} = p_e$ for all k. Hence the optimal value of $T = p_e$. In reality we cannot perfectly estimate p_e, so we have to analyze two cases:

1. If $T < p_e$ then we have that $\tilde{p}_{i\ nT}^{(k)} = 1$ for $k \geq 1$, so cooperation will never emerge.

2. If $T > p_e$ then player $-i$ will be perceived to be cooperative as long as it drops packets with probability:

$$p_{-i}^{(k)} \leq \frac{T - p_e}{1 - p_e}.$$

Therefore, since p_e is unknown, any choice of threshold other than $T = p_e$ results in either all packets being dropped or some fraction of packets being dropped. In other words, full cooperation is never the Nash equilibrium point with trigger strategies.

Variations on this strategy have been used in several reputation mechanisms, where the different proposals focus on ideas on how to detect selfish behavior and then proceed to isolate selfish nodes: Catch [16], CONFIDANT [13], OCEAN [14], and the reputation-based mechanism in [18] are

among them.

2.3.2 Tit For Tat

A second alternative is to use a *Tit For Tat* (TFT) strategy [21]. It was generalized in [22] for the wireless context as follows:

$$\tilde{p}_{i\ TFT}^{(0)} = 0$$
$$\tilde{p}_{i\ TFT}^{(k)} = \hat{p}_{-i}^{(k-1)}\ for\ k \geq 1.$$

However, Milan *et al.* [22] proved that this strategy does not provide the right incentive either for cooperation in wireless networks.

In RMS [19] the selfishness of a node is classified into one of several different levels, and punishment is given according to the level. Such a strategy can then be considered to implement a discretized version of TFT, as opposed to the continuous version presented here.

2.3.3 Generous Tit For Tat

The problem with TFT is that it does not take into account the fact that it is not always possible to determine whether a packet was relayed or not due to collisions. A way to deal with this is by using a generosity factor g that allows cooperation to be restored. Such a strategy is known as *Generous TFT* (GTFT) [23] and in the case of wireless networks it can be defined [22] as follows[1]:

$$\tilde{p}_{i\ GTFT}^{(0)} = 0$$
$$\tilde{p}_{i\ GTFT}^{(k)} = \max\{\hat{p}_{-i}^{(k-1)} - g, 0\}\ for\ k \geq 1.$$

Lemma 2. *If both nodes do not deviate from the GTFT strategy, then the generosity factor that maximizes the discounted average payoff is* $g^* \geq p_e$. $\diamond$

Proof. If $g \geq p_e$ then from (2.1) we have for all $k \geq 0$ and $i \in \{1, 2\}$ that $p_i^{(k)} = 0$. Using (2.2) and (2.3) we obtain:

$$U_i^{(0)} = \frac{1}{1-w}. \tag{2.4}$$

[1]Note that this definition corresponds to a reputation-based mechanism, not to be confused with the credit-based mechanism proposed in [6] that bears the same name.

In the case $g < p_e$ we obtain:

$$p_i^{(0)} = 0$$

and for $k \geq 1$:

$$p_i^{(k)} = (p_e - g)\sum_{n=0}^{k-1}(1-p_e)^n = (p_e - g)\frac{1-(1-p_e)^k}{p_e}.$$

So the stage payoffs for $k \geq 1$ are:

$$u_i^{(k)} = \frac{1}{p_e}\left[g + (p_e - g)(1-p_e)^k\right].$$

Therefore the discounted average payoff is:

$$U_i^{(0)} = 1 + \frac{w}{p_e}\left[\frac{g}{1-w} + \frac{(p_e-g)(1-p_e)}{1-w(1-p_e)}\right]. \tag{2.5}$$

It can easily be checked that the payoff (2.5) is strictly less than the payoff (2.4). □

It is important to highlight that in the case $g > p_e$ GTFT is not a Nash equilibrium since for player $-i$, it pays to deviate dropping packets with a probability

$$p_{-i}^{(k)} \leq \frac{g - p_e}{1 - p_e}.$$

The following theorem and corollary tell us that if the interaction between two nodes lasts long enough, then GTFT is a robust strategy where no node can gain by deviating from the expected behavior, even if it is not able to achieve full cooperation.

Theorem 1. *GTFT is subgame perfect if and only if*

$$g \leq p_e \text{ and } w > \frac{1}{2\alpha(1-p_e)}.$$

◇

(See the proof in Appendix A.)

Corollary 1. *If both nodes use GTFT then cooperation is achieved on the equilibrium path if and only if* $g = p_e$. ◇

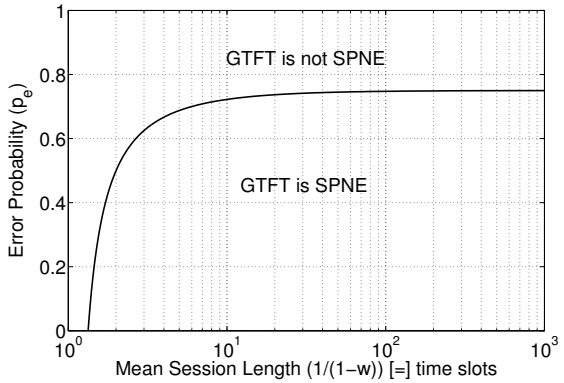

Figure 2.1: GTFT's subgame perfect Nash equilibrium (SPNE) region for $\alpha = 2$

Note that in [22] a proof was done for the case $g = p_e$ but only considering the equilibrium path. The subgame perfect region of GTFT is plotted in Fig. 2.1 for $\alpha = 2$. Figure 2.2 shows how the shape of this region is affected by different values of α. Note that when the value of a packet grows larger compared to the actual cost of transmitting it, then cooperation has a better chance to emerge since being connected is more important than reducing the cost of helping other nodes. In summary, GTFT is not satisfactory because in order to achieve full cooperation we need a perfect estimate of p_e. Such a strategy has been used in SORI [15] to punish selfish behavior.

2.4 DARWIN

In this section we introduce our algorithm, and we prove that our strategy is subgame perfect, achieves cooperation on the equilibrium path, and can cope with a group of colluding nodes. We end the section discussing some possible security issues and how they can be solved.

2.4.1 Definition

Our goal is to propose a reputation strategy that does not depend on a perfect estimation of p_e to achieve full cooperation and that is also more robust

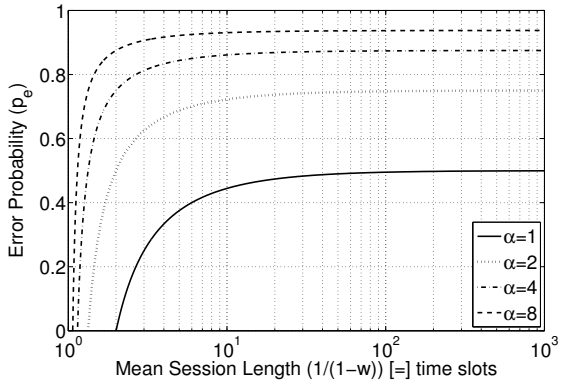

Figure 2.2: Sensitivity of GTFT's subgame perfect region for different values of α

than previously proposed strategies. For the iterated Prisoner's Dilemma, a modification of TFT known as Contrite Tit For Tat (CTFT) [24, 25] has been proposed based on the idea of contriteness: a player that made a mistake and unintentionally defected should exercise contrition and try to correct the error instead of going into a retaliation situation. This strategy depends on the notion of good standing and is defined as follows. A player is always in good standing in the first stage. It remains in good standing as long as it cooperates when CTFT specifies that it should cooperate. If an individual is in bad standing it can recover good standing by cooperating in one stage. Then CTFT specifies that a player should cooperate if it is in bad standing, or if its opponent is in good standing; otherwise the player should defect. Inspired by this strategy, for the case of wireless networks we define the following strategy:

$$\tilde{p}_{i\ DARWIN}^{(k)} = \left[\gamma\left(q_{-i}^{(k-1)} - q_i^{(k-1)}\right)\right]_0^1 \ for\ k \geq 0, \tag{2.6}$$

where we define for $i = \{1, 2\}$:

$$q_i^{(k)} = \begin{cases} \left[\hat{p}_i^{(k)} - \tilde{p}_{i\ DARWIN}^{(k)}\right]_0^1 & for \quad k \geq 0 \\ 0 & for \quad k = -1. \end{cases} \tag{2.7}$$

Additionally we define the function:

$$[x]_0^1 = \begin{cases} 1 & if \quad x \geq 1 \\ x & if \quad 0 < x < 1 \\ 0 & if \quad x \leq 0 \end{cases} .$$

Recall that $\hat{p}_i^{(k)}$ denotes the estimated dropping probability and $\tilde{p}_{i\ DARWIN}^{(k)}$ is the dropping probability under DARWIN. Thus, if $\hat{p}_i^{(k)} > \tilde{p}_{i\ DARWIN}^{(k)}$, it means node i is perceived to be dropping more packets than it should under DARWIN. The parameter $q_i^{(k)}$ measures this deviation. In this case $q_i^{(k)}$ acts as a measurement of the standing of a node, and only the player that has better standing should punish its opponent in proportion to the *difference* between the two standings instead of the *absolute* value of the standing of its opponent. The limitation on any strategy is that it requires that the interaction between the nodes last long enough for the reputation mechanism to be effective. This is translated in a feasible set for the parameter w, the probability that both nodes continue to interact after each time slot. In the case of DARWIN, γ determines the set of feasible values of w: the larger the punishment factor γ, up to an upper bound, the shorter the interaction between the nodes can be. This relationship will be quantitatively presented in Theorem 2.

2.4.2 Performance Guarantees

The following theorem proves that when the interaction between two nodes lasts long enough, DARWIN is a robust strategy where no node can gain by deviating from the expected behavior.

Theorem 2. *Assuming* $1 < \gamma < p_e^{-1}$*, DARWIN is subgame perfect if and only if*

$$w > \max\left\{\frac{1}{\gamma}, \frac{1}{2\alpha(1 - p_e\gamma) + p_e\gamma}\right\}. \tag{2.8}$$

◇

(See the proof in Appendix B.)

From (2.8) it is clear that the optimum value of γ that minimizes this bound is a function of α and p_e. Since one cannot estimate α, a suboptimal

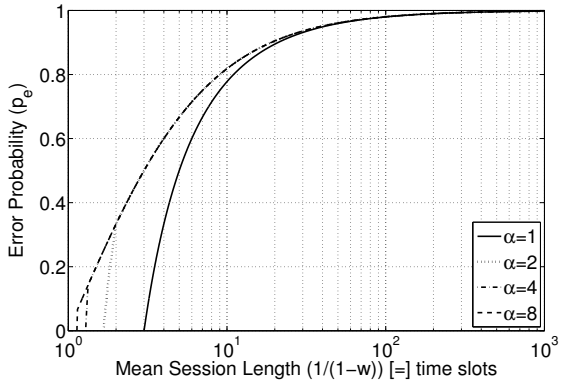

Figure 2.3: Sensitivity of DARWIN's subgame perfect region for different values of α, assuming (2.9) holds

strategy could be to choose γ to be the average of the interval $(1, p_e^{-1})$:

$$\gamma = \frac{1 + p_e^{-1}}{2} = \frac{1 + p_e}{2p_e}. \tag{2.9}$$

Figure 2.3 shows the subgame perfect region of DARWIN for different values of α, assuming (2.9) holds, which is not significantly different from the subgame perfect region if we had used the optimal value of γ.

It must be highlighted that if both nodes use DARWIN then full cooperation is achieved. This can easily be checked using (2.1) and the definition of DARWIN to observe the game evolution.

Lemma 3. *If both nodes use DARWIN then cooperation is achieved on the equilibrium path. That is,* $p_i^{(k)} = p_{-i}^{(k)} = 0$ *for all* $k \geq 0$. ◇

Since this is the best any strategy S can achieve, we have that:

$$U_{i\ S}^{(0)} \leq U_{i\ DARWIN}^{(0)} \ for\ any\ strategy\ S. \tag{2.10}$$

It is also important to remember that for DARWIN to be subgame perfect we need to estimate p_e in order to achieve the bound $\gamma < p_e^{-1}$. Since we cannot do perfect estimation, we have that the estimated error probability $p_e^{(e)}$ is equal to

$$p_e^{(e)} = p_e + \Delta,$$

where $\Delta \in (-p_e, 1-p_e)$ is the estimation error. If we choose γ using (2.9) we have:

$$\gamma = \frac{1+p_e^{(e)}}{2p_e^{(e)}} = \frac{1+p_e+\Delta}{2p_e+2\Delta}.$$

So we have that $\gamma < p_e^{-1}$ if and only if:

$$\Delta > -p_e\left(\frac{1-p_e}{2-p_e}\right).$$

Thus, for the DARWIN strategy, one does not need a precise estimate of p_e; an estimator that overestimates p_e is sufficient for Theorem 2 to hold.

2.4.3 Collusion Resistance

We now consider the case when a group of colluding nodes work together to maximize their own benefit regardless of the social optimum. Define $U_{i\ S_i|S_{-i}}^{(0)}$ to be the discounted average payoff of player i using strategy S_i when it plays against player $-i$ using strategy S_{-i}. Hence (2.10) can be rewritten as:

$$U_{i\ S|S}^{(0)} \leq U_{i\ D|D}^{(0)} \textit{ for any strategy } S. \tag{2.11}$$

Also, a consequence of Theorem 2 is

$$U_{i\ S|D}^{(0)} < U_{i\ D|D}^{(0)} \tag{2.12}$$

for any strategy $S \neq D$ =DARWIN. Assume a group of colluding nodes implementing strategy S enters the network. Define $p_S \in (0,1)$ to be the probability that a node that implements DARWIN interacts with a colluding node. Therefore the average payoff to a cooperative node will be:

$$U(D) = p_S U_{i\ D|S}^{(0)} + (1-p_S)U_{i\ D|D}^{(0)}.$$

Similarly, if $p_D \in (0,1)$ is the probability that a colluding node interacts with a node implementing DARWIN, we have:

$$U(S) = p_D U_{i\ S|D}^{(0)} + (1-p_D)U_{i\ S|S}^{(0)}.$$

We have that the average payoff is bounded by

$$U(S) \leq \max\left\{U_{i\ S|D}^{(0)}, U_{i\ S|S}^{(0)}\right\}. \tag{2.13}$$

So a group of colluding nodes cannot gain from unilaterally deviating if and only if $U(S) < U(D)$. Equivalently,

$$p_S\left[U_{i\ D|D}^{(0)} - U_{i\ D|S}^{(0)}\right] < U_{i\ D|D}^{(0)} - U(S). \tag{2.14}$$

From (2.11), (2.12) and (2.13) we know that

$$U_{i\ D|D}^{(0)} - U(S) \geq 0.$$

Definition 5. *Strategy S is a* naive strategy *if*

$$U_{i\ D|D}^{(0)} < U_{i\ D|S}^{(0)}. \tag{2.15}$$

That is, strategy S is exploited when matched against DARWIN. Furthermore, a non-naive strategy *is one such that (2.15) does not hold.* ◇

From (2.14), we have proved the following theorem:

Theorem 3. *DARWIN is collusion-resistant against a naive strategy. Furthermore, it is resistant against a non-naive strategy if and only if*

$$p_S < \frac{U_{i\ D|D}^{(0)} - U(S)}{U_{i\ D|D}^{(0)} - U_{i\ D|S}^{(0)}}.$$

◇

Thus if cooperative nodes mostly interact among each other then DARWIN can resist group attacks.

2.4.4 Security Issues

In this section, we will comment on possible security issues in implementing DARWIN. Since our solutions to these issues rely on other works, our discussion will be brief.

Short Term Identities and Sybil Attacks

Nodes can change identities to avoid detection or to help spread false values to improve their own reputation. To cope with this we can use a proof-of-effort approach, first suggested for ad hoc networks in [14]: a node that claims to be entering the network for the first time must show that it has spent some effort creating its identity, otherwise it is not allowed to connect. Since memory access speeds vary across machines much less than CPU speeds, the approach uses a memory-bound function [26,27]. Its two main properties are that its computing time is determined by the memory access speed and not the CPU speed, and that it is moderately hard to compute but very easy to verify. This approach tends to be more egalitarian and tries to avoid the problem posed to the network by selfish users with high-end computers, since they could potentially spend less CPU time with the burden of creating new identities.

Node Impersonation

Selfish nodes can try to impersonate cooperative nodes in order to boost their reputation or to request other nodes to forward their own packets. This problem can be solved generating a shared secret key among each pair of nodes and using it in conjunction with a message authentication code. The key can be safely exchanged over an insecure channel using the Diffie-Hellman key exchange algorithm [28].

2.5 Simulations

In this section we first present a possible implementation of our reputation mechanism and later we will present the settings and results of our simulation study of the performance of DARWIN against the strategies studied in Section 2.3.

2.5.1 Algorithm Implementation

Let $N_i^{(k)}$ denote the set of one-hop neighbors that node i has discovered in time interval k by overhearing packet transmissions. For every node $j \in N_i^{(k)}$, node i keeps two counters, one for the number of messages sent to j for forwarding ($S_{ij}^{(k)}$) in time slot k and another for the number of messages j actually forwarded ($F_{ij}^{(k)}$) in time interval k. At the end of the time slot it computes the ratio

$$c_{ij}^{(k)} = \frac{F_{ij}^{(k)}}{S_{ij}^{(k)}}$$

and proceeds to send $c_{ij}^{(k)}$ to its neighbors. With the values gathered, node i estimates j's average connectivity ratio

$$\hat{c}_j^{(k)} = \frac{\sum\limits_{\substack{m \in N_i^{(k)} \cup \{i\} \\ m \neq j}} c_{im}^{(k)} \times c_{mj}^{(k)}}{\sum\limits_{\substack{m \in N_i^{(k)} \cup \{i\} \\ m \neq j}} c_{im}^{(k)}},$$

where by definition $c_{ii}^{(k)} = 1$ for all k. It must be noted that the average is weighted with the perceived connectivity ratio that node i measured from node m. This helps to avoid sybil attacks that spread false values in order to improve a selfish node's reputation since all its other identities have low connectivity, too, and therefore a small impact on the average. In a similar way, node i will find $\hat{c}_i^{(k)}$, the average connectivity ratio its one-hop neighborhood perceived from it during time slot k. We define $\hat{p}_j^{(k)} = 1 - \hat{c}_j^{(k)}$ and use (2.6) and (2.7) to find the dropping probability that node i will use while forwarding packets for node j in time interval $k+1$.

Since we need $\gamma < p_e^{-1}$, we need to estimate p_e. An interesting solution was proposed in [16] probing a node with anonymous messages, but it increases the overhead of the protocol. Instead, note that p_e is the probability that at least one terminal in $N_i^{(k)}$ transmits when node j transmits. Thus we estimate p_e by measuring the fraction of time during which at least one node other than j transmits. Call it $\hat{p}_{ej}$. Mathematically, if $T_j^{(k)}$ is the fraction of time during which node j has transmitted up to time interval k and $T_c^{(k)}$ is the fraction of time during which a collision occurred up to time interval k

we have:

$$\hat{p}_{ej} = T_c^{(k)} + \sum_{\substack{n \in N_i^{(k)} \\ n \neq j}} T_n^{(k)}.$$

In case the MAC layer uses a CSMA/CA protocol, and due to the exposed terminal problem, we will have that $\hat{p}_{ej} \geq p_e$. This overestimation is not a problem for our algorithm since

$$\gamma < \frac{1}{\hat{p}_{ej}} \leq \frac{1}{p_e}.$$

2.5.2 Settings

Our goal is to study the network performance of the different strategies presented in Section 2.3 and how they compare against DARWIN. To do that we used the network simulator *ns-2*. For the propagation we used the two-ray ground reflection model, while the IEEE 802.11 Distributed Coordination Function (DCF) was used at the MAC layer. Nodes had a physical radio range of 250 m and a raw bandwidth of 2 Mbps. Routing was performed by the Dynamic Source Routing (DSR) protocol. We simulated a network of 50 nodes randomly placed in an area of 670×670 m^2 that implement a reputation mechanism in a given simulation run, where we randomly selected five nodes that do not implement such strategy and behave selfishly by dropping all packets that are not destined to them. In the rest of this section, a selfish node will be taken to mean a node that does not implement the reputation mechanism and a cooperative node one that does. Unless otherwise noted, there are 14 source-destination pairs and each source transmits at a constant bit rate (CBR) of 2 packets/s, with a packet size of 512 bytes. The simulation time is 800 s, where the time intervals used are 60 s long. Each figure presents an average of 120 randomly generated runs.

Since our goal is to study the different strategies and not specific implementations, all cooperative nodes use the implementation suggested in Section 2.5.1 to test node behavior and share reputation values. The only exception is the strategy used to punish selfish behavior. For the case when nodes implement the n-step trigger strategy, the threshold T is set to be 0.2, while $n = 5$. For GTFT we set the parameter g to be 0.1, while for DARWIN we set γ to be 2.

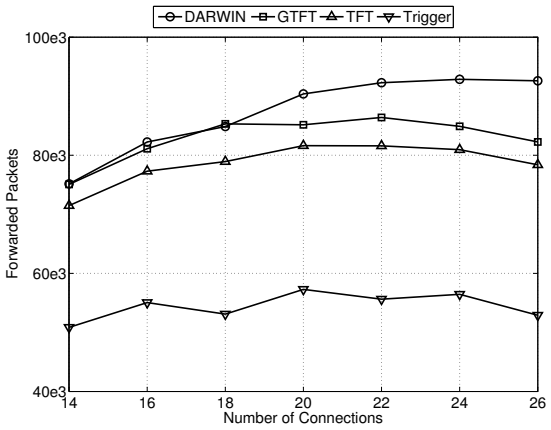

Figure 2.4: Number of forwarded packets for different numbers of source-destination pairs

2.5.3 Results

Before presenting the results of our simulation study, we would like to emphasize the fact that, as proved in Sections 2.3 and 2.4, DARWIN can help restore cooperation under a larger set of conditions on nodes interactions, is more robust against imperfect knowledge of network parameters compared to other strategies, and is a self-enforcing strategy where no node or group of colluding nodes can obtain a gain from deviating from our strategy. These desirable characteristics for any reputation mechanism mean that the chances that a rational user deviates from the strategy and behaves selfishly are smaller under DARWIN than under the other strategies studied; averting such behavior is the ultimate goal of a mechanism that tries to incentivize cooperation. The simulations complement these theoretical conclusions by assuming that some nodes are rogue users and behave selfishly.

To evaluate network performance, we measure the total number of forwarded packets. In Figure 2.4 we explore the effect of varying the total number of source-destination pairs. It is apparent that the throughput gap for cooperative nodes increases with the number of connections. The reason for this is that when there are more active connections the probability that a node does not listen when a packet is being forwarded increases, leading to an increased number of misunderstandings where cooperative nodes are deemed to be acting selfishly. This increases the level of retaliation situations in TFT and the n-step trigger strategies. It can be noted that when the

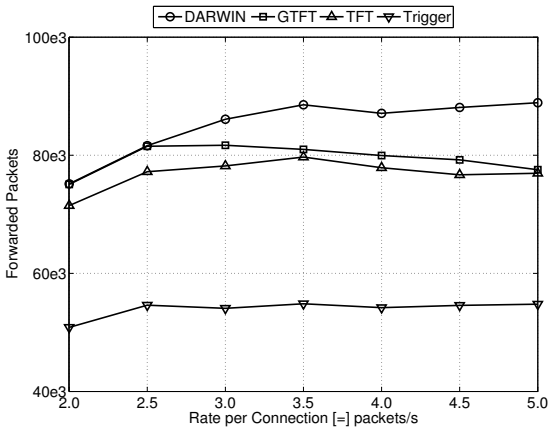

Figure 2.5: Number of forwarded packets for different connection rates (for a packet size of 512 bytes)

number of connections is greater than 18 there is a decrease in throughput in GTFT compared to DARWIN. As explained in Section 2.3.3, to achieve full cooperation GTFT requires a perfect estimation of the probability p_e that a packet that has been forwarded was not overheard by the originating node. Since we keep constant the generosity factor g in our simulations, once $p_e > g$ when the number of connections is large enough, we have that network throughput starts to decrease. This behavior is also evident in Figure 2.5, where the relationship between source rate and the number of forwarded packets is presented. Since DARWIN does not require a perfect estimate of p_e but an overestimation suffices, as explained in Section 2.4.2, and since it compensates for the misunderstandings between cooperative nodes, we see that the advantage of using DARWIN over other strategies is more apparent when the network becomes heavily congested.

Figure 2.6 explores the impact of the fraction of selfish nodes, where the figure presented is the average of 240 randomly generated runs instead of 120 as in the rest of the plots, showing the average number of forwarded packets per node for both selfish and cooperative nodes. This was done because the confidence intervals for this plot tended to be larger than for the other plots when we only used 120 runs. Since the goal of this plot is to highlight the difference in throughput between cooperative and selfish nodes, and not the throughput difference between competing strategies as it has already been studied in Figures 2.4 and 2.5, we run our simulations in the low traffic

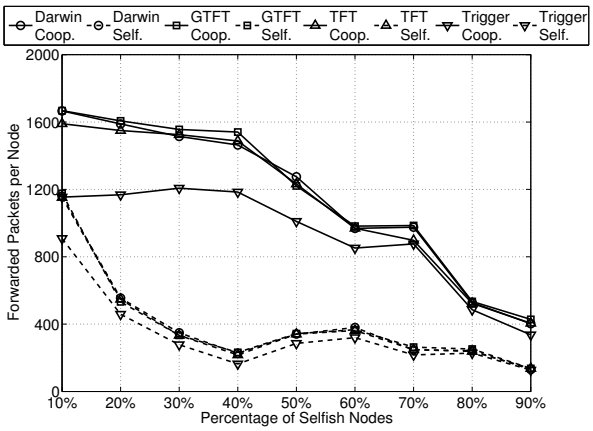

Figure 2.6: Number of forwarded packets per node for different numbers of selfish nodes

regime.

As expected, the total throughput of cooperative nodes decreases proportionally when the number of selfish nodes increases. This is due to the fact that since the number of selfish nodes increases, the total number of packets being dropped increases proportionally. In this case, it can be seen that the average number of forwarded packets for cooperative nodes is larger than the one for selfish nodes, even when 90% of the nodes act selfishly. The fact that the difference between the throughput decreases is less relevant than the fact that selfishness does not improve performance. It can also be noted that of all the strategies simulated, the n-step trigger has the harshest punishment for selfish behavior, but at the cost of significantly decreasing network throughput for cooperative nodes.

