RICE UNIVERSITY

Efficient Resource Allocation in Multiflow Wireless Networks

by

Gareth B. Middleton

A THESIS SUBMITTED
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE

Doctor of Philosophy

APPROVED, THESIS COMMITTEE:

[illegible]

Dr. Behnaam Aazhang, *Chair*
J.S. Abercrombie Professor of Electrical and Computer Engineering

[illegible]

Dr. Edward W. Knightly
Professor of Electrical and Computer Engineering

[illegible]

Dr. Ashutosh Sabharwal
Associate Professor of Electrical and Computer Engineering

[illegible]

Dr. Yin Zhang
Professor of Computational and Applied Mathematics

[illegible]

Dr. Jorma Lilleberg
Adjunct Professor of Electrical and Computer Engineering

HOUSTON, TEXAS
MAY 2011

ABSTRACT

Efficient Resource Allocation in Multiflow Wireless Networks

by

Gareth B. Middleton

We consider the problem of allocating resources in large wireless networks in which multiple information flows must be accommodated. In particular, we seek a method for selecting schedules, routes, and power allocations for networks with terminals capable of user-cooperation at the signal level. To that end, we adopt a general information-theoretic communications model, in which the datarate of a wireless link is purely a function of transmission power, pathloss and interference.

We begin by studying the case of resource allocation when only point-to-point links are available. The problem is NP-hard in this case, requiring an exponentially-complex exhaustive search to guarantee an optimal solution. This is prohibitively difficult for anything but the smallest of networks, leading us to approximate the problem using a decomposition approach. We construct the solution iteratively, developing polynomial-time algorithms to optimally allocate resources on a per-frame basis. We then update the network graph to reflect the resources consumed by the allocated frame. To manage this decomposition, we present a novel tool, termed the Network-Flow Interaction Chart. By representing the network in both space and time, our techniques trade off interference with throughput for each frame, offering considerable performance gains over schemes of similar complexity. Recognizing that our approach requires a large amount of overhead, we go on to develop a method in which it may be decentralized. We find that while the overhead is considerably lower, the limited solution space results in suboptimal solutions in a throughput sense.

We conclude with a generalization of the Network-Flow Interaction Chart to address cooperative resource allocation. We represent cooperative links using "metanodes," which are made available to the allocation algorithms alongside point-to-point links and will be selected only if they offer higher throughput. The data-carrying capability of the cooperative links is modeled using Decode-and-Forward achievable rates, which are functions of transmit power and interference, and so may be incorporated directly into our framework. We demonstrate that allocations incorporating cooperation results in significant performance gains as compared to using point-to-point links alone.

Acknowledgements

In large measure, my graduate career at Rice was the result of the opportunity offered to me by the Center for Multimedia Communication, in particular by its directors Behnaam Aazhang, Ashu Sabharwal, and Joe Cavallaro. My early experience with wireless communication in their lab convinced me of the relevance and promise of this exciting field. Their steady encouragement and honest criticisms over the years have forced me beyond my comfort zones as both a researcher and an individual, and I will take with me many lessons from my years as their student. From my thesis committee, I greatly appreciate the help of Nokia (now Renesas) research scientist Jorma Lilleberg, who has offered an industry perspective throughout the years I've studied this problem (along with two great summers in Oulu!). Professors Edward Knightly and Yin Zhang fine-tuned this work, keeping me honest in their respective fields with careful critique and advice.

As a hopeful teacher-to-be, I would like to thank those who have most inspired me during the ten years I've spent at Rice: Jim Brown of Economics and Don Johnson of ECE are without question two of the best lecturers I've ever had the privilege of learning from. I'd like to thank the 2008 class of ELEC 241, the first group of students I've ever taught, for their encouragement and feedback during my earliest efforts at the blackboard. I'd also like to take a moment to recognize the professor who first introduced me to real research, Tissa Illangasekare of the Environmental Science Department at the Colorado School of Mines. His gentle nature and quiet confidence in the face of significant challenges is unlike any other I've seen.

I owe a great deal to the good friends and colleagues at Rice whom I've had the privilege and pleasure of knowing over the years. In particular, thanks to Matt Nokleby, Chris Steger, Chris Hunter and David Kao for their friendship, perspectives and suggestions. In matters large and small, professional and personal, I never had any doubt that we would figure things out over a beer and a laugh at "The Perch" above Valhalla. To my friends "beyond-the-hedges" Kevin & Laura Grahmann and Chris Ponder (and Stuart!), thanks for the glimpse into–and view from–the real world. Those many evenings at the Porch-on-Byrne are among my best memories from this time.

The unwavering support of my parents and brother from the very first to the very last is the integral element in this achievement: for 27 years, I have been blessed with the most loving family anyone could ever ask for. Their unconditional encouragement and advice has meant more to me than I could ever express, and so as a token of my gratitude I dedicate this work to the Middletons: Nigel, Glyn and Chris.

Contents

List of Figures

List of Tables

List of Symbols

α	Pathloss parameter
$\bar{R}$	Minimum neighborhood rate
$\cdot_B$	Subscript indicating cooperative broadcast
$\cdot_M$	Subscript indicating cooperative multiple-access
η^f	Throughput of flow f
$\gamma_t(y)$	Interference at terminal y in timeslot t
λ	Intensity of Poisson point process
$\mathbb{I}$	Space of binary values
$\mathbf{p}$	Vector of transmission powers
$\mathcal{C}_i$	Set of terminals in cluster i
$\mathcal{E}$	Set of NFIC edges
$\mathcal{F}$	Set of source-destination flows
$\mathcal{L}$	Set of all routes between the same source and destination
$\mathcal{N}$	Set of network terminals
$\mathcal{N}_x$	Set of neighbors of terminal x
$\mathcal{P}$	Set of feasible transmission powers

$\mathcal{S}$	Set of feasible schedules
$\mathcal{V}$	Set of NFIC nodes
$\mathrm{I}_\oplus$	Semiring summary identity element
$\mathrm{I}_\otimes$	Semiring extension annihilator element
$\oplus$	Semiring summary operator
$\otimes$	Semiring extension operator
$\widehat{R}_t^k$	Rate of frame affected by interference from future frames
A	Side length of network region
$A(\cdot)$	Resource allocation function
a_k	Arrival slot of frame k
B_t^k	Binary variable indicating cooperative broadcast of frame k at time t
C	Number of clusters in the network
$d(k)$	Destination terminal of frame k
$D_{\bar{R}}$	Radius of rate-based neighborhood
$D_{x,y}$	Euclidean distance between terminals x and y
E^{C}	NFIC Metanode edge mapping
E^{NC}	Non-cooperative edge mapping
$e^{x,y}$	NFIC edge between nodes x and y at time t
F	Number of source-destination flows
f	Index of flows
$g_{i,j}$	Gateway terminal from cluster i to cluster j
$h^{\cdot,\cdot}_{\{B,M\}}$	Consolidated channel variable
$h_t(y)$	NFIC history variable for node y at time t

$I_{\cdot}^{\cdot}(\cdot,\cdot)$	Binary link activation variable
K	Total number of frames to be transmitted
k	Index of frames
L^k	Route of frame k
L_t^k	Terminal transmitting frame k at time t
M_t^k	Binary variable indicating cooperative multiple-access of frame k at time t
N	Number of terminals in a network realization
N_o	Thermal noise
P_{Tx}	Maximum terminal transmission power
$P_t(x)$	Transmission power of terminal x at time t
R^k	Final rate of frame k
R_{Int}	Fraction of rate loss allowed due to interference
R_t^k	Rate of frame k at time t
S	Schedule
$s(k)$	Source terminal of frame k
T	Number of total timeslots
t	Timeslot index variable
$t_i^a(k)$	Timeslot frame k enters cluster i
$t_i^d(k)$	Timeslot frame k departs cluster i
T_k	Arrival time of k^{th} frame
$U(\cdot)$	General utility function
V^{C}	NFIC Metanode node mapping
V^{NC}	Non-cooperative terminal mapping

CHAPTER 1

Introduction

As the nature of point-to-point wireless links has become more comprehensively characterized, the wireless research community has broadened the focus of its study to include more than two terminals. This has been motivated largely by the rise of modern cellular networks and the struggle to deliver data to an ever-more thirsty subscriber pool. The traditional cellular architecture–widely in use today for both telephony and wireless computer data–is nothing more than a collection of point-to-point links. The channels between all of the users and a common access point are nearly perfectly orthogonalized in some manner: either through time-division (as in the case of GSM cellular or 802.11), or through code division (IS-95, 3GPP, WCDMA).

As a consequence of this orthogonalization, the delivery of higher throughput to users has been accomplished to date in a brute-force manner: the deployment of more access points. Each one offers more timeslots or codes to serve the increasing user demand. However, even this has its limits: dense urban environments like New York pose geographical challenges to system designers, leading to situations in which the user base cannot be satisfied despite massive capital investment [95].

An alternative approach recognizes that at any given time, it is unlikely that *all*

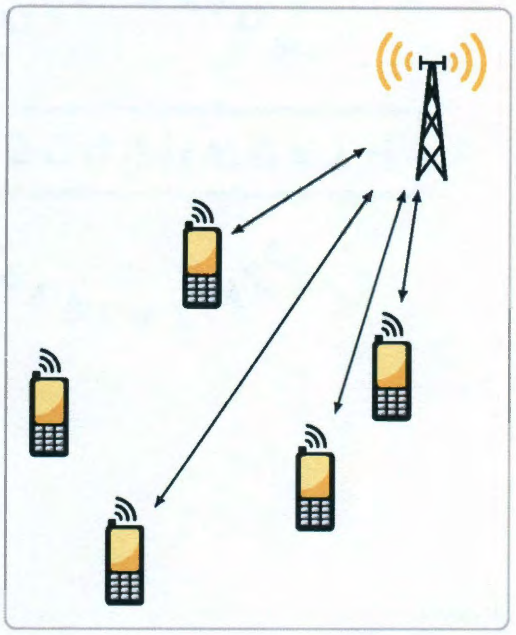


Figure 1.1: A cellular network in which all links are between a central base station and individual mobile terminals.

users in a network are actively transmitting data. In particular, there often exist many "idle" devices: those which are powered, registered with the network, but not acting as a data sink or source. A system architecture may be developed around this notion, in which there are relatively few access points; rather, idle terminals may *forward* information between end terminals and the backhauled access points.

The orthogonalization in the traditional cellular architecture of Figure 1.1 results fundamentally in a series of point-to-point links. However, relaxing the orthogonal, centralized network constraints leads to the *ad-hoc* architecture suggested above and visualized in Figure 1.2. Here, we are dealing with a collection of interfering multi-hop links, across which many different streams of data must flow. The orthogonal point-to-point case has been widely studied in the literature, across all levels from basic signal design to abstract information-theoretic capacity. The interfering multi-terminal network, on the other hand, has proven to be much more difficult to analyze.

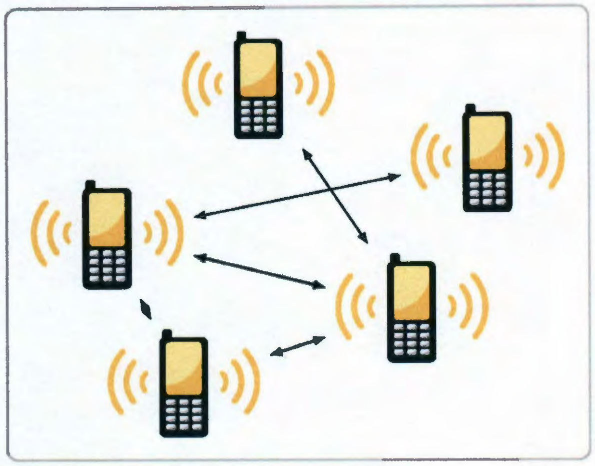


Figure 1.2: An ad-hoc network in which links exist between all devices in the network, not just between all devices and a common base station.

At its most basic, this is the consequence of the *interfering* nature of the network. Simultaneous transmissions sharing a common bandwidth affect each other in unpredictable and mathematically troublesome ways. The overall performance is a combination of the channels between the terminals, the signalling scheme used to transmit the information, and the power used at each transmitter. The difficulty in characterizing these effects and interactions increases exponentially with the size of the network, resulting in an *NP-Hard* optimization program.

Some multiterminal architectures have admitted analysis; perhaps the most famous is the *relay* channel, in which two end-point terminals communicate with the help of a third relay terminal. However, this architecture has been studied primarily in isolation. That is, the additional interference caused by the relay terminal has not been studied as to its effects on other transmissions in a larger network. This is partly due to the interfering nature of large networks in general, but more a result of the specific constraints the relay architecture imposes. In particular, the transmissions of the source and relay are coupled both temporally and spatially; relay gains are the consequence of this coordinated action of two terminals.

It is in this type of network that we propose to perform resource allocation: all terminals share a common bandwidth to transmit multiple information flows, and may potentially cooperate with each other on the signal level as described. We set the goal of determining, on a *fine-grained timescale*, the behavior of the network such that the throughput of the data flows involved is optimized. Put concisely, we seek to *jointly* determine *who* is transmitting (scheduling), *where* those transmissions are bound (routing), at what power level those transmissions are happening (power control), and whether or not signal-level cooperation should be employed.

We make several contributions towards answering these questions in both the modeling and algorithmic domains, which this document will describe in detail. Throughout this thesis, we will consider the most general case of a large ad-hoc wireless network: one in which there are several sources of information bound for different destinations, and in which *all* transmissions interfere.

Graphical models have long been used to describe terminals and interconnections, with origins in the wired networks deployed in the middle part of the last century. In these models–commonly referred to as *network graphs*, vertices are used to represent the terminals themselves, and edges represent the connections between them. Problems in these types of networks are typically of a routing and congestion-control nature, i.e. what is the best path to move data through the network, and how can the load in main links (trunk lines) be managed so that all flows get some sense of a fair share.

In the wireless environment, the network graph becomes much more complex, since devices are "connected" to all terminals in the near vicinity, whether they mean to be or not. The notion of "connection" depends on the physical layer model in use, which describes the radio technology used to transmit information between devices over the air. Network graphs for wireless systems therefore require a variety of supporting assumptions in contrast to wired networks, in which there either is or is not a connection between terminals (so that an edge exists or not in the network graph). Wireless networks, due to the nature of the medium, may have connections between terminals but representing differing data rates, which must somehow be modeled in the graph. Further, the state of connections between terminals may be evolving with time, meaning that the network graph can be constantly changing.

The structure of the network graph–its size, directionality, connectivity, etc–can be viewed as the fundamental premises from which a research study begins. There are many different approaches taken in the literature, from the information-theoretic asymptotic analysis of infinitely large networks to the protocol development for small random-access networks. To place our work in context, we now discuss the state of art in this area.

1.1 State of the Art

Resource allocation in wireless networks has been studied from two main viewpoints: large-scale information theoretic analyses and medium-access control studies both lend very different flavors to the same problem, giving very different results. The order analysis of large networks, pioneered by Gupta and Kumar [32], opened the door to a variety of information-theoretic network studies. Defining the *transport capacity* of the network is the product of all bits and the distances they have traveled, this paper relates the asymptotic size of the network to the overall data-carrying capacity. This work and its extension [103] assume simplistic transmission schemes, precluding multihop transmissions and cooperative technologies, resulting in an interference-limited regime of operation and a consequent degrading of transport capacity as the network grows large. More recently, [64] relaxed some of these constraints, permitting cooperative technologies and interference-cancellation techniques. However, their order analysis still shows achievable rates tending to zero as the network becomes large. With a view to studying achievable transport capacities for networks with fixed size, the upper bound on transport capacity been established for the multihop case with interference [47], and a hierarchical scheme for achieving it has been developed [84]. This line of study has also been extended to fading channels [104], where it is shown that in fast-fading environments, multi-hop techniques can allow linear growth in transport capacity even without full channel-state information (CSI). The study of networks without full CSI has been continued through the application of percolation theory [24] and physics-inspired channel models, both of which can determine the limits of network capacity [25].