In Figure 2.7 we study the effect of mobility on the effectiveness of the punishment mechanisms, where the sources transmit at a rate of 4 packets/s. The mobility model used is the random waypoint model, where a node moves to a random destination at a speed uniformly distributed between 0 and 20 m/s, and once it reaches the destination it remains there for a specified pause time before choosing its next destination. To complement our simulation study, in this figure we include in the comparison CORE [12, 17]. As can be observed, the more the nodes move, the less throughput they get, since the routing algorithm sends packets to stalled routes, which eventually leads to packet dropping. However, the performance of DARWIN remains superior to that of

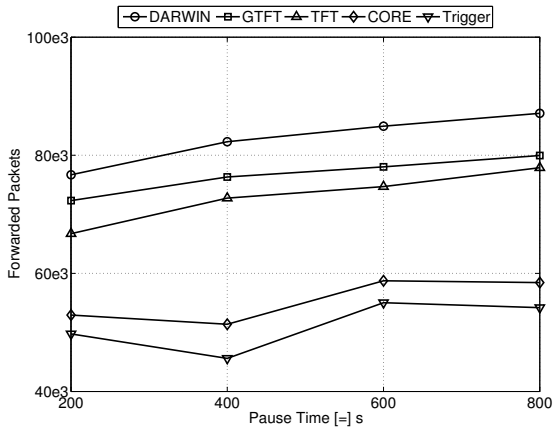

Figure 2.7: Number of forwarded packets per node for different pause times

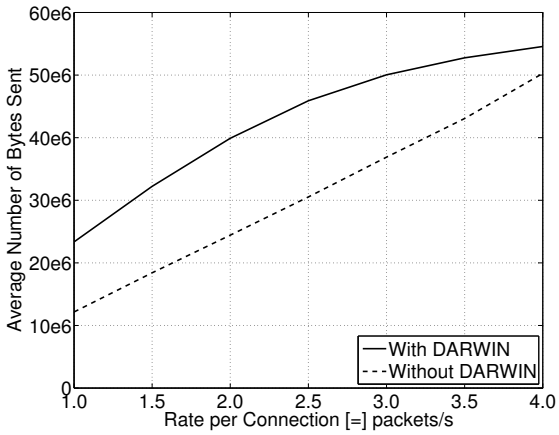

Figure 2.8: Overhead of DARWIN

the other strategies, showing that the mechanism is better able to incentivize cooperation even in the case when nodes roam.

In summary, nodes that are selfish are punished similarly by most protocols, but nodes that implement DARWIN get much better throughput than nodes that implement n-step trigger or TFT strategies. Furthermore, the throughput of DARWIN is better than that of GTFT in heavily loaded networks.

One important aspect of every protocol is the overhead that results from its implementation. Figure 2.8 explores this for different source rates when all nodes implement DARWIN compared to the same network when all nodes are cooperative and do not run DARWIN. DARWIN (and, for that matter,

any other reputation mechanism) incurs a certain fixed overhead associated with sharing and processing reputation information; thus, as expected, the fraction of overhead packets to data packets is higher at low loads but smaller at high loads. This feature is desirable since resources are at a premium at high loads.

2.6 Related Work

Incentive mechanisms can be broadly divided in two categories, according to the techniques they use to enforce cooperation: credit-based schemes and reputation-based schemes. Here we present a review of the previous work done in both areas.

2.6.1 Credit-Based Schemes

The strategy proposed in [8] is based on a nuglet counter that increases every time a node forwards a packet, and decreases by the estimated number of intermediate nodes every time a packet is sent. A node is only allowed to send a packet if its nuglet counter remains positive after the operation. Therefore, if a node wants to be able to transmit, it has to cooperate. This scheme requires tamper resistant hardware, but this kind of hardware must be trusted with caution [29]. Sprite [7] avoids the use of tamper resistant hardware by storing receipts of forwarded packets, and later they are cleared with a central trusted authority that distributes the credits to cooperative nodes. The drawback is the need for an infrastructure to operate, which may hinder its ability to gain widespread acceptance, e.g., in post-disaster networks.

An algorithm called Generous Tit For Tat (GTFT) has been proposed in [6]: a node agrees to forward packets in a session if and only if the throughput received by the node from the network so far is greater than the throughput given to the network minus a generosity factor; if a node decides to reject a session, it informs the source so it can establish another path. Assuming that misbehaving nodes do not lie about their actual actions, Srinivasan et al. [6] proved that no node has an incentive to unilaterally deviate from the GTFT strategy.

In [9], a pricing mechanism is studied through fluid-level simulations. It has been demonstrated that users' prices and credit balances stabilize for fixed ad hoc networks, where nodes in the center of the network have an advantage since they can act as relay nodes for a larger number of routes. Ad hoc VCG [10] is a reactive routing protocol that implements a variation of the VCG mechanism. As is mentioned in [10], the protocol has considerable overhead on the route discovery phase if communication sessions between two nodes are usually short or the routing path frequently changes during a session; additionally, to guarantee truthfulness and cost-efficiency it is required that every node have complete and up-to-date knowledge of the underlying graph, so techniques such as route caches are not suitable with this protocol.

2.6.2 Reputation-Based Schemes

In [11] the reputation mechanism performs two functions: (i) identifies misbehaving nodes by monitoring packet forwarding, and (ii) helps the routing protocol avoid those nodes by informing the source node that there are selfish nodes on its path. The source node can use this information to find alternate paths. Hence, the mechanism only tries to avoid selfish nodes, but the behavior is not discouraged. CONFIDANT [13] goes one step further and after a selfish node is detected, it is isolated; however, it relies on building a "friends" list that is imprinted in every node on a user-to-user basis.

In SORI [15] and Catch [16] the spreading of reputation information is limited to one-hop neighbors. SORI evaluates the reputation of a node by weighting the information of all its neighbors and then punishes it, if necessary, with a Generous Tit For Tat strategy. Catch uses control messages to reduce the impact of collisions on estimating reputation, and a trigger strategy to punish selfish behavior.

CORE [12,17] keeps a counter to keep track of the neighbor's last B actions, where the counter is increased by 1 every time the node cooperates, and decreased by 1 every time it defects. If the counter is positive, CORE will cooperate; otherwise it will punish its neighbor by defecting.

CHAPTER 3

ADMISSION CONTROL AND ROUTING IN MULTI-HOP WIRELESS NETWORKS

Future multi-hop wireless networks will carry a host of multimedia services such as voice calls and video conferencing. A common feature of such services is that they require quality of service (QoS) guarantees; specifically we consider services that require a pre-specified bandwidth between the endpoints of the flow. To support such services, the network must be equipped with a protocol to decide whether or not to accept a new request, and to find a route with sufficient bandwidth for an admitted flow. Optimal admission control and routing for pre-specified bandwidth flows has been extensively studied for wireline networks. In the case of wireless networks, a number of papers have highlighted the difficulties in designing good QoS routing algorithms. In particular, the importance of taking contention count into consideration for available bandwidth estimation has been recognized in [30, 31] and the importance of load balancing to maximize the number of admitted flows has been highlighted in [32]. However, no provably optimal algorithm has been developed to the best of our knowledge.

The idea of achieving provably good performance without any modeling assumptions on the arrival requests was proposed in [33–35], based on the concept of *competitive ratio* [36–39]. Informally, competitive ratio measures the performance loss of a given algorithm caused by imperfect decisions due to the fact that it is oblivious of the future when compared against an off-line algorithm that has complete *a priori* knowledge of the sequence of requests, including the future, and can therefore make perfect decisions. For reasons that we will describe next, it is difficult to immediately adapt the competitive ratio-based routing algorithms to wireless networks.

The wireless channel is a shared resource, so there is resource contention among transmissions from different nodes. As a result, even if a flow requests a pre-specified bandwidth from the network, the load imposed by a flow on a node is a function of the topology of the network (for example, the number

of neighbors and hidden terminals) and the choice of the route itself. Hence, unlike a wireline network where the bandwidth consumed by a flow along a link is known at the arrival time of a flow, in a wireless network, the load imposed by a flow on a node can only be determined during route discovery. Therefore, it is not immediately obvious that the techniques for deriving optimal QoS routing algorithms for wireline networks can be applied to wireless networks.

In this chapter, our contributions are as follows:

- We first develop a model for QoS routing in multi-hop wireless networks which allows us to derive an admission control and routing algorithm with provable performance guarantees using competitive analysis.
- We then show that no other algorithm performs better in an asymptotic sense to be described later. The proof of this result is more involved and relies on the unique characteristics of wireless networks.
- The optimal algorithm is not in a form that is amenable to distributed implementation. Thus, an important contribution is to convert the algorithm into a form that allows the use of standard shortest-path algorithms.

The rest of the chapter is organized as follows. Section 3.1 presents an overview of previous related work. Competitive analysis theory is presented in Section 3.2. The network model and definitions are described in Section 3.3, while in Section 3.4 we introduce our algorithm and use the competitive analysis theory to prove performance guarantees. Section 3.5 proves that our algorithm is asymptotically optimal with respect to the competitive ratio. We show how the algorithm can be implemented in a distributed fashion in Section 3.6. The proofs are included in the appendices.

3.1 Related Work

Finding algorithms to support quality of service in multi-hop networks has been an active topic of research in the last several years. Reference [31] studies the problem of bandwidth estimation at a node while [30, 40] study the problem of estimating the impact of contention in the available bandwidth

in a multi-hop path. In [41, 42] some heuristics are presented to support QoS but the effect of contention in the admission process is ignored. The work in [43] takes into account contention under the implicit assumption that the interference range of a node is equal to its transmission range.

In [44] the use of packet scheduling to guarantee QoS in multi-hop networks is studied. Some proposals rely on a central algorithm to do admission control [45, 46], while others have proposed strategies assuming a TDMA [47–49] or CDMA over TDMA [50, 51] layer. The work in [52] explores how to provide implicit synchronization in CSMA/CA networks to achieve TDM-like performance.

A solution for multichannel multi-hop networks has been proposed in [32] under the assumption that requests can be split; if requests are non-splittable a heuristic is introduced where requests are routed in the least-congested, minimum-hop path. To the best of our knowledge, our algorithm is the first provably optimal for general networks that allows a distributed implementation.

3.2 Competitive Analysis

The concept of competitive ratio was first introduced by [36] and further developed by [37–39]. Here we will present a brief overview of this theory.

In many situations we must develop efficient algorithms which have to deal with a sequence of tasks one at a time, where the efficiency of current decisions is affected by future tasks. One classical example is the well-known *ski rental problem*, where a ski enthusiast plans on skiing for several days while weather permits. Our ski fan is faced with two options every day: either rent skis for the day at a price of r or buy them at cost b. If we knew that the total number of days to ski is d, then the optimal decision algorithm would be to buy skis if $b < rd$. Since we have no knowledge of the future, we have to develop an algorithm that has to make decisions on a day by day basis and still performs well. To do that we introduce the concepts of *on-line* and *off-line* algorithm.

Definition 6. *An* on-line algorithm *is an algorithm that has to deal with a sequence of requests one at a time, without having the entire sequence available from the beginning.* ◇

Definition 7. *An* off-line algorithm *is an algorithm that has complete a priori knowledge of the entire request sequence, including future requests, before it outputs its answer to solve the problem at hand.* ◇

One way to measure the performance of an on-line algorithm is by comparing it against the best possible off-line algorithm when both have to deal with the same set of requests. The *competitive ratio* then measures the performance loss of an on-line algorithm caused by imperfect decisions when compared against an off-line algorithm that can make perfect decisions since it has complete knowledge of the request sequence.

Definition 8. *The* competitive ratio *of an on-line algorithm is the supremum over all possible request sequences of the performance ratio between the best possible off-line algorithm and the on-line algorithm.* ◇

This means that if an on-line algorithm has a competitive ratio of c then its performance is at least $1/c$ the performance of the best possible off-line algorithm for any request sequence, and for a given performance measure.

3.3 Network Model

The network is composed of a set $\mathcal{N}$ of N nodes, where node $n \in \mathcal{N}$ has capacity $u(n)$. Without loss of generality, in the rest of the chapter we will assume that

$$u(n) = 1 \ for \ all \ n \in \mathcal{N}. \tag{3.1}$$

The input to the algorithm is a set of flow requests $\mathcal{F} = \{f_1, f_2, \ldots, f_k\}$, where flow j is specified by

$$f_j = \left\{ s_j, d_j, r_j(t), t_j^S, t_j^F, \rho_j \right\}.$$

Nodes s_j and d_j are the source and destination respectively of a unidirectional flow.[1] Flow j requests a bandwidth $r_j(t)$ at time t, where we define $r_j(t) = 0$ for $t \notin [\, t_j^S, t_j^F \,)$. Thus, t_j^S and t_j^F are the start and finish times of flow j. For simplicity, and without loss of generality, we assume that these times are integers. A profit of ρ_j is accrued if the flow is admitted into the

[1]For the case of bidirectional flows, we simply need to split the request in two unidirectional flows that need to be accepted/rejected simultaneously.

network. Given $f_j \in \mathcal{F}$, our algorithm will output a path P_j assigned for the request, with the understanding that $P_j = \emptyset$ if it is rejected.

Let $\lambda_n(t)$ be the relative load on node n at time t, which is a function of the flows currently admitted by our algorithm. Flow j's holding time is denoted by $T_j = t_j^F - t_j^S$, where we define the maximum holding time

$$T = \max_j \{T_j\}.$$

As mentioned at the beginning of the chapter, the wireless channel is a shared resource and contention among transmissions from different nodes implies that when admitting a flow we must take into account the impact of a flow in the network. To better understand this, let us use the simple scenario of Fig. 3.1. Our linear network of 5 nodes is such that any node can only communicate with its nearest neighbors, and the interference range for each node is illustrated as a concentric circle around each node. There is a flow from node B to node D that requires a rate of r. Due to the exposed terminal problem, node A cannot transmit while B is transmitting, so node A's available capacity is

$$u(A) - r \overset{(3.1)}{=} 1 - r,$$

where $\overset{(3.1)}{=}$ means that the equality follows from equation (3.1). We use this notation throughout the chapter.

Thus, when scheduling this flow, we must reserve a rate of r in node A for it to remain idle. Similarly, node B must transmit at a rate r and must remain silent while C is relaying a packet for this flow, which implies a resource reservation of rate $2r$ at B. Because of the hidden terminal problem, node E must remain idle while D is receiving a packet, so we have to reserve a rate of r at this node. Following the lines of this argument, it can be checked that we must reserve a rate of $2r$ at nodes C and D in order to be able to schedule this flow. These values are shown above each node in Fig. 3.1.

This example gives the intuition for the following definition. Let $Q_n(P_j)$ be the impact of flow j on node n if path P_j is used. Formally,

$$Q_n(P_j) = \sum_{n' \in \mathcal{N}_n} I_{\{n' \in P_j\}} \left(I_{\{n' \neq d_j\}} + I_{\{n' \in HT_n(P_j)\}} \right), \tag{3.2}$$

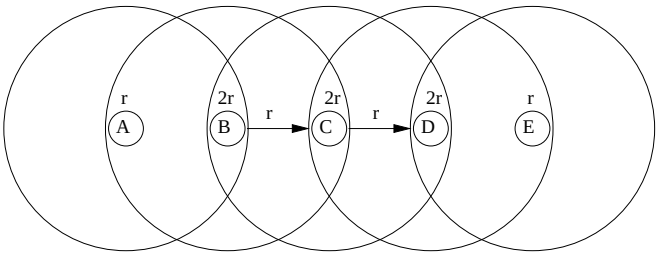

Figure 3.1: Example of resource contention among nodes

where $\mathcal{N}_n$ is the set of nodes within interference range of node n (including node n itself), $HT_n(P_j)$ is the set of nodes in $\mathcal{N}_n$ that need to receive a transmission for flow j that node n cannot sense—that is, hidden terminal transmissions—and $I_{\{\}}$ is the indicator function defined as

$$I_{\{statement\}} = \begin{cases} 1 & if \quad statement = TRUE \\ 0 & if \quad statement = FALSE \end{cases}.$$

As an example, in Fig. 3.1 we have that for path $P = \{B, C, D\}$, $Q_A(P) = 1$, $Q_C(P) = 2$, and $Q_E(P) = 1$. It must be noted that this definition is only an upper bound on the actual impact of a flow in a given node, and can only be improved by assuming a specific, possibly ideal, transmission scheduling algorithm. To illustrate this, in Fig. 3.2 we assume that there is a flow from node A to E and that we schedule node transmissions such that nodes A and D simultaneously transmit at time t_0, node B transmits at t_1, and C is scheduled to transmit at time t_2. In this case, the impact of the flow on node C is 3 instead of 4, as estimated using $Q_n(P_j)$.[2]

Finding methods for estimating $Q_n(P_j)$ has been an active topic of research. For related work, the reader is referred to [30, 40, 43].

Define $Q_T(P_j)$ to be the total impact of flow j (routed on path P_j) on the network. Thus,

$$Q_T(P_j) = \sum_{n \in \mathcal{N}} Q_n(P_j). \tag{3.3}$$

[2]It must be highlighted that, for ease of explanation, we have assumed that due to the exposed terminal problem a node cannot transmit while another in its interference neighborhood is transmitting. If a certain physical layer technology does not preclude such transmissions, the definition of $Q_n(P_j)$ should be modified accordingly and the results in the rest of the chapter still apply with minor modifications.

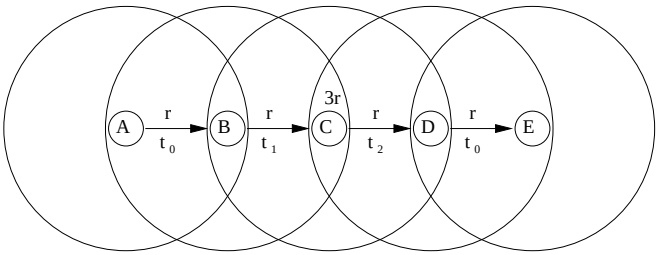

Figure 3.2: Example of resource contention among nodes under the assumption of perfect packet transmission scheduling

Additionally, define Q_T and Q as follows:

$$Q_T = \max_j \{Q_T(P_j)\}$$

$$Q = \max_{j,n} \{Q_n(P_j)\}. \tag{3.4}$$

We normalize the cost such that, for any flow $f_j \in \mathcal{F}$ and any time such that $r_j(t) > 0$,

$$1 \leq \frac{\rho_j}{Q_T r_j(t) T_j} \leq F \tag{3.5}$$

for F large enough. For example, if we have that $r_j(t) = r_j$ for $t \in [\, t_j^S, t_j^F\,)$ and the profit is defined to be proportional to the bandwidth-holding time product, i.e. throughput, then we can make $\rho_j = Q_T r_j T_j$ and let $F = 1$.

Finally, we assume that rate requirements are small enough compared to node capacity. Specifically,

$$r_j(t) \leq \frac{\min_n \{u(n)\}}{Q \log \mu} \overset{(3.1)}{=} \frac{1}{Q \log \mu} \tag{3.6}$$

where

$$\mu = 2\,(1 + Q_T T F)\,, \tag{3.7}$$

and log means $\log_2$.

Later we will prove in Section 3.5 that (3.6) is a necessary condition for any algorithm to achieve logarithmic competitive ratio.

3.4 Algorithm

The main goal is to develop an admission control and routing algorithm that enforces capacity constraints, that is

$$\lambda_n(t) \leq 1 \; for \; all \; t \; and \; n \in \mathcal{N}, \tag{3.8}$$

and maximizes profit:

$$\sum_{j:P_j \neq \emptyset} \rho_j.$$

Furthermore, the algorithm cannot rely on knowledge about future flows to make admission decisions and once a flow has been admitted no rerouting is allowed and no flow is to be interrupted. To do that, it will sequentially analyze flows from $\mathcal{F}$ and decide whether to admit them or not.

3.4.1 Definition

The decision rule for admitting flow $f_j \in \mathcal{F}$ and assigning a path is:

1. For all $t \in [\, t_j^S, t_j^F \,)$, $n \in \mathcal{N}$ let the cost of node n at time t be

$$c_n(t) = u(n) \left[\mu^{\lambda_n(t)} - 1\right] \overset{(3.1)}{=} \mu^{\lambda_n(t)} - 1. \tag{3.9}$$

2. If there exists a path P_j from node s_j to d_j such that

$$\sum_{n \in \mathcal{N}} \sum_{t_j^S \leq t < t_j^F} Q_n(P_j) \frac{r_j(t)}{u(n)} c_n(t) \leq \rho_j \tag{3.10}$$

then route f_j using P_j and set

$$\lambda_n(t) \leftarrow \lambda_n(t) + Q_n(P_j) \frac{r_j(t)}{u(n)} \tag{3.11}$$

for all $n \in \mathcal{N}$, $t_j^S \leq t < t_j^F$.

Note that

$$Q_n(P_j) \frac{r_j(t)}{u(n)}$$

is the fraction of node n's capacity that would be used by flow j. Thus, the algorithm compares the link cost weighted by this quantity to the profit and admits the call if the cost is less than or equal to the profit.

3.4.2 Performance Guarantees

Now that the wireless model and the algorithm are defined, we can use the techniques developed for wireline networks in [33]. We will first proceed to prove that our algorithm enforces capacity constraints, which from now on we will call the *Admission Control and Routing (ACR) algorithm*. In other words, if the ACR algorithm decides to admit a flow request, then there is sufficient available capacity.

Lemma 4. *Capacity constraints are enforced by the ACR algorithm. That is,*

$$\lambda_n(t) \leq 1 \ for \ all \ t \ and \ n \in \mathcal{N}.$$

◇

(See the proof in Appendix C.)

The proof of the competitiveness of the ACR algorithm is done in two steps. In Lemma 5 we prove that the total network cost is at most within a factor of the accrued gain, and in Lemma 6 we prove that the profit due to requests rejected by the ACR algorithm and accepted by the optimal off-line algorithm is bounded by the total network cost. These two results then imply that the profit of the ACR algorithm is within a factor of the profit accrued by the off-line algorithm, and hence the competitive ratio is bounded.

For the following two lemmas, define $\lambda_n(t, j)$ to be the relative load on node n at time t when only the first $j - 1$ flow requests have been either admitted or rejected. That is,

$$\lambda_n(t, j) \overset{(3.11)}{=} \sum_{i<j} Q_n(P_i) \frac{r_i(t)}{u(n)} \overset{(3.1)}{=} \sum_{i<j} Q_n(P_i) r_i(t), \tag{3.12}$$

with the understanding that $P_i = \emptyset$ if $f_i \in \{f_1, f_2, \ldots, f_{j-1}\}$ is rejected.

Similarly, and from (3.9), let $c_n(t, j)$ be defined as

$$c_n(t, j) = \mu^{\lambda_n(t,j)} - 1. \tag{3.13}$$

Lemma 5. *Let $\mathcal{A}_{ACR}$ be the set of indices of accepted flows by the ACR algorithm and k be the index of the last flow request in $\mathcal{F}$. Then,*

$$2\log\mu \sum_{j\in\mathcal{A}_{ACR}} \rho_j \geq \sum_t \sum_n c_n(t, k+1).$$

◇

(See the proof in Appendix D.)

Lemma 6. *Let $\mathcal{A}_{O\setminus A}$ be the set of indices of accepted requests by the optimal off-line algorithm but rejected by the ACR algorithm. Let $m = \max\{\mathcal{A}_{O\setminus A}\}$. Then*

$$\sum_{j\in\mathcal{A}_{O\setminus A}} \rho_j \leq \sum_n \sum_t c_n(t, m).$$

◇

(See the proof in Appendix E.)

Now, we are ready to prove the following:

Theorem 4. *The ACR algorithm enforces capacity constraints and achieves a competitive ratio of $O(\log(Q_T TF))$.* ◇

(See the proof in Appendix F.)

Remark: It is important to highlight that for the lemmas and theorem of this section we only assume that $Q_n(P_j) \geq 1$, but the precise definition given in (3.2), which depends on the assumptions about the physical layer, is only used in the following sections.

3.5 Optimality

Now we will prove that no other algorithm can achieve a better competitive ratio than the ACR algorithm in an asymptotic sense and that assumption (3.6) is a necessary condition to achieve a good competitive ratio. To do that, we will first show that even if flow rates are small enough, the profit of the optimal off-line algorithm will exceed the profit of any on-line algorithm that is oblivious to the future by a factor of $\Omega(\log(Q_T TF))$. Finally, we will present stronger bounds for the case when flow rates are allowed to be

relatively large, showing that the competitive ratio degrades when we allow large rates.

The techniques used here are again similar to the ones developed for wireline networks in [33], but rely on the unique characteristics of wireless networks, especially to find worst case scenarios in Lemmas 7 and 10.

Lemma 7. *Any on-line algorithm has a competitive ratio of* $\Omega(\log Q_T)$. $\diamond$

(See the proof in Appendix G.)

Lemma 8. *Any on-line algorithm has a competitive ratio of* $\Omega(\log T)$. $\diamond$

(See the proof in Appendix H.)

Lemma 9. *Any on-line algorithm has a competitive ratio of* $\Omega(\log F)$. $\diamond$

(See the proof in Appendix I.)

Hence, we have just proved the following.

Theorem 5. *Any on-line algorithm has competitive ratio of* $\Omega(\log(Q_T T F))$. $\diamond$

Proof. It is a direct consequence of Lemmas 7, 8 and 9. $\square$

For the proof of Theorem 5 we assume that (3.6) holds. We will now proceed to prove that if this bound does not hold then no on-line algorithm can achieve logarithmic competitive ratio.

Lemma 10. *If we allow requests of rate up to* $\frac{1}{4k}$ *then the competitive ratio is* $\Omega(Q_T^{\frac{1}{4k}})$ *for any positive integer* k. $\diamond$

(See the proof in Appendix J.)

Lemma 11. *If we allow requests of rate up to* $\frac{1}{k}$ *then the competitive ratio is* $\Omega(T^{\frac{1}{k}})$ *for any positive integer* k. $\diamond$

(See the proof in Appendix K.)

Lemma 12. *If we allow requests of rate up to* $\frac{1}{k}$ *then the competitive ratio is* $\Omega(F^{\frac{1}{k}})$ *for any positive integer* k. $\diamond$

(See the proof in Appendix L.)

Therefore, we have the following theorem.

Theorem 6. *If we allow requests of rate up to* $\frac{1}{4k}$ *then the competitive ratio is* $\Omega(Q_T^{\frac{1}{4k}} + T^{\frac{1}{4k}} + F^{\frac{1}{4k}})$ *for any positive integer* k. $\diamond$

Proof. This follows from Lemmas 10, 11 and 12. $\square$

Thus, in order to achieve logarithmic competitive ratio we need to let k be greater than

$$\frac{\log(Q_T TF)}{4\log(\log(Q_T TF))}.$$

3.6 Distributed Implementation

In its present form, checking

$$\sum_{n\in\mathcal{N}} \sum_{t_j^S \le t < t_j^F} Q_n(P_j) \frac{r_j(t)}{u(n)} c_n(t) \le \rho_j$$

in the ACR algorithm requires first specifying a path P_j from source to destination, and then its associated cost can be calculated. Since we ideally would like to use the minimum cost path, this means that it would be required to first identify all possible paths and then find the cost for each one.

The contribution of this section is to show how this can be implemented using a distributed algorithm that can find the minimum cost path without first identifying all possible solutions, and where every node only needs to get access to information from a local neighborhood.

Define the directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is the set of nodes in the wireless network and $\mathcal{E}$ is the set of all directed transmission edges. Formally, $e \in \mathcal{E}$ if and only if $e = (s(e), d(e))$, where $s(e), d(e) \in \mathcal{N}$, $s(e) \neq d(e)$, and node $s(e)$ can transmit to node $d(e)$.

Furthermore, define $c_n^a(j)$ to be the aggregate cost that node $n \in \mathcal{N}$ has during the holding time of flow request $f_j \in \mathcal{F}$. That is,

$$c_n^a(j) = \sum_{t_j^S \le t < t_j^F} \frac{r_j(t)}{u(n)} c_n(t).$$

With these definitions we have the following:

$$\begin{aligned}
&\sum_{n\in\mathcal{N}}\sum_{t_j^S\le t<t_j^F} Q_n(P_j)\frac{r_j(t)}{u(n)}c_n(t) = \sum_{n\in\mathcal{N}} Q_n(P_j)c_n^a(j) \\
&\overset{(3.2)}{=} \sum_{n\in\mathcal{N}}\sum_{n'\in\mathcal{N}_n} I_{\{n'\in P_j\}}c_n^a(j)\left[I_{\{n'\neq d_j\}} + I_{\{n'\in HT_n(P_j)\}}\right] \\
&= \sum_{e\in P_j}\sum_{n\in\mathcal{N}_{s(e)}\bigcup\mathcal{N}_{d(e)}} c_n^a(j) \qquad (3.14) \\
&= \sum_{e\in P_j} C_e(j),
\end{aligned}$$

where (3.14) is simply summation reordering and

$$C_e(j) = \sum_{n\in\mathcal{N}_{s(e)}\bigcup\mathcal{N}_{d(e)}} c_n^a(j) \qquad (3.15)$$

is the cost of using edge e for routing flow f_j. It must be noted that (3.15) decouples the cost of an edge from the path cost, allowing a distributed implementation of a shortest path algorithm to find the optimal route. Furthermore, for every edge $e \in \mathcal{E}$ we only need to gather information from the local set $\mathcal{N}_{s(e)}\bigcup\mathcal{N}_{d(e)}$ to find $C_e(j)$. It must be noted that for *any* admission algorithm this is the minimal information that must be gathered and updated in order to check resource availability, since the transmission in this link will affect the load of all nodes in the set $\mathcal{N}_{s(e)}\bigcup\mathcal{N}_{d(e)}$.

CHAPTER 4

OPTIMAL SCHEDULING FOR FAIR RESOURCE ALLOCATION IN AD HOC NETWORKS WITH ELASTIC AND INELASTIC TRAFFIC

As wireless networks become more prevalent, they will be expected to support a wide variety of services, including best-effort and real-time traffic. Such networks will have to serve flows that require quality of service requirements, such as minimum bandwidth and maximum delay constraints, while at the same time keeping the network queues stable for data traffic and guaranteeing throughput optimality. For the case of wireless networks with best-effort traffic only, optimization-based algorithms which naturally map into different layers of the protocol stack have been proposed in the last few years [53–58]; see [59] for a survey. However, these models do not take into account strict per-packet delay bounds.

Scheduling packets with strict deadlines has been studied in [60–63], but all of these papers provide approximate solutions. The model that we study in this paper builds upon the recent work in [2,3,64] on admission control and scheduling for inelastic flows in collocated wireless networks, i.e., networks where all links interfere with each other. Among the many contributions in these papers is a key modeling innovation whereby the network is studied in frames, where a frame is a contiguous set of time-slots of fixed duration. Packets with deadlines are assumed to arrive at the beginning of a frame and have to be served by the end of the frame. In this chapter, we explore this modeling paradigm further to study the design of resource allocation algorithms for ad hoc networks. The frame-based model allows us to incorporate delay deadlines in the optimization framework for very general network models, and somewhat surprisingly, allows us to design a common framework for handling both elastic and inelastic flows.

The main contributions of the paper are as follows:

1. We present an optimization framework for resource allocation in a wireless network consisting of both best-effort flows and flows that generate

traffic with per-packet delay constraints. The framework allows for very general interference, channel and arrival models.

2. Using a dual decomposition approach, we derive an optimal scheduling and congestion control algorithm that fairly allocates resources and ensures that a required fraction of each inelastic flow's packets are delivered on time by appealing to connections between Lagrange multipliers, queues, and service deficits. The scheduling algorithm seamlessly integrates inelastic and elastic traffic into an unified max-weight scheduling framework, extending the well-known results in [1].

3. The convergence of the above algorithm in an appropriate stochastic sense is proved and it is also shown that the network is stable.

4. We prove that the use of per-slot feedback does not increase the capacity region for stationary policies, but it can potentially simplify the optimal scheduling algorithm.

4.1 Network Model

The network is represented by a directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{L})$, where $\mathcal{N}$ is the set of nodes and $\mathcal{L}$ is the set of directional links such that for all $n_1, n_2 \in \mathcal{N}$ if $(n_1, n_2) \in \mathcal{L}$ then node n_1 can transmit to node n_2. The links are numbered 1 through $|\mathcal{L}|$, and by abusing notation, we sometimes use $l \in \mathcal{L}$ to mean $l \in \{1, 2, \ldots, |\mathcal{L}|\}$.

Traffic is assumed to be a mixture of elastic and inelastic flows, where an inelastic flow is one that has maximum per-packet delay requirements. In contrast, elastic flows do not have such requirements.

Time is divided in slots, where a set of T consecutive time slots makes a *frame*. We assume that packet arrivals only occur at the beginning of a frame, and every inelastic packet has a deadline of T time slots. If a packet misses its deadline it is discarded, and it is required that the loss probability at link $l \in \mathcal{L}$ due to deadline expiry must be no more than p_l. For elastic traffic we associate a utility function $U_l(x_l)$ which is a function of the mean elastic arrival rate per frame x_l. We assume that $U_l(.)$ is a concave function.

For a given frame, we denote by the vector $a_i = (a_{il})_{l \in \mathcal{L}}$ the number of inelastic packet arrivals at every link, where a_{il} is a random variable with

mean λ_l and variance σ_{il}^2. We further assume that arrivals are independent between different frames and that $Pr(a_{il} = 0) > 0$ and $Pr(a_{il} = 1) > 0$. The last two assumptions are used to guarantee that the Markov chain we define later is both irreducible and aperiodic, although these can be replaced by other similar assumptions. Similarly, we define $a_e = (a_{el})_{l \in \mathcal{L}}$ to be the number of elastic packet arrivals at every link in a given frame.

The channel state is assumed to be independent between different frames, and independent of arrivals. This paper studies two cases: first, it is assumed that the channel state is fixed for any given frame; second, the channel state is allowed to change from time slot to time slot. In the fixed channel case, the vector $c = (c_l)_{l \in \mathcal{L}}$ denotes the number of packets link l can successfully transmit on a time slot in a given frame. In the case of the channel state changing every time slot, the matrix $c = (c_{l,t})$ denotes the number of packets link $l \in \mathcal{L}$ can successfully transmit on time slot $t \in \{1, 2, \ldots, T\}$ for a given frame.

Depending on the wireless technology used, we can have some channel feedback before or after a transmission occurs. If channel estimation is performed before transmitting, we can determine the optimal rate at which we can successfully transmit. Alternatively, feedback from the receiver after the transmission can be used to detect if a transmission is successful or not. In this paper we try to capture the different scenarios in the following cases:

1. Known channel state: It is assumed that c_l is a *non-negative* random variable with mean $\bar{c}_l$ and variance σ_{cl}^2, and we get to know the channel state at the *beginning* of the frame.

2. Unknown channel state, per-frame feedback: It is assumed that c_l is a *Bernoulli* random variable with mean $\bar{c}_l$ and we only get to know the channel state at the *end* of the frame.

3. Unknown channel state, per-slot feedback: It is assumed that $c_{l,t}$ is a *Bernoulli* random variable with mean $\bar{c}_l$ and we get to know the channel state at the *end* of the time slot.

In the known channel state case where we do channel estimation to determine the optimal transmission rate, we can potentially send more than one packet in a time slot at higher rates. This is captured by the fact that we

make no assumptions on the values c_l can take since it will be determined by the particular wireless technology used. In the case of unknown channel state we assume that we only get the binary feedback of acknowledgments, which is reflected in the Bernoulli assumption on c_l and $c_{l,t}$. In this case, and without any loss of generality, we assume only one packet can be transmitted per time slot per link.

4.2 Known Channel State

4.2.1 Problem Formulation

We first formulate the problem as a static optimization problem. Using decomposition theory, we will then obtain a dynamic solution to this problem and prove its stability using stochastic Lyapunov techniques.

A feasible schedule $s = (s_{il,t}, s_{el,t})$ is such that $s_{il,t}$, $s_{el,t}$ respectively denote the number of inelastic and elastic packets that can be scheduled for transmission at link $l \in \mathcal{L}$ and time slot $t \in \{1, 2, \ldots, T\}$; thus, $s_{il,t} + s_{el,t} > 0$ means that link l is scheduled to transmit in time slot t of the frame. Furthermore, for any t, if $s_{il_1,t} + s_{el_1,t} > 0$ and $s_{il_2,t} + s_{el_2,t} > 0$ then links l_1 and l_2 can be scheduled to simultaneously transmit without interfering with each other. Assuming the inelastic arrivals and the channel state are given by a_i and c respectively, we have the following constraints:

$$\sum_{t=1}^{T} s_{il,t} \leq a_{il} \text{ for all } l \in \mathcal{L} \text{ and} \tag{4.1}$$

$$s_{il,t} + s_{el,t} \leq c_l \text{ for all } l \in \mathcal{L} \text{ and } t \in \{1, 2, \ldots, T\}. \tag{4.2}$$

We denote by $\mathcal{S}(a_i, c)$ the set of all feasible schedules when the arrival state is a_i and the channel state is c; thus, $\mathcal{S}(a_i, c)$ captures any interference constraints we have on our network and satisfies (4.1) and (4.2).

At the beginning of any frame we must choose a feasible schedule to serve all links and decide how many elastic packets are allowed to be injected in the network. Therefore, our goal is to find a function $Pr(s|a_i, c)$ which is the probability of using schedule $s \in \mathcal{S}(a_i, c)$ when the inelastic arrivals are given by a_i and the channel state is c, subject to the constraint that the loss

probability at link $l \in \mathcal{L}$ due to deadline expiry cannot exceed p_l. For elastic traffic, we want to select the vector a_e such that we maximize the network utility while keeping the queues stable.

To properly formulate the problem, let us first define $\mu_i(a_i, c)$ to be the expected number of inelastic packets served if the number of packet arrivals is given by a_i and the channel state is c. Similarly, $\mu_e(a_i, c)$ denotes the expected number of elastic packets that can be served. Therefore, we have the following constraints:

$$\mu_{il}(a_i, c) \leq \sum_{s \in \mathcal{S}(a_i,c)} \sum_{t=1}^{T} s_{il,t} Pr(s|a_i, c)$$

$$\mu_{el}(a_i, c) \leq \sum_{s \in \mathcal{S}(a_i,c)} \sum_{t=1}^{T} s_{el,t} Pr(s|a_i, c).$$

The expected service for mixed traffic at link l is then given by

$$\mu_{il} \stackrel{def}{=} \sum_{a_i} \sum_{c} \mu_{il}(a_i, c) Pr(c) Pr(a_i)$$

$$\mu_{el} \stackrel{def}{=} \sum_{a_i} \sum_{c} \mu_{el}(a_i, c) Pr(c) Pr(a_i)$$

and due to QoS requirements and capacity constraints, we require that

$$\mu_{il} \geq \lambda_l(1 - p_l) \text{ and } x_l \leq \mu_{el}.$$

We will focus on maximizing the following objective for some given vector $w \in \mathbb{R}_+^{|\mathcal{L}|}$:

$$\max_{\substack{\mu_i(a_i,c), \mu_e(a_i,c), \\ \mu_i, \mu_e, x, Pr(s|a_i,c)}} \sum_{l \in \mathcal{L}} U_l(x_l) + w_l \mu_{il} \tag{4.3}$$

subject to

$$\mu_{il}(a_i, c) \leq \sum_{s} \sum_{t=1}^{T} s_{il,t} Pr(s|a_i, c) \text{ for all } l \in \mathcal{L}, a_i, c$$

$$\mu_{el}(a_i, c) \leq \sum_{s} \sum_{t=1}^{T} s_{el,t} Pr(s|a_i, c) \text{ for all } l \in \mathcal{L}, a_i, c$$

$$\mu_{il} = \sum_{a_i} \sum_{c} \mu_{il}(a_i, c) Pr(c) Pr(a_i) \text{ for all } l \in \mathcal{L}$$

$$\mu_{el} = \sum_{a_i} \sum_{c} \mu_{el}(a_i, c) Pr(c) Pr(a_i) \text{ for all } l \in \mathcal{L}$$

$$\mu_{il} \geq \lambda_l (1 - p_l) \text{ for all } l \in \mathcal{L}$$

$$0 \leq x_l \leq \mu_{el} \text{ for all } l \in \mathcal{L}$$

$$Pr(s|a_i, c) \geq 0 \text{ for all } s \in \mathcal{S}(a_i, c), a_i, c$$

$$\sum_{s} Pr(s|a_i, c) \leq 1 \text{ for all } a_i, c.$$

The vector w can be used to allocate additional bandwidth fairly to inelastic flows beyond what is required to meet their QoS needs. Other uses for w will be explored in the simulations section. We will assume that the arrivals and loss probability requirements are feasible and thus the optimization problem has a solution (x^*, μ_i^*).

4.2.2 Solution Using Dual Decomposition

Using the definition of the dual function [65], we have that $D(\delta_i, \delta_e) =$

$$\max_{\substack{\mu_i(a_i,c), \mu_e(a_i,c), \\ \mu_i, \mu_e, x, Pr(s|a_i,c)}} \sum_{l \in \mathcal{L}} U_l(x_l) + w_l \mu_{il} - \delta_{el}[x_l - \mu_{el}] - \delta_{il}[\lambda_l(1 - p_l) - \mu_{il}]$$

subject to

$$\mu_{il}(a_i, c) \leq \sum_{s} \sum_{t=1}^{T} s_{il,t} Pr(s|a_i, c) \text{ for all } l \in \mathcal{L}, a_i, c$$

$$\mu_{el}(a_i, c) \leq \sum_{s} \sum_{t=1}^{T} s_{el,t} Pr(s|a_i, c) \text{ for all } l \in \mathcal{L}, a_i, c$$

$$\mu_{il} = \sum_{a_i} \sum_{c} \mu_{il}(a_i, c) Pr(c) Pr(a_i) \text{ for all } l \in \mathcal{L}$$

$$\mu_{el} = \sum_{a_i} \sum_{c} \mu_{el}(a_i, c) Pr(c) Pr(a_i) \text{ for all } l \in \mathcal{L}$$

$$x_l \geq 0 \text{ for all } l \in \mathcal{L}$$

$$Pr(s|a_i, c) \geq 0 \text{ for all } s \in \mathcal{S}(a_i, c), a_i, c$$

$$\sum_s Pr(s|a_i, c) \leq 1 \text{ for all } a_i, c.$$

Slater's condition [66] states that, since the objective is concave and the constraints are affine functions, the duality gap is zero and therefore $D(\delta_i^*, \delta_e^*) = \sum_{l \in \mathcal{L}} U_l(x_l^*) + w_l \mu_{il}^*$, where

$$(\delta_i^*, \delta_e^*) \in \underset{\delta_{il} \geq 0, \delta_{el} \geq 0}{\arg\min}\ D(\delta_i, \delta_e).$$

We are interested in finding (x^*, μ_i^*) but not the value $D(\delta_i^*, \delta_e^*)$, so if we rewrite the objective in the dual function as

$$\max_{\substack{\mu_i(a_i,c), \mu_e(a_i,c), \\ \mu_i, \mu_e, x, Pr(s|a_i,c)}} \left\{ \begin{array}{l} \sum_{l \in \mathcal{L}} U_l(x_l) - \delta_{el} x_l \\ + \sum_{l \in \mathcal{L}} (w_l + \delta_{il}) \mu_{il} + \delta_{el} \mu_{el} \\ - \sum_{l \in \mathcal{L}} \delta_{il} \lambda_l (1 - p_l) \end{array} \right\}$$

we notice that the problem can be decomposed into the following subproblems:

$$\max_{x_l \geq 0} U_l(x_l) - \delta_{el} x_l$$

and

$$\max_{\substack{\mu_i(a_i,c), \mu_e(a_i,c), \\ \mu_i, \mu_e, Pr(s|a_i,c)}} \sum_{l \in \mathcal{L}} (w_l + \delta_{il}) \mu_{il} + \delta_{el} \mu_{el} \tag{4.4}$$

subject to

$$\mu_{il}(a_i, c) \leq \sum_s \sum_{t=1}^{T} s_{il,t} Pr(s|a_i, c) \text{ for all } l \in \mathcal{L}, a_i, c$$

$$\mu_{el}(a_i, c) \leq \sum_s \sum_{t=1}^{T} s_{el,t} Pr(s|a_i, c) \text{ for all } l \in \mathcal{L}, a_i, c$$

$$\mu_{il} = \sum_{a_i} \sum_c \mu_{il}(a_i, c) Pr(c) Pr(a_i) \text{ for all } l \in \mathcal{L}$$

$$\mu_{el} = \sum_{a_i} \sum_c \mu_{el}(a_i, c) Pr(c) Pr(a_i) \text{ for all } l \in \mathcal{L}$$

$$Pr(s|a_i, c) \geq 0 \text{ for all } s \in \mathcal{S}(a_i, c), a_i, c$$

$$\sum_s Pr(s|a_i, c) \leq 1 \text{ for all } a_i, c.$$

Furthermore, since we are interested in solving the problem for non-negative values of δ_{il} and δ_{el}, it must be the case that μ_i^* and μ_e^* are as large as possible, and since the upper bounds for $\mu_{il}^*(a_i, c)$ and $\mu_{el}^*(a_i, c)$ are expressed as a convex combination, and the objective function in (4.4) is linear, the problem can be decomposed into the following subproblems for fixed a_i and c:

$$\max_{s \in \mathcal{S}(a_i,c)} \sum_{l \in \mathcal{L}} \left[(w_l + \delta_{il}) \sum_{t=1}^{T} s_{il,t} + \delta_{el} \sum_{t=1}^{T} s_{el,t} \right].$$

This suggests the following iterative algorithm to find the solution to our optimization problem, where k is the step index and $X_{max} > \max_{l \in \mathcal{L}} x_l^*$ is a fixed parameter:

$$\tilde{x}_l^*(k) \in \underset{0 \leq x_l \leq X_{max}}{\arg\max}\ U_l(x_l) - \delta_{el}(k) x_l$$

$$\tilde{s}^*(a_i, c, k) \in$$
$$\underset{s \in \mathcal{S}(a_i,c)}{\arg\max} \sum_{l \in \mathcal{L}} \left\{ [w_l + \delta_{il}(k)] \sum_{t=1}^{T} s_{il,t} + \delta_{el}(k) \sum_{t=1}^{T} s_{el,t} \right\}$$

$$\tilde{\mu}_{il}^*(k) = \sum_{a_i} \sum_{c} \sum_{t=1}^{T} \tilde{s}_{il,t}^*(a_i, c, k) Pr(c) Pr(a_i)$$

$$\tilde{\mu}_{el}^*(k) = \sum_{a_i} \sum_{c} \sum_{t=1}^{T} \tilde{s}_{el,t}^*(a_i, c, k) Pr(c) Pr(a_i).$$

We update the Lagrange multipliers $\delta_i(k)$, $\delta_e(k)$ at every step according to the following equations:

$$\delta_{il}(k+1) = \{\delta_{il}(k) + \epsilon[\lambda_l(1 - p_l) - \tilde{\mu}_{il}^*(k)]\}^+$$

and

$$\delta_{el}(k+1) = \{\delta_{el}(k) + \epsilon[\tilde{x}_l^*(k) - \tilde{\mu}_{el}^*(k)]\}^+$$

where $\epsilon > 0$ is a fixed step-size parameter, and for any $\alpha \in \mathbb{R}$, $\alpha^+ \stackrel{def}{=} \max\{\alpha, 0\}$.

Making the change of variables $\epsilon \hat{d}(k) = \delta_i(k)$ and $\epsilon \hat{q}(k) = \delta_e(k)$, we have

that our iterative algorithm can be rewritten as

$$\tilde{x}_l^*(k) \in \underset{0 \le x_l \le X_{max}}{\arg\max} \frac{1}{\epsilon} U_l(x_l) - \hat{q}_l(k) x_l$$

$$\tilde{s}^*(a_i, c, k) \in$$
$$\underset{s \in \mathcal{S}(a_i,c)}{\arg\max} \sum_{l \in \mathcal{L}} \left\{ \left[\frac{1}{\epsilon} w_l + \hat{d}_l(k) \right] \sum_{t=1}^{T} s_{il,t} + \hat{q}_l(k) \sum_{t=1}^{T} s_{el,t} \right\}$$

$$\tilde{\mu}_{il}^*(k) = \sum_{a_i} \sum_{c} \sum_{t=1}^{T} \tilde{s}_{il,t}^*(a_i, c, k) Pr(c) Pr(a_i)$$

$$\tilde{\mu}_{el}^*(k) = \sum_{a_i} \sum_{c} \sum_{t=1}^{T} \tilde{s}_{el,t}^*(a_i, c, k) Pr(c) Pr(a_i)$$

with update equations

$$\hat{d}_l(k+1) = [\hat{d}_l(k) + \lambda_l(1 - p_l) - \tilde{\mu}_{il}^*(k)]^+$$

$$\hat{q}_l(k+1) = [\hat{q}_l(k) + \tilde{x}_l^*(k) - \tilde{\mu}_{el}^*(k)]^+.$$

It should be noted that due to the change of variables $\hat{d}_l(k)$ can be interpreted as a queue that has $\lambda_l(1 - p_l)$ arrivals and $\tilde{\mu}_{il}^*(k)$ departures at step k; $\hat{q}_l(k)$ can have a similar queue interpretation. The dual decomposition approach only provides an intuition behind the solution but the real network has stochastic and dynamic arrivals and channel state conditions. In the next section, we present the complete solution which takes into account these dynamics and also establishes its convergence properties.

4.2.3 Dynamic Algorithm and Its Convergence Analysis

Scheduler and Congestion Controller

To implement the algorithm on-line, we propose the following congestion control algorithm in frame k, where the queue length at link l is given by $q_l(k)$:

$$\tilde{x}_l^*(k) \in \underset{0 \le x_l \le X_{max}}{\arg\max} \frac{1}{\epsilon} U_l(x_l) - q_l(k) x_l. \tag{4.5}$$

We need to convert this elastic arrival rate, which in general is a non-negative real number, into a non-negative integer indicating the number of elastic packets allowed to enter the network in a given frame. This conversion can be made in many different ways: we assume the elastic arrivals at link l, $\tilde{a}_{el}(k)$, are a random variable with mean $\tilde{x}_l^*(k)$ and variance upper-bounded by σ_e^2, and are such that $Pr(\tilde{a}_{el}(k) = 0) > 0$ and $Pr(\tilde{a}_{el}(k) = 1) > 0$ for all $l \in \mathcal{L}$ and all k. The last two assumptions are used to guarantee the Markov chain we define below is both irreducible and aperiodic, although these can be replaced by other similar assumptions.

Letting the number of inelastic arrivals be denoted by $a_i(k)$ and the channel state by $c(k)$, we propose the following scheduling algorithm:

$$\tilde{s}^*(a_i(k), c(k), d(k), q(k)) \in \tag{4.6}$$
$$\arg\max_{s \in \mathcal{S}(a_i(k), c(k))} \sum_{l \in \mathcal{L}} \left\{ \left[\frac{1}{\epsilon} w_l + d_l(k) \right] \sum_{t=1}^{T} s_{il,t} + q_l(k) \sum_{t=1}^{T} s_{el,t} \right\}.$$

The vectors $d(k)$ and $q(k)$ are updated from frame to frame as follows:

$$d_l(k+1) = [d_l(k) + \tilde{a}_{il}(k) - I_{il}^*(a_i(k), c(k), d(k), q(k))]^+$$

$$q_l(k+1) = [q_l(k) + \tilde{a}_{el}(k) - I_{el}^*(a_i(k), c(k), d(k), q(k))]^+,$$

where

$$I_{il}^*(a_i(k), c(k), d(k), q(k)) = \sum_{t=1}^{T} \tilde{s}_{il,t}^*(a_i(k), c(k), d(k), q(k))$$

$$I_{el}^*(a_i(k), c(k), d(k), q(k)) = \sum_{t=1}^{T} \tilde{s}_{el,t}^*(a_i(k), c(k), d(k), q(k))$$

and $\tilde{a}_{il}(k)$ is a binomial random variable with parameters $a_{il}(k)$ and $1 - p_l$. The quantity $\tilde{a}_{il}(k)$ can be generated by the network as follows: upon each inelastic packet arrival, toss a coin with probability of *heads* equal to $1 - p_l$, and if the outcome is *heads*, add a one to the deficit counter.