Information theoretic analysis is useful for bounding the performance of large networks, but it fails to address the practical problems of scheduling and routing, which must be resolved before a system can be implemented. This requires limiting the network size, allowing the terminals and their connections to be represented as a directed graph. While routing data through a directed graph may be accomplished in $O(N^3)$ time [17, 6], computing optimal schedules is known to be NP-Hard [13, 77]. As such, throughput bounds for non-asymptotic networks are considerably different from those found under order analysis, as demonstrated by the approximation methods used in [10].

1.1.1 Network Connectivity Models

In the formulation of the graph-based network, a critical choice is the interference model applied to the transmissions; in this vein there are two commonly-used models: the *protocol* model, in which links are said to fully interfere if the receiver is within some distance of another transmitter, and the *physical* model, in which a link exists if the signal-to-interference and noise ratio (SINR) at the receiver remains above some threshold. As a result of the narrowing of network parameters and the introduction of a more stringent interference model than in the order-analysis, the metric of network performance must change as well. Instead of studying the transport capacity, limited-size network analyses generally consider the *rate region* of a network, which is the set of vectors of link rates which are stable and achievable for a given traffic demand under some scheduling algorithm [54]. Of note here is that a common assumption across interference models is that of constant packet sizes, such that if the link exists, a certain amount of data is successfully transmitted probability tending to one.

Both the protocol model and the physical model result in directed-graph networks, which can be mapped to a conflict graph [43], from which scheduling becomes a coloring problem. The distinction between the two lies in whether the algorithm is required to be iterative, as the conflict graph updates as different links are activated under the physical model. In this paradigm, the challenge is to identify matchings in the graph corresponding to feasible schedules. The first polynomial-time link scheduling algorithm appeared in [36], though distributed algorithms recently have appeared; [97] employs a token-passing technique, and [76] schedules a directed-graph network with guaranteed maximum throughput using gossip algorithms. Applying the protocol model to random-access networks, [46] proposes a distributed algorithm guaranteed to deliver a bounded fraction of a throughput vector on the convex hull of the rate region for arbitrary networks. Incorporating carrier-sensing multiple access (CSMA) allows graph-theoretic matching techniques to be applied in a distributed sense, as shown in [90]. The flexibility of the protocol model also allows link pricing techniques to be used to generate schedules, as recently done in [105]. These results generally assume that the network topology must be known to a centralized scheduler, which may be impractical in rapidly evolving networks. Recently, [108] has relaxed this assumption, presenting a randomized, distributed algorithm which computes schedules capable of delivering rate vectors at any point on a lattice in the feasible rate region.

The scheduling problem alone assumes that traffic demands and terminal resources are fixed. By contrast, there is the "cross layer" problem [67, 29] of choosing schedules, routes and power allocations together. This optimization was first studied by Cruz [15], who solved it for the protocol model using a Lagrangian dual decomposition. The same approach extends to the SINR model [22] and also to other objective functions, including power [7] and congestion [11].

Lagrangian-dual algorithms suffer from worst-case exponential complexity in the number of terminals, which led to efforts to approach the joint-resource allocation problem using multicommodity flow techniques. This has been done for the general large network [71], and also for the specific problem of scheduling and routing in limited-size networks [102]. These techniques require relaxation of the integer constraints on routing and scheduling, which must be re-enforced through a randomized rounding approach [89].

Our problem is one of cross-layer design, in which physical layer parameters interact with network layer decisions. Jointly allocating resources at both layers was investigated first by Cruz et. al. [15] in the point-to-point context, where a linear program approach is used to solve the scheduling and power allocation problems, the solutions to which are fed into a routing algorithm. This line of inquiry continues in other work, namely in [54] and [22], and most recently in [56] and [66]. There are two common themes in these works: although each presents a method to solve the joint allocation problem, all of the solution techniques assume point-to-point links. Further, each assumes a *fixed* packet size, usually by way of enforcing a minimum SINR constraint. This precludes fine-grained physical layer rate control, which is a key element in making cooperation possible in large networks.

1.1.2 Allocation with Cooperation

Although our approach is general to any characterizable multiterminal scheme, we will restrict our attention in this paper to the *relay channel* [98, 14], shown in Figure 6.1. Throughout, we will describe the relay system in terms of its three terminals and its two phases. The three terminals are uniquely the source, relay, and destination–we note the uniqueness because the data capacity of the cooperative link will change if the terminals change roles. The first of the two phases is the *broadcast* (BC) phase, in which the source alone transmits, but both the relay and destination receive. Note here that two atomic links are active, one between the source and relay, and another between the source and destination. The second phase is the *multiple access* (MA) phase, in which the source and relay transmit simultaneously to the destination. The destination, because it receives three packets over two timeslots, is then able to receive more information from the source than it would have if the source alone had transmitted. We will view this coordination action as a single physical layer unit, which can be used by the MAC layer to aid data transfer in the larger network.

The study of joint resource allocation for cooperative links has become an area of research relatively recently. Routing for cooperation has been studied in [60], where cooperative nodes were used to improve transmission reliability rather than datarates. Cooperative routing (without the scheduling component) has been studied in [110, 111] and [50] through the notion of a "virtual link," and in [107] in the form of a layered network. From the medium-access layer perspective, [4] presents routing protocols based on location estimation. [18] presents solutions to the routing and power allocation problem in networks with cooperative links, but ignores the temporal scheduling aspect. Game theoretic tools are introduced in [37] to aid in selection of cooperative terminals for routing.

While the study of cooperation in this thesis is with a view to throughput improvements, it has also been used to reduce energy consumption in the network, since cooperative diversity has been shown to overcome the fading environment. In this area, several works address cooperative routing for energy efficiency [50, 110, 41], but do not optimize for throughput. Recent work in [16] does address throughput optimization, modeling cooperative links as being capable of multipoint beamforming in order to increase the overall throughput of a route. In this work, the topologies are of a regular nature and scheduling is not considered; this thesis aims to fill that gap, presenting a technique for allocation cooperative resources in polynomial time regardless of the network topology and traffic demands.

In all of these cases, the allocation algorithms have had high complexity, limiting their usefulness in fast-changing networks. This thesis will address all three issues–allocating space, time, and power–in one optimization formulation, solved in cubic time.

1.2 Problem Statement and Contributions

As we have seen, there has been a considerable amount of research into resource allocations in large networks, much of it in the wireless networking paradigm. However, as we have also seen, optimality guarantees have traditionally been available only in asymptotic cases, or with high complexity. To our knowledge, efficient techniques for determining the behavior of the *entire* network at all time instants are not known.

We seek to address this issue in the context of large networks with many multihop, cooperative information flows. Generally, the question we wish to answer is the following:

Question: Given a large half-duplex wireless network with varying connectivity between terminals and a number of unique independent information flows, what is the optimal scheduling and routing of packets, what is the optimal power allocation, and when should signal-level cooperation be incorporated?

This question requires considerable refinement before we can begin to answer it. First, we must formulate a network model founded upon the fundamentals of wireless communication technology. From there, we must study ways to address the complexity of the problem, in a theoretical as well as algorithmic sense. We may then verify our solutions via simulation. As such, our contributions fall into three broad areas:

1. **Problem Decomposition**: We present a method of reformulating the original resource allocation problem as a series of smaller, easier problems.

2. **Data Structure**: Due to the nature of our system model, in which we study the evolution of network conditions as a function of time, we must find a temporally-aware representation of the network. Our second contribution is such a data structure, constructed from the network topology but also a function of time. It will be shown that this data structure is naturally suited to representing cooperative communication alongside traditional point-to-point links.

3. **Algorithms**: The sequence of subproblems we develop above may be solved algorithmically, but such algorithms must be aware of the context of the subproblem, i.e. the initial conditions in each subproblem are consequences of the solutions to the previous subproblem. Our third contribution is a set of context-aware algorithms, which solve the subproblem sequence in cubic time.

Our first contribution is the problem decomposition. As we shall see, the joint problem we define is exponentially complex and non-convex, resulting in prohibitive complexity. We will leverage the specifics of our problem–in particular the frame-based nature of wireless communications–to rewrite this problem as a sequence of optimizations for each data frame, which we can then solve optimally in cubic time.

Our decomposition trades off solution-complexity with data-management complexity. Each subproblem must be initialized, and the solution to that subproblem becomes a part of the initial conditions for the next one. As such, we require a method in which to manage this data. This is our second contribution, a representation of the wireless network in graphical form. Our graphical model, termed the *Network Flow Interaction Chart* (NFIC), is designed around the notion of discretization in both spatial and temporal dimensions. We must impose a time-slotted system constraint in order for this to hold, but under this general assumption, we are able to model both the changing nature of the network as well as the differing quality of wireless connections between nodes. We will show that the NFIC is also able to represent *groups* of terminals acting together in a *single* transmission, as in the case of the physical-layer relay network discussed earlier. In this way, our graphical model is able to capture key features of the wireless landscape which then serve as inputs to our optimization algorithms.

Our third contribution is an algorithmic structure which, under only mild assumptions on the operating characteristics and optimality criteria in the network, delivers solutions in cubic time. Our central algorithm relies on the decomposition technique, in which individual pieces of data in the network are handled separately. The solutions to these problems are cataloged on the NFIC for use as inputs to the subsequent problems in the decomposition.

In constructing these results, we will need to major assumption that we have all of the network state information: that is, we have instantaneous knowledge of all N^2 channels between the N terminals, for T timeslots. While we employ this assumption to help us gain insight into the approaches most suited to this problem, we immediately recognize that it is unrealistic in all but the smallest networks. As such, we explore methods for *decentralization*, such that our techniques require less information about the network to be effective. This is our fourth contribution: a methodology by which our original, full-information technique can be deployed across the large network without requiring full coordination. With nothing more than a sense of *locality*, which we will define as terminals of such a proximity to support some target transmission rate $\bar{R}$, we will present techniques for fine-grained resource allocation.

Our final contribution is, in our view, the most meaningful and in some sense, the most unifying. Drawing on our previous results–network representation, optimization algorithms, distributed methods–we offer a generalization to permit resource allocation for *multiterminal links.* Under study for more than three decades (as the three-terminal relay link), multi-terminal links show great promise towards increasing the throughput of a system, but with the downside of increased interference in the larger network.

Our framework will allow, for the first time, a methodology with which to study the usefulness of multiterminal links within the broader network. Within that class we will consider only three-terminal relay links, but we will show that our framework is sufficiently general so as to handle any variety of multiterminal link for which the throughput can be characterized using information theory.

Our model and algorithm are extremely general by design. In this thesis, we will consider a fully connected network in which data throughput is our metric, but we will show that this is just one of many different ways in which a network may be optimized using our technique.

Our approach to the general problem is outlined in Figure 1.3, which also gives the outline of this document. The physical network topology, described in Chapter

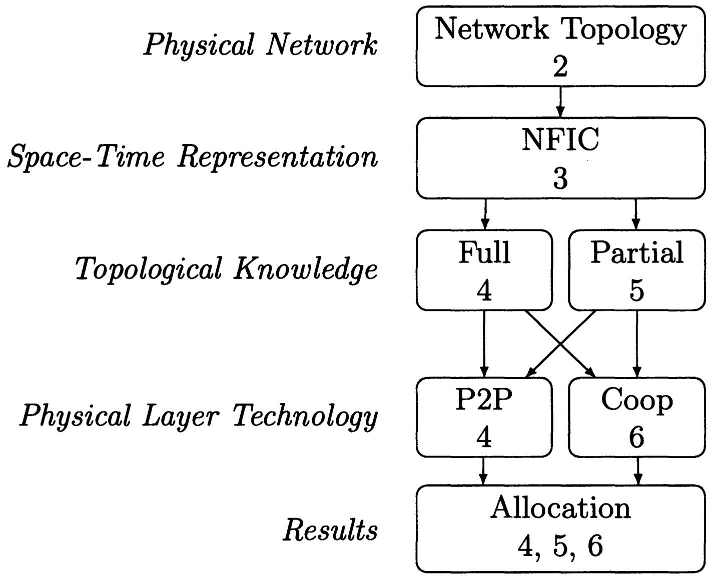


Figure 1.3: Organization of the document and our approach: the network topology leads to an NFIC representation, which may be the result of either full or partial network-state information. Transmission technologies range from simple point-to-point to cooperative techniques, leading to a particular network resource allocation.

2, gives rise to our space-time network representation, the NFIC, which we detail in Chapter 3. The amount of knowledge available to the NFIC is either full (Chapter 4) or partial (Chapter 4), and the communications technologies available to the system include Point-to-Point (Chapter 3) or Cooperative (Chapter 6). The result is a resource allocation, which appear in all three chapters. In Chapter 7, we summarize our results and present a unifying example: a decentralized multiflow allocation utilizing cooperative links.

CHAPTER 2

System Configuration

Before we can precisely define our problem, we must specify the system model under consideration. There are two key parts: the *network* model and the *physical* model. The network model specifies the region and how terminals are distributed within it, and the traffic flows the system must serve. The physical model specifies how the terminals exchange information on a signal-level.

We precisely define each of these elements of our model here, then go on to use those definitions to formulate our problem.

2.1 Network Configuration

We consider a network in a square region of size A^2, in which we position terminals according to a 2-dimensional Poisson process of intensity λ. We randomly define F source-destination pairs from the terminals of the network, establishing sources and sinks for F information flows denoted by $\mathcal{F}$.[1] We assume a common clock and synchronized timeslots.

Data is transferred for the flows in a stream of arbitrarily many *frames*, where

[1]We consider unidirectional flows here, but bidirectional data may be accommodated in our framework by defining F more flows operating in the reverse direction.

a frame is composed of small packets, the number of which is dictated by channel conditions. As such, frames crossing good channels can carry a larger amount of data than those being sent over poorer channels, leading us to model frame sizes as being from the positive real line: any value between 0 and $+\infty$.[1] We assume that frames are fully transmitted within one timeslot, so that the temporal unit of allocation is one timeslot.

Flow sources are infinitely backlogged,[2] and terminals are capable of only half-duplex communication, but can store frames in their memories for later transmission. We will denote the set of N terminals by $\mathcal{N}$, and will allocate resources over T timeslots, where T may become large. Each terminal may use a maximum transmit power P_{Tx}, and we denote the space of feasible powers with $\mathcal{P}$.[3] We will allocate resources for K frames, where the frames-per-flow $\frac{K}{F}$ is an integer. An example of this configuration is shown in Figure 2.1, for $N = 49$ and $F = 3$.

The flows transmit their data in a stream of many frames, which are indexed by k. We regard a frame as being data transmitted between two terminals in a single timeslot, making its way through the network from source terminal $s(k)$ to destination terminal $d(k)$. Any terminal in the network may act in a forwarding manner, and may also store data for future transmission. As such, we can define a *schedule* S specifying which transmitter and receiver pairs are active in each timeslot. A schedule specifying T timeslots exists in the binary space $\mathbb{I}^{N\times N\times T}$, since each timeslot has N^2 binary variables specifying the transmission state of each link.