In our notation we make explicit the fact that for fixed ϵ and w, the optimal scheduler (4.6) is a function of $a_i(k)$, $c(k)$, $d(k)$, and $q(k)$. We interpret $d_l(k)$ as a virtual queue that counts the deficit in service for link l to achieve a loss probability due to deadline expiry less than or equal to p_l. This deficit queue

was first used in the inelastic traffic context in [2] for the case of collocated networks; the connection to the dual decomposition approach now provides a Lagrange multiplier interpretation to it and allows the extension to general ad hoc networks. Note that $q_l(k)$ is just the queue size for elastic packets at link l.

Convergence Results

For ease of readability, we present the main results in this section, but the proofs are deferred to the appendixes. We start by noting that $(d(k), q(k))$ defines an irreducible and aperiodic Markov chain. To prove that our dynamic algorithm achieves the optimal solution to the static problem (4.3) in some average sense and fulfills all links' requirements, we will first bound the expected drift of $(d(k), q(k))$ for a suitable Lyapunov function.

Lemma 13. *Consider the Lyapunov function* $V(d,q) = \frac{1}{2}\sum_{l\in\mathcal{L}} d_l^2 + q_l^2$. *If* $\mu_{il}^* > \lambda_l(1-p_l)$ *and* $\mu_{el}^* > x_l^*$ *for all* $l \in \mathcal{L}$, *then*

$$\begin{aligned}
&E\left[V(d(k+1), q(k+1))|d(k)=d, q(k)=q\right] - V(d,q)\\
\leq& B_1 - B_2\sum_{l\in\mathcal{L}} d_l - B_3\sum_{l\in\mathcal{L}} q_l - \frac{1}{\epsilon}\sum_{l\in\mathcal{L}}[U_l(x_l^*) - U_l(\tilde{x}_l^*(k))]\\
&-\frac{1}{\epsilon}\sum_{l\in\mathcal{L}} w_l\mu_{il}^* - w_l E\left[I_{il}^*(a_i(k), c(k), d, q)\right]
\end{aligned}$$

for some positive constants B_1, B_2, B_3, *any* $\epsilon > 0$, *where* (x^*, μ_i^*) *is the solution to (4.3),* $\tilde{x}^*(k)$ *is the solution to (4.5), and* $I_i^*(a_i(k), c(k), d, q)$ *is obtained from the solution to (4.6).* $\diamond$

It is important to note that since the last two terms in the right-hand side of the inequality can be upper-bounded, Lemma 13 implies that $(d(k), q(k))$ is positive recurrent since the expected drift is negative but for a finite set of values of $(d(k), q(k))$. As a direct consequence of this fact, we note that the total service deficit and queue length have a $O(1/\epsilon)$ bound.

Corollary 2. *If* $\mu_{il}^* > \lambda_l(1-p_l)$ *and* $\mu_{el}^* > x_l^*$ *for all* $l \in \mathcal{L}$, *then the total expected service deficit and network queue length is upper-bounded by*

$$\limsup_{k\to\infty} E\left[\sum_{l\in\mathcal{L}} d_l(k) + q_l(k)\right] \leq B_4 + \frac{1}{\epsilon}B_5$$

for all $l \in \mathcal{L}$ *and*

$$B_4 = \frac{B_1}{\min\{B_2, B_3\}}$$

and

$$B_5 \leq \frac{\sum_{l\in\mathcal{L}} \max_{0\leq x_l \leq X_{max}} 2|U_l(x_l)| + w_l\lambda_l}{\min\{B_2, B_3\}}.$$

◇

This also implies that the scheduling and congestion control algorithm fulfills all links' inelastic requirements.

Corollary 3. *If* $\mu^*_{il} > \lambda_l(1-p_l)$ *and* $\mu^*_{el} > x^*_l$ *for all* $l \in \mathcal{L}$*, then the on-line algorithm fulfills all the inelastic constraints. That is:*

$$\liminf_{K\to\infty} E\left[\frac{1}{K}\sum_{k=1}^{K} I^*_{il}(a_i(k), c(k), d(k), q(k))\right] \geq \lambda_l(1-p_l)$$

for all $l \in \mathcal{L}$*.* ◇

The above corollary simply states that the arrival rate into the deficit counter is less than or equal to the departure rate. This result is an obvious consequence of the stability of the deficit counters and so a formal proof is not provided here.

Now we are ready to prove that our on-line algorithm is within $O(\epsilon)$ of the optimal value.

Theorem 7. *For any* $\epsilon > 0$*, if* $\mu^*_{il} > \lambda_l(1-p_l)$ *and* $\mu^*_{el} > x^*_l$ *for all* $l \in \mathcal{L}$*, then*

$$\limsup_{K\to\infty} E\left[\sum_{l\in\mathcal{L}} U_l(x^*_l) + w_l\mu^*_{il} - \sum_{l\in\mathcal{L}} \frac{1}{K}\sum_{k=1}^{K} U_l(\tilde{x}^*_l(k)) \right.$$
$$\left. - \sum_{l\in\mathcal{L}} \frac{1}{K}\sum_{k=1}^{K} w_l I^*_{il}(a_i(k), c(k), d(k), q(k))\right] \leq B\epsilon$$

for some $B > 0$*, where* (x^*, μ^*_i) *is the solution to (4.3),* $\tilde{x}^*(k)$ *is the solution to (4.5), and* $I^*_i(a_i(k), c(k), d(k), q(k))$ *is obtained from the solution to (4.6).* ◇

In conclusion, there is a trade-off in choosing the parameter ϵ: smaller values will achieve a solution closer to the optimal, but at the same time the

deficit in service at the links and the aggregate queue length increase. The statement and the proof of Theorem 7 follows the techniques in [55]. The result can also be derived, in a slightly different form, using the techniques in [56]. A closely related result can be obtained using the methods in [53].

4.3 Unknown Channel State, Per-Frame Feedback

The analysis for the unknown channel case is similar to the one we presented for the known channel case, so in this section we will only highlight the differences.

A feasible schedule $s = (s_{il,t}, s_{el,t})$ is such that $s_{il,t}$, $s_{el,t}$ respectively denote the number of inelastic and elastic packets that can be scheduled for transmission at link $l \in \mathcal{L}$ and time $t \in \{1, 2, \ldots, T\}$ without violating any interference constraints. Assuming the inelastic arrivals are given by a_i, and since we can only schedule at most one packet per link at every time slot, we have the following constraints:

$$\sum_{t=1}^{T} s_{il,t} \leq a_{il} \text{ for all } l \in \mathcal{L} \text{ and} \tag{4.7}$$

$$s_{il,t} + s_{el,t} \leq 1 \text{ for all } l \in \mathcal{L} \text{ and } t \in \{1, 2, \ldots, T\}. \tag{4.8}$$

We denote by $\mathcal{S}(a_i)$ the set of all feasible schedules for fixed arrivals, capturing any interference constraints we have on our network, and satisfying (4.7) and (4.8).

Our goal now is to find a function $Pr(s|a_i)$ which is the probability of using schedule $s \in \mathcal{S}(a_i)$ when the inelastic arrivals are given by a_i, subject to the constraint that the loss probability at link $l \in \mathcal{L}$ due to deadline expiry cannot exceed p_l. For elastic traffic, we still want to select the vector a_e such that we maximize the total utility while keeping the queues stable.

For a given distribution $Pr(s|a_i)$ we have that $\mu_{il}(a_i)$ is the expected number of attempted inelastic transmissions if arrivals are given by a_i. Similarly, $\mu_{el}(a_i)$ denotes the expected number of times link l is scheduled to serve elas-

tic packets in a given frame. As before, we have the following constraints:

$$\mu_{il}(a_i) \leq \sum_s \sum_{t=1}^{T} s_{il,t} Pr(s|a_i)$$

$$\mu_{el}(a_i) \leq \sum_s \sum_{t=1}^{T} s_{el,t} Pr(s|a_i)$$

When the (unknown) channel state is c, we have that $c_l \mu_{il}(a_i)$ is the expected number of successful inelastic transmissions per frame at link l for fixed arrivals, while $c_l \mu_{el}(a_i)$ is the expected service to link l for inelastic arrivals. Thus, the expected service for mixed traffic at link l is given by

$$\mu_{il} \stackrel{def}{=} \sum_{a_i} \sum_{c} c_l \mu_{il}(a_i) Pr(c) Pr(a_i)$$

$$\mu_{el} \stackrel{def}{=} \sum_{a_i} \sum_{c} c_l \mu_{el}(a_i) Pr(c) Pr(a_i).$$

Simplifying both expressions we get

$$\mu_{il} = \sum_{a_i} \bar{c}_l \mu_{il}(a_i) Pr(a_i)$$

$$\mu_{el} = \sum_{a_i} \bar{c}_l \mu_{el}(a_i) Pr(a_i).$$

Due to service requirements and capacity constraints we need that

$$\mu_{il} \geq \lambda_l (1 - p_l) \text{ and } x_l \leq \mu_{el}.$$

With the definitions and constraints stated above we can formulate the optimization problem in a similar way as in (4.3).

The only difference with the known channel state case is the scheduling algorithm. Assuming inelastic arrivals are given by $a_i(k)$ the scheduling algorithm is given by

$$\tilde{s}^*(a_i(k), d(k), q(k)) \in$$
$$\arg\max_{s \in \mathcal{S}(a_i(k))} \sum_{l \in \mathcal{L}} \left\{ \left[\frac{1}{\epsilon} w_l + d_l(k) \right] \bar{c}_l \sum_{t=1}^{T} s_{il,t} + q_l(k) \bar{c}_l \sum_{t=1}^{T} s_{el,t} \right\}.$$

The main difference in the scheduling algorithm compared to the known channel state case is that the network now uses the expected channel state in making scheduling decisions. Thus, the network needs to know or estimate $\bar{c}_l$ as in [2].

Similar results can be proved for this algorithm using the techniques developed in Section 4.2.3, whereby one can show that the algorithm meets all the inelastic QoS constraints, the total expected service deficits and the queue lengths have a $O(1/\epsilon)$ bound, and the mean value of the objective is within $O(\epsilon)$ of the optimal value.

4.4 Unknown Channel State, Per-Slot Feedback

4.4.1 Problem Formulation

In this section we assume that there is only inelastic traffic and that at every frame and at every link there is one packet arrival. This simplification in the problem formulation allows us to develop the main ideas behind this case, while the generalization to more complex scenarios is similar in nature to the development done in previous sections.

We first formulate the problem as a static optimization problem. Using decomposition theory, we will then obtain a dynamic solution that makes per-frame decisions and prove its stability using stochastic Lyapunov techniques. Later we will prove that feedback at every slot does not improve the capacity region compared to the utility maximization framework, but it can help simplify the scheduling algorithm in certain scenarios.

A feasible schedule $s = (s_{l,t})$ denotes the number of inelastic packets that can be scheduled for transmission at link $l \in \mathcal{L}$ and time slot $t \in \{1, 2, \ldots, T\}$; thus, $s_{l,t} > 0$ means that link l is scheduled to transmit in time slot t of the frame. Furthermore, for any t, if $s_{l_1,t} > 0$ and $s_{l_2,t} > 0$ then links l_1 and l_2 can be scheduled simultaneously to transmit without interfering with each other. Since we assume we can only schedule at most one packet per link at every time slot, we have the following constraint:

$$s_{l,t} \leq 1 \text{ for all } l \in \mathcal{L} \text{ and } t \in \{1, 2, \ldots, T\}. \tag{4.9}$$

We denote by $\mathcal{S}$ the set of all feasible schedules, capturing any interference constraints we have on our network, and satisfying (4.9).

At the beginning of any frame we must choose a feasible schedule to serve all links. Therefore, our goal is to find a function $Pr(s)$ which is the probability of using schedule $s \in \mathcal{S}$, subject to the constraint that the loss probability at link $l \in \mathcal{L}$ due to deadline expiry cannot exceed p_l.

To properly formulate the problem, define $\tau = (\tau_{l,s})$ to be the number of slots assigned to link $l \in \mathcal{L}$ under schedule $s \in \mathcal{S}$. That is,

$$\tau_{l,s} = \sum_{t=1}^{T} s_{l,t}.$$

Denoting by $\mu = (\mu_l)_{l \in \mathcal{L}}$ the expected service to link l, we have that

$$\mu_l \leq \sum_{s \in \mathcal{S}} \left[1 - (1 - \bar{c}_l)^{\tau_{l,s}}\right] Pr(s).$$

Due to QoS requirements, we require that

$$\mu_l \geq (1 - p_l).$$

We will focus on maximizing the following objective for some given constant A:

$$\max_{\tau, \mu, Pr(s)} A \tag{4.10}$$

subject to

$$\tau_{l,s} = \sum_{t=1}^{T} s_{l,t} \text{ for all } l \in \mathcal{L} \text{ and } s \in \mathcal{S}$$

$$\mu_l \leq \sum_{s \in \mathcal{S}} \left[1 - (1 - \bar{c}_l)^{\tau_{l,s}}\right] Pr(s) \text{ for all } l \in \mathcal{L}$$

$$\mu_l \geq (1 - p_l) \text{ for all } l \in \mathcal{L}$$

$$Pr(s) \geq 0 \text{ for all } s \in \mathcal{S}$$

$$\sum_{s \in \mathcal{S}} Pr(s) \leq 1.$$

The objective in the above optimization formulation is a dummy objective; the real goal is to find a feasible resource allocation without any further considerations. We will assume that the arrivals and loss probability require-

ments are feasible and thus the optimization problem has a solution μ^*.

4.4.2 Solution Using Dual Decomposition

Using the definition of the dual function [65], we have that $D(\delta) =$

$$\max_{\tau,\mu,Pr(s)} A - \sum_{l\in\mathcal{L}} \delta_l[(1-p_l) - \mu_l]$$

subject to

$$\tau_{l,s} = \sum_{t=1}^{T} s_{l,t} \text{ for all } l \in \mathcal{L} \text{ and } s \in \mathcal{S}$$

$$\mu_l \leq \sum_{s\in\mathcal{S}} \left[1 - (1-\bar{c}_l)^{\tau_{l,s}}\right] Pr(s) \text{ for all } l \in \mathcal{L}$$

$$Pr(s) \geq 0 \text{ for all } s \in \mathcal{S}$$

$$\sum_{s\in\mathcal{S}} Pr(s) \leq 1.$$

We are interested in finding μ^* but not the value $D(\delta^*)$, so if we rewrite the objective in the dual function as

$$\max_{\tau,\mu,Pr(s)} A - \sum_{l\in\mathcal{L}} \delta_l(1-p_l) + \sum_{l\in\mathcal{L}} \delta_l\mu_l$$

we note that the problem can be simplified to

$$\max_{\tau,\mu,Pr(s)} \sum_{l\in\mathcal{L}} \delta_l\mu_l \tag{4.11}$$

subject to

$$\tau_{l,s} = \sum_{t=1}^{T} s_{l,t} \text{ for all } l \in \mathcal{L} \text{ and } s \in \mathcal{S}$$

$$\mu_l \leq \sum_{s\in\mathcal{S}} \left[1 - (1-\bar{c}_l)^{\tau_{l,s}}\right] Pr(s) \text{ for all } l \in \mathcal{L}$$

$$Pr(s) \geq 0 \text{ for all } s \in \mathcal{S}$$

$$\sum_{s\in\mathcal{S}} Pr(s) \leq 1.$$

Since we are interested in solving this problem for non-negative values of δ_l,

it must be the case that μ_l^* is as large as the constraints allow. Furthermore, since the upper bound for μ_l is expressed as a convex combination, and the objective function in (4.11) is linear, the problem can be further simplified to

$$\max_{s\in\mathcal{S}} \sum_{l\in\mathcal{L}} \delta_l \left[1-(1-\bar{c}_l)^{\tau_{l,s}}\right],$$

where

$$\tau_{l,s} = \sum_{t=1}^{T} s_{l,t} \text{ for all } l \in \mathcal{L} \text{ and } s \in \mathcal{S}.$$

This suggests the following iterative algorithm to find the solution to our optimization problem, where k is the step index:

$$\tilde{s}^*(k) \in \arg\max_{s\in\mathcal{S}} \sum_{l\in\mathcal{L}} \delta_l(k) \left[1-(1-\bar{c}_l)^{\tau_{l,s}}\right].$$

We update the Lagrange multipliers $\delta_l(k)$ at every step according to the following equation:

$$\delta_l(k+1) = \{\delta_l(k) + \epsilon[(1-p_l) - \tilde{\mu}_l^*(k)]\}^+$$

where

$$\tilde{\mu}_l^*(k) = \left[1-(1-\bar{c}_l)^{\tilde{\tau}_{l,s}^*(k)}\right],$$

$$\tilde{\tau}_{l,s}^*(k) = \sum_{t=1}^{T} \tilde{s}_{l,t}^*(k) \text{ for all } l \in \mathcal{L},$$

$\epsilon > 0$ is a fixed step-size parameter, and for any $\alpha \in \mathbb{R}$, $\alpha^+ \stackrel{def}{=} \max\{\alpha, 0\}$.

Making the change of variables $\epsilon\hat{d}(k) = \delta(k)$, we have that our iterative algorithm can be rewritten as

$$\tilde{s}^*(k) \in \arg\max_{s\in\mathcal{S}} \sum_{l\in\mathcal{L}} \hat{d}_l(k) \left[1-(1-\bar{c}_l)^{\tau_{l,s}}\right]$$

with update equations

$$\hat{d}_l(k+1) = [\hat{d}_l(k) + (1-p_l) - \tilde{\mu}_l^*(k)]^+.$$

It should be noted that due to the change of variables $\hat{d}_l(k)$ can be interpreted as a queue that has $(1-p_l)$ arrivals and $\tilde{\mu}_l^*(k)$ departures at step

k. The dual decomposition approach only provides an intuition behind the solution, but the real network has stochastic and dynamic arrivals and channel state conditions. In the next section, we present the complete solution which takes into account these dynamics and we also establish its convergence properties.

4.4.3 Dynamic Algorithm and Its Convergence Analysis

Scheduler and Congestion Controller

To implement the algorithm on-line, we propose the following scheduling algorithm in frame k, where we denote the channel state by $c(k)$:

$$\tilde{s}^*(d(k)) \in \arg\max_{s\in\mathcal{S}} \sum_{l\in\mathcal{L}} d_l(k)\left[1-(1-\bar{c}_l)^{\tau_{l,s}}\right] \tag{4.12}$$

where

$$\tau_{l,s} = \sum_{t=1}^{T} s_{l,t} \text{ for all } l \in \mathcal{L} \text{ and } s \in \mathcal{S}.$$

The vector $d(k)$ is updated from frame to frame as follows:

$$d_l(k+1) = [d_l(k) + \tilde{a}_l(k) - I_l^*(c(k), d(k))]^+$$

where

$$I_l^*(c(k), d(k)) = \min\left\{\sum_{t=1}^{T} c_{l,t}(k)\tilde{s}_{l,t}^*(d(k)), 1\right\},$$

and $\tilde{a}_l(k)$ is a Bernoulli random variable with parameter $1-p_l$. The quantity $\tilde{a}_l(k)$ can be generated by the network as follows: upon each packet arrival, toss a coin with probability of *heads* equal to $1-p_l$, and if the outcome is *heads*, add a one to the deficit counter.

In our notation we make explicit the fact that the optimal scheduler (4.12) is a function of $d(k)$. We interpret $d_l(k)$ as a virtual queue that counts the deficit in service for link l to achieve a loss probability due to deadline expiry less than or equal to p_l. This deficit queue was first used in the inelastic traffic context in [2] for the case of collocated networks; the connection to the dual decomposition approach now provides a Lagrange multiplier interpretation to it and allows the extension to general ad hoc networks.

Convergence Results

For readability, we present the main results in this section, but the proof of Lemma 14 is deferred to Appendix O. We start by noting that $d(k)$ defines an irreducible and aperiodic Markov chain. To prove that our dynamic algorithm makes the deficit counters stable and fulfills all links' requirements, we first bound the expected drift of $d(k)$ for a suitable Lyapunov function.

Lemma 14. *Consider the Lyapunov function* $V(d) = \frac{1}{2}\sum_{l\in\mathcal{L}} d_l^2$. *If* $\mu_l^* > (1-p_l)$ *for all* $l \in \mathcal{L}$, *then*

$$E\left[V(d(k+1))|d(k)=d\right] - V(d) \leq B_1 - B_2 \sum_{l\in\mathcal{L}} d_l$$

for some positive constants B_1, B_2. ◇

It is important to note that Lemma 14 implies that $d(k)$ is positive recurrent since the expected drift is negative but for a finite set of values of $d(k)$. As a direct consequence of this fact, we note that the total service deficit is upper-bounded.

Corollary 4. *If* $\mu_l^* > (1-p_l)$ *for all* $l \in \mathcal{L}$ *then the total expected service deficit is upper-bounded by*

$$\limsup_{k\to\infty} E\left[\sum_{l\in\mathcal{L}} d_l(k)\right] \leq B_3$$

for all $l \in \mathcal{L}$ *and*

$$B_3 = \frac{B_1}{B_2}.$$

◇

This also implies that the scheduling algorithm fulfills all links' inelastic requirements.

Corollary 5. *If* $\mu_l^* > (1-p_l)$ *for all* $l \in \mathcal{L}$, *then the on-line algorithm fulfills all the inelastic constraints. That is:*

$$\liminf_{K\to\infty} E\left[\frac{1}{K}\sum_{k=1}^{K} I_l^*(c(k),d(k))\right] \geq (1-p_l)$$

for all $l \in \mathcal{L}$. ◇

The above corollary simply states that the arrival rate into the deficit counter is less than or equal to the departure rate. This result is an obvious consequence of the stability of the deficit counters and so a formal proof is not provided here.

4.4.4 Per-Slot Feedback and the Capacity Region

So far we have developed a framework that allows us to choose a schedule at the beginning of every frame and ignore any per-slot feedback we get, while fulfilling all the inelastic constraints. It remains to be shown that we cannot increase the capacity region of the network by using per-slot feedback in our scheduling decisions. To prove that, we will restrict ourselves to the set of stabilizing frame-stationary policies.

Definition 9. *A* stabilizing frame-stationary policy *is a scheduling algorithm with the following properties:*

1. *The policy makes scheduling decisions at every time slot based only on the current value of the deficit counters, and on the success or failure of its decisions at previous slots of the current frame. In particular, the policy is not a function of events in previous frames.*

2. *The policy renders the deficit counters at frame boundaries stable. In other words, the Markov chain $d(k)$ is positive recurrent.* ◇

Theorem 8. *Any stabilizing frame-stationary policy has a per-frame stationary distribution.* ◇

Proof. First, we highlight the fact that due to Property 2 in Definition 9, we know that the Markov chain $d(k)$ has a stationary distribution since it is positive recurrent. Second, from Property 1 in Definition 9 we know that the scheduling policy at time slot t in a given frame depends only on the history of slots 1 through $t-1$. That is, the scheduling decision at time slot t is given by:

$$Pr\left[(s_{l,t})_{l\in\mathcal{L}}\middle|d(k),(s_{l,j})_{l\in\mathcal{L}}^{j\in\{1,\ldots,t-1\}},(c_{l,j}s_{l,j})_{l\in\mathcal{L}}^{j\in\{1,\ldots,t-1\}}\right],$$

where $c_{l,j}s_{l,j}$ is an indicator that expresses the success or failure of the scheduling decisions.

Since we assume that $c_{l,t}$ is a Bernoulli random variable, and since $d(k)$ has a stationary distribution, it is clear that we can explicitly write the per-frame stationary distribution $Pr(s)$ of any stabilizing frame-stationary policy. $\square$

From Theorem 8 we note that there is no loss of generality in our utility maximization framework because the formulation already takes into account the set of stabilizing frame-stationary policies, and therefore we cannot improve the capacity region by allowing per-slot feedback to affect scheduling decisions. However, we also note that the use of per-slot feedback can potentially decrease the complexity of the scheduling algorithm as shown in [2].

4.4.5 A Greedy Strategy for Collocated Networks

To show how feedback can help simplify the on-line algorithm (4.12), in this section we will prove that a strategy that makes greedy decisions at every time slot suffices for networks where only one link can transmit at any given time, thus recovering the results in [2] for access-point networks.

Fact 1. *The expected service to link l depends not on which specific time slots the link is scheduled, but on the total number of slots it is scheduled.* $\diamond$

This implies that for a feasible schedule s with expected service given by $[1-(1-\bar{c}_l)^{\tau_{l,s}}]$ at link l, we can generate another feasible schedule $\tilde{s}$ with the same expected service at every link by doing a permutation in the time slot index of s. For the case of collocated networks, Fact 1 allows us to restrict our attention, without any loss of generality, to the set of feasible schedules s such that

$$\sum_{l\in\mathcal{L}} s_{l,t} \leq 1 \text{ for all } t \in \{1,2,\ldots,T\}$$

where links can only be scheduled in decreasing order of the priorities $d_l\bar{c}_l$, given that the deficit counters at the beginning of the frame are given by d. We denote by $\mathcal{S}(d)$ the set of feasible schedules with the defined priorities.

We first prove that the best one can do in time slot 1 is to schedule the link with the highest priority. For ease of readability, we defer the proofs to the appendixes.

Lemma 15. *If at a given frame the deficit counters are given by d, and if we restrict the optimization (4.12) to the set $\mathcal{S}(d)$, then at time slot 1 we have*

$$\tilde{s}^*_{l,1}(d) = \begin{cases} 1 \text{ for a } \tilde{l} \in \mathcal{L} \text{ such that } d_{\tilde{l}}\bar{c}_{\tilde{l}} \geq d_l\bar{c}_l \text{ for all } l \in \mathcal{L} \\ 0 \text{ for } l \in \mathcal{L} \setminus \{\tilde{l}\}. \end{cases} \tag{4.13}$$

◇

Definition 10. *A* greedy strategy *for collocated networks is a scheduling policy that at every time slot schedules the link with the highest priority $d_l\bar{c}_l$ among the links that have a packet that remains to be transmitted.* ◇

This means that the greedy strategy makes its decisions based on the value of the deficit counters and the channel state at the previous time slots. At the end of the time frame, the greedy algorithm would have picked a schedule that we denote by $s^g(c,d)$, when the deficit counters at the beginning of the frame were given by d and the channel state in the frame was c.

We now show that using the greedy scheduler can only improve the total weighted expected service compared to the on-line algorithm in (4.12).

Lemma 16. *Assuming that the deficit counters at the beginning of the frame k are given by $d(k) = d$, we have that*

$$\sum_{l\in\mathcal{L}} d_l E\left[\sum_{t=1}^{T} c_{l,t}s^g_{l,t}(c,d)\middle| d(k)=d\right] \geq \sum_{l\in\mathcal{L}} d_l\left[1-(1-\bar{c}_l)^{\tau^*_{l,s}}\right]$$

where

$$\tau^*_{l,s} = \sum_{t=1}^{T} \tilde{s}^*_{l,t}(d),$$

and $\tilde{s}^*(d)$ is the solution to (4.12). ◇

This result allows us to bound the expected drift of the deficit counters for a suitable Lyapunov function when the greedy strategy is used.

Lemma 17. *Consider the Lyapunov function $V(d) = \frac{1}{2}\sum_{l\in\mathcal{L}} d_l^2$. If $\mu^*_l > (1-p_l)$ for all $l \in \mathcal{L}$, and the greedy strategy is used, we have*

$$E\left[V(d(k+1))|d(k)=d\right] - V(d) \leq B_1 - B_2\sum_{l\in\mathcal{L}} d_l$$

for some positive constants B_1, B_2. ◇

Similar to the development in Section 4.4.3, one can show that the greedy algorithm meets all the inelastic QoS constraints and the total expected service deficit is upper-bounded.

4.4.6 Probabilistic Arrivals and Fairness

In this section we extend the formulation of Section 4.4.1 for the case of probabilistic arrivals and weighted throughput fairness to allocate bandwidth to links beyond what is required to meet their QoS needs. Since the analysis is similar to the one we already presented, we only highlight the differences.

We denote by the vector $a = (a_l)_{l \in \mathcal{L}}$ the number of inelastic packet arrivals at every link, where a_l is a random variable with mean λ_l and variance σ_l^2. We assume that arrivals are independent between frames and independent of the channel state and that $Pr(a_l = 0) > 0$ and $Pr(a_l = 1) > 0$. The last two assumptions are used to guarantee that the Markov chain we define later is both irreducible and aperiodic, although these can be replaced by other similar assumptions.

Our goal is to find a function $Pr(s|a)$ which is the probability of using schedule $s \in \mathcal{S}$ when the inelastic arrivals are given by a, subject to the constraint that the loss probability at link $l \in \mathcal{L}$ due to deadline expiry cannot exceed p_l.

The expected service at link l when the schedule is $s \in \mathcal{S}$ and the arrivals at the frame are given by a is

$$\mu_l(s, a) \leq \sum_c \min \left\{ \sum_{t=1}^{T} c_{l,t} s_{l,t}, a_l \right\} Pr(c).$$

Thus, the expected service at link l when using policy $Pr(s|a)$ is

$$\mu_l \stackrel{def}{=} \sum_a \sum_{s \in \mathcal{S}} \mu_l(s, a) Pr(s|a) Pr(a).$$

Due to service requirements and capacity constraints we need that

$$\mu_l \geq \lambda_l (1 - p_l).$$

We will focus on maximizing the following objective for some given vector

$w \in \mathbb{R}_+^{|\mathcal{L}|}$:

$$\max_{Pr(s|a),\mu(s,a),\mu} \sum_{l\in\mathcal{L}} w_l \mu_l \tag{4.14}$$

subject to

$$\mu_l(s,a) \leq \sum_c \min\left\{\sum_{t=1}^{T} c_{l,t} s_{l,t}, a_l\right\} Pr(c) \text{ for all } l \in \mathcal{L}, s \in \mathcal{S}, a$$

$$\mu_l = \sum_a \sum_{s\in\mathcal{S}} \mu_l(s,a) Pr(s|a) Pr(a) \text{ for all } l \in \mathcal{L}$$

$$\mu_l \geq \lambda_l(1-p_l) \text{ for all } l \in \mathcal{L}$$

$$Pr(s|a) \geq 0 \text{ for all } s \in \mathcal{S}, a$$

$$\sum_{s\in\mathcal{S}} Pr(s|a) \leq 1 \text{ for all } a.$$

We will assume that the arrivals and loss probability requirements are feasible and thus the optimization problem has a solution μ^*.

With this formulation we can use a dual decomposition approach to find the on-line scheduler. Assuming inelastic arrivals in frame k are given by $a(k)$ and the channel state is $c(k)$, the algorithm is given by

$$\tilde{s}^*(a(k), d(k)) \in \arg\max_{s\in\mathcal{S}} \sum_{l\in\mathcal{L}} \left[\frac{1}{\epsilon} w_l + d_l(k)\right] \mu_l(s, a(k)),$$

where

$$\mu_l(s, a(k)) = \sum_c \min\left\{\sum_{t=1}^{T} c_{l,t} s_{l,t}, a_l(k)\right\} Pr(c),$$

and the deficit counters $d(k)$ have the following update equation:

$$d_l(k+1) = [d_l(k) + \tilde{a}_l(k) - I_l^*(a(k), c(k), d(k))]^+,$$

where

$$I_l^*(a(k), c(k), d(k)) = \min\left\{\sum_{t=1}^{T} c_{l,t}(k) \tilde{s}_{l,t}^*(a(k), d(k)), a_l(k)\right\}.$$

Similar results can be proved for this algorithm using the techniques developed in Section 4.2.3, whereby one can show that the algorithm meets all the

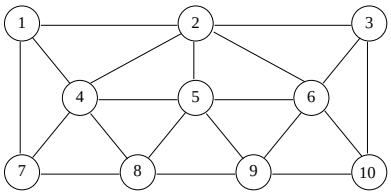

Figure 4.1: Interference graph used in the simulations

inelastic QoS constraints, the total expected service deficits have a $O(1/\epsilon)$ bound, and the mean value of the objective is within $O(\epsilon)$ of the optimal value. Furthermore, Section 4.4.4 can be similarly modified to take into account probabilistic arrivals to prove equivalent results. Section 4.4.5 can be modified to take into account probabilistic arrivals for the case when a_l is a Bernoulli random variable, and when we do not use weighted throughput fairness as our objective function.

4.5 Simulations

The purpose of this simulation study is to understand how the parameter ϵ and the link weights w_l impact the performance of the algorithm, and how a greedy heuristic can be used to implement the optimal scheduler. We simulate a 10-link network with an interference graph given by Fig. 4.1, where each node represents a link and each edge means that the two adjacent links cannot be scheduled simultaneously. For example, if link 1 is scheduled, then links 2, 4, and 7 cannot be activated. The required loss probability due to deadline expiry of inelastic packets is set to 0.1, the link arrivals are assumed to have a Bernoulli distribution with mean 0.6 packets/frame, and there are 3 time slots per frame. The channel for every link is assumed to have a Bernoulli distribution with mean 0.96, and we get to know the channel state at the beginning of the frame. We set $U_l(x_l) = \log(x_l)$ for all links. The simulation time was 10^6 frames.

As can be noted from (4.6), the max-weight scheduler requires that we do an exhaustive search to find the optimal schedule at every frame. For large networks this can become a burden due to the large search space; thus we explore a greedy heuristic and check how close it is to the optimal solution: at any given time slot, the greedy scheduler orders all links according to their weights. The greedy scheduler adds one of the links with the largest

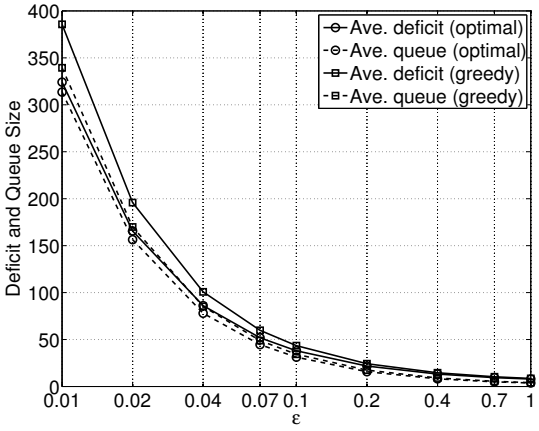

Figure 4.2: Deficit size and queue length when $w_l = 0$

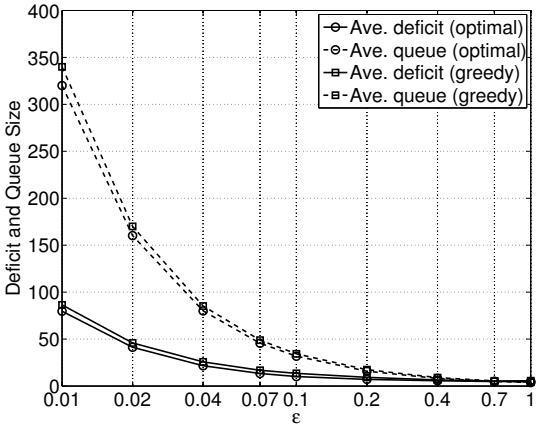

Figure 4.3: Deficit size and queue length when $w_l = 3$

weight to the schedule, then removes all links that interfere with this link from the graph, then schedules a link with the largest weight among the remaining links, and so on. This procedure continues until no more links can be scheduled.

In Figs. 4.2, 4.3, and 4.4, we plot the expected values of the deficit counters and queues per link for various values of w_l, and compare their evolution for both the scheduler with optimal decisions and the greedy scheduler.

We see that as w_l increases, the deficit counters become small. The upper bound in Corollary 2 only suggests that the sum of the deficit counters and queues is $O(1/\epsilon)$. Thus, it is interesting to note that by changing w_l, one can nearly eliminate the backlog in deficit for inelastic traffic while maintaining

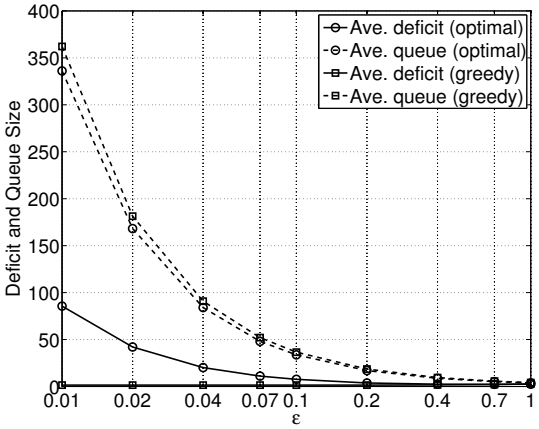

Figure 4.4: Deficit size and queue length when $w_l = 6$

the same order of queue sizes. The reason for this can be understood by examining the scheduling algorithm (4.6). Note that the algorithm gives priority to elastic traffic if queues are larger than counters. When w_l is small compared to ϵ, the effect of w_l is negligible in the scheduling algorithm. On the other hand, when w_l is $O(1)$, w_l/ϵ is $O(1/\epsilon)$ which is comparable to the queue lengths and hence, the deficit does not have to be large to provide service to inelastic traffic under algorithm (4.6).

It must be noted that small deficit counters mean that there is a small backlog in providing acceptable service to inelastic arrivals. For the case of real-time traffic this is a desirable property, since we do not want to have large variations in the service provided that could affect the perceived quality. Thus, even if fair allocation of bandwidth beyond the minimum is not required for inelastic flows, choosing w_l an order of magnitude larger than ϵ is desirable to maintain small deficits.

As can be noted, the greedy scheduler seems to give lower deficit values than the optimal scheduler for larger values of w_l. We believe that the reason is that weights given to inelastic flows increase with increasing w_l and therefore the greedy scheduler picks them first. However, our optimality goal is given by (4.3) which is determined by the rates received by the various flows. The rates achieved by the two schedulers are quite close in the simulations, as seen in Figs. 4.5, 4.6, and 4.7, while keeping the dropping probabilities below the requirement, as shown in Figs. 4.8, 4.9, and 4.10.

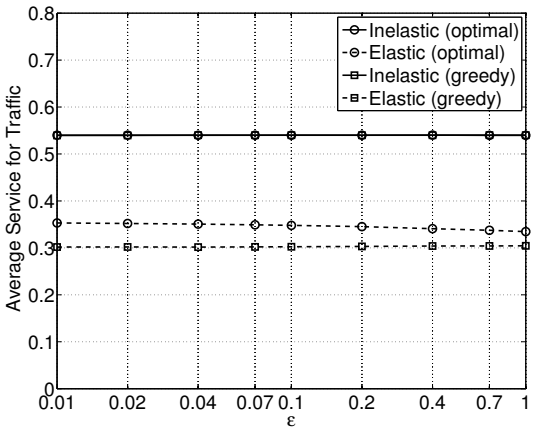

Figure 4.5: Average service when $w_l = 0$

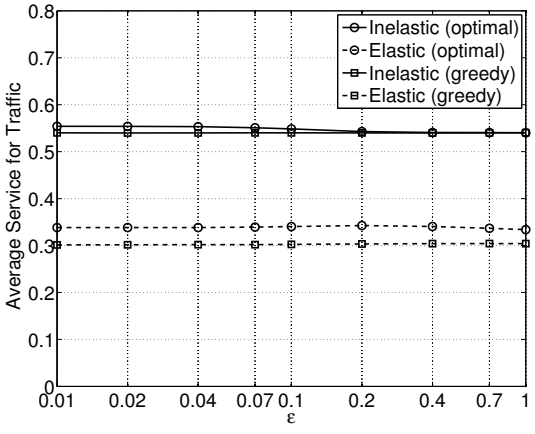

Figure 4.6: Average service when $w_l = 3$

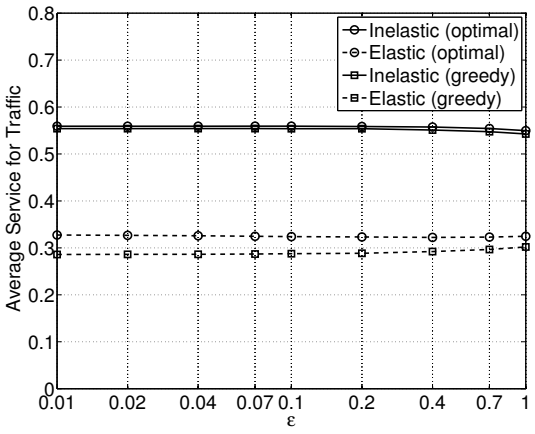

Figure 4.7: Average service when $w_l = 6$

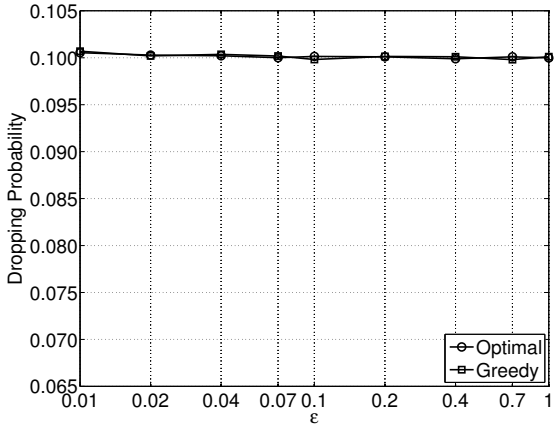

Figure 4.8: Dropping probability when $w_l = 0$

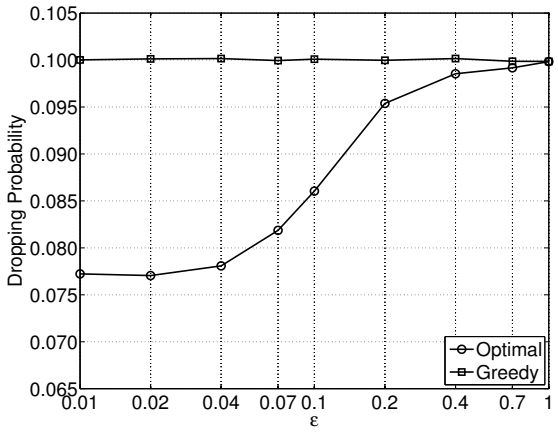

Figure 4.9: Dropping probability when $w_l = 3$

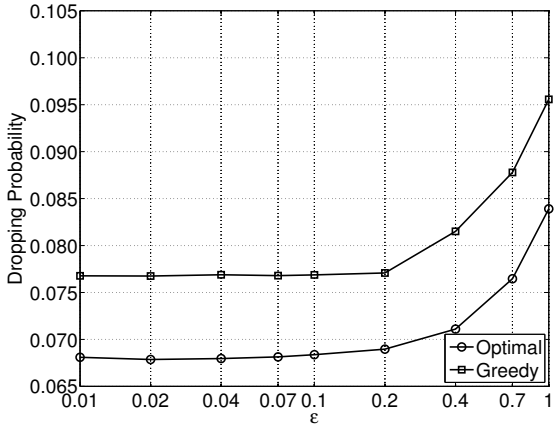

Figure 4.10: Dropping probability when $w_l = 6$

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

In this dissertation we have studied key design problems in wireless ad hoc networks. We showed how reputation-based mechanisms can help cooperation emerge among selfish users, studied the properties of previously proposed schemes, and with the insight gained from such understanding, we proposed a new mechanism called DARWIN, which is robust to imperfect measurements, is collusion-resistant and is able to achieve full cooperation.

We developed a model for admission control and routing with minimum-bandwidth requirements which allows us to derive an algorithm with provable performance guarantees using competitive analysis. We proved that our algorithm has a performance close to that of an omniscient off-line algorithm that has complete *a priori* knowledge of the entire sequence of flow arrivals (including the future) and their bandwidth requests. Our algorithm makes no statistical assumptions on the flow arrival pattern or other parameters of the arriving requests. We proved that no other algorithm performs better in an asymptotic sense of the competitive ratio. We also showed that our algorithm is amenable to a distributed implementation.

We presented an optimization framework for the problem of congestion control and scheduling of elastic and inelastic traffic. The model was developed for general interference graphs, general arrivals and time-varying channels. Using a dual function approach we presented a decomposition of the problem into an on-line algorithm that is able to make optimal decisions while keeping the network stable and fulfilling the inelastic flow's per-packet delay constraints. A key result is that, through the use of deficit counters, one can treat the scheduling problem for elastic and inelastic flows in a common framework.

Several problems arise as natural extensions to our work:

- Coping with Liars in Reputation Mechanisms

In the definition of DARWIN it is assumed that nodes share the perceived dropping probability with each other. This assumption is made in order to facilitate the theoretical analysis by isolating a pair of nodes, but in an implementation a mechanism is required to guarantee that even if a node lies, the reputation scheme still works. To do that we must rely on other cooperative nodes to tell the actual perceived dropping probability of a node in order to minimize the impact of liars. In [67,68] it is proved that if the connectivity of the network is at least $2f + 1$, then using linear iterations it is possible for all nodes to share some initial values to calculate an arbitrary function on them when there are up to f malicious nodes in the network. In principle, such a scheme can be used in our problem. The study of this in the context of wireless networks is an interesting topic for future research.

- Scheduling Under Heterogeneous Delay Constraints

  In this work we assumed that all flows have the same per-packet delay. To be able to cope with heterogeneous services, it is necessary to develop a framework that allows different users to have different deadlines. When scheduling, there can be scenarios where we must choose between transmitting a packet for a flow that has a small service deficit that is about to expire or a packet from a flow with a large deficit that has a large time to live. The characterization of this trade-off will help design efficient algorithms that are able to fulfill real-time requirements.

- Delay Guarantees on Wireless Multi-hop Networks

  When dealing with scheduling in multi-hop networks, we must also tackle the problem of delay partitioning, in which an end-to-end delay is divided in per-link delays. This problem has proven hard to solve so far because the techniques used do not take into account the feedback from the network, but it would be interesting to investigate whether queue-length-based scheduling algorithms can help solve the problem. For example, when service deficits decrease in a particular link this can potentially be taken as an indicator that the delay constraint in that link may be decreased without having an impact on the end-to-end service requirements.

- Guaranteeing QoS in Distributed Scheduling

Our scheduling algorithm has been designed to achieve the capacity region, but to do that it relies on centralized decisions. To achieve a scheduler suitable for practical wireless ad hoc networks it is necessary to develop a distributed mechanism. Our algorithm can be used as a benchmark on which distributed scheduling algorithms can be studied and compared. Of great importance is investigation into whether distributed implementations can achieve the capacity region.

APPENDIX A

PROOF OF THEOREM 1

In Section 2.3.3 we have already seen that if $g > p_e$ then GTFT is not a Nash equilibrium, so for the rest of the proof we will assume $g \leq p_e$. It must be noted that GTFT is a one-stage history strategy because it only needs to take into account what happened in the previous stage. With that in mind, and without loss of generality, let us assume that any history h^n is represented as $p_i^{(0)} = p_i$ for $i \in \{1, 2\}$. If both nodes use GTFT then using (2.1) we have the following subgame evolution:

k	$p_i^{(k)}$
0	p_i
1	$p_{-i}(1-p_e) + p_e - g$
2	$p_i(1-p_e)^2 + (p_e - g)\sum_{n=0}^{1}(1-p_e)^n$
3	$p_{-i}(1-p_e)^3 + (p_e - g)\sum_{n=0}^{2}(1-p_e)^n$
$\vdots$	$\vdots$

or equivalently for $k \geq 1$:

$$p_i^{(k)} = \theta_i^{(k)}(1-p_e)^k + \frac{(p_e - g)}{p_e}\left[1 - (1-p_e)^k\right]$$

where

$$\theta_i^{(k)} = \begin{cases} p_i & if\ k\ is\ even \\ p_{-i} & if\ k\ is\ odd. \end{cases}$$

Therefore from (2.2) the stage payoffs for $k \geq 1$ are:

$$u_i^{(k)} = 1 + \frac{1}{2\alpha - 1}p_i^{(k)} - \frac{2\alpha}{2\alpha - 1}p_{-i}^{(k)}.$$

If player i deviates at stage 1 using

$$p_{i\delta}^{(1)} = \tilde{p}_{i\ GTFT}^{(1)} + \delta$$

for some $\delta > 0$ and later conforms to GTFT, we have the following dropping probabilities:

k	$p_{i\delta}^{(k)}$
0	p_i
1	$p_{-i}(1-p_e) + (p_e - g) + \delta$
2	$p_i(1-p_e)^2 + (p_e - g)\sum_{n=0}^{1}(1-p_e)^n$
3	$p_{-i}(1-p_e)^3 + (p_e - g)\sum_{n=0}^{2}(1-p_e)^n + \delta(1-p_e)^2$
$\vdots$	$\vdots$

or equivalently:

$$p_{i\delta}^{(2m+1)} = p_{-i}(1-p_e)^{2m+1} + \frac{(p_e - g)}{p_e}\left[1 - (1-p_e)^{2m+1}\right] + \delta(1-p_e)^{2m}$$

$$p_{i\delta}^{(2m)} = p_i(1-p_e)^{2m} + \frac{(p_e - g)}{p_e}\left[1 - (1-p_e)^{2m}\right].$$

So we have:

$$p_{i\delta}^{(2m+1)} = p_i^{(2m+1)} + \delta(1-p_e)^{2m} \; for \; m \geq 0$$

$$p_{i\delta}^{(2m)} = p_i^{(2m)} \; for \; m \geq 1.$$

Which leads to the following stage payoffs:

$$u_{i\delta}^{(2m+1)} = u_i^{(2m+1)} + \frac{1}{2\alpha - 1}\delta(1-p_e)^{2m} \; for \; m \geq 0$$

$$u_{i\delta}^{(2m)} = u_i^{(2m)} - \frac{2\alpha}{2\alpha - 1}\delta(1-p_e)^{2m-1} \; for \; m \geq 1.$$

Since the stage payoff received at stage 0 is independent of the action player i takes at stage 1, we are only interested in finding the following discounted average payoff:

$$U_{i\delta}^{(1)} = \sum_{k=1}^{\infty} w^{k-1} u_{i\delta}^{(k)} = U_i^{(1)} + \frac{\delta\left[1 - 2\alpha w(1-p_e)\right]}{(2\alpha - 1)\left[1 - w^2(1-p_e)^2\right]},$$

where $U_i^{(1)}$ is the discounted payoff received if $\delta = 0$. Since we assume that $\alpha \geq 1$, it does not pay to deviate if:

$$1 - 2\alpha w(1-p_e) < 0.$$

But this is true if and only if:

$$w > \frac{1}{2\alpha(1-p_e)}.$$

Then, by the One-Stage Deviation Principle, GTFT is subgame perfect. $\square$

APPENDIX B

PROOF OF THEOREM 2

The line of reasoning is similar to that presented for Theorem 1. DARWIN is a one-stage history strategy because it only needs to take into account what happened in the previous stage. Hence, and without loss of generality, any history h^n can be represented as $q_i^{(0)} = q_i$ for $i \in \{1, 2\}$. If both nodes do not deviate from DARWIN, then using (2.1) we have for $k \geq 1$ the following subgame evolution:

If $q_i \geq q_{-i}$ then:

$$\begin{aligned}
p_i^{(k)} &= 0 \\
p_{-i}^{(k)} &= p_e^{k-1}\gamma^{k-1}\min\{1, \gamma(q_i - q_{-i})\} \\
\hat{p}_i^{(k)} &= p_e \\
\hat{p}_{-i}^{(k)} &= p_e + p_e^{k-1}\gamma^{k-1}(1 - p_e)\min\{1, \gamma(q_i - q_{-i})\} \\
q_i^{(k)} &= p_e \\
q_{-i}^{(k)} &= p_e - p_e^{k}\gamma^{k-1}\min\{1, \gamma(q_i - q_{-i})\}
\end{aligned}$$

From (2.2) the stage payoffs for $k \geq 1$ are:

$$u_{i\ a}^{(k)} = 1 - \frac{2\alpha}{2\alpha - 1}\left[(p_e\gamma)^{k-1}\min\left\{1, \gamma(q_i - q_{-i})\right\}\right].$$