A schedule S is termed *feasible* if it specifies utilization of network terminals such that frames travel from their sources to their destinations without violating the routing and duplexing constraints of the devices. Defining the space of feasible

[1]As will be seen, quantization of frame size to a finite set of values is also possible in our model.
[2]This assumption is made for clarity but is not required.
[3]While not shown here for notational clarity, each terminal may be assigned a unique P_{Tx}.

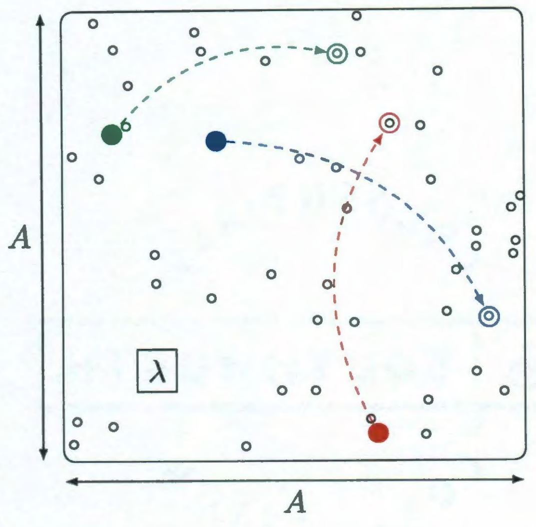


Figure 2.1: Picture of network configuration. Terminals are deployed in an $A \times A$ area according to a 2D Poisson process of intensity λ. F source (solid) and destination (open) pairs are defined, between which resources for an integer number of frames will be allocated. Here, $A = 20, \lambda = 0.01, N = 49$ and $F = 3$.

schedules requires the introduction of an auxiliary binary variable.

Definition 1. *The allocation variable $I_t^k(x, y)$ is 1 if frame k is being transmitted from terminal x to terminal y in timeslot t, and is zero otherwise.*

Note that $I_t^k(x, y)$ is indexed in both time t and space (x, y). Using this variable, we can define the set of feasible schedules $\mathcal{S}$. It is constrained by the following equations. The first is that all frames must be accounted for at all times:

$$\sum_{x,y \in \mathcal{N}} I_t^k(x, y) = 1 \quad \forall \quad k, \quad t \geq 1 \tag{2.1}$$

Frames must originate from their sources at time 1 and terminate at their destinations at some time $T_k > 1$. T_k is a timeslot index, specific to each frame, after which the frame has arrived at its destination and is no longer in transit across the network.

$$\sum_{y\in\mathcal{N}} I_1^k(s(k),y) \;=\; 1 \quad \forall \quad k \tag{2.2}$$
$$\sum_{x\in\mathcal{N}} I_{T_k}^k(x,d(k)) \;=\; 1 \quad \forall \quad k,\, T_k > 1 \tag{2.3}$$

The frame must not ever be received by its source or transmitted by its destination; this eliminates the possibility of "circular routing" in the solution.

$$\sum_{x\in\mathcal{N}} I_t^k(x,s(k)) \;=\; 0 \quad \forall \quad k, \quad t \geq 1 \tag{2.4}$$
$$\sum_{y\in\mathcal{N}} I_{T_k}^k(d(k),y) \;=\; 0 \quad \forall \quad k,\, T_k \geq 1 \tag{2.5}$$

We require continuity of routing: if terminal y received a frame in time t, it must transmit it again in time $t+1$, even if it "transmits it to itself," $I_t^k(y,y) = 1$, corresponding to data storage:

$$\sum_{x\in\mathcal{N}} I_t^k(x,y) = \sum_{z\in\mathcal{N}} I_{t+1}^k(y,z) \quad \forall \quad k,\, y\in\mathcal{N},\, t\geq 1 \tag{2.6}$$

Last, we require that each terminal operate in a half-duplex mode and serve only one frame per timeslot:

$$\sum_k \left[\underbrace{\sum_{z\in\mathcal{N}\setminus(x,y)} \left(I_t^k(x,z) + I_t^k(z,x)\right)}_{\text{Tx must be half-duplex}} + \underbrace{\sum_{z\in\mathcal{N}\setminus(x,y)} \left(I_t^k(y,z) + I_t^k(z,y)\right)}_{\text{Rx must be half-duplex}} \right] \leq 1 \quad \forall \quad (x,y)\in\mathcal{N} \tag{2.7}$$

These constraints apply to all frames k and are specified in terms of source-

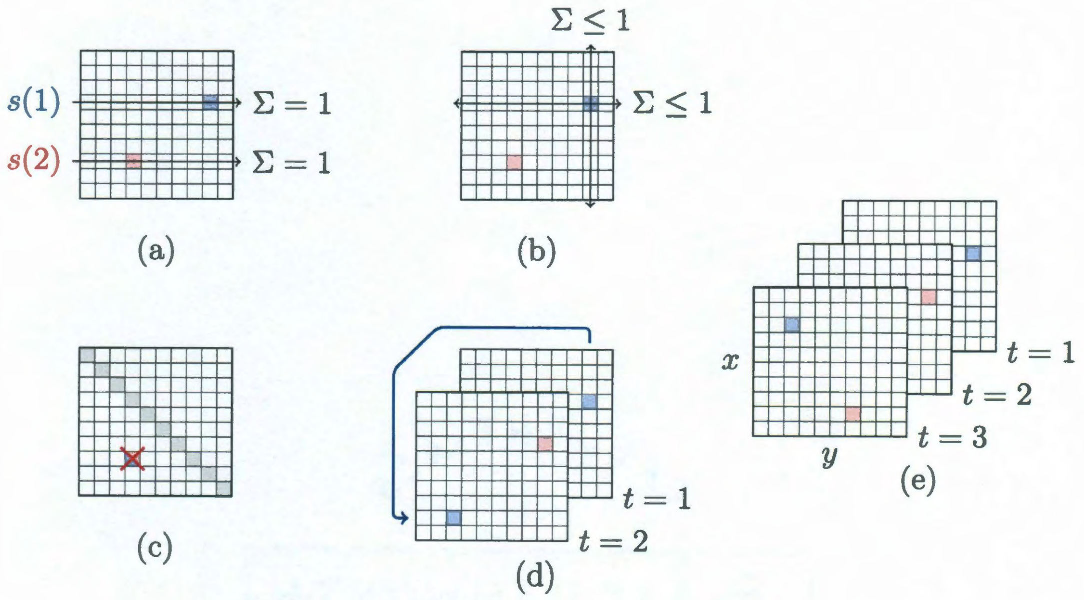


Figure 2.2: (a) Frames must originate at their source in slot 1, so there must be a '1' somewhere in the row corresponding to $s(k)$. (b) A terminal can handle only one frame at a time, so rows and columns can have only a single '1'. (c) A transmission cannot accomodate more than one frame at a time, unless it is in storage. (d) Routes must be continuous. (e) Visualization of how $I_t^k(x, y)$ evolves over time for three frames.

destination pairs (x, y) and a third terminal z. An illustration of $I_t^k(x, y)$ and the constraints is shown in Figure 2.2. I may be viewed as a matrix, with the transmitting terminals labeling the rows and receiving terminals labeling the columns. These illustrations depict a network with $N = 10$ terminals and $F = 2$ flows. A colored box indicates a '1' for the flow corresponding to that color. In pane (a), we show constraint (2.2), where in timeslot 1, each of the flows must have a '1' in the row corresponding to their sources. A similar figure may be drawn with vertical arrows for (2.3). Pane (b) shows that for a given transmit-receive pair, there can be no other active links in the corresponding row and column. This is the duplexing-multicast constraint (2.7). A link can only handle one frame unless it is a "storage" link, as shown in pane (c). Continuity of routing is shown in pane (d), where terminal 9 received the blue frame in timeslot 1 and must now transmit it in timeslot 2. An overall visualization of how $I_t^k(x, y)$ evolves with time is shown in pane (e).

We may also formalize the set of feasible power allocations $\mathcal{P}$, which is simply an N-dimensional space in which all terminals are allocated power levels between 0 and the maximum terminal transmission power P_{Tx}. The feasible region of powers is defined as:

$$0 \le P_t(x) \le P_{\text{Tx}} \quad \forall \quad t \le T \tag{2.8}$$

$$P_t(x) \le P_{\text{Tx}} \cdot \sum_k \sum_{y \in \mathcal{N} \setminus x} I_t^k(x, y) \quad \forall \quad x \in \mathcal{N} \tag{2.9}$$

where the second constraint guarantees that if terminal x is not transmitting at time t, its power assignment in that timeslot is exactly zero. This precludes spurious solutions in which terminals expend power but without transmitting data. With these definitions and constraints in mind, we make precise the related notions of schedules and routes.

Definition 2. *A* schedule *S is the set of variables I_t^k for all frames k and times t*

$$S = \bigcup_{t,k} I_t^k \tag{2.10}$$

This set indicates which links are active during each time slot, and is a synthesis of I across both time and space.

Definition 3. *For frame k, a* route *is the sequence of terminals it uses as it traverses the network. At time t, the terminal transmitting for frame k is written as*

$$L_t^k = \{x \mid I_t^k(x,y) = 1\} \tag{2.11}$$

Thus the route for frame k is

$$L^k = \bigcup_{t} L_t^k \tag{2.12}$$

In this way, specifying the variable $I_t^k(x,y)$ determines both the schedule at each timeslot and the routes for all frames *simultaneously.*

2.2 Physical-Layer Model

We consider an extremely general physical layer model, one in which the size of a frame is determined by the *information-theoretic capacity* of the link between two terminals. This allows us to capture the notion of interfering–rather than colliding–transmissions, introducing *rate control* and allowing a study of interference management at both the physical and medium-access layers together. Our notion of a frame is a collection of small packets, the fraction of which are successful being determined by the interference level. As such frames may be of different sizes, depending on channel conditions.

We use a pathloss-dominated additive white-Gaussian noise channel,[1] and follow the worst-case assumption that transmissions interfere with each other purely as noise.[2] All terminals are subject to a peak power constraint P_{Tx}, the thermal noise level is N_{o}, and the pathloss exponent is α. With $D_{x,y}$ we denote the Euclidean distance between terminals x and y, such that the rate R on the link between the terminals at time t for frame k is upper-bounded by Shannon capacity [93]:

$$R_t^k(x,y) = \log_2\left(1 + \frac{P_t(x)D_{x,y}^{-\alpha}}{N_{\mathrm{o}} + \gamma_t(y)}\right) \cdot I_t^k(x,y) \tag{2.13}$$

where the interference term

$$\gamma_t(y) = \sum_{z \in \mathcal{N}\setminus x} P_t(z) D_{z,y}^{-\alpha} \tag{2.14}$$

Here and after, we use $P_t(x)$ to denote the power used by terminal x in timeslot t. We will collect all terminal transmission powers at time t into a vector $\mathbf{p}_t$, which exists in $\mathcal{P}$.

Without loss of generality, we will consider a normalized bandwidth of 1 Hz and timeslots of 1 second, such that R_t^k is the size in bits of frame k at time t, or equivalently, the spectral efficiency fo the transmission in b/s/Hz. If the link between x and y is not active, the rate is necessarily zero, as is $P_t(x)$. This coupling of link-activation and power appears in (2.9), where $P_t(x)$ is multiplied by the link activation variable I.

A critical element of our model is the notion that a frame cannot be larger than what the bottleneck link in its route can support. This assumption means that we perform an end-to-end allocation of a frame, without needing to break it in the middle

[1] Although the large-geographical area we consider here leads to pathloss-dominant channels, our approach readily extends to fading environments.

[2] This assumption is made for simplicity of expressions; a more intelligent interference-cancellation scheme may be considered.

of its network traversal because a subsequent link is not strong enough to support it. As such we define the size of frame k as $R^k = \min_t R_t^k$.

2.3 Joint Problem Definition

Resource allocation entails managing interference in two ways: first by specifying the transmission set I_t^k for all frames and timeslots, and second by selecting the power level each terminal uses while transmitting so as to optimize some network-wide parameter. Consider power $\mathbf{p} \in \mathbb{R}^N$ in use at the terminals at time t as given in the vector $\mathbf{p}_t$. We refer to the set of vectors over all timeslots as $\mathbf{p}$, where each element in a vector corresponds to one terminal. The space $\mathcal{P}$ is the space of feasible power allocations, which may be smaller than $\mathcal{P}_{\text{Tx}}$ if interference management constraints prevent nearby terminals from using maximum power.

Network utility is a function of scheduling decisions and available power, according to a mapping $U : \mathbb{R}^{N\times T} \times \mathbb{I}^{N\times N\times T} \to \mathbb{R}$. This function may be defined in terms of throughput, aggregate delay, or some other relevant performance metric. Computing the optimal utility U^* requires finding the optimal schedule S^* and power allocations $\mathbf{p}^*$. If $\mathcal{P}$ resources are available, we write a vector-valued allocation function $A : \mathbb{R}^{N\times T} \to \mathbb{R} \times \mathbb{R}^{N\times T} \times \mathbb{I}^{N\times N\times T}$ as $A(\mathcal{P}) = (U^*, \mathbf{p}^*, S^*)$ where

$$\begin{aligned} U^* &= \max_{\mathbf{p}\in\mathcal{P},\, S\in\mathcal{S}} U(\mathbf{p}, S) \\ (\mathbf{p}^*, S^*) &= \arg\max_{\mathbf{p}\in\mathcal{P},\, S\in\mathcal{S}} U(\mathbf{p}, S) \end{aligned} \tag{2.15}$$

In our network model, evaluating $A(\mathcal{P})$ amounts to populating $I(x, y)$ for all frames from the F flows over all timeslots, and choosing transmission power levels $\mathbf{p}$. This is an NP-Hard mixed-integer program in general, and since the convexity of U is not guaranteed, the search space may have many local extrema in the continuous domain.

As the utility function U, we choose minimum throughput across the flows:

$$U(\mathbf{p}, S) = \min_{f \in \mathcal{F}} \eta^f \tag{2.16}$$

where the throughput η^f is defined as the sum of the data transmitted scaled by the time in transit:

$$\eta^f = \frac{1}{T} \sum_{k=\{f, f+F, f+2F, \ldots\}} R^k \tag{2.17}$$

where we sum only the frames carrying data for frame f. As such, evaluating (2.15) with this U results in computing the max-min throughput allocation in the network. This is NP-hard in general, and because the equation determining the throughput of U contains a fractional form inside the logarithm in (2.13), U is non-convex. Replacing (2.16) into (2.15), we have the joint minimax optimization problem:

$$\max_{\mathbf{p} \in \mathcal{P}, S \in \mathcal{S}} \min_{f} \eta^f \tag{2.18}$$

The remainder of this thesis will address the solution to this problem, which is an NP-hard, mixed integer, non-convex problem.

2.4 Remarks

As we mentioned in the Introduction, one of the aims of this thesis is to relax the notion that all transmissions must be orthogonalized. As such, our model is not strict Time Division Multiple Access (TDMA), but rather simply Time Division Duplex (TDD). The TDD system permits interference, prohibited by the TDMA implementation commonly found in the slot-synchronized systems such as the Global System for Mobile Communication (GSM). While this may appear to be a departure from established system-design norms in a literal sense, we point out that interference is a fundamental concern in *any* communications system. In the case of the GSM, the interference is managed by frequency reuse across cells, often resulting in wasted resources.