If $q_i < q_{-i}$ then:

$$\begin{aligned}
p_i^{(k)} &= p_e^{k-1}\gamma^{k-1}\min\{1, \gamma(q_{-i} - q_i)\} \\
p_{-i}^{(k)} &= 0 \\
\hat{p}_i^{(k)} &= p_e + p_e^{k-1}\gamma^{k-1}(1 - p_e)\min\{1, \gamma(q_{-i} - q_i)\} \\
\hat{p}_{-i}^{(k)} &= p_e \\
q_i^{(k)} &= p_e - p_e^{k}\gamma^{k-1}\min\{1, \gamma(q_{-i} - q_i)\} \\
q_{-i}^{(k)} &= p_e
\end{aligned}$$

From (2.2) the stage payoffs for $k \geq 1$ are:

$$u_{i\ b}^{(k)} = 1 + \frac{1}{2\alpha - 1}p_e^{k-1}\gamma^{k-1}\min\left\{1, \gamma(q_{-i} - q_i)\right\}.$$

If player i deviates at stage 1 using

$$p_{i\delta}^{(1)} = \tilde{p}_{i\ DARWIN}^{(1)} + \delta$$

for some $\delta > 0$ and later conforms to DARWIN, we have the following game evolution:

If $q_i \geq q_{-i}$ then:

$$\begin{aligned}
p_i^{(1)} &= \delta \\
p_i^{(k)} &= 0 \\
p_{-i}^{(1)} &= \min\{1, \gamma(q_i - q_{-i})\} \\
p_{-i}^{(k)} &= (p_e\gamma)^{k-2} \min\{1, \gamma\delta(1-p_e) + p_e\gamma p_{-i}^{(1)}\} \\
\hat{p}_i^{(1)} &= p_e + \delta(1-p_e) \\
\hat{p}_i^{(k)} &= p_e \\
\hat{p}_{-i}^{(1)} &= p_e + (1-p_e)p_{-i}^{(1)} \\
\hat{p}_{-i}^{(k)} &= p_e + (p_e\gamma)^{k-2}(1-p_e) \min\{1, \gamma\delta(1-p_e) + p_e\gamma p_{-i}^{(1)}\} \\
q_i^{(1)} &= p_e + \delta(1-p_e) \\
q_i^{(k)} &= p_e \\
q_{-i}^{(1)} &= p_e - p_e p_{-i}^{(1)} \\
q_{-i}^{(k)} &= p_e - p_e^{k-1}\gamma^{k-2} \min\{1, \gamma\delta(1-p_e) + p_e\gamma p_{-i}^{(1)}\}
\end{aligned}$$

Therefore from (2.2) the stage payoffs are:

$$u_{i\delta}^{(1)} = u_{i\ a}^{(1)} + \frac{\delta}{2\alpha - 1}$$

$$u_{i\delta}^{(k)} = u_{i\ a}^{(k)} - \frac{2\alpha(p_e\gamma)^{k-2}}{2\alpha - 1} \min\{1 - p_e\gamma p_{-i}^{(1)}, \gamma\delta(1-p_e)\}.$$

Since the stage payoff received at stage 0 is independent of the action player i takes at stage 1, we are only interested in finding the discounted average payoff

$$\begin{aligned}
U_{i\delta}^{(1)} &= \sum_{k=1}^{\infty} w^{k-1} u_{i\delta}^{(k)} \\
&= U_{i\ a}^{(1)} + \frac{1}{2\alpha - 1}\left[\delta - \frac{2\alpha w}{1 - wp_e\gamma} \min\{1 - p_e\gamma p_{-i}^{(1)}, \gamma\delta(1-p_e)\}\right],
\end{aligned}$$

where $U_{i\ a}^{(1)}$ is the discounted payoff received if $\delta = 0$. It does not pay to deviate if $U_{i\delta}^{(1)} < U_{i\ a}^{(1)}$. Since we assume that $\alpha \geq 1$, we only have to check two cases:

1. If $1 - p_e\gamma p_{-i}^{(1)} < \gamma\delta(1 - p_e)$ we need the condition

$$\delta - \frac{2\alpha w(1 - p_e\gamma p_{-i}^{(1)})}{1 - wp_e\gamma} < 0$$

to be true for any δ. Equivalently:

$$w > \max_{0\leq\delta\leq 1}\left\{\frac{\delta}{2\alpha(1 - p_e\gamma p_{-i}^{(1)}) + p_e\gamma\delta}\right\}.$$

So we get the bound:

$$w > \frac{1}{2\alpha(1 - p_e\gamma p_{-i}^{(1)}) + p_e\gamma}. \tag{B.1}$$

2. If $1 - p_e\gamma p_{-i}^{(1)} \geq \gamma\delta(1 - p_e)$ we need the following condition:

$$\delta - \frac{2\alpha w\gamma\delta(1 - p_e)}{1 - wp_e\gamma} < 0.$$

Thus we have the bound:

$$w > \frac{1}{2\alpha\gamma(1 - p_e) + p_e\gamma}. \tag{B.2}$$

For the case $q_i < q_{-i}$ the analysis has to be more detailed. In stage 1 according to DARWIN, player i has to drop player $-i$'s packets with probability

$$\tilde{p}_{i\ DARWIN}^{(1)} = \min\{1, \gamma(q_{-i} - q_i)\}.$$

So if $\gamma \geq \frac{1}{q_{-i}-q_i}$ then player i cannot deviate at stage 1 for any value of w. In the case that $\gamma < \frac{1}{q_{-i}-q_i}$ we can only increase δ up to:

$$\delta \leq 1 - \gamma(q_{-i} - q_i).$$

Now the rest of the analysis will consider the following two cases:

$$\delta \leq \min\left\{1 - \gamma(q_{-i} - q_i), \frac{p_e\gamma(q_{-i} - q_i)}{1 - p_e}\right\} \tag{B.3}$$

$$\frac{p_e\gamma(q_{-i} - q_i)}{1 - p_e} < \delta \leq 1 - \gamma(q_{-i} - q_i) \tag{B.4}$$

For the case when (B.3) is true we have the following evolution of the game:

$$
\begin{aligned}
p_i^{(1)} &= \gamma(q_{-i} - q_i) + \delta \\
p_i^{(k)} &= p_e^{k-1}\gamma^k(q_{-i} - q_i) - \delta p_e^{k-2}\gamma^{k-1}(1 - p_e) \\
p_{-i}^{(1)} &= 0 \\
p_{-i}^{(k)} &= 0 \\
\hat{p}_i^{(1)} &= p_e + \gamma(q_{-i} - q_i)(1 - p_e) + \delta(1 - p_e) \\
\hat{p}_i^{(k)} &= p_e + p_e^{k-1}\gamma^k(q_{-i} - q_i)(1 - p_e) - \delta p_e^{k-2}\gamma^{k-1}(1 - p_e)^2 \\
\hat{p}_{-i}^{(1)} &= p_e \\
\hat{p}_{-i}^{(k)} &= p_e \\
q_i^{(1)} &= p_e - p_e\gamma(q_{-i} - q_i) + \delta(1 - p_e) \\
q_i^{(k)} &= p_e - p_e^k\gamma^k(q_{-i} - q_i) + \delta p_e^{k-1}\gamma^{k-1}(1 - p_e) \\
q_{-i}^{(1)} &= p_e \\
q_{-i}^{(k)} &= p_e
\end{aligned}
$$

In this case, and from (2.2), the stage payoffs are:

$$u_{i\delta}^{(1)} = u_{i\ b}^{(1)} + \frac{\delta}{2\alpha - 1}$$

$$u_{i\delta}^{(k)} = u_{i\ b}^{(k)} - \frac{\delta p_e^{k-2}\gamma^{k-1}(1 - p_e)}{2\alpha - 1}.$$

And the discounted average payoff starting from stage 1 is:

$$U_{i\delta}^{(1)} = \sum_{k=1}^{\infty} w^{k-1}u_{i\delta}^{(k)} = U_{i\ b}^{(1)} + \frac{\delta}{2\alpha - 1}\left[1 - \frac{w\gamma(1 - p_e)}{1 - wp_e\gamma}\right].$$

Since $\alpha \geq 1$ it does not pay to deviate if:

$$1 - \frac{w\gamma(1 - p_e)}{1 - wp_e\gamma} < 0.$$

Which leads to the following bound on w:

$$w > \frac{1}{\gamma}. \tag{B.5}$$

For the case when (B.4) is true we have the following game evolution:

$$
\begin{aligned}
p_i^{(1)} &= \gamma(q_{-i} - q_i) + \delta \\
p_i^{(k)} &= 0 \\
p_{-i}^{(1)} &= 0 \\
p_{-i}^{(k)} &= p_e^{k-2}\gamma^{k-2}\min\{1, \gamma\delta(1-p_e) - p_e\gamma^2(q_{-i} - q_i)\} \\
\hat{p}_i^{(1)} &= p_e + \gamma(q_{-i} - q_i)(1-p_e) + \delta(1-p_e) \\
\hat{p}_i^{(k)} &= p_e \\
\hat{p}_{-i}^{(1)} &= p_e \\
\hat{p}_{-i}^{(k)} &= p_e + (1-p_e)p_{-i}^{(k)} \\
q_i^{(1)} &= p_e - p_e\gamma(q_{-i} - q_i) + \delta(1-p_e) \\
q_i^{(k)} &= p_e \\
q_{-i}^{(1)} &= p_e \\
q_{-i}^{(k)} &= p_e - p_e p_{-i}^{(k)}
\end{aligned}
$$

From (2.2), the respective stage payoffs are:

$$u_{i\delta}^{(1)} = u_{i\ b}^{(1)} + \frac{\delta}{2\alpha - 1}$$

$$u_{i\delta}^{(k)} = u_{i\ b}^{(k)} - \frac{(p_e\gamma)^{k-2}}{2\alpha - 1}\left[p_e\gamma^2(q_{-i} - q_i) + 2\alpha\min\{1, \gamma\delta(1-p_e) - p_e\gamma^2(q_{-i} - q_i)\}\right].$$

The discounted average payoff starting from stage 1 is:

$$
\begin{aligned}
U_{i\delta}^{(1)} &= \sum_{k=1}^{\infty} w^{k-1}u_{i\delta}^{(k)} \\
&= U_{i\ b}^{(1)} + \frac{1}{2\alpha - 1}\left\{\delta - \frac{w\left[p_e\gamma^2 Q + 2\alpha\min\{1, \gamma\delta(1-p_e) - p_e\gamma^2 Q\}\right]}{1 - wp_e\gamma}\right\},
\end{aligned}
$$

where $Q = q_{-i} - q_i$ and $U_{i\ b}^{(1)}$ is the discounted payoff received if player i does not deviate. It does not pay to deviate if $U_{i\delta}^{(1)} < U_{i\ b}^{(1)}$. Since we assume $\alpha \geq 1$, we have:

1. If $\gamma\delta(1-p_e) - p_e\gamma^2(q_{-i} - q_i) > 1$, we need the condition

$$\delta - \frac{w[2\alpha + p_e\gamma^2(q_{-i} - q_i)]}{1 - wp_e\gamma} < 0$$

to be true for any δ. Equivalently:

$$w > \max_{\delta}\left\{\frac{\delta}{2\alpha + p_e\gamma^2(q_{-i} - q_i) + p_e\gamma\delta}\right\}.$$

Since δ is bounded by (B.4) we get:

$$w > \frac{1-\gamma(q_{-i}-q_i)}{2\alpha + p_e\gamma^2(q_{-i}-q_i) + p_e\gamma[1-\gamma(q_{-i}-q_i)]}.$$

Simplifying:

$$w > \frac{1-\gamma(q_{-i}-q_i)}{2\alpha + p_e\gamma}. \tag{B.6}$$

2. If $\gamma\delta(1-p_e) - p_e\gamma^2(q_{-i}-q_i) \leq 1$, we need the following condition:

$$\delta - \frac{w\left[p_e\gamma^2 Q + 2\alpha\gamma\delta(1-p_e) - 2\alpha p_e\gamma^2 Q\right]}{1-wp_e\gamma} < 0,$$

where Q was defined above. Thus we have the bound:

$$w > \max_{\delta}\left\{\frac{\delta}{\delta\left[2\alpha\gamma(1-p_e)+p_e\gamma\right] - (2\alpha-1)p_e\gamma^2 Q}\right\}.$$

Since δ is bounded by (B.4) we get:

$$w > \frac{1}{\gamma}. \tag{B.7}$$

So for a given history h^n we have found five bounds that w has to fulfill in order for DARWIN to be a Nash equilibrium in a given subgame. We first start noting that (B.5) and (B.7) are identical, so we really have four bounds, two of which are dependent on h^n. In order to find the conditions under which DARWIN is subgame perfect, we need to find bounds that are history independent. In the case of (B.1) the bound is maximized by:

$$w > \frac{1}{2\alpha(1-p_e\gamma) + p_e\gamma}. \tag{B.8}$$

Similarly, (B.6) is maximized by:

$$w > \frac{1}{2\alpha + p_e\gamma}. \tag{B.9}$$

Comparing (B.2), (B.8) and (B.9) it is easy to check that (B.8) is the strictest bound since we assumed $\gamma > 1$. In summary, we have the following bound

on w for DARWIN:

$$w > \max\left\{\frac{1}{\gamma}, \frac{1}{2\alpha(1-p_e\gamma)+p_e\gamma}\right\}.$$

Thus if the bound holds true, by the One-Stage Deviation Principle, DARWIN is subgame perfect. □

APPENDIX C

PROOF OF LEMMA 4

We prove the result by contradiction. Let $f_j \in \mathcal{F}$ be the first admitted flow request that violates capacity constraints when admitted. Thus, there is a node $n \in \mathcal{N}$ and time $t \in [\, t_j^S, t_j^F \,)$ such that

$$\lambda_n(t) + Q_n(P_j) r_j(t) > 1$$

which necessarily means that $Q_n(P_j) \geq 1$ by the definition of $Q_n(P_j)$.

Therefore:

$$\begin{aligned}
c_n(t) &\overset{(3.9)}{=} \mu^{\lambda_n(t)} - 1 \\
&> \mu^{1 - Q_n(P_j) r_j(t)} - 1 \\
&\overset{(3.4)}{\geq} \mu^{1 - Q r_j(t)} - 1 \\
&\overset{(3.6)}{\geq} \mu^{1 - \frac{1}{\log \mu}} - 1 \\
&= \frac{\mu}{2} - 1 \\
&\overset{(3.7)}{=} Q_T T F.
\end{aligned}$$

So we have

$$c_n(t) > Q_T T F,$$

which implies

$$\begin{aligned}
Q_n(P_j) r_j(t) c_n(t) &> Q_n(P_j) r_j(t) Q_T T F \\
&\geq r_j(t) Q_T T F \\
&\overset{(3.5)}{\geq} \rho_j.
\end{aligned}$$

So flow f_j should not have been admitted in the first place according to the ACR algorithm. □

APPENDIX D

PROOF OF LEMMA 5

The proof will proceed by induction over k. The case $k = 0$ is trivially true since both sides of the inequality are zero. If a request is rejected, then neither side of the inequality changes, so we only need to prove that for any accepted flow f_i we have:

$$2\rho_i \log \mu \geq \sum_t \sum_n \left[c_n(t, i+1) - c_n(t, i) \right].$$

Consider any node $n \in \mathcal{N}$. Using the fact that

$$2^x - 1 \leq x \ for \ x \in [0, 1] \tag{D.1}$$

we have the following:

$$\begin{aligned}
c_n(t, i+1) - c_n(t, i) &\overset{(3.13)}{=} \mu^{\lambda_n(t,i+1)} - \mu^{\lambda_n(t,i)} \\
&\overset{(3.12)}{=} \mu^{\lambda_n(t,i)} \left[\mu^{Q_n(P_i) r_i(t)} - 1 \right] \\
&= \mu^{\lambda_n(t,i)} \left[2^{Q_n(P_i) r_i(t) \log \mu} - 1 \right] \\
&\overset{(3.6)(D.1)}{\leq} \mu^{\lambda_n(t,i)} Q_n(P_i) r_i(t) \log \mu \\
&\overset{(3.13)}{=} \left[c_n(t, i) + 1 \right] Q_n(P_i) r_i(t) \log \mu \\
&= \log \mu \left[Q_n(P_i) r_i(t) c_n(t, i) + Q_n(P_i) r_i(t) \right].
\end{aligned}$$

Therefore, we have

$$\begin{aligned}
& \sum_t \sum_n [c_n(t, i+1) - c_n(t, i)] \\
\leq \quad & \log \mu \left[\sum_t \sum_n Q_n(P_i) r_i(t) c_n(t, i) + \sum_t \sum_n Q_n(P_i) r_i(t) \right] \\
\overset{(3.10)}{\leq} \quad & \log \mu \left[\rho_i + \sum_n Q_n(P_i) \sum_t r_i(t) \right] \\
\overset{(3.3)}{=} \quad & \log \mu \left[\rho_i + Q_T(P_i) \sum_t r_i(t) \right] \\
\overset{(3.5)}{\leq} \quad & 2\rho_i \log \mu.
\end{aligned}$$

□

APPENDIX E

PROOF OF LEMMA 6

Since $j \in \mathcal{A}_{O\setminus A}$ was rejected by the ACR algorithm it must be the case that (3.10) was false. Due to the monotonicity of $c_n(t,j)$ on i we have

$$\begin{aligned} \rho_j &< \sum_n \sum_t Q_n(P_j) r_j(t) c_n(t,j) \\ &\leq \sum_n \sum_t Q_n(P_j) r_j(t) c_n(t,m). \end{aligned}$$

Summing up over $j \in \mathcal{A}_{O\setminus A}$, and taking into account that the off-line algorithm must enforce capacity constraints, we have:

$$\begin{aligned} \sum_{j\in\mathcal{A}_{O\setminus A}} \rho_j &\leq \sum_{j\in\mathcal{A}_{O\setminus A}} \sum_n \sum_t Q_n(P_j) r_j(t) c_n(t,m) \\ &= \sum_n \sum_t \left[c_n(t,m) \sum_{j\in\mathcal{A}_{O\setminus A}} Q_n(P_j) r_j(t) \right] \\ &\overset{(3.12)(3.8)}{\leq} \sum_n \sum_t c_n(t,m). \end{aligned}$$

□

APPENDIX F

PROOF OF THEOREM 4

Using the monotonicity of $c_n(t,i)$ on i, we have that the profit accrued by the optimal off-line algorithm can be bounded as follows:

$$\begin{aligned}
\sum_{j\in\mathcal{A}_{OFF}} \rho_j &\leq \sum_{j\in\mathcal{A}_{O\setminus A}} \rho_j + \sum_{j\in\mathcal{A}_{ACR}} \rho_j \\
&\overset{Lemma\ 6}{\leq} \sum_{n}\sum_{t} c_n(t,m) + \sum_{j\in\mathcal{A}_{ACR}} \rho_j \\
&\leq \sum_{n}\sum_{t} c_n(t,k+1) + \sum_{j\in\mathcal{A}_{ACR}} \rho_j \\
&\overset{Lemma\ 5}{\leq} 2\log\mu \sum_{j\in\mathcal{A}_{ACR}} \rho_j + \sum_{j\in\mathcal{A}_{ACR}} \rho_j \\
&= (2\log\mu + 1) \sum_{j\in\mathcal{A}_{ACR}} \rho_j .
\end{aligned}$$

Therefore:

$$\sum_{j\in\mathcal{A}_{OFF}} \rho_j = O(\log(Q_T TF)) \sum_{j\in\mathcal{A}_{ACR}} \rho_j .$$

And using Lemma 4 the proof is done. □

APPENDIX G

PROOF OF LEMMA 7

The idea behind the proof is simple. We need to construct a sequence of requests that have unit holding time, and each offers the same profit, but its path length is exponentially decreasing; thus, future requests generate greater profit per unit of resource consumed. Since no on-line algorithm has knowledge about future requests, we generate requests until the ratio between the profits of the off-line and on-line algorithms is maximized.

Assume that all requests appear at the beginning and request some fixed rate r that is small enough according to (3.6). Define $W(Q_T)$ to be a wireless network of $Q_T/2$ nodes arranged in a ring such that any node can only communicate and interfere with its nearest neighbors. Number the nodes from 0 to $Q_T/2 - 1$ and for convenience let Q_T be a power of 2.

Consider the following sequence of requests of unit duration consisting of $\log(Q_T/2)$ phases, where phase p ($1 \leq p \leq \log(Q_T/2)$) consists of 2^p groups of requests.

A request in phase p, group g ($0 \leq g \leq 2^p - 1$), has as source node $g(Q_T/2)2^{-p}$ and destination node $[(g+1)(Q_T/2)2^{-p}] \mod (Q_T/2)$. Each group consists of $1/4r$ identical requests, requesting rate r and offering the same profit, e.g., rQ_T.

Let u_p be the profit the on-line algorithm receives due to requests in phase p. From the definitions of $Q_n(P_j)$ and $Q_T(P_j)$, and using (3.12) we know that in order to achieve a unit of profit we need to use $2^{-(p-1)}$ units of capacity, which implies that at phase p we have used $u_p 2^{-(p-1)}$ units of capacity. Since total network capacity is $Q_T/2$ we have:

$$\sum_{p=1}^{\log(Q_T/2)} \frac{u_p}{2^{p-1}} \leq \frac{Q_T}{2}. \tag{G.1}$$

Define $S_p = \frac{1}{2^p} \sum_{i=1}^{p} u_i$. Thus we have:

$$\begin{aligned}
\sum_{p=1}^{\log(Q_T/2)} S_p &= \sum_{p=1}^{\log(Q_T/2)} \sum_{i=1}^{p} \frac{u_i}{2^p} \\
&\leq \sum_{p=1}^{\log(Q_T/2)} \frac{u_p}{2^{p-1}} \\
&\overset{(G.1)}{\leq} \frac{Q_T}{2}.
\end{aligned}$$

Hence, for any on-line algorithm there exists k such that

$$S_k \leq \frac{Q_T/2}{\log(Q_T/2)}.$$

Now consider a new set of requests consisting only of the first k phases. The profit accrued by the on-line algorithm is then

$$\sum_{p=1}^{k} u_p = 2^k S_k \leq 2^k \frac{Q_T/2}{\log(Q_T/2)}.$$

The off-line algorithm can reject all the requests but those in phase k, achieving a profit of $2^k Q_T/4$. So the competitive ratio for this topology will be greater than or equal to $\frac{1}{2}\log(Q_T/2)$. Thus, the competitive ratio is $\Omega(\log Q_T)$ for any on-line algorithm. □

APPENDIX H

PROOF OF LEMMA 8

To proceed to prove this lemma we use the same ideas as in Lemma 7, by considering a wireless network consisting of two nodes that are within reception range of each other and constructing a sequence of requests where each request offers the same profit and has an exponentially decreasing holding time.

Assume that all requests appear at the beginning and request some fixed rate r that is small enough according to (3.6). Consider a wireless network comprised of two nodes that are within reception range of each other, and a sequence of requests consisting of $\log T$ phases, where T is a power of 2 and phase p ($1 \leq p \leq \log T$) consists of 2^p groups of requests. Each group consists of $1/r$ requests, where each request requires a rate r and provides the same profit, e.g. $Q_T r T$. In this case we have $Q_T = 2$.

A request in phase p, group g ($0 \leq g \leq 2^p - 1$), has as starting time $gT2^{-p}$ and finishing time $(g+1)T2^{-p}$.

Let u_p be the profit the on-line algorithm receives due to requests in phase p. From the definitions of $Q_n(P_j)$ and using (3.12) we know that in order to achieve a unit of profit we need to send $2^{-(p+1)}$ bits, which implies that at phase p we have sent $u_p 2^{-(p+1)}$ bits. Since total available bits is T we have:

$$\sum_{p=1}^{\log T} \frac{u_p}{2^{p+1}} \leq T. \tag{H.1}$$

Define $S_p = \frac{1}{2^{p+2}} \sum_{i=1}^{p} u_i$. Thus, we have:

$$\sum_{p=1}^{\log T} S_p = \sum_{p=1}^{\log T} \sum_{i=1}^{p} \frac{u_i}{2^{p+2}} \leq \sum_{p=1}^{\log T} \frac{u_p}{2^{p+1}} \overset{(H.1)}{\leq} T.$$

Hence, for any on-line algorithm there exists k such that

$$S_k \leq \frac{T}{\log T}.$$

Now consider a new set of requests consisting only of the first k phases. The profit accrued by the on-line algorithm is then

$$\sum_{p=1}^{k} u_p = 2^{k+2} S_k \leq \frac{2^{k+2} T}{\log T}.$$

The off-line algorithm can reject all the requests but those in phase k, achieving a profit of $2^{k+1}T$. So the competitive ratio for this topology will be greater than or equal to $\frac{1}{2}\log T = \Omega(\log T)$. □

APPENDIX I

PROOF OF LEMMA 9

Borrowing from the ideas in Lemma 7, to prove this lemma, consider a wireless network consisting of two nodes that are within reception range of each other and construct a sequence of requests of unit holding times that have an exponentially increasing profit.

Assume that all requests appear at the beginning and request some fixed rate r that is small enough according to (3.6). Consider a wireless network comprised of two nodes that are within reception range of each other, and a sequence of requests consisting of $\log F$ phases, where F is a power of 2 and phase p ($1 \leq p \leq \log F$) consists of $1/r$ requests of unit duration, where each request requires a rate r. In this case we have $Q_T = 2$.

A request in phase p provides a profit of $r2^{p+1}$. It is easy to check that such a profit complies with (3.5).

Let u_p be the profit the on-line algorithm receives due to requests in phase p. From the definitions of $Q_n(P_j)$ and $Q_T(P_j)$, and using (3.12) we know that in order to achieve a unit of profit we need to use 2^{-p} units of capacity, which implies that at phase p we have used $u_p 2^{-p}$ units of capacity. Since total network capacity is 2 we have:

$$\sum_{p=1}^{\log F} \frac{u_p}{2^p} \leq 2. \tag{I.1}$$

Define $S_p = \frac{1}{2^{p+1}} \sum_{i=1}^{p} u_i$. Thus, we have:

$$\sum_{p=1}^{\log F} S_p = \sum_{p=1}^{\log F} \sum_{i=1}^{p} \frac{u_i}{2^{p+1}} \leq \sum_{p=1}^{\log F} \frac{u_p}{2^p} \overset{(I.1)}{\leq} 2.$$

Hence, for any on-line algorithm there exists k such that

$$S_k \leq \frac{2}{\log F}.$$

Now consider a new set of requests consisting only of the first k phases. The profit accrued by the on-line algorithm is then

$$\sum_{p=1}^{k} u_p = 2^{k+1} S_k \leq \frac{2^{k+2}}{\log F}.$$

The off-line algorithm can reject all the requests but those in phase k, achieving a profit of 2^{k+1}. So the competitive ratio for this topology will be greater than or equal to $\frac{1}{2} \log F = \Omega(\log F)$. □

APPENDIX J

PROOF OF LEMMA 10

Similar to Lemma 7, the idea is to construct a sequence of requests such that future requests generate greater profit per unit of resource used. Since no on-line algorithm has knowledge about future requests, we generate requests until we maximize the competitive ratio.

Define W(Q_T) to be a wireless network of $Q_T/4+3$ nodes uniformly spaced on a line segment such that any node can only communicate with its nearest neighbors, and assume that Q_T is such that $Q_T^{\frac{1}{4k}}$ is an even number greater than or equal to 4.

Number the nodes from 0 to $Q_T/4+2$ and consider the following sequence of requests of unit duration and rate $1/4k$ consisting of $k+1$ phases, where each request offers the same profit, e.g., $Q_T/4k$.

In phase 0 there is a request from node 1 to node $Q_T/4+1$. The first request must be accepted because it can be the only one. Using the definition of $Q_T(P_j)$ and $Q_n(P_j)$ it is easy to check that $Q_T(P_0) = Q_T$. This means that before phase 1 the load of any node in the range $1 + Q_T^{1-\frac{1}{4k}}/4$ to $1 + (Q_T^{\frac{1}{4k}} - 1)Q_T^{1-\frac{1}{4k}}/4$ is $\lambda_n = 1/k$, with the profit accrued being $Q_T/4k$.

We claim that for phases 1 through k the following statement holds true: Either the lower bound has already been proved, or before phase j the load of any node in the range $i_j + Q_T^{1-\frac{j}{4k}}/4$ to $i_j + (Q_T^{\frac{1}{4k}} - 1)Q_T^{1-\frac{j}{4k}}/4$ for some i_j is $\lambda_n = j/k$ and the total profit accrued so far is $jQ_T/4k$ by the on-line algorithm.

To prove the claim we use induction. For phase 1 let $i_1 = 1$. We now assume that the statement is true for phase j and some i_j and prove it for $j+1$ and some i_{j+1}.

We now make k requests with source node $i_j + mQ_T^{1-\frac{j}{4k}}/4$ and destination $i_j + (m+1)Q_T^{1-\frac{j}{4k}}/4$ for $1 \leq m \leq Q_T^{\frac{1}{4k}} - 2$. If all the requests are rejected by the on-line algorithm then the lower bound has been proved since the off-line algorithm can reject all requests but those in phase j and achieve a

profit of $\left(Q_T^{\frac{1}{4k}} - 2\right) Q_T/4$ while the on-line algorithm only accrued a profit of $jQ_T/4k \leq Q_T/4$.

However, if there is an accepted request we stop phase j after such a request. That request is from node $i_j + mQ_T^{1-\frac{j}{4k}}/4$ to $i_j + (m+1)Q_T^{1-\frac{j}{4k}}/4$ for some $1 \leq m \leq Q_T^{\frac{1}{4k}} - 2$. Define $i_{j+1} = i_j + mQ_T^{1-\frac{j}{4k}}/4$. We have that the total profit accrued by the on-line algorithm before phase $j+1$ is $jQ_T/4k + Q_T/4k = (j+1)\,Q_T/4k$ and the load of the nodes from $i_{j+1} + Q_T^{1-\frac{j+1}{4k}}/4$ to $i_{j+1} + (Q_T^{\frac{1}{4k}} - 1)Q_T^{1-\frac{j+1}{4k}}/4$ is $4(j+1)/4k = (j+1)/k$, which proves the claim.

The statement means that at the start of phase k either the lower bound has already been proved or the load of any node in the range $i_k + Q_T^{\frac{3}{4}}/4$ to $i_k + (Q_T^{\frac{1}{4k}} - 1)Q_T^{\frac{3}{4}}/4$ is $\lambda_n = 1$. Thus, the on-line algorithm has to reject all the k requests from node $i_k + mQ_T^{\frac{3}{4}}/4$ to $i_k + (m+1)Q_T^{\frac{3}{4}}/4$ for $1 \leq m \leq Q_T^{\frac{1}{4k}} - 2$, while the off-line algorithm can reject all requests but those at phase k, and achieve a total profit of $Q_T\left(Q_T^{\frac{1}{4k}} - 2\right)/4$. The total profit of the on-line algorithm is bounded by $Q_T/4$, which proves the lower bound. $\square$

APPENDIX K

PROOF OF LEMMA 11

The proof follows the same line of reasoning as in Lemma 10, where we consider a wireless network consisting of two nodes that are within reception range of each other and construct a sequence of requests, each of which offers the same profit and has an exponentially decreasing holding time.

Consider a wireless network comprised of two nodes that are within reception range of each other and assume that T is such that $T^{\frac{1}{k}}$ is an integer greater than or equal to 2.

Furthermore, consider the following sequence of requests consisting of $k+1$ phases, where each request asks for a rate of $\frac{1}{k}$ and offers the same profit, e.g., TQ_T/k. In this case we have $Q_T = 2$.

In phase 0 there is a request with starting time 0 and finishing time T. The first request must be accepted because it can be the only one. This means that before phase 1 the load of both nodes from time 0 to T is $1/k$, with an accrued profit of $2T/k$.

We claim that for phases 1 through k the following statement holds true: Either the lower bound has already been proved, or before phase j the load of both nodes from time t_j to $t_j + T^{1-\frac{j-1}{k}}$ is j/k, with an accrued profit of $2Tj/k$ for the on-line algorithm.

To prove the claim we use induction. For phase 1 let $t_j = 0$. We now assume that the statement is true for phase j and some t_j and prove it for $j+1$ and some t_{j+1}.

We now make k requests from time $t_j + mT^{1-\frac{j}{k}}$ to $t_j + (m+1)T^{1-\frac{j}{k}}$ for $0 \leq m \leq T^{\frac{1}{k}} - 1$. If all the requests are rejected by the on-line algorithm then the lower bound has been proved since the off-line algorithm can reject all requests but those in phase j and achieve a profit of $2T^{1+\frac{1}{k}}$ while the on-line algorithm only accrued a profit of $2Tj/k \leq 2T$.

However, if there is an accepted request we stop phase j after such a request. That request is from time $t_j + mT^{1-\frac{j}{k}}$ to $t_j + (m+1)T^{1-\frac{j}{k}}$ for

some $0 \leq m \leq T^{\frac{1}{k}} - 1$. Define $t_{j+1} = t_j + mT^{1-\frac{j}{k}}$. We have that the total profit accrued by the on-line algorithm before phase $j+1$ is $2Tj/k + 2T/k = 2T(j+1)/k$ and the load of both nodes from time t_{j+1} to $t_{j+1} + T^{1-\frac{j}{k}}$ is $(j+1)/k$, which proves the claim.

The statement means that at the start of phase k either the lower bound has already been proved or the load of both nodes between time t_k and $t_k + T^{\frac{1}{k}}$ is 1. Thus, the on-line algorithm has to reject all the k requests from $t_k + m$ to $t_k + m + 1$ for $0 \leq m \leq T^{\frac{1}{k}} - 1$, while the off-line algorithm can reject all requests but the ones at phase k, achieving a profit of $2T^{1+\frac{1}{k}}$. The profit of the on-line algorithm is bounded by $2T$, which proves the lower bound. $\square$

APPENDIX L

PROOF OF LEMMA 12

To prove this lemma we proceed along the same lines as in Lemma 10, by considering a pair of nodes that are within reception range of each other and constructing a sequence of requests of unit duration where the profit increases exponentially.

Consider a wireless network comprised of two nodes that are within reception range of each other, and the following sequence of requests of unit duration consisting of $k+1$ phases, where each request asks for a rate of $1/k$. In this case we have $Q_T = 2$. A request in phase j provides a profit of $2F^{\frac{j}{k}}/k$ for $0 \leq j \leq k$.

In phase 0 there is a request, which must be accepted because it can be the only one. This means that before phase 1 the load of both nodes is $1/k$ and the profit is $2/k$.

We claim that for phases 1 through k the following statement holds true: Either the lower bound has already been proved, or before phase j the load of both nodes is j/k, with an accrued profit of

$$\sum_{i=0}^{j-1} \frac{2F^{\frac{i}{k}}}{k}$$

for the on-line algorithm.

To prove the claim we use induction. For phase 1 the claim is clearly true, so we assume that the statement is true for phase j and prove it for phase $j+1$.

We now make k requests. If all the requests are rejected by the on-line algorithm, then the lower bound has been proved since the off-line algorithm can reject all requests but those in phase j and achieve a profit of $2F^{\frac{j}{k}}$ while

the on-line algorithm only accrued a profit of

$$\sum_{i=0}^{j-1} \frac{2F^{\frac{i}{k}}}{k} \leq \frac{2jF^{\frac{j-1}{k}}}{k} \leq 2F^{\frac{j-1}{k}}.$$

However, if there is an accepted request we stop phase j after such a request. We have that the total profit accrued by the on-line algorithm before phase $j+1$ is

$$\sum_{i=0}^{j} \frac{2F^{\frac{i}{k}}}{k}$$

and the load of both nodes is $(j+1)/k$, which proves the claim.

The statement means that at the start of phase k either the lower bound has already been proved or the load of both nodes is 1. Thus, the on-line algorithm has to reject all the k requests in the last phase, while the off-line algorithm can reject all requests but those at phase k, achieving a profit of $2F$. The profit of the on-line algorithm is bounded by $2F^{1-\frac{1}{k}}$, which proves the lower bound. □

APPENDIX M

PROOF OF LEMMA 13

To prove Lemma 13, we start by first proving two auxiliary lemmas and then stating a fact.

Lemma 18. *Given that at frame k we have the event $d(k) = d$ and $q(k) = q$, then*

$$
\begin{aligned}
&E\left[\frac{1}{2}\sum_{l\in\mathcal{L}}\{[d_l + \tilde{a}_{il}(k) - I^*_{il}(a_i(k), c(k), d, q)]^+\}^2\right] - \sum_{l\in\mathcal{L}}\frac{d_l^2}{2} \\
\leq & B_6 + \sum_{l\in\mathcal{L}} d_l\lambda_l(1-p_l) - E\left[\sum_{l\in\mathcal{L}}\left(\frac{1}{\epsilon}w_l + d_l\right) I^*_{il}(a_i(k), c(k), d, q)\right. \\
& \left. - \sum_{l\in\mathcal{L}}\frac{1}{\epsilon}w_l I^*_{il}(a_i(k), c(k), d, q)\right]
\end{aligned}
$$

for some non-negative constant B_6, and where $I^*_{il}(a_i(k), c(k), d, q)$ is given by the solution to (4.6). ◇

Proof.

$$
\begin{aligned}
&E\left[\frac{1}{2}\sum_{l\in\mathcal{L}}\{[d_l+\tilde{a}_{il}(k)-I^*_{il}(a_i(k),c(k),d,q)]^+\}^2\right]-\sum_{l\in\mathcal{L}}\frac{d_l^2}{2}\\
\leq&E\left[\frac{1}{2}\sum_{l\in\mathcal{L}}[d_l+\tilde{a}_{il}(k)-I^*_{il}(a_i(k),c(k),d,q)]^2\right]-\sum_{l\in\mathcal{L}}\frac{d_l^2}{2}\\
=&E\left[\sum_{l\in\mathcal{L}}d_l[\tilde{a}_{il}(k)-I^*_{il}(a_i(k),c(k),d,q)]\right.\\
&\left.+\frac{1}{2}\sum_{l\in\mathcal{L}}[\tilde{a}_{il}(k)-I^*_{il}(a_i(k),c(k),d,q)]^2\right]\\
\leq&E\left[\sum_{l\in\mathcal{L}}d_l\tilde{a}_{il}(k)-d_lI^*_{il}(a_i(k),c(k),d,q)+\frac{1}{2}\sum_{l\in\mathcal{L}}\tilde{a}^2_{il}(k)+a^2_{il}(k)\right] \qquad \text{(M.1)}\\
\leq&B_6+\sum_{l\in\mathcal{L}}d_l\lambda_l(1-p_l)\\
&-E\left[\sum_{l\in\mathcal{L}}\left(\frac{1}{\epsilon}w_l+d_l\right)I^*_{il}(a_i(k),c(k),d,q)-\sum_{l\in\mathcal{L}}\frac{1}{\epsilon}w_lI^*_{il}(a_i(k),c(k),d,q)\right]
\end{aligned}
$$

where (M.1) follows from the definition of $I^*_{il}(a_i(k),c(k),d,q)$ and

$$B_6=\frac{1}{2}\sum_{l\in\mathcal{L}}(\lambda_l^2+\sigma_{il}^2)[1+(1-p_l)^2]+\lambda_lp_l(1-p_l).$$

□

Lemma 19. *Given that at frame k we have the event $d(k)=d$ and $q(k)=q$, then*

$$
\begin{aligned}
&E\left[\frac{1}{2}\sum_{l\in\mathcal{L}}\{[q_l+\tilde{a}_{el}(k)-I^*_{el}(a_i(k),c(k),d,q)]^+\}^2\right]-\sum_{l\in\mathcal{L}}\frac{q_l^2}{2}\\
\leq&B_7-\frac{1}{\epsilon}\sum_{l\in\mathcal{L}}[U_l(x^*_l)-U_l(\tilde{x}^*_l(k))]-\sum_{l\in\mathcal{L}}q_l\left\{E\left[I^*_{el}(a_i(k),c(k),d,q)\right]-x^*_l\right\}
\end{aligned}
$$

for some constant $B_7>0$, where x^* and $\tilde{x}^*(k)$ are the solutions to (4.3) and (4.5) respectively. ◇

Proof.

$$
\begin{aligned}
&E\left[\frac{1}{2}\sum_{l\in\mathcal{L}}\{[q_l+\tilde{a}_{el}(k)-I^*_{el}(a_i(k),c(k),d,q)]^+\}^2\right]-\sum_{l\in\mathcal{L}}\frac{q_l^2}{2}\\
\leq&E\left[\frac{1}{2}\sum_{l\in\mathcal{L}}[q_l+\tilde{a}_{el}(k)-I^*_{el}(a_i(k),c(k),d,q)]^2\right]-\sum_{l\in\mathcal{L}}\frac{q_l^2}{2}\\
=&E\left[\sum_{l\in\mathcal{L}}q_l[\tilde{a}_{el}(k)-I^*_{el}(a_i(k),c(k),d,q)]\right.\\
&\left.+\frac{1}{2}\sum_{l\in\mathcal{L}}[\tilde{a}_{el}(k)-I^*_{el}(a_i(k),c(k),d,q)]^2\right]\\
\leq&E\left[\sum_{l\in\mathcal{L}}q_l\tilde{a}_{el}(k)-q_lI^*_{el}(a_i(k),c(k),d,q)+\frac{1}{2}\sum_{l\in\mathcal{L}}(\tilde{a}^2_{el}(k)+c_l^2T^2)\right] \qquad \text{(M.2)}\\
\leq&B_7+\sum_{l\in\mathcal{L}}-[\frac{1}{\epsilon}U_l(\tilde{x}^*_l(k))-q_l\tilde{x}^*_l(k)]+\sum_{l\in\mathcal{L}}\frac{1}{\epsilon}U_l(\tilde{x}^*_l(k))\\
&-\sum_{l\in\mathcal{L}}q_lE\left[I^*_{el}(a_i(k),c(k),d,q)\right]\\
\leq&B_7+\sum_{l\in\mathcal{L}}-[\frac{1}{\epsilon}U_l(x^*_l)-q_lx^*_l]+\sum_{l\in\mathcal{L}}\frac{1}{\epsilon}U_l(\tilde{x}^*_l(k)) \qquad \text{(M.3)}\\
&-\sum_{l\in\mathcal{L}}q_lE\left[I^*_{el}(a_i(k),c(k),d,q)\right]\\
=&B_7-\frac{1}{\epsilon}\sum_{l\in\mathcal{L}}[U_l(x^*_l)-U_l(\tilde{x}^*_l(k))]-\sum_{l\in\mathcal{L}}q_l\left\{E\left[I^*_{el}(a_i(k),c(k),d,q)\right]-x^*_l\right\}
\end{aligned}
$$

where (M.2) follows from the definition of $I^*_{el}(a_i(k),c(k),d,q)$,

$$B_7=\frac{1}{2}\sum_{l\in\mathcal{L}}X^2_{max}+\sigma_e^2+(\bar{c}_l^2+\sigma_{cl}^2)T^2,$$

and (M.3) follows from the fact that $\tilde{x}^*(k)$ is the optimal point of (4.5). □

Fact 2. *The optimization in (4.6) can be performed over $\mathcal{S}(a_i(k),c(k))_{\mathcal{CH}}$, the convex hull of $\mathcal{S}(a_i(k),c(k))$; that is,*

$$
\begin{aligned}
&\max_{s\in\mathcal{S}(a_i(k),c(k))}\sum_{l\in\mathcal{L}}\left\{\left[\frac{1}{\epsilon}w_l+d_l(k)\right]\sum_{t=1}^{T}s_{il,t}+q_l(k)\sum_{t=1}^{T}s_{el,t}\right\}=\\
&\max_{s\in\mathcal{S}(a_i(k),c(k))_{\mathcal{CH}}}\sum_{l\in\mathcal{L}}\left\{\left[\frac{1}{\epsilon}w_l+d_l(k)\right]\sum_{t=1}^{T}s_{il,t}+q_l(k)\sum_{t=1}^{T}s_{el,t}\right\}.
\end{aligned}
$$

The reason for this comes from the fact that the objective function is linear and therefore there must be an optimal point $s^*(a_i(k), c(k), d(k), q(k))) \in \mathcal{S}(a_i(k), c(k))$. $\diamond$

Proof of Lemma 13. For the purpose of this proof, we define the capacity region for fixed arrival and channel states a_i and c as follows:

$$\mathcal{C}(a_i, c) \stackrel{def}{=} \left\{ \begin{array}{l} (\bar{\mu}_{il}, \bar{\mu}_{el})_{l\in\mathcal{L}} : \text{there exists } \bar{s} \in \mathcal{S}(a_i, c)_{\mathcal{CH}}, \\ \bar{\mu}_{il} \leq \sum_{t=1}^{T} \bar{s}_{il,t} \text{ and } \bar{\mu}_{el} \leq \sum_{t=1}^{T} \bar{s}_{el,t} \end{array} \right\}.$$

Then, the overall capacity of the network is defined as $\mathcal{C} \stackrel{def}{=}$

$$\left\{ \begin{array}{l} (\mu_{il}, \mu_{el})_{l\in\mathcal{L}} : \text{there exists } (\bar{\mu}_{il}(a_i, c), \bar{\mu}_{el}(a_i, c))_{l\in\mathcal{L}} \in \mathcal{C}(a_i, c) \text{ for all } a_i, c \\ \text{ and } \mu_{il} = E[\bar{\mu}_{il}(a_i, c)],\ \mu_{el} = E[\bar{\mu}_{el}(a_i, c)] \text{ for all } l \in \mathcal{L} \end{array} \right\}.$$

From Lemmas 18 and 19 we have:

$$\begin{aligned}
&E\left[V(d(k+1), q(k+1))|d(k) = d, q(k) = q\right] - V(d, q) \\
\leq& B_1 + \sum_{l\in\mathcal{L}} d_l \lambda_l (1 - p_l) + \sum_{l\in\mathcal{L}} q_l x_l^* - \frac{1}{\epsilon} \sum_{l\in\mathcal{L}} [U_l(x_l^*) - U_l(\tilde{x}_l^*(k))] \\
&- E\left[\sum_{l\in\mathcal{L}} \left(\frac{1}{\epsilon} w_l + d_l\right) I_{il}^*(a_i(k), c(k), d, q) + \sum_{l\in\mathcal{L}} q_l I_{el}^*(a_i(k), c(k), d, q)\right] \\
&+ \sum_{l\in\mathcal{L}} \frac{1}{\epsilon} w_l E\left[I_{il}^*(a_i(k), c(k), d, q)\right] \\
\leq& B_1 + \sum_{l\in\mathcal{L}} d_l \lambda_l (1 - p_l) + \sum_{l\in\mathcal{L}} q_l x_l^* - \frac{1}{\epsilon} \sum_{l\in\mathcal{L}} [U_l(x_l^*) - U_l(\tilde{x}_l^*(k))] \qquad \text{(M.4)} \\
&- E\left[\sum_{l\in\mathcal{L}} \left(\frac{1}{\epsilon} w_l + d_l\right) \bar{\mu}_{il}(a_i(k), c(k)) + \sum_{l\in\mathcal{L}} q_l \bar{\mu}_{el}(a_i(k), c(k))\right] \\
&+ \sum_{l\in\mathcal{L}} \frac{1}{\epsilon} w_l E\left[I_{il}^*(a_i(k), c(k), d, q)\right] \\
=& B_1 - \sum_{l\in\mathcal{L}} d_l [\mu_{il} - \lambda_l (1 - p_l)] - \sum_{l\in\mathcal{L}} q_l (\mu_{el} - x_l^*) \qquad \text{(M.5)} \\
&- \frac{1}{\epsilon} \sum_{l\in\mathcal{L}} [U_l(x_l^*) - U_l(\tilde{x}_l^*(k))] - \frac{1}{\epsilon} \sum_{l\in\mathcal{L}} w_l \mu_{il} - w_l E\left[I_{il}^*(a_i(k), c(k), d, q)\right]
\end{aligned}$$

where $B_1 = B_6 + B_7$, (M.4) follows for any $(\bar{\mu}_{il}(a_i(k), c(k)), \bar{\mu}_{el}(a_i(k), c(k)))_{l\in\mathcal{L}} \in \mathcal{C}(a_i(k), c(k))$ as was explained in Fact 2, and (M.5) holds for any $(\mu_{il}, \mu_{el})_{l\in\mathcal{L}} \in \mathcal{C}$. It should be clear that $(\mu_{il}^*, \mu_{el}^*)_{l\in\mathcal{L}} \in \mathcal{C}$, where $(\mu_{il}^*, \mu_{el}^*)_{l\in\mathcal{L}}$ is the solution

to (4.3). Thus we have the following:

$$\begin{aligned}
&E\left[V(d(k+1),q(k+1))|d(k)=d,q(k)=q\right]-V(d,q)\\
\leq& B_1-B_2\sum_{l\in\mathcal{L}}d_l-B_3\sum_{l\in\mathcal{L}}q_l-\frac{1}{\epsilon}\sum_{l\in\mathcal{L}}[U_l(x_l^*)-U_l(\tilde{x}_l^*(k))]\\
&-\frac{1}{\epsilon}\sum_{l\in\mathcal{L}}w_l\mu_{il}^*-w_lE\left[I_{il}^*(a_i(k),c(k),d,q)\right]
\end{aligned}$$

where

$$B_2=\min_{l\in\mathcal{L}}\left\{\mu_{il}^*-\lambda_l(1-p_l)\right\}$$

and

$$B_3=\min_{l\in\mathcal{L}}\left\{\mu_{el}^*-x_l^*\right\}.$$

□

APPENDIX N

PROOF OF THEOREM 7

From Lemma 13 we know that

$$\begin{aligned}
&\frac{1}{\epsilon}\sum_{l\in\mathcal{L}} U_l(x_l^*) - U_l(\tilde{x}_l^*(k)) + \frac{1}{\epsilon}\sum_{l\in\mathcal{L}} w_l\mu_{il}^* - w_l E\left[I_{il}^*(a_i(k), c(k), d, q)\right] \\
\leq & B_1 - B_2\sum_{l\in\mathcal{L}} d_l - B_3\sum_{l\in\mathcal{L}} q_l + V(d,q) \\
& - E\left[V(d(k+1), q(k+1))|d(k)=d, q(k)=q\right] \\
\leq & B_1 + V(d,q) - E\left[V(d(k+1), q(k+1))|d(k)=d, q(k)=q\right]
\end{aligned}$$

since $B_2\sum_{l\in\mathcal{L}} d_l + B_3\sum_{l\in\mathcal{L}} q_l \geq 0$ for all d. Taking expectations:

$$\begin{aligned}
&\frac{1}{\epsilon}E\left[\sum_{l\in\mathcal{L}} U_l(x_l^*) - U_l(\tilde{x}_l^*(k)) + \sum_{l\in\mathcal{L}} w_l\mu_{il}^* - w_l I_{il}^*(a_i(k), c(k), d(k), q(k))\right] \\
\leq & B_1 - E\left[V(d(k+1), q(k+1))\right] + E\left[V(d(k), q(k))\right].
\end{aligned}$$

Adding the terms for $k = \{1, \ldots, K\}$ and dividing by K we get:

$$\begin{aligned}
&\frac{1}{\epsilon}E\left[\sum_{l\in\mathcal{L}} U_l(x_l^*) + w_l\mu_{il}^*\right. \\
&\left. - \sum_{l\in\mathcal{L}}\frac{1}{K}\sum_{k=1}^{K} U_l(\tilde{x}_l^*(k)) + w_l I_{il}^*(a_i(k), c(k), d(k), q(k))\right] \\
&\leq B_1 - \frac{E\left[V(d(K+1), q(K+1))\right]}{K} + \frac{E\left[V(d(1), q(1))\right]}{K} \\
&\leq B_1 + \frac{E\left[V(d(1), q(1))\right]}{K} \qquad \text{(N.1)}
\end{aligned}$$

where (N.1) follows from the fact that the Lyapunov function V is non-negative.