Recognizing this, schemes have been developed to exploit temporal and spatial "holes" in traditional cellular communications. Our motivation in the general model we propose here is not to *leverage* a design consequence, but rather to *design* systems from the ground up in which interference is optimally managed across the wide area.

Our choice of $\log_2(1 + \mathrm{SINR})$ may be viewed as an optimistic accounting of communications technology. We recognize that while this expression does not precisely specify the capacity of our channels (which would require timeslots of infinite length or infinite bandwidth), advances in coding techniques have brought this limit to the forefront of possibility in today's systems. In particular, Low Density Parity Check Codes (LDPC) are able to come within fractions of a decibel of this limit with modest block length, on the order of tens of thousands of bits.

Chapter 3

Decomposition Approach and Tools

We wish to select a schedule S and power allocation $\mathbf{p}$ which solves the maxmin problem

$$\max_{\mathbf{p}\in\mathcal{P}, S\in\mathcal{S}} \min_{f} \eta^{f} \tag{3.1}$$

This chapter will discuss the approaches one might take to solve the problem posed in Section 2.3. We will discuss the complexity of the problem, followed by a computationally-intensive "brute-force" solution technique. We will see how this method breaks down for anything beyond the smallest networks, leading us to our technique: a decomposition and associated data-management tool, used to both solve and catalog the decomposition.

3.1 Complexity and Convexity

The schedule S is an integer variable specified by $I_t^k(\cdot,\cdot)$, and the power $\mathbf{p}$ must be chosen to balance interference and throughput as specified by (2.13). We first consider the complexity of selecting the schedule S.

A system model similar to ours is considered in [86], though without the rate-control element. In this work, the authors show that computing a routing schedule with minimum *makespan* (arrival time of last packet) is polynomially reducible to the proven NP-Hard 3-BOUNDED-3SAT problem. This is the Boolean satisfiability problem with three literals per clause, and in which each variable appears at most three times in the expression. From an intuitive perspective, this is because link-activation is inherently binary. The notion of terminals appearing only three times and being grouped in triples is the result of the graph-coloring theorem. This reduction implies that computing the routing schedule for each frame is NP-Hard. We refer the reader to [86] for details on the proof, but emphasize here the computational difficulty in computing the routing schedule.

As a result, we must check *all* feasible schedules and routes the frames may take in order to have an optimality guarantee for our solution, which becomes prohibitively complex for networks with many terminals. For a given source-destination link in a network of N terminals, there are $N-1$ possible first-hop links. From each of those possible first hops, there are $N-2$ possible second hops, leading to $N!$ enumerable routes between the source and destination. Because

$$(N/3)^N < N! < (N/2)^N \tag{3.2}$$

the complexity of the integer problem–computing the routing schedule–alone grows as $O(N^N)$ for large N.

Even if we fix a schedule and route S, we must still compute the power allocation $\mathbf{p}$ to determine the rate. This calculation is non-convex, as we demonstrate with a simple example. Consider the network shown on the left in Figure 3.1, where two flows are active in one timeslot: $A \to B$ and $X \to Y$. On the right, we plot

$$\overline{T} = \frac{1}{2}\log_2\left(1 + \frac{P_A}{P_X}\right) + \frac{1}{2}\log_2\left(1 + \frac{P_X}{P_A}\right) \tag{3.3}$$

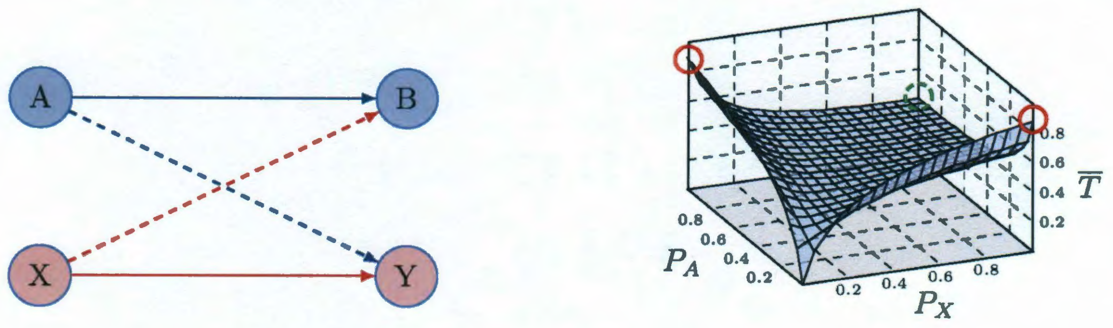


Figure 3.1: A simple two-flow network on the left, and the corresponding mean throughput shown on the left as function of transmission power.

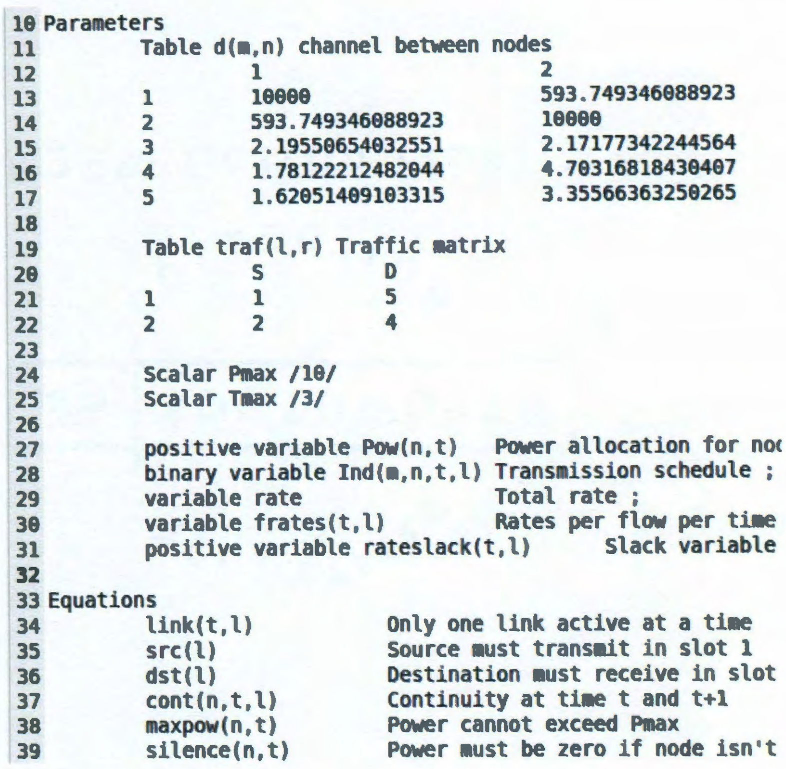
as a function of the transmission powers P_A and P_X.

Solution techniques seeking points at which the KKT conditions are satisfied will tend to the circles on the extreme edges of graph, while the max-min point is the green circle at the back of the plot. At these locations, complimentarity constraints will bind such that only one flow is transmitting, and the other uses zero power. This unfortunate situation is fundamentally caused by the fractional term the logarithm, which immediately gives a non-convex system, clearly seen in the shape of the surface.

The joint problem is therefore extremely difficult to solve: the integer program is NP-Hard, and the continuous program is non-convex. Any technique guaranteeing the solution would necessarily check all N^N possible integer solutions, and *for each one* evaluate a non-convex continuous program in potentially hundreds of variables.

3.2 Numerical Solution Techniques

One method to find the solution to the joint problem is to employ a successive computational approach, using branch-and-cut techniques to first trim the integer program and then randomized-restarts to address the nonconvexity of the continuous program. The computational resources required for this type of approach are enormous, yet the



```
Parameters
        Table d(m,n) channel between nodes
                 1                      2
        1        10000                  593.749346088923
        2        593.749346088923       10000
        3        2.19550654032551       2.17177342244564
        4        1.78122212482044       4.70316818430407
        5        1.62051409103315       3.35566363250265

        Table traf(l,r) Traffic matrix
                 S          D
        1        1          5
        2        2          4

        Scalar Pmax /10/
        Scalar Tmax /3/

        positive variable Pow(n,t)   Power allocation for no
        binary variable Ind(m,n,t,l) Transmission schedule ;
        variable rate                Total rate ;
        variable frates(t,l)         Rates per flow per time
        positive variable rateslack(t,l)      Slack variable

Equations
        link(t,l)           Only one link active at a time
        src(l)              Source must transmit in slot 1
        dst(l)              Destination must receive in slot
        cont(n,t,l)         Continuity at time t and t+1
        maxpow(n,t)         Power cannot exceed Pmax
        silence(n,t)        Power must be zero if node isn't
```

Figure 3.2: Generalized Algebraic Modeling System (GAMS) code for the problem at hand. Here there are five terminals and two traffic flows, with a series of equations constraining the solution to the scheduling feasiblity space defined in Chapter 2.

problem can be defined in a way that such brute-force techniques can be deployed.

When written in the Generalized Algebraic Modeling System (GAMS), our problem takes no more than twenty lines of description, as shown in Figure 3.2. This description can then be fed into any of a variety of computational solvers, such as the MOSEK, KNITRO, and LINDO Global [69] packages. These are all proprietary solvers, made available on National Labs NEOS servers.

Because it may be easier for humans to formulate the problem in a series of related yet possibly redundant constraints, the original system (while consistent) may be overdetermined. To reduce this overdetermined system to its smallest (yet possibly non-intuitive) form, the solvers first execute reformulation routines prior to attempting a solution search. In Table 3.1, we report the computational resources

System Parameters	Equations	Continuous Variables	Integer Variables
$N = 5, F = 1, T = 3$	33	91	75
$N = 5, F = 2, T = 3$	151	178	150
$N = 10, F = 3, T = 2$	297	633	600

Table 3.1: Computational complexity of our problem after reduction by the LINDO-Global numerical solver. National Labs servers processing these problems limit the integer program to 600 variables, meaning that the $(10, 3, 2)$ system is the largest we can study using this approach.

consumed by small systems, *after* reduction, when solved numerically. All dimensions in our model–number of terminals, number of flows, and number of timeslots–are at least an order of magnitude below those we wish to address, yet the size of the reduced system is considerable. In particular, we reach the upper limit of the public-domain servers for the third network, since we have a system with 600 integer variables–the maximum permitted on the NEOS cluster.

In section 4.8.3, we will report the solution to a simple problem using the LINDO computational solver, and will compare it to the novel approach we now propose.

3.3 Decomposition of the Joint Problem

The problem described in Section 2.3 is, as we've shown, prohibitively complex for anything beyond the smallest of networks. In this section, we will describe the decomposition approach we take to bring this problem into a tractable domain.

3.3.1 Overview

We propose a decomposition of this problem across the frames k. Fundamentally, we study the problem of finding a time-space allocation $I_t^k(\cdot, \cdot)$ for *one* frame which optimizes the utility gained from adding it to the network, and we then update the resource availability $\mathcal{P}$ to reflect the power resources remaining after frame k has been allocated. As such, we break problem (3.1) into a sequence of subproblems concerning only one frame, where the constraints $I_t^k(\cdot,\cdot)$ and resources $\mathbf{p}$ are updated iteratively as we solve the subproblems for each frame, conditioned on the solution for all previous frames. We treat frames from the flows in a round-robin manner, such that frame k and frame $k+F$ carry data for the same flow. Our algorithm, to be discussed in detail in Chapter 4, is thus summarized as follows for the general case:

1. Given resource availability $\mathcal{P}$, select the time-space allocation $I_t^k(x,y)$ for frame k that maximizes the network utility derived from frame k, U_k^*. That is, evaluate $(U_k^*, \mathbf{p}_k^*, S_k^*) = A(\mathcal{P})$ as if k were the only frame requiring network resources. The result will populate $I_t^k(x,y)$ with 1's to indicate active links in the schedule, which implicitly defines the route for that frame. $\mathbf{p}^*$ represents power allocation for terminals on that route.

2. Remove the resources used by frame k as it traversed the network according to schedule S_k^*. That is, solve $\mathcal{P} = A^{-1}(U_k^*, S_k^*, \mathbf{p}_k^*)$. Intuitively, this makes the terminals used by frame k unable to offer resources to a subsequent frame, enforcing constraints (2.1) - (2.7) as well as limiting transmission power of other terminals to forbid future frames from excessively interfering with those already in the network.

3. Use this updated resource set and repeat for frame $k+1$.

Efficiency and complexity management motivate this choice of decomposition. We will show that evaluation of $A(\mathcal{P})$ may be accomplished in polynomial time for only one frame with one source and one destination; similarly, the resource update in Step 2 above is also a polynomial time operation. We require the existence of solution to $\max_{\mathbf{p},S} U(\mathbf{p}, S)$, which means that the image of $\mathbf{p}$ and S under U must be closed and bounded. Assuming finite resources $\mathcal{P}$ and since S is constrained as in Section 2.1, we can consider any smooth and bounded U. Last, we require that A is reversible given some schedule S^*, power allocation $\mathbf{p}^*$ and utility U^*. Here, we mean that given a level of network utility and a resource allocation, we must be able to identify the level of resources *remaining* in the network; this requires that A^{-1} exist, such that at any time t, the mapping between network utility and required resources is invertible.

There are network architectures for which A^{-1} is not available because A is not one-to-one. For instance, consider the case of the *cooperative* allocation, in which a single link may be composed of several simultaneous, coordinated transmissions each of which is defined by several values in $\mathbf{p}$. In this case A is many-to-one and cannot be uniquely inverted. This will be addressed in Chapter 6.

While these concepts have been presented in the abstract to illustrate their generality, we will confine our attention to the *throughput maximization* problem described in Section 2.3.

3.3.2 Decomposition

As the utility function U, we choose mininum throughput across the flows:

$$U(\mathbf{p}, S) = \min_{f \in \mathcal{F}} \eta^f \tag{3.4}$$

where the throughput η^f is defined as the sum of the data transmitted scaled by the time in transit:

$$\eta^f = \frac{1}{T} \sum_{k=\{f, f+F, f+2F, \ldots\}} R^k \tag{3.5}$$

where we sum only the frames carrying data for flow f, since frames are sequenced in a round robin manner.

We employ a frame-wise decomposition of this problem, solving a sequence of problems (3.1) for each frame k. This requires enforcing the data flow constraints in the variable $I(\cdot,\cdot)$, while optimizing power over a space $\mathcal{P}$ defined such that excess interference is not caused for existing frames in the network. Each frame may be viewed as contributing an incremental element to the throughput of flow f, as such we seek to maximize this element. Denote with a_k the arrival slot of the k^{th} frame, such that a_{k-F} is the arrival slot of the last frame carrying data for flow f. For each frame in the sequence we seek to solve

$$\max_{\mathbf{p}\in\mathcal{P},\mathcal{S}\in\mathcal{S}} \frac{1}{a_k - a_{k-F}} R^k \tag{3.6}$$

which is the throughput contribution for this frame: the amount of new data divided by the number of timeslots elapsed since the previous frame arrived. This contribution is the result of the separability of the sum in (3.5).

We may now compare our iterative, per-frame solution to the full problem expressed in (3.1) above. In Figure 3.3, we have exposed in the abstract the three dimensions of the problem: power allocation $\mathbf{p}$, link activation decisions $I(\cdot,\cdot)$, and frames k. The figure shows the joint problem, in which decisions made for each frame directly affect those made for the other frames–hence the bidirectional arrows between allocation points. For the purpose of exposition, the blue panes represent the schedules and power allocations feasible for each frame; in the joint allocation, all combinations are feasible subject to decisions made for other frames.

Our decomposition is shown in Figure 3.4. When problem (3.6) is solved for frame $k = 1$, all schedules and power allocations are feasible. After choosing the optimal sequence of terminals $I(\cdot,\cdot)$ for that frame, power allocations for terminals involved are reduced such that all links operate at the same rate. This schedule and corresponding power allocation then determine feasible schedules and power allocations for the second frame, which is a smaller space than was available for the first. This

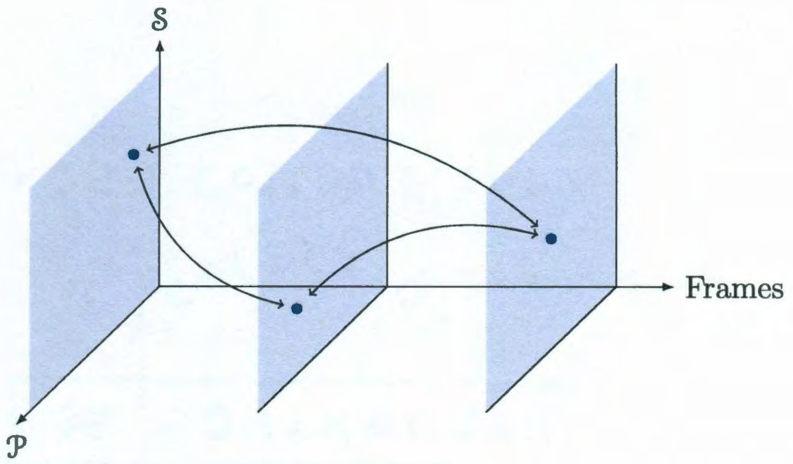


Figure 3.3: The joint optimization: the selection of schedule and power allocation for each frame directly influences the feasible regions for all others.

process then repeats for subsequent frames.

While the decomposition makes our problem tractable in general, we must efficiently manage the data in the sequence of sub-problems. We now describe the data structure we have created for this purpose. We name it the *Network-Flow Interaction Chart* (NFIC), because–put succinctly–it represents exactly how data from network flows interacts with the terminals in the network, and with other data.

3.4 Network-Flow Interaction Chart

Traditional network graphs lay out the network in a geographical manner, drawing edges between nodes to represent connectivity. In our problem, we require a network representation in which the connections between terminals are permitted to change with time. Further, we would like construct a graph in which the evolution of these connections is easy to monitor.

The frame-based nature of our network allows for a natural discretization in time, and since terminals are physically separated in space, they are also discrete elements.

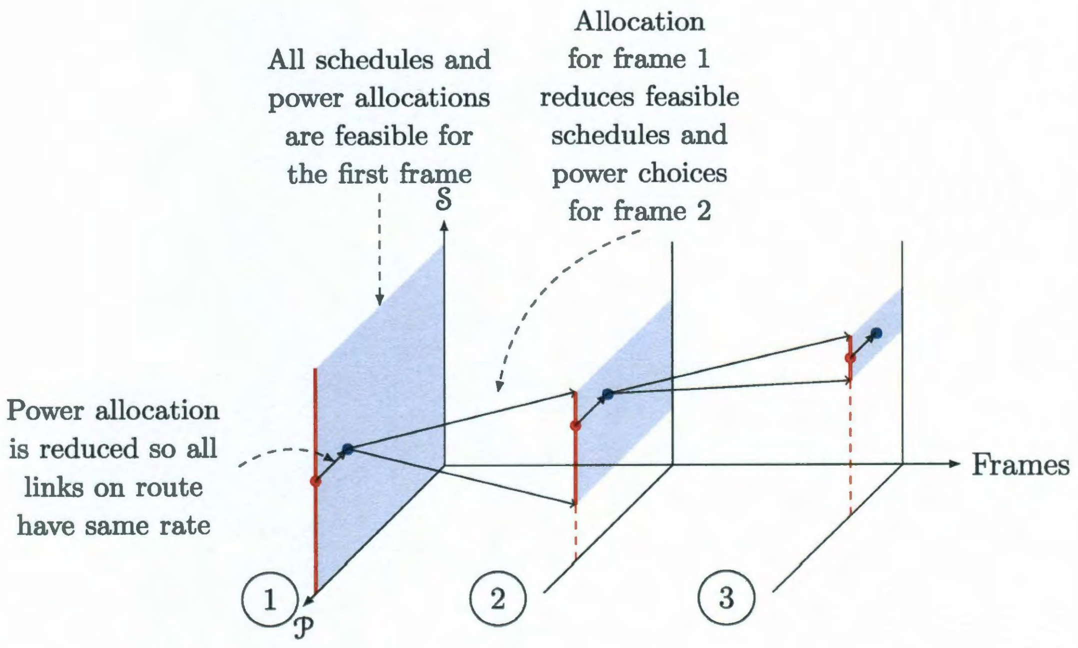

All schedules and power allocations are feasible for the first frame
Allocation for frame 1 reduces feasible schedules and power choices for frame 2
Power allocation is reduced so all links on route have same rate
$\mathcal{S}$
$\mathcal{P}$
Frames
1
2
3

Figure 3.4: Optimization decomposition: the first frame is optimized among all feasible schedules with high power, then power is reduced such that all links operate at the bottleneck rate. The feasible region for the frame packet is a consequence of the allocation for the first, and the algorithm repeats.

As such, we have two discrete axes to consider, which we will map into the two dimensions of the graphical model. We discretize time in the x direction and terminals in the y direction. Terminals are represented as *nodes* in the NFIC in the vertical direction, such that their discrete nature is represented by the whitespace between them in that dimension.[1]

Definition 4. *The* Network-Flow Interaction Chart *is a graphical data structure, comprising of a set of nodes $\mathcal{V}$, replicated across timeslots t as the sets $\mathcal{V}_t$. The sets of nodes $\mathcal{V}_t$ and $\mathcal{V}_{t+1}$ are connected with a set of directed edges $\mathcal{E}_t$. The NFIC represents T timeslots of network activity. The nodes and edges in the NFIC correspond to the terminals and channels of the network at a given time t.*

The nodes and edges in the NFIC correspond to the terminals and channels of the network at a given time t, where time is indexed from slot 1.

Definition 5. *Define the* non-cooperative terminal mapping $V^{\mathrm{NC}} : \mathcal{N} \times T \to \mathcal{V}^{\mathrm{NC}} \times T$, *which maps single terminals from the physical network to nodes in the NFIC.*

3.4.1 Edges

The second component of our model are the *edges* of the graph, representing the ways terminals are connected with each other at time t. For the time being, we restrict the NFIC to be a *directed* graph, though in the future one may consider a *hypergraph* in which edges may have one origin and many destinations.

At time t, the network has a particular state: terminals x and y are able to interact in a way determined by channel conditions and also the actions of other terminals, since all transmissions interact. If there is an interaction between terminals x and y, we draw an *edge* in the NFIC, labeled $e_t^{x,y}$. In particular, because transmissions are unidirectional, the originating terminal is x and the receiving terminal is y.

[1]We refer to *terminals* and in the physical network and *nodes* in the NFIC.

Definition 6. *Define the* non-cooperative rate mapping $E^{\mathrm{NC}} : \mathcal{N}^2 \times \mathcal{P} \times T \rightarrow \mathcal{E}^{\mathrm{NC}} \times T$*, mapping power availability at network terminals to edge weights in the NFIC. Explicitly, for the channel between terminals x and y at time t, the edge $e_t^{x,y}$ in the NFIC is assigned a weight given by the mapping as:*

$$e_t^{x,y} = \log_2 \left(1 + \frac{P_t(x) D_{x,y}^{-\alpha}}{\sum_{z \in N \backslash x} P_t(z) D_{z,y}^{-\alpha} + N_o} \right) \tag{3.7}$$

These mappings and the corresponding NFIC are illustrated for a small network in Figure 3.5.

3.4.2 Node and Edge Weights

Directed edges in the NFIC represent data flowing between a pair of terminals, and are given meaning with a *weighting value.* They may be drawn from any of the following notions:

- **Channel capacity:** This is the case we study here, where edge weights represent the Shannon capacity of a point-to-point link in the network, as specified in Definition 6.

- **Channel state:** In the protocol model, a channel may be said to exist perfectly or not at all, depending on the interference at the receiver. In this case, edge weights will be drawn from the binary field.

- **Delay of the link:** In cases of delay optimization, an edge weight may represent the time a frame spends enqueued at the source terminal before being transmitted to the next terminal.

- **Cost of the link:** This general notion may be a function of power and time needed for a transmission, when optimizations tradeoff energy expenditure with

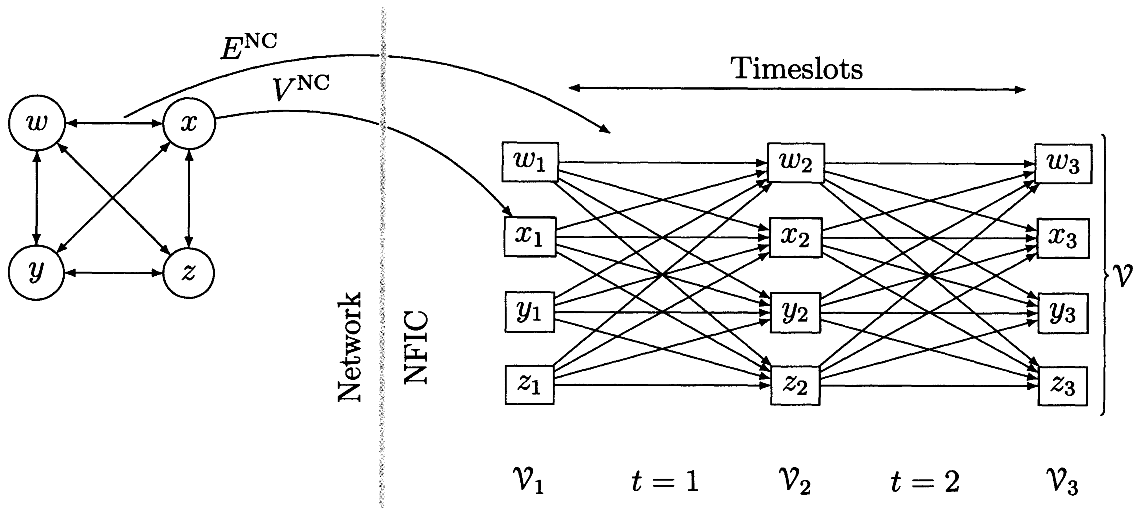


Figure 3.5: An example network (left) and the corresponding NFIC (right). The mappings E^{NC} and V^{NC} define nodes and edges corresponding to point-to-point links.

delay.

This list is not inclusive; the edge weight may also represent a function of all of these things, reflecting the *utility* of using the link. Node weights will also be assigned from the same field as the edge weights, in a manner determined by the execution of resource allocation algorithms.

3.5 Semirings: Optimal Subproblem Property

This data structure leads to our main result, an algorithm which does not require an exhaustive search to find near-optimal routes and schedules for the network. Reaching this goal requires exploiting the *Optimal Subproblem Property*, which allows us to decompose the overall single-flow routing-scheduling problem into a series of smaller, easier-to-solve problems. Specifically, we will decompose the routing-scheduling problem into a series of optimizations executed over only one time period.

Problem (3.6) will have this factorization property if the problem parameters meet some basic algebraic requirements. Namely, we must specify the set over which we are optimizing and the update functions which affect our solution such that they form a monotone semiring. If this requirement is not met, then the solutions obtained by the dynamic programming approach cannot be guaranteed. In what follows, we draw on notions from group theory [13].

Definition 7. *By a* semiring *we mean a set A with summary and extension operators $\oplus$ and $\otimes$ and their corresponding identity and annihilator elements. We denote the semiring as*

$$K = \{A, \oplus, \otimes, \mathrm{I}_\oplus, \mathrm{I}_\otimes\} \tag{3.8}$$

As it relates to our model, we will assign weights to the nodes and edges from the set A, and we will use $\oplus$ and $\otimes$ to update these weights as we progress through the chart. Namely, the node weights are updated according to

$$x_t = \bigoplus_{y \in \mathcal{V}_{t-1}} e_{t-1}^{y,x} \tag{3.9}$$

which is the semiring summary of all incoming edges, and the edge weights are updated according to

$$e_t^{x,y} = e_t^{x,y} \otimes x_t \tag{3.10}$$

which is the semiring extension of the initial edge weight by its origination node weight. Note that (3.9) does not apply to the set of nodes at time $t = 1$.

Definition 8. *We define a* path L^k *for a frame k as a sequence of T edges $L^k = \{e_1^{s(k),x}, e_2^{x,y}, \ldots, e_T^{z,d(k)}\}$ where the indices $s(k)$ and $d(k)$ denote those of the source and destination for the frame.*

We will use $\mathcal{L}$ to denote the collection of all unique paths between $s(k)$ and $d(k)$, since there will likely be more than one depending on the network topology. We now illuminate these concepts with two examples of network-centric semiring definitions.

Example 1. *Consider the case of a capacity maximization. Here, we wish to maximize the capacity of all the links in the route, which is to say we wish to maximize the minimum capacity in the route. The corresponding semiring is*

$$K_C = \{\mathbb{R}^+, \max, \min, 0, +\infty\} \tag{3.11}$$

The edge weights represent the capacity between two nodes, such that when optimizing for flow f, we can use equation (3.7) to write

$$e_t^{x,y} = R_t^k(x, y) \tag{3.12}$$

and the general optimization problem is:

$$\max_l \min_t \{e_t : e_t \in L_l,\ L_l \in \mathcal{L}\} \tag{3.13}$$

which corresponds to choosing the path with the largest minimum capacity.

Example 2. *Suppose we wish to minimize the delay along a route. Here the edge weights represent delay in transmitting from node x to node y. The corresponding semiring is*

$$K_D = \{\mathbb{R}^+, \min, +, +\infty, 0\} \tag{3.14}$$

and the general optimization problem is:

$$\min_l \sum_t e_t,\ e_t \in L_l,\ L_l \in \mathcal{L} \tag{3.15}$$

which corresponds to choosing the path with minimum total delay (more generally, the widely-studied "minimum cost path").

Lemma 1. *If K is a monotone semiring, i.e. is equipped with a partial ordering $\leq$ and has $a \oplus a = a$ for all $a \in A$, and if values from K are assigned as weights to the edges of a graph G, then the sequence of edges L from node s to node d which extremizes the extension*

$$\bigotimes_t e_t \quad e_t \in L \tag{3.16}$$

also extremizes the same extension for the sequence of edges from s to k, where k is an intermediate node on the path L.

In short, this lemma states that the optimal path between s and d is also the optimal path between s and any of the points on that path. This means that at each step in the chart, we must make a local decision on optimality; the optimal route will never involve a suboptimal decision at any step. This is the key to our algorithm, presented in the following chapter.

3.6 Specialized Applications of the NFIC

The NFIC may be modified to represent several specialized network configurations, a few of which we discuss below.

- **Wired Backhaul.** In the case of a mesh network, there are several "super terminals" in the networks which are connected to a wired backhaul of very high bandwidth and essentially zero delay. The NFIC can represent these special terminals as a type of "supernode," connected to each terminal in the network but with the weight on the edge being calculated as between that terminal and the *nearest* backhaul. Duplexing constraints are somewhat relaxed on this supernode, since it functionally represents many different access points.