Assuming $E\left[V(d(1), q(1))\right] < \infty$ we get the following limit expression:

$$\limsup_{K\to\infty} E\left[\sum_{l\in\mathcal{L}} U_l(x_l^*) + w_l\mu_{il}^* - \sum_{l\in\mathcal{L}} \frac{1}{K}\sum_{k=1}^{K} U_l(\tilde{x}_l^*(k)) \right.$$
$$\left. - \sum_{l\in\mathcal{L}} \frac{1}{K}\sum_{k=1}^{K} w_l I_{il}^*(a_i(k), c(k), d(k), q(k))\right] \leq B\epsilon$$

where $B = B_1$. □

APPENDIX O

PROOF OF LEMMA 14

To prove Lemma 14, we start by first stating a fact.

Fact 3. *The following optimization problems have the same optimal value:*

$$\max_{s\in\mathcal{S}} \sum_{l\in\mathcal{L}} d_l \left[1-(1-\bar{c}_l)^{\tau_{l,s}}\right]$$

and

$$\max_{\tau,\mu,Pr(s)} \sum_{l\in\mathcal{L}} d_l \mu_l$$

subject to

$$\tau_{l,s} = \sum_{t=1}^{T} s_{l,t} \text{ for all } l\in\mathcal{L} \text{ and } s\in\mathcal{S}$$

$$\mu_l \leq \sum_{s\in\mathcal{S}} \left[1-(1-\bar{c}_l)^{\tau_{l,s}}\right] Pr(s) \text{ for all } l\in\mathcal{L}$$

$$Pr(s) \geq 0 \text{ for all } s\in\mathcal{S}$$

$$\sum_{s\in\mathcal{S}} Pr(s) \leq 1.$$

◇

The reason for the equivalence of these optimization problems follows the same logic used in Section 4.4.2 for the case $\delta = d$.

Proof of Lemma 14.

$$
\begin{aligned}
&E\left[V(d(k+1))|d(k)=d\right]-V(d)\\
=&E\left[\frac{1}{2}\sum_{l\in\mathcal{L}}\{[d_l+\tilde{a}_l(k)-I_l^*(c(k),d)]^+\}^2\right]-\sum_{l\in\mathcal{L}}\frac{d_l^2}{2}\\
\leq&E\left[\frac{1}{2}\sum_{l\in\mathcal{L}}[d_l+\tilde{a}_l(k)-I_l^*(c(k),d)]^2\right]-\sum_{l\in\mathcal{L}}\frac{d_l^2}{2}\\
=&E\left[\sum_{l\in\mathcal{L}}d_l[\tilde{a}_l(k)-I_l^*(c(k),d)]+\frac{1}{2}\sum_{l\in\mathcal{L}}[\tilde{a}_l(k)-I_l^*(c(k),d)]^2\right]\\
\leq&E\left[\sum_{l\in\mathcal{L}}d_l\left[\tilde{a}_l(k)-I_l^*(c(k),d)\right]+\frac{1}{2}\sum_{l\in\mathcal{L}}[\tilde{a}_l^2(k)+1]\right] \qquad \text{(O.1)}\\
=&\sum_{l\in\mathcal{L}}d_l(1-p_l)-\sum_{l\in\mathcal{L}}d_l\left[1-(1-\bar{c}_l)^{\tau_{l,s}^*(k)}\right]\\
&+\frac{1}{2}\sum_{l\in\mathcal{L}}\left[(1-p_l)^2+p_l(1-p_l)+1\right]\\
\leq&B_1+\sum_{l\in\mathcal{L}}d_l(1-p_l)-\sum_{l\in\mathcal{L}}d_l\mu_l^* \qquad \text{(O.2)}\\
\leq&B_1-B_2\sum_{l\in\mathcal{L}}d_l
\end{aligned}
$$

where (O.1) follows from the definition of $I_l^*(c(k),d)$, $\tau_{l,s}^*(k)$ is given by the solution to (4.12), μ^* is a solution to (4.10), (O.2) follows from Fact 3 and

$$B_1=\frac{1}{2}\sum_{l\in\mathcal{L}}(2-p_l)$$

$$B_2=\min_{l\in\mathcal{L}}\left\{\mu_l^*-(1-p_l)\right\}.$$

□

APPENDIX P

PROOF OF LEMMA 15

Assume there is some policy

$$s^*(d) \in \arg\max_{s \in \mathcal{S}(d)} \sum_{l \in \mathcal{L}} d_l \left[1 - (1 - \bar{c}_l)^{\tau_{l,s}}\right]$$

such that (4.13) is false.

Define the schedule $\tilde{s}$ such that $\tilde{s}_{l,t} = s^*_{l,t}(d)$ for $t \in \{2, \ldots, T\}$ and $l \in \mathcal{L}$ and $\tilde{s}_{l,1}$ is chosen such that (4.13) holds true. Additionally, let $\tilde{l} = \arg\max_{l \in \mathcal{L}} \tilde{s}_{l,1}$ and $l^* = \arg\max_{l \in \mathcal{L}} s^*_{l,1}(d)$. In this case we have:

$$\begin{aligned}
&\sum_{l \in \mathcal{L}} d_l \left[1 - (1 - \bar{c}_l)^{\tau_{l,\tilde{s}}}\right] - \sum_{l \in \mathcal{L}} d_l \left[1 - (1 - \bar{c}_l)^{\tau_{l,s^*}}\right] \\
&= d_{\tilde{l}} \left[1 - (1 - \bar{c}_{\tilde{l}})\right] + d_{l^*} \left[1 - (1 - \bar{c}_{l^*})^{\tau_{l^*,s^*}-1}\right] - d_{l^*} \left[1 - (1 - \bar{c}_{l^*})^{\tau_{l^*,s^*}}\right] \\
&= d_{\tilde{l}} \bar{c}_{\tilde{l}} + d_{l^*} \left[(1 - \bar{c}_{l^*})^{\tau_{l^*,s^*}} - (1 - \bar{c}_{l^*})^{\tau_{l^*,s^*}-1}\right] \\
&= d_{\tilde{l}} \bar{c}_{\tilde{l}} - d_{l^*} \bar{c}_{l^*} (1 - \bar{c}_{l^*})^{\tau_{l^*,s^*}-1} \\
&\geq d_{\tilde{l}} \bar{c}_{\tilde{l}} - d_{l^*} \bar{c}_{l^*} \\
&> 0.
\end{aligned}$$

So we have found a scheduling policy that improves on $s^*(d)$, contradicting the optimality assumption. □

APPENDIX Q

PROOF OF LEMMA 16

The proof is done by induction: we will show that at every time slot the best strategy is to schedule the link with the largest weight $d_l \bar{c}_l$ among the links that have a packet that remains to be transmitted.

From Lemma 15 we know that at time slot 1 the greedy strategy is the best strategy. We now assume that up to time slot j the best strategy is to use the greedy strategy, and prove that at time slot $j+1$ the best strategy is to use again the greedy strategy.

Denote by $\mathcal{L}_r(j+1)$ the set of links that have a packet that remains to be transmitted at time slot $j+1$. We note that the history up to time j can be completely described by d and $\mathcal{L}_r(j+1)$. Therefore, in time slot $j+1$ we need to solve the following optimization problem:

$$(\tilde{s}^*_{l,t}(d))_{l\in\mathcal{L}}^{t\in\{j+1,\ldots,T\}} \in \arg\max_{s\in\mathcal{S}(d)} \sum_{l\in\mathcal{L}_r(j+1)} d_l \left[1-(1-\bar{c}_l)^{\tau_{l,s}(j+1)}\right],$$

where

$$\tau_{l,s}(j+1) = \sum_{t=j+1}^{T} s_{l,t} \text{ for all } l \in \mathcal{L} \text{ and } s \in \mathcal{S}.$$

But this optimization is equivalent to the problem we studied in Lemma 15 for the case that we have $T-j$ time slots in a frame and $\mathcal{L} = \mathcal{L}_r(j+1)$, so the best strategy in time slot $j+1$ is to use the greedy strategy. $\square$

APPENDIX R

PROOF OF LEMMA 17

Assuming that the channel state at frame k is given by c(k), define

$$I_l^g(c(k),d) \stackrel{def}{=} \sum_{t=1}^{T} c_{l,t}(k) s_{l,t}^g(c(k),d)$$

to be the service given to link l by the greedy strategy. In this case we have

$$\begin{aligned}
&E\left[V(d(k+1))|d(k)=d\right]-V(d)\\
=&E\left[\frac{1}{2}\sum_{l\in\mathcal{L}}\{[d_l+\tilde{a}_l(k)-I_l^g(c(k),d)]^+\}^2\right]-\sum_{l\in\mathcal{L}}\frac{d_l^2}{2}\\
\leq&E\left[\frac{1}{2}\sum_{l\in\mathcal{L}}[d_l+\tilde{a}_l(k)-I_l^g(c(k),d)]^2\right]-\sum_{l\in\mathcal{L}}\frac{d_l^2}{2}\\
=&E\left[\sum_{l\in\mathcal{L}}d_l[\tilde{a}_l(k)-I_l^g(c(k),d)]+\frac{1}{2}\sum_{l\in\mathcal{L}}[\tilde{a}_l(k)-I_l^g(c(k),d)]^2\right]\\
\leq&E\left[\sum_{l\in\mathcal{L}}d_l\left[\tilde{a}_l(k)-I_l^g(c(k),d)\right]+\frac{1}{2}\sum_{l\in\mathcal{L}}[\tilde{a}_l^2(k)+1]\right] \qquad \text{(R.1)}\\
=&\sum_{l\in\mathcal{L}}d_l(1-p_l)+\frac{1}{2}\sum_{l\in\mathcal{L}}\left[(1-p_l)^2+p_l(1-p_l)+1\right]\\
&-\sum_{l\in\mathcal{L}}d_lE\left[\sum_{t=1}^{T}c_{l,t}(k)s_{l,t}^g(c(k),d)\middle|d(k)=d\right]\\
\leq&\sum_{l\in\mathcal{L}}d_l(1-p_l)-\sum_{l\in\mathcal{L}}d_l\left[1-(1-\bar{c}_l)^{\tau_{l,s}^*(k)}\right] \qquad \text{(R.2)}\\
&+\frac{1}{2}\sum_{l\in\mathcal{L}}\left[(1-p_l)^2+p_l(1-p_l)+1\right]\\
\leq&B_1-B_2\sum_{l\in\mathcal{L}}d_l \qquad \text{(R.3)}
\end{aligned}$$

where (R.1) is a consequence of the definition of $I_l^g(c(k), d)$, (R.2) follows from Lemma 16, and (R.3) comes from the proof of Lemma 14. $\square$

REFERENCES

[1] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Transactions on Automatic Control*, vol. 37, no. 12, pp. 1936–1948, Dec. 1992.

[2] I.-H. Hou, V. Borkar, and P. R. Kumar, "A theory of QoS for wireless," in *IEEE INFOCOM*, Rio de Janeiro, Brazil, Apr. 19–25, 2009, pp. 486–494.

[3] I.-H. Hou and P. R. Kumar, "Admission control and scheduling for QoS guarantees for variable-bit-rate applications on wireless channels," in *10th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, New Orleans, LA, USA, May 18–21, 2009, pp. 175–184.

[4] L. Buttyán and J.-P. Hubaux, "Enforcing service availability in mobile ad-hoc WANs," in *Proc. International Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc '00)*, Boston, MA, Aug. 2000, pp. 87–96.

[5] V. Srinivasan, P. Nuggehalli, C. F. Chiasserini, and R. R. Rao, "Energy efficiency of ad hoc wireless networks with selfish users," in *Proc. European Wireless Conference*, Florence, Italy, Feb. 2002.

[6] V. Srinivasan, P. Nuggehalli, C. F. Chiasserini, and R. R. Rao, "Cooperation in wireless ad hoc networks," in *Proc. IEEE INFOCOM '03*, vol. 2, San Francisco, CA, Mar./Apr. 2003, pp. 808–817.

[7] S. Zhong, J. Chen, and Y. R. Yang, "Sprite: A simple, cheat-proof, credit-based system for mobile ad-hoc networks," in *Proc. IEEE INFOCOM '03*, vol. 3, San Francisco, CA, Mar./Apr. 2003, pp. 1987–1997.

[8] L. Buttyán and J.-P. Hubaux, "Stimulating cooperation in self-organizing mobile ad hoc networks," *ACM/Kluwer Mobile Networks and Applications*, vol. 8, no. 5, pp. 579–592, Oct. 2003.

[9] J. Crowcroft, R. Gibbens, F. Kelly, and S. Östring, "Modelling incentives for collaboration in mobile ad hoc networks," in *Proc. WiOpt '03*, France, Mar. 2003.

[10] L. Anderegg and S. Eidenbenz, "Ad hoc-VCG: A truthful and cost-efficient routing protocol for mobile ad hoc networks with selfish agents," in *Proc. ACM Ninth Annual International Conference on Mobile Computing and Networking (MobiCom '03)*, San Diego, CA, Sep. 2003, pp. 245–259.

[11] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," in *Proc. Sixth Annual International Conference on Mobile Computing and Networking (MobiCom '00)*, Boston, MA, Aug. 2000, pp. 255–265.

[12] P. Michiardi and R. Molva, "CORE: A collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks," in *Proc. Communications and Multimedia Security Conference (CMS '02)*, Portoroz, Slovenia, Sep. 2002.

[13] S. Buchegger and J.-Y. Le Boudec, "Performance analysis of the CONFIDANT protocol (cooperation of nodes: Fairness in dynamic ad-hoc networks)," in *Proc. International Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc '02)*, Lausanne, Switzerland, Jun. 2002, pp. 226–236.

[14] S. Bansal and M. Baker, "Observation-based cooperation enforcement in ad hoc networks," arXiv, Tech. Rep. cs/0307012, 2003. [Online]. Available: http://arxiv.org/abs/cs/0307012

[15] Q. He, D. Wu, and P. Khosla, "SORI: A secure and objective reputation-based incentive scheme for ad-hoc networks," in *Proc. IEEE Wireless Communications and Networking Conference (WCNC '04)*, vol. 2, Atlanta, GA, Mar. 2004, pp. 825–830.

[16] R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan, "Sustaining cooperation in multi-hop wireless networks," in *Proc. Second Symposium on Networked Systems Design and Implementation (NSDI '05)*, Boston, MA, May 2005, pp. 231–244.

[17] P. Michiardi and R. Molva, "Analysis of coalition formation and cooperation strategies in mobile ad hoc networks," *Ad Hoc Networks*, vol. 3, no. 2, pp. 193–219, Mar. 2005.

[18] M. T. Refaei, V. Srivastava, L. DaSilva, and M. Eltoweissy, "A reputation-based mechanism for isolating selfish nodes in ad hoc networks," in *Proc. IEEE Second Annual International Conference on Mo-*

bile and Ubiquitous Systems: Networking and Services (MobiQuitous '05), San Diego, CA, Jul. 2005, pp. 3–11.

[19] T. Anantvalee and J. Wu, "Reputation-based system for encouraging the cooperation of nodes in mobile ad hoc networks," in *Proc. IEEE International Conference on Communications (ICC)*, Glasgow, Scotland, Jun. 24-28, 2007, pp. 3383–3388.

[20] D. Fudenberg and J. Tirole, *Game Theory*. Cambridge, MA: The MIT Press, Aug. 1991.

[21] R. Axelrod, "The emergence of cooperation among egoists," *The American Political Science Review*, vol. 75, no. 2, pp. 306–318, Jun. 1981.

[22] F. Milan, J. J. Jaramillo, and R. Srikant, "Achieving cooperation in multihop wireless networks of selfish nodes," in *Workshop on Game Theory for Networks (GameNets 2006)*, Pisa, Italy, Oct. 14, 2006.

[23] J. Wu and R. Axelrod, "How to cope with noise in the iterated prisoner's dilemma," *The Journal of Conflict Resolution*, vol. 39, no. 1, pp. 183–189, Mar. 1995.

[24] R. Sugden, *The Economics of Rights, Cooperation and Welfare*. Blackwell Publishing, 1986.

[25] R. Boyd, "Mistakes allow evolutionary stability in the repeated prisoner's dilemma game," *Journal of Theoretical Biology*, vol. 136, no. 1, pp. 47–56, Jan. 1989.

[26] M. Abadi, M. Burrows, M. Manasse, and T. Wobber, "Moderately hard, memory-bound functions," *ACM Transactions on Internet Technology*, vol. 5, no. 2, pp. 299–327, May 2005.

[27] C. Dwork, A. Goldberg, and M. Naor, "On memory-bound functions for fighting spam," in *Proc. 23rd Annual International Cryptology Conference (CRYPTO '03)*, Santa Barbara, CA, Aug. 2003.

[28] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. IT-22, no. 6, pp. 644–654, Nov. 1976.

[29] R. Anderson and M. Kuhn, "Tamper resistance - a cautionary note," in *Proc. Second USENIX Workshop on Electronic Commerce*, Oakland, CA, Nov. 1996, pp. 1–11.

[30] K. Sanzgiri, I. D. Chakeres, and E. M. Belding-Royer, "Pre-reply probe and route request tail: Approaches for calculation of intra-flow contention in multihop wireless networks," *Mobile Networks and Applications*, vol. 11, no. 1, pp. 21–35, Feb. 2006.

[31] I. D. Chakeres, E. M. Belding-Royer, and J. P. Macker, "Perceptive admission control for wireless network quality of service," *Ad Hoc Networks*, vol. 5, no. 7, pp. 1129–1148, Sep. 2007.

[32] J. Tang, G. Xue, and W. Zhang, "Interference-aware topology control and QoS routing in multi-channel wireless mesh networks," in *Proc. 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, Urbana-Champaign, IL, USA, May 25–27, 2005, pp. 68–77.

[33] B. Awerbuch, Y. Azar, and S. Plotkin, "Throughput-competitive on-line routing," in *Proc. 34th Annual Symposium on Foundations of Computer Science*, Palo Alto, CA, USA, Nov. 3–5, 1993, pp. 32–40.

[34] B. Awerbuch, Y. Azar, S. Plotkin, and O. Waarts, "Competitive routing of virtual circuits with unknown duration," in *Proc. 5h Annual ACM-SIAM Symposium on Discrete Algorithms*, Arlington, VA, USA, Jan. 23–25, 1994, pp. 321–327.

[35] S. Plotkin, "Competitive routing of virtual circuits in ATM networks," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 6, pp. 1128 – 1136, Aug. 1995.

[36] D. D. Sleator and R. E. Tarjan, "Amortized efficiency of list update and paging rules," *Communications of the ACM*, vol. 28, no. 2, pp. 202–208, Feb. 1985.

[37] A. R. Karlin, M. S. Manasse, L. Rudolph, and D. D. Sleator, "Competitive snoopy caching," *Algorithmica*, vol. 3, no. 1, pp. 79–119, Mar. 1988.

[38] M. Manasse, L. McGeoch, and D. Sleator, "Competitive algorithms for on-line problems," in *Proc. 20th Annual ACM Symposium on Theory of Computing*, Chicago, IL, USA, 1988, pp. 322–333.

[39] A. Borodin, N. Linial, and M. E. Saks, "An optimal on-line algorithm for metrical task system," *Journal of the ACM*, vol. 39, no. 4, pp. 745–763, Oct. 1992.

[40] Y. Yang and R. Kravets, "Contention-aware admission control for ad hoc networks," *IEEE Transactions on Mobile Computing*, vol. 4, no. 4, pp. 363–377, Apr./May 2005.

[41] S.-B. Lee, G.-S. Ahn, X. Zhang, and A. T. Campbell, "INSIGNIA: An IP-based quality of service framework for mobile ad hoc networks," *Journal of Parallel and Distributed Computing*, vol. 60, no. 4, pp. 374–406, Apr. 2000.

[42] G.-S. Ahn, A. T. Campbell, A. Veres, and L.-H. Sun, "SWAN: Service differentiation in stateless wireless ad hoc networks," in *Proc. IEEE INFOCOM*, vol. 2, New York, NY, USA, Jun. 23–27, 2002, pp. 457–466.

[43] Q. Xue and A. Ganz, "Ad hoc QoS on-demand routing (AQOR) in mobile ad hoc networks," *Journal of Parallel and Distributed Computing*, vol. 63, no. 2, pp. 154–165, Feb. 2003.

[44] H. Luo, S. Lu, V. Bharghavan, J. Cheng, and G. Zhong, "A packet scheduling approach to QoS support in multihop wireless networks," *Mobile Networks and Applications (MONET)*, vol. 9, no. 3, pp. 193–206, Jun. 2004.

[45] R. Ramanathan and M. Steenstrup, "Hierarchically-organized, multihop mobile wireless networks for quality-of-service support," *Mobile Networks and Applications (MONET)*, vol. 3, no. 1, pp. 101–119, Jun. 1998.

[46] S. H. Shah and K. Nahrstedt, "Guaranteeing throughput for real-time traffic in multi-hop IEEE 802.11 wireless networks," in *Proc. Military Communication Conference (Milcom)*, Atlantic City, NJ, USA, Oct. 17–20, 2005.

[47] W.-H. Liao, Y.-C. Tseng, and K.-P. Shih, "A TDMA-based bandwidth reservation protocol for QoS routing in a wireless mobile ad hoc network," in *Proc. IEEE International Conference on Communications (ICC)*, vol. 5, New York, NY, USA, Apr. 28/May 2, 2002, pp. 3186–3190.

[48] C. Zhu and M. S. Corson, "QoS routing for mobile ad hoc networks," in *Proc. IEEE INFOCOM*, vol. 2, New York, NY, USA, Jun. 23–27, 2002, pp. 958–967.

[49] S. Guo and O. Yang, "QoS-aware minimum energy multicast tree construction in wireless ad hoc networks," *Ad Hoc Networks*, vol. 2, no. 3, pp. 217–229, Jul. 2004.

[50] C. R. Lin and J.-S. Liu, "QoS routing in ad hoc wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, pp. 1426–1438, Aug. 1999.

[51] H. Liu, X. Jia, D. Li, and C. Lee, "Bandwidth guaranteed call admission in TDMA/CDMA ad hoc wireless networks," *Ad Hoc Networks*, vol. 3, no. 6, pp. 689–701, Nov. 2005.

[52] S. Singh, P. A. K. Acharya, U. Madhow, and E. M. Belding-Royer, "Sticky CSMA/CA: Implicit synchronization and real-time QoS in mesh networks," *Ad Hoc Networks*, vol. 5, no. 6, pp. 744–768, Aug. 2007.

[53] A. Eryilmaz and R. Srikant, "Fair resource allocation in wireless networks using queue-length-based scheduling and congestion control," in *IEEE INFOCOM*, vol. 3, Miami, FL, USA, Mar. 13–17, 2005, pp. 1794–1803.

[54] X. Lin and N. B. Shroff, "Joint rate control and scheduling in multihop wireless networks," in *43rd IEEE Conference on Decision and Control (CDC)*, vol. 2, Atlantis, Paradise Island, Bahamas, Dec. 14–17, 2004, pp. 1484–1489.

[55] M. J. Neely, E. Modiano, and C.-P. Li, "Fairness and optimal stochastic control for heterogeneous networks," in *IEEE INFOCOM*, vol. 3, Miami, FL, USA, Mar. 13–17, 2005, pp. 1723–1734.

[56] A. Stolyar, "Maximizing queueing network utility subject to stability: Greedy primal-dual algorithm," *Queueing Systems*, vol. 50, no. 4, pp. 401–457, Aug. 2005.

[57] A. Eryilmaz and R. Srikant, "Joint congestion control, routing and mac for stability and fairness in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1514–1524, Aug. 2006.

[58] L. Chen, S. H. Low, M. Chiang, and J. C. Doyle, "Cross-layer congestion control, routing and scheduling design in ad hoc wireless networks," in *IEEE INFOCOM*, Barcelona, Catalunya, Spain, Apr. 23–29, 2006.

[59] X. Lin, N. B. Shroff, and R. Srikant, "A tutorial on cross-layer optimization in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1452– 1463, Jun. 2006.

[60] S. Shakkottai and R. Srikant, "Scheduling real-time traffic with deadlines over a wireless channel," *Wireless Networks*, vol. 8, no. 1, pp. 13–26, Jan. 2002.

[61] V. Raghunathan, V. Borkar, M. Cao, and P. R. Kumar, "Index policies for real-time multicast scheduling for wireless broadcast systems," in *IEEE INFOCOM*, Phoenix, AZ, USA, Apr. 13–18, 2008, pp. 1570–1578.

[62] A. Dua and N. Bambos, "Downlink wireless packet scheduling with deadlines," *IEEE Transactions on Mobile Computing*, vol. 6, no. 12, pp. 1410–1425, Dec. 2007.

[63] Q. Liu, X. Wang, and G. B. Giannakis, "A cross-layer scheduling algorithm with QoS support in wireless networks," *IEEE Transactions on Vehicular Technology*, vol. 55, no. 3, pp. 839–847, May 2006.

[64] I.-H. Hou and P. R. Kumar, "Scheduling heterogeneous real-time traffic over fading wireless channels," arXiv, Tech. Rep. 0908.0587, 2009. [Online]. Available: http://arxiv.org/abs/0908.0587

[65] D. G. Luenberger, *Linear and Nonlinear Programming*, 2nd ed. Norwell, MA: Kluwer Academic Publishers, 2003.

[66] S. Boyd and L. Vandenberghe, *Convex Optimization*, 1st ed. New York, NY: Cambridge University Press, Mar. 2004.

[67] S. Sundaram and C. N. Hadjicostis, "Distributed function calculation via linear iterations in the presence of malicious agents - Part I: Attacking the network," in *Proc. 27th American Control Conference (ACC)*, Seattle, WA, Jun. 11-13, 2008, pp. 1350–1355.

[68] S. Sundaram and C. N. Hadjicostis, "Distributed function calculation via linear iterations in the presence of malicious agents - Part II: Overcoming malicious behavior," in *Proc. 27th American Control Conference (ACC)*, Seattle, WA, Jun. 11-13, 2008, pp. 1356–1361.

AUTHOR'S BIOGRAPHY

Juan Jose Jaramillo Jimenez received his B.S. degree (summa cum laude) from Universidad Pontificia Bolivariana, Colombia, in 1998, his M.S. and Ph.D. degrees from the University of Illinois at Urbana-Champaign in 2005 and 2010, respectively, all in electrical engineering. From 1999 to 2003 he worked at Empresas Publicas de Medellin in Colombia. He is the recipient of a Fulbright fellowship. His research interests include communication networks and game theory.