- **Terminal Arrival and Departure.** If terminals are known to be joining or leaving the network at predetermined times, they can be phased into or out of the NFIC accordingly. This has application in, for instance, the addition of congestion-relieving backhauls at peak traffic times.

- **Warm Start.** In this work we assume an empty network, but this need not be the case. If the state of the network is partially known, i.e. if some frames are already in the network and en route to their destinations while the NFIC is being initialized, those allocations can be represented in the initial conditions of the NFIC.

These are a few of the extensions possible on the NFIC. While we do not consider any of these explicitly in the discussions that follow, we mention them as possible applications of NFIC tools.

CHAPTER 4

Resource Allocation Algorithm

We now present the solution technique for problem (3.6) discussed in the previous chapter. Namely, we must establish how $I(\cdot,\cdot)$ is populated, and how the space of available transmission powers $\mathcal{P}$ evolves as frames are allocated.

4.1 Frame Sequencing

As discussed in Section 3.5, we will use shortest-path programming techniques to solve (3.1) optimally for one frame at a time. To determine the sequence in which frames should be allocated, we recall the max-min nature of the optimization formulation in (3.4). Since the successive allocation of frames reduces available network resources, we allocate the frame we expect to require the most resources prior to those expected to require fewer.

We must determine this sequence without solving the joint problem. The pathloss-dominant nature of our environment provides such a method: since terminals are distributed according to a Poisson process, the average distance between terminals is $\frac{1}{\sqrt{\lambda}}$. A frame requiring more hops is more likely to experience a rate-limiting long hop, limiting its throughput. As such, we estimate that the flow with the longest end-

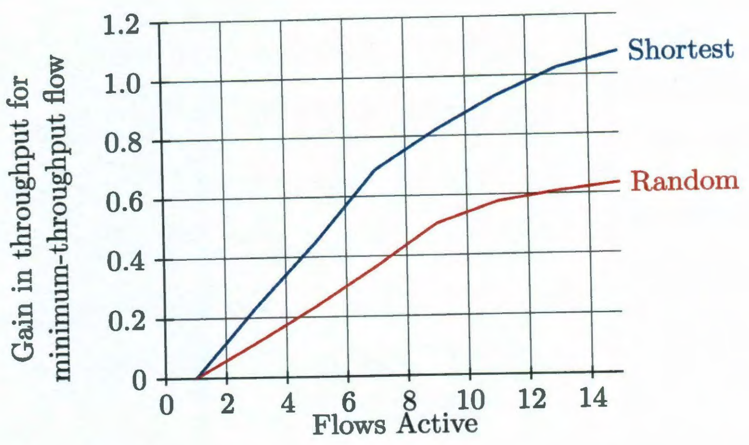


Figure 4.1: Relative gains of scheduling the flow in descending order of Euclidean distance over random sequencing and increasing-distance sequencing. Throughput gain is reported for the minimum flow.

to-end distance should be considered for allocation first, followed by the one with the second longest end-to-end distance and so on. After a frame has been allocated for all F flows in this order, the sequence repeats for frame $F+1$. This means that frame k is carrying information for flow $k \bmod F$, since we increment the frame counter for each individual frame considered.[1] Recall that we denote the terminal sourcing frame k as $s(k)$ and its destination as $d(k)$.

In Figure 4.1 we report the relative increase in minimum throughput when using the maximum-distance frame sequencing described above over both random and minimum-distance sequencing. We note that minimum-distance sequencing performs worst, followed by random sequencing. This confirms our intuition that the longest route is likely to experience the minimum throughput and therefore should be given priority in the resource allocation. Note in particular that the throughput gain increases as flows are added to the network, highlighting how the resources consumed by many short flows being allocated before the longer one reduce the throughput on

[1]We consider round-robin allocation for streaming flows here, although an interesting optimization problem would be to attempt a more optimal sequencing.

the longer flow.

4.2 Frame Allocation: Link Activation and Rate Control

We explicitly describe how to solve (3.6) for the first frame. Prior to allocating any frames, $I_t^k(\cdot,\cdot)$ is zero for all terminals and $\mathcal{P} = \mathcal{P}_{\text{Tx}}$, since there is no interference constraint in an empty network. We initialize the NFIC edge weights corresponding to the capacity that link represents, as

$$e_t^{x,y} \leftarrow \log_2\left(1 + \frac{P_{\text{Tx}}}{N_{\text{o}} + 0} D_{x,y}^{-\alpha}\right) \tag{4.1}$$

for all t. This corresponds to the best rate (with normalized 1 Hz bandwidth and 1 second slots, the maximum frame size) between any terminals x and y, since terminals may use maximum power and, since no resources have been allocated, any single transmission would be interference-free. We now assign the source node for frame 1, $s(1)_1$, a value of ∞ at time 1 corresponding to the infinite amount of data this source wishes to send. If the source had less data to transmit, this would be reflected in this initial source node weighting. All other nodes receive weight $\text{I}_{\otimes} = 0$ at time 1, because only one frame is being considered. The dynamic programming shortest-path procedure now executes on the NFIC: edges are updated according to the minimum of the source node weight and the edge initialization weight, which corresponds intuitively to the rate-control notion that a frame cannot be larger than the channel can bear, nor should it be larger than the amount of data at the source of the link:

$$e_1^{x,y} \leftarrow \min\{e_1^{x,y}, x_1\} \quad \forall \quad (x,y) \in \mathcal{N} \tag{4.2}$$

The node weights for timeslot 2 are assigned according to the summary operation $\oplus$, maximization. We also record *which* edge gives rise to this maximum in the auxiliary "history" variable h_t^y:

$$\begin{aligned} y_2 &= \max_x e_1^{x,y} \quad \forall \quad y \in \mathcal{N} && (4.3) \\ h_2(y) &= \arg\max_x e_1^{x,y} \quad \forall \quad y \in \mathcal{N} && (4.4) \end{aligned}$$

This corresponds to terminal y receiving data originating at $s(1)$ from the intermediate terminal which offers the most data. In subsequent timeslots, this two-step process of node-weight and edge-weight updates is repeated until all node weights have stabilized, i.e. until $x_t = x_{t+1}$ for all $x \in \mathcal{N}$. When this occurs, all *data maximizing* routes between $s(1)$ and all terminals have been determined. At any time-period t, the shortest-path algorithm guarantees that the weight at the destination node $d(1)_t$ for frame 1 is the maximum amount of data that could be transmitted over a route with $t-1$ hops. As such, to calculate the throughput-maximizing route for a general frame k, we evaluate the throughput-contribution for routes of lengths t as

$$\eta_t^k = \frac{d(k)_t}{t - a_{k-F} - 1} \tag{4.5}$$

which is the amount of data at the destination at time t scaled by how much time has passed since the previous frame arrived. The arrival time of the previous frame for this flow is a_{k-F}, zero for $k \leq F$. Searching η_t^k across timeslots for the maximum value gives the frame contributing most to the throughput for this flow:

$$t_{\text{opt}} = \arg\max_t \eta_t^k \tag{4.6}$$

The route L^k is then constructed by tracking back through the NFIC starting at the destination node $d(k)_{t_{\text{opt}}}$, using the history variable $h_t(x)$. The last terminal in the route is the destination: $L^k_{t_{\text{opt}}} = d(k)$. The others are found using the recursive relation $L^k_t = h_{t+1}(L^k_{t+1})$. The binary link-state variable $I^k_t(\cdot,\cdot)$ may now be updated as well:

$$I^k_t(L^k_t, L^k_{t+1}) = 1 \quad \forall \quad t < t_{\text{opt}} \tag{4.7}$$

This technique will solve (3.6) for an individual frame k, populating I and calculating the power allocation $\mathbf{p}$. Note how the execution of the dynamic programming algorithm implicitly ensures that the schedule chosen will be feasible: in the solution for frame k, only one link will be active per timeslot (2.1), there will be continuity (2.6), and the source and destination will be visited only once (2.4) , (2.5). We now show how resources are removed from the NFIC to enforce the routing and duplexing constraints on I for subsequent frames, step 2 in the general algorithm outlined above.

An extremely simple execution of this algorithm is illustrated in Figure 4.2, where one frame from s to d is allocated in a network of four terminals.

4.3 Optimality of the Solution Algorithm

Before showing how we allocate power to each transmitter in frame k's route, we show the throughput-optimality of our algorithm as it acts on the NFIC for frame k. Note in particular that the proof concerns the NFIC *only*, where cooperative terminals are represented as individual nodes and edges, just as for the non-cooperative links.

Theorem 1. *An allocation for frame k determined by the algorithm given above is throughput-optimal for frame k, given an NFIC $(\mathcal{V}, \mathcal{E})$ and that the set from which edge weights are drawn and the node- and edge-update operations together with their*

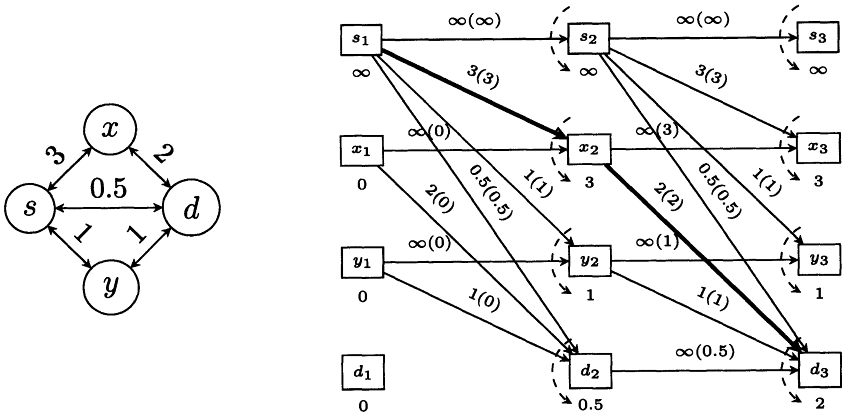


Figure 4.2: Example NFIC traversal for the network shown on the left. Initial node weights are shown under the nodes in the first column, and initial edge weights are shown on the edges. Updated edge weights are shown in parens. The optimal route turns out to be from s to d through x, since two bits received over two timeslots is greater than half a bit received in one timeslot.

identity elements form a monotone semiring.

Proof: We denote our monotone semiring as $K = \{\mathbb{R}^+, \max, \min, \infty, 0\}$, where max is the summary operation, and min is the extension operation with ∞ and 0 the corresponding identity elements. Our set is the non-negative real line, since achievable rates of channels cannot be negative.

Restrict attention to paths of t timeslots, and consider a path formed by a sequence of edges in the NFIC from the source node in the NFIC at time 1, s_1, and the destination node at time t, d_t. Label that path p_i, where i indexes all possible sequences of length t.

The sequence offering the highest throughput is the one yielding the most data in the $t-1$ slots the frame used to cross the network. The amount of data path p_i can sustain is limited by its bottleneck. To represent the amount of data corresponding to this path, assign label l_i to the path as $l_i = \min\{e_1^{s,x}, e_2^{x,y}, \ldots, e_t^{z,d}\}$, where $e_j^{x,y}$ are the edges comprising path p_i from s to d. l_i is therefore the minimum edge weight, which from definitions 6 and 12 is the smallest data rate a link in the route can support.

The throughput-maximizing path of length t is the solution to

$$\arg\max_i \left\{ \frac{l_i}{t-1} \right\} \tag{4.8}$$

This is (3.6) written in terms of NFIC edge-sequences, since the amount of data is represented by l_i and we scale by the number of edges, each corresponding to one timeslot. Replacing the definition of l_i into (4.8) gives a maximization over a set of minimizations, or

$$\arg\max_i \left\{ \frac{\min_t\{e_1^{s,x(i)}, e_2^{x(i),y(i)}, \ldots, e_t^{z(i),d}\}}{t-1} \right\} \tag{4.9}$$

where $x(i)$ and $y(i)$ are the first and second terminals in edge-sequence p_i. Since we are working with a monotone semiring, we may distribute the max over the min without affecting the result of the calculation. This gives us the algorithm above: maxima are calculated at each node over the edge weights, which are themselves the result of minimizations. This holds for paths of all lengths, so that we identify the optimal path of length t. Sweeping across timeslots for the maximum throughput then concludes the proof.

4.4 Power Allocation Algorithm

Once the optimal route for frame k has been chosen, we must minimize the power required to sustain the rate on that route, which in-turn minimizes interference in the network. If the rate for frame k is R^k, then we update the power at each node in the route according to

$$P_t(x) = (N_o + \gamma_t(y))\, D_{x,y}^{\alpha} \cdot 2^{R^k - 1} \tag{4.10}$$

which is nothing but the inverse of Shannon's capacity formula, solved for the transmit power. Here $\gamma_t(y)$ denotes the interference at terminal y during time t. Due to the nature of the resource-allocation algorithm, no more than one of the nodes will transmit at maximum power. Because we assume a pathloss-dominant channel, it turns out that the longest or most-interefered hop will be the one which requires the maximum power.

4.5 Edge-Weight Update

Now that we have scheduled and routed one frame (referred to with generality as frame k here), we must update the NFIC to reflect this frame's impact on resource availability. The NFIC managed our application of dynamic programming for the first frame, now we demonstrate how it can also manage resource sharing among multiple flows. Specifically, as we update the NFIC in preparation for allocating the next flow, we must enforce the following rules:

- A terminal can transmit or receive from only one other terminal at a time.
- All terminals must remain half-duplex.
- Interference caused by a new flow cannot reduce the rate of an existing flow by more than some tolerance ratio R_{Int}.

In the following discussion, suppose frame k follows a route through the nodes $\{n_1, \ldots, n_T\}$. The first constraint is the easiest to apply. We assign 0 to all edges departing from or arriving at an active node, as shown below. The assignment of $\mathrm{I}_{\otimes} = 0$ is not arbitrary; it guarantees that this edge will never be selected in the max operation during node updates, effectively making the link unavailable at this timeslot. The summation of I_t^k over all nodes will be 1 if there is an active edge arriving at or departing from node n_t, which causes zero to be assigned to the other edges. We update for all $y \in \mathcal{N} \setminus n_t$, $t \leq T_k$ to enforce the multiple-access and half-duplex constraints:

$$e_t^{y,n_t} \leftarrow e_t^{y,n_t} \cdot \left[1 - \sum_{z \in \mathcal{N} \setminus n_t} I_t^k(z, n_t)\right] \tag{4.11}$$

$$e_t^{n_t,y} \leftarrow e_t^{n_t,y} \cdot \left[1 - \sum_{z \in \mathcal{N} \setminus n_t} I_t^k(n_t, z)\right] \tag{4.12}$$

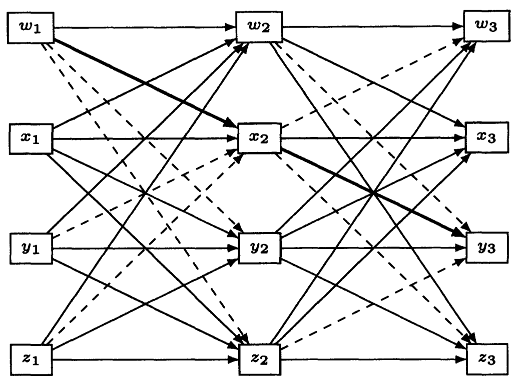
The first update specifies that if terminal n_t is receiving at time t, it cannot receive another frame. The second update is similar except for transmissions. Next, we enforce half-duplex constraints by assigning 0 to all edges arriving at a node in time t if it is transmitting in that time, and zeroing all edges departing from a node in t if it is receiving at that time. This guarantees a half-duplex system:

$$e_t^{y,n_t} \leftarrow e_t^{y,n_t} \cdot \left[1 - \sum_{z \in \mathcal{N} \setminus n_t} I_t^k(n_t, z)\right] \tag{4.13}$$

$$e_t^{n_t,y} \leftarrow e_t^{n_t,y} \cdot \left[1 - \sum_{z \in \mathcal{N} \setminus n_t} I_t^k(z, n_t)\right] \tag{4.14}$$

These updates are visualized in Figure 4.3, where the first set of updates (multiple access and broadcast) are visualized in the left pane and the second set (half-duplex) in the right pane.

All other edges with nonzero weights are available for subsequent frames. However, because we consider a fully interfering network, previously-allocated frames will be impacted by any newly-scheduled simultaneous transmission, meaning that interference must be taken into account. Given channel conditions, interference is purely



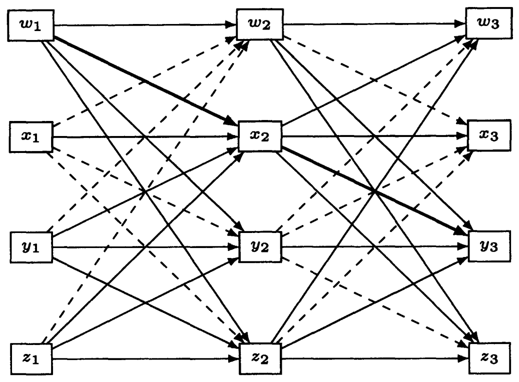


Figure 4.3: Example of enforcing constraints on $I(\cdot,\cdot)$ in the NFIC. On the left, dashed edges are those removed (weighted with $\mathrm{I}_{\otimes} = 0$) because a terminal may transmit and receive only one frame at a time. On the right, dashed edges are removed due to the half-duplex constraint, i.e. terminal w cannot receive in the first timeslot because it is transmitting, and terminal x cannot transmit because it is receiving.

a function of transmission power, which also exactly determines the size of a frame on a link. Managing feasible sizes of subsequent frames is therefore equivalent to managing interference imposed on frames already in the network.

We define a loss ratio $R_{\mathrm{Int}} \in [0,1]$, the fraction of data rate loss permitted on existing frames as a result of the introduction of new ones. Consider the leg from x to y at time t in frames k's path, which is the most recently allocated frame. This leg operates at rate R_t^k, which would drop to $\widehat{R}_t^k$ in the presence of the new frame $k+1$:

$$\begin{aligned} R_{\mathrm{Int}} \cdot R_t^k &= R_t^k - \widehat{R}_t^k & (4.15)\\ &= \log_2\left(\frac{1+\frac{P_t(x)}{N_o+\gamma_t(y)}D_{x,y}^{-\alpha}}{1+\frac{P_t(x)}{N_o+\hat{\gamma}_t(y)}D_{x,y}^{-\alpha}}\right) & (4.16)\end{aligned}$$

where $\gamma_t(y)$ is the interference frame k is experiencing at terminal y from other existing frames $\{1,\ldots,k-1\}$, and $\hat{\gamma}_t(y)$ is the interference it would experience if another frame is added to the network at time t. This equation can be manipulated to find the maximum tolerated interference $\hat{\gamma}_t(y)$ in terms of frame k's allocated rate and the tolerated loss fraction R_{Int}:

$$\hat{\gamma}_t(y) = \left[P_t(x) \left[2^{-(R_{\text{Int}} \cdot R_t^k)} \left(1 + \frac{P_t(x)}{N_\text{o} + \gamma_t(y)} \right) - 1 \right]^{-1} \cdot D_{x,y}^{\alpha} - N_\text{o} \right]^{+} \tag{4.17}$$

By scaling the parameter R_{Int}, we can select between full time-orthogonalization ($R_{\text{Int}} = 0$) and full interference ($R_{\text{Int}} = 1$). Calculation of an optimal value for R_{Int} is topologically specific, though such a value can be determined experimentally. We discuss this in more detail in the next section.

Since we have employed frame-wise decomposition, the next frame to be allocated in the NFIC will be considered in isolation; hence we know that at most one new link will be introducing interference to frame k. As such, we update NFIC edge weights in time t as if any edge is used in isolation; to do so, we calculate the maximum power any node $z \neq x$ could use without causing more than $\hat{\gamma}_t(y)$ interference at y. x is excluded from this calculation because it is already in use for frame k.

$$\hat{P}_t(z) \leq \hat{\gamma}_t(y) \cdot D_{z,y}^{\alpha} \quad \forall \quad z \in \mathcal{N} \tag{4.18}$$

Evaluating (4.18) specifies the maximum power any terminal may use at time t so as not to cause too much interference on frame k. However, there are $k-1$ other frames which must be considered; some of which may be closer to terminal z than frame k in this timeslot, further limiting z's power availability. We therefore update the power as the minimum of the previous maximum power or the maximum power as calculated above: $P_t(z) \leftarrow \min\{P_t(z), \hat{P}_t(z)\}$. We can now update all edges from

z at time t according to

$$e_t^{z,w} \leftarrow \log_2\left(1 + \frac{P_t(z)}{N_o + \gamma_t(w)} D_{z,w}^{-\alpha}\right) \quad \forall \quad z, w \in \mathcal{N} \tag{4.19}$$

which is repeated for all t in L^k. This redefines the space of feasible powers $\mathcal{P}$, as was illustrated in Figure 3.4. With the NFIC so updated, we are ready to repeat the procedure and allocate resources for frame $k+1$. In this way–allocating resources for one frame optimally, then removing them from availability before allocating the next frame–we construct the master resource allocation for all frames in the network. In particular, the schedule is constructed from the sequence of routes:

$$S = \bigcup_k L^k = \bigcup_{t,k} I_t^k \tag{4.20}$$

and the power allocation is the union of allocations at each timeslot, for each terminal:

$$\mathbf{p} = \bigcup_{t,x} P_t(x) \tag{4.21}$$

The resulting allocation is the approximate solution to (3.1) above, in which we have calculated $\mathbf{p}$ and S to maximize the minimum throughput among our flows. A graphical depiction of this algorithm is given in Figure 4.4. Starting with initial edge weights and frame and timeslot counters, we proceed through the steps discussed above: assigning ∞ to the source node, updating edge and node weights until the chart converges, and tracking back through the history to determine the allocation. Power levels are selected, leading to updated edge weights, which then becomes the initial condition for the allocation of the next frame.

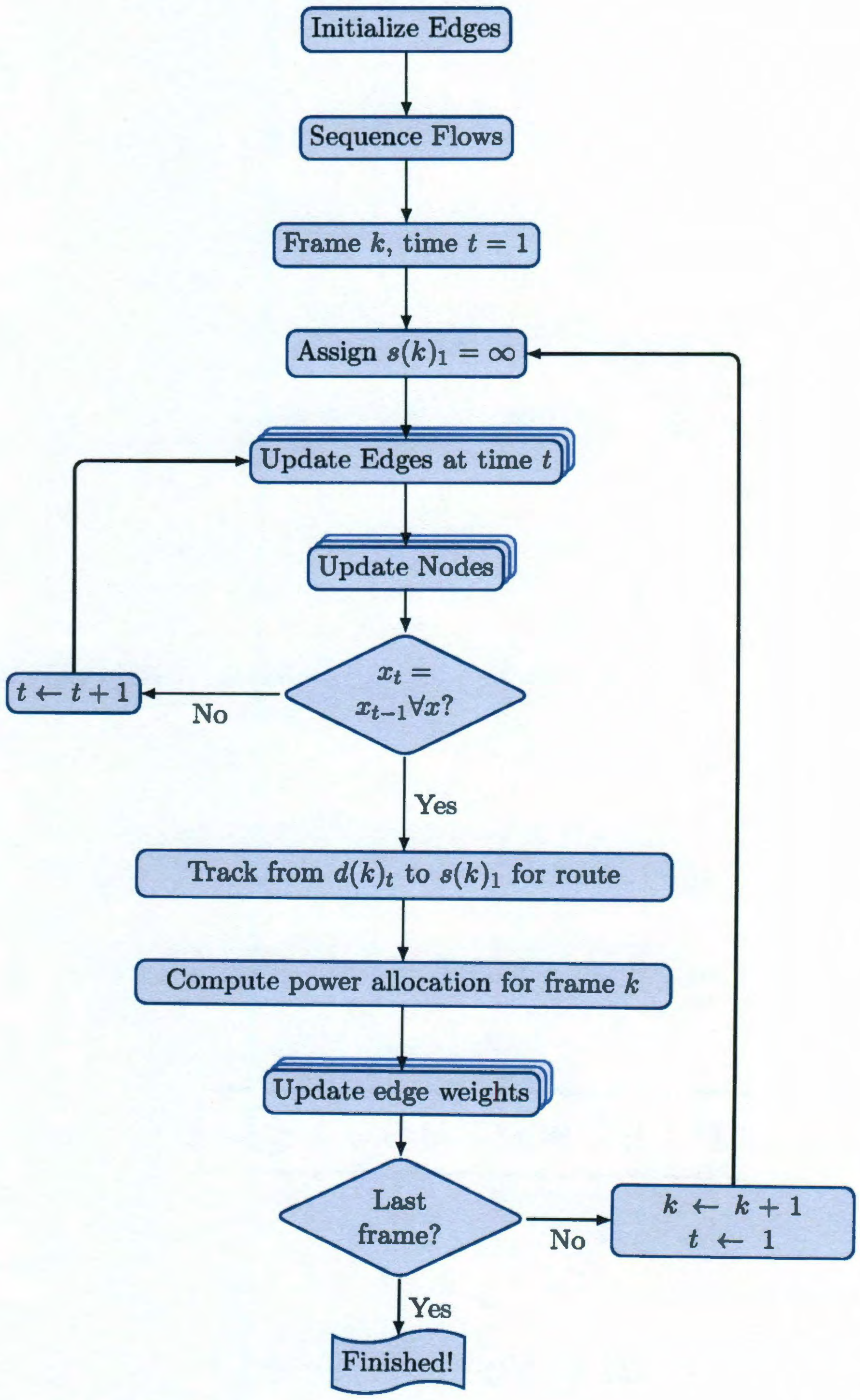


Figure 4.4: Flowchart of the general resource allocation algorithm.

4.6 Selection of R_{Int}

By scaling R_{Int}, we can select between forcing full time-orthogonalization ($R_{\text{Int}} = 0$) and allowing full interference ($R_{\text{Int}} = 1$). In the full-interference case, this corresponds to previously-allocated frames constraining the solution space in a *scheduling sense only.* By this we mean that some terminals will be unavailable, but only because they are handling a previously-allocated frame: not because they are too close to other to be able to transmit without excessively interfering. For small networks, this happens automatically; with very few terminals available, only a few frames may occupy the medium at a particular time.

Calculation of an optimal value for R_{Int} is topologically specific, though a mean value can be determined experimentally. We carry out such a simulation in both low and high pathloss environments, for three information flows of six frames each in networks with an average of 40 terminals. The results are reported in Figure 4.5. The value T_{gain} is the increase over orthogonal $R_{\text{Int}} = 0$ allocation. Lower values of R_{Int} do not permit enough interference to allow the algorithm to exploit all degrees of freedom, while allowing too much interference results in a "decoupling" of frame allocation such that previously allocated frames are not considered in subsequent allocations. Network isolation in the high-pathloss case means we require a lower R_{Int} to see good network performance.

Most interesting is the difference in peak locations across pathloss value. Since throughput is being optimized, transit time in the network–number of hops–is a dominating factor. With $\alpha = 2$, the availability of short-hop routes is contingent on allowing flows and frames to interfere, which is why we require a higher R_{Int} in that case. This makes a variety of routes available to frames in the same flow, mitigating the duplex penalty. With $\alpha = 4$, there are fewer short-hop routes, meaning that more frames following the same route, separated only by the duplexing penalty. Higher val-

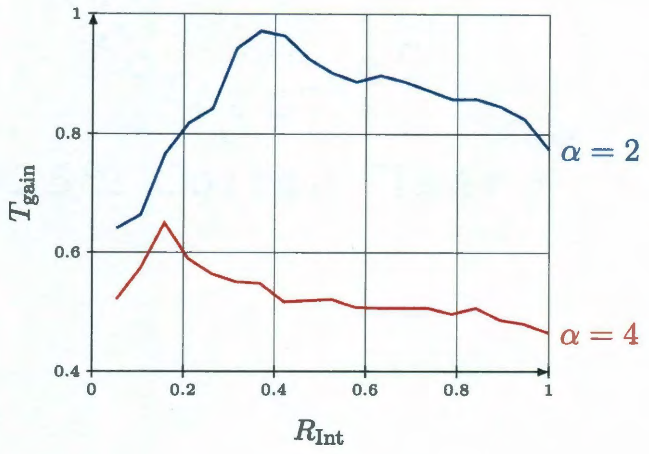


Figure 4.5: Selection of R_{Int} for a two different network environments. T_{gain} is the relative gain in throughput of allowing interference up to the fraction specified for R_{Int} over the interference-free allocation. Because network isolation is higher for $\alpha = 4$, less interference margin is required to see network gains.

ues of R_{Int} permit frames on the same route to excessively interfere, lowering overall throughput, hence the peak appears at a lower value of R_{Int}.

4.7 Computational Complexity

A strength of our approach is its low complexity. Here we calculate and analyze the computational and storage requirements of the NFIC.

4.7.1 Calculations

Assume the overall schedule uses T time slots. At each time slot, we must do node weight updates for all N nodes. In the worst case (fully interconnected or "full-interference" network), this requires N $\oplus$ operations, which here are minimizations. We must then do $N \cdot (N-1)$ edge updates for all outgoing edges. If we require T timeslots for the schedule, this becomes $2T(N^2 - N)$ node- and edge-weight updates. After computing the schedule, we must solve for the optimal power allocation, requiring exactly T computations, and then update all other the edges in the graph accordingly. Since there are N^2 edges in each time slot, this update takes TN^2 steps. We repeat this for all K frames scheduled, and arrive at the total

$$K(2T(N^2 - N) + TN^2) = KTN(3N - 2) \tag{4.22}$$

In the worst case, each of the K frames requires a hop through all N terminals. Here, the length of the schedule (total number of timeslots used) is on the order of KN, so that the complexity is $K(3N^3 - 2N^2)$, which is $O(N^3)$, considerably less than the $O(N^N)$ required for the exhaustive search, and also on the order of the best-known single-flow routing algorithms. The improvements we see over the exhaustive search result from two aspects of our approach:

- Decomposition by flow: This decoupling allows us to route each flow with minimal complexity, though it does introduce some sub-optimality. This step enables use to take advantage of dynamic programming methods.
- Dynamic Programming: routing individual flows on the NFIC using dynamic programming techniques gives the $O(N^3)$ improvement. This is a direct consequence of our problem possessing the Optimal Subproblem Property.

4.7.2 Storage Requirements

The N nodes of the NFIC at any time period are associated with a weight and a history variable, and each of the N^2 edges are also weighted. As such the memory requirements for T time periods are $T(2N + N^2)$, linear in the number of timeslots used and like the square of the number of terminals in the network. This is most easily seen when recalling the matrix-representation of $I_t^k(\cdot, \cdot)$.

The complexity discussed here is relevant only at the central authority, where the scheduling computation is being performed. In this discussion, we have assumed the availability of full network-state information at the central authority. With this full information we have been able to efficiently calculate solutions to the problem, albeit in using a decomposition approach.

4.8 Optimality of NFIC Solution

We now discuss the relative optimality of our decomposition-based approach. Recall that we already showed the optimality of the per-frame allocation in Section 3.5. As we mentioned in Chapter 1, the complexity of the joint problem precludes the computation of an optimal solution against which we can compare our decomposition approach. As such, we will consider two means by which to evaluate the relative optimality. First, we will examine how our technique performs in the "bounding cases" of very large or very small networks, environments in which the optimal solution is known. Second, we will construct an upper bound by calculating the one hop solution.

4.8.1 Extreme Cases

In the case of an extremely large network with many terminals and few flows, network isolation results in little interference among flows; hence in the asymptote, as $N \to \infty$ and $A \to \infty$.[1] For a given number of flows F, network isolation results in the separability of the overall optimization problem. In particular, $\gamma_t(x) \to 0$, decoupling the transmissions from one-another.

The result is a series of resource allocation problems for each flow, which is exactly the approach we take in our decomposition. Since network isolation is high, the allocation of one flow before another will have very little effect on the resources

[1]This corresponds to an ever increasing network region with a fixed density of terminals.

available for the next flow. As such, in this environment, we can claim that our approach is asymptotically near to the optimal solution. Note that this is true only for the allocation of one frame per flow, since multiple frames are now in close proximity and the notion of network-isolation does not hold.

Now, consider the case of a very small network, consisting of $N \leq 5$ terminals. This can be considered the "strong interference" case, where $\gamma_t(x)$ is large at all terminals. In this case, it is known [9] that TDMA is the optimal solution, since the spatial degrees of freedom are so tightly limited.[1] Our algorithm, with the appropriately tuned value of R_{Int}, reduces exactly to this case. After allocating resources for the first frame, it will make virtually all network resources unavailable for the timeslots in which that frame is in the network. Following that, the next frame will be allocated a similar manner. The result is the optimal, TDMA solution.

Consideration of these cases leads us to feel confident in the optimality of our approach–at either extreme of the network architecture, our technique results in the optimal allocation. As yet undefined, however, is the performance of our scheme in the "middle" region: relatively large networks, where interference must be managed in both space in time. In an effort to understand the optimal solution in this case *without* computing it explicitly, we consider a relevant bound.

4.8.2 Bounds

While our technique is near-optimal in the boundary cases, it is in cases where interference is most problematic–large networks with many flows–that we cannot evaluate optimality directly. Cutset bounds may be calculated for our network, but their generality does not impose the scheduling and routing constraints we consider, resulting

[1]Note that as mentioned in [9], TDMA is *not* optimal in low SNR environments. Here we consider terminals sufficiently near to one another as to be termed "high SNR".

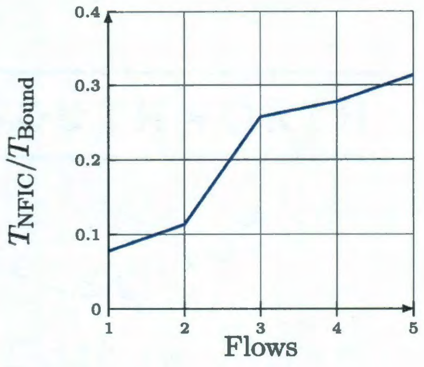


Figure 4.6: We show the difference between NFIC allocation and the one-hop bound described in Section 4.8.2. As the number of flows in the network increases, our allocation approaches the 1-hop bound.

in a bound too loose to be of use[1]. This routing constraint inspires the bounding result we present in Figure 4.6. Here, we bound the sum throughput for our flows by optimizing the first hop from their sources. A consequence of the Optimal Subproblem Property is that if the first hop is optimized, no other allocation can possibly outperform it, hence this is an upper bound on performance. To establish this value, we search over all N^F possible first hops and solve the power allocation problem (which is non-convex) using a random-restart technique. To illustrate the complexity of even this problem, a network of $N = 40$ and $F = 5$, the parameters we use in the majority our simulations, has approximately 102 million possible first hops.

The result is shown in Figure 4.6, where we show the difference in NFIC data forwarding capacity compared to that established by the one-hop bound, assuming that all subsequent hops are *as good* as the initial one hop. Here we report the sum throughput ratio in bits/sec/Hz, where we assume that the optimal schedule is the same length as the NFIC schedule. Note that for only one flow being allocated, the optimal first hop is the bottleneck part of the route we choose 10% of the time; the bound is calculating throughput to *any* other terminal (i.e. there is no routing), hence

[1]Moreover, the cutset bound is combinatorial in the number of cuts which must be considered.

the result is always the nearest neighbor terminal. However, as we allocate resources for more flows, the rising interference in the network reduces the amount of data that can be transferred simultaneously in one timeslot. Because the NFIC manages interference efficiently, our throughputs rise as more flows are allocated. Although we cannot compute it directly, we conjecture that as the number of flows tends to infinity, we will converge to this one-hop bound. This will manifest as TDMA in high utilization cases, which is what the optimal solution will be when computing in the one-hop bound.

4.8.3 LINDO Global Solution

As discussed in Section 3.2, we can enlist the aid of a computational solver to produce resource allocations for these networks. We consider the network shown Figure 4.7, with $N = 5$ and $F = 2$. As we mentioned earlier, the computational limits of the servers implementing the solvers intentionally restrict our problem size, so we model only three timeslots: $T = 3$.

Our max-min throughput problem was given to the LINDO Global solver exactly as presented in Section 3.2, which then worked to return a guaranteed-optimal solution. Unfortunately, even in this limited network and with a relatively small number of system variables, the computational resource ceiling was encountered with every run of the solver. As such, the solution we present below is *not* guaranteed by LINDO to be an optimal result. It is, however, on the gradient towards the optimal solution and can be considered better than any previously processed allocations. In this example, the computational solver evaluated 976,748 different solutions prior to timing-out. By comparison, the NFIC technique required 15 nodes, 50 edges, and 140 update operations. The solution is calculated virtually instantly.

In Table 4.1 we report the solution returned by our NFIC, comparing it to the

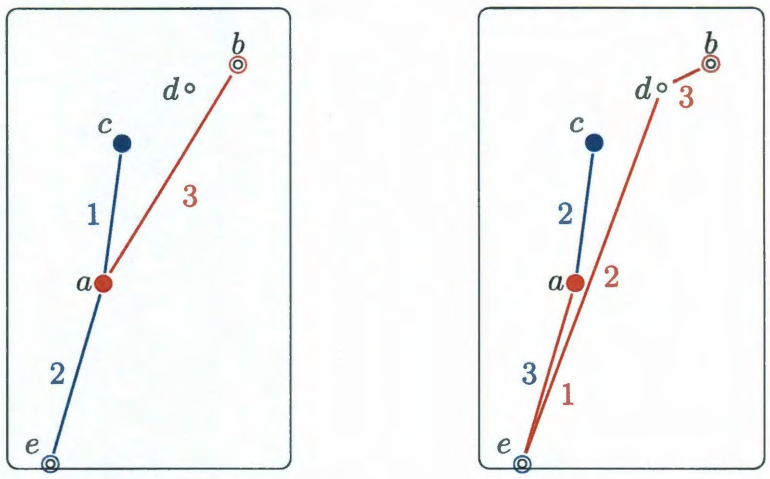


Figure 4.7: The small network we address using the LINDO Global solver. Here, we limit the search space to 3 timeslots. Even so, we exhaust the computational limits of our equipment. On the left is the NFIC solution, on the right is the LINDO solution.

LINDO solution reported in Table 4.2. We identify which link is active in each timeslot, and what the rate of the frames are. In the NFIC solution, all constraints are met: continuity of route and also matching of rate across hops. While on the gradient, the LINDO result violates the rate matching constraint–by orders of magnitude for both frames. The integer program is still within the solution space, since continuity of routing, duplexing, and source-destination requirements hold. However, the suboptimality of this solution is obvious: Frame II first begins by moving *away* from its destination before turning around.

This small, seemingly trivial example highlights the extreme difficulty in computing solutions to this class of problems. We therefore abandon any further efforts at computing a guaranteed-optimal solution, prefering instead to evaluate our methods on wide area networks against other low complexity techniques.

	Frame I	R_t^1	Frame II	R_t^2
$t=1$	$c \rightarrow a$	0.201	-	-
$t=2$	$a \rightarrow e$	0.201	-	-
$t=3$	-	-	$a \rightarrow b$	0.11

Table 4.1: Routing solution resulting from the NFIC method.

	Frame I	R_t^1	Frame II	R_t^2
$t=1$	-	-	$a \rightarrow e$	0.0468
$t=2$	$c \rightarrow a$	0.004	$e \rightarrow d$	0.0467
$t=3$	$a \rightarrow e$	0.04	$d \rightarrow b$	0.40

Table 4.2: Routing solution resulting from the LINDO Global system. Since LINDO did not converge, this solution has no guarantee of optimality.

4.9 Simulation Results

In the preceding sections, we have presented a low complexity algorithm to perform resource allocation in large networks. Here we show the performance of our techniques relative to other low-complexity approaches, and also present some network-layer consequences of the NFIC allocation algorithm.

4.9.1 Scheduling and Interference Constraints

To illuminate the effects of the constraints on terminal use $I(\cdot,\cdot)$ and also the interference management control captured by R_{Int}, we present an example allocation for one frame from each of three flows in Figure 4.8. The lines represent active links, enumerated by timeslot. For example, the green flow is active in timeslots $\{1, 2, 5, 6\}$. The longest flow (dotted blue) is allocated first, and follows a more or less straight-line path from its source to its destination. After that allocation, constraints on $I(\cdot,\cdot)$ begin to bind, preventing the second (dashed red) flow from using any terminals occupied by the blue frame. For the red flow, the interference constraint is more binding; because as the blue frame is making its way 'south,' the red frame would have to lower its power if it took the direct path towards its destination, to avoid interfering with the blue frame. Instead, the NFIC algorithm determines that a better route is a detour to the left, which allows the red frame to use higher power and achieve better throughput–spatially avoiding the blue frame.

The allocation for the third flow (solid green) demonstrates the scheduling constraints on $I(\cdot,\cdot)$ binding. Because terminal x is handling the red frame at times 3 and 4, the sum in constraint (2.7) evaluates to 1 for both of those timeslots. As a result, the green frame must remain in terminal x's memory while the red frame occupies its radio resources, before finally being transmitted in timeslot 5. Relating to the NFIC, edges into and out of x were weighted with $\mathbf{I}_{\otimes} = 0$ during timeslots 3 and 4.

4.9.2 Throughput and Power Gains from NFIC Allocation

To compare our scheme with another low-complexity scheme, we choose a round-robin (RR) allocation in which each flow is given one dedicated timeslot to transmit data directly between its source and destination at full power P_{Tx} (recall that because our network is fully connected, this is always possible). We consider networks for which $A = 20$ and terminal intensity is $\lambda = .1$, giving roughly 40 terminals per 2-D Poisson realization. For each network realization, we compute the flow throughputs T_{NFIC} and T_{RR} using the two allocation techniques. The relative gain is calculated as:

$$T_{\text{gain}} = \frac{1}{F}\frac{T_{\text{NFIC}} - T_{\text{RR}}}{T_{\text{RR}}} \tag{4.23}$$

Note that the throughput variables are vectors, so the subtraction and division should be viewed as elementwise operations. Because the NFIC allocation removes resources only from those terminals already used by another frame and only *scales* resource availability at other terminals, simultaneous scheduling is possible subject to

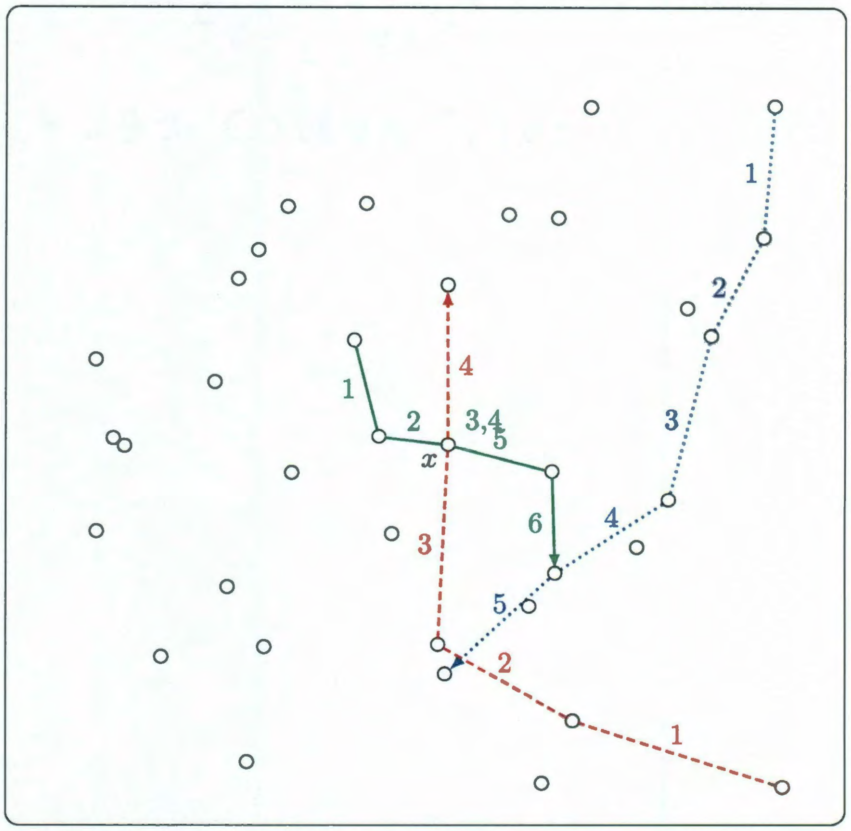


Figure 4.8: Example NFIC resource allocation for three flows. The blue dotted flow is allocated first, and proceeds directly to its destination. The red dashed flow is allocated second, and proceeds on a doglegged path to the left to avoid interfering with the blue frame. Duplexing constraints force the solid green flow to remain in storage at terminal x during timeslots 3 and 4 while waiting for the dashed flow to pass.

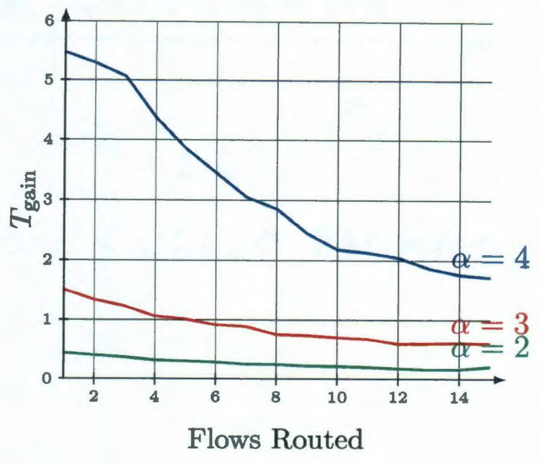


Figure 4.9: Throughput gains as a function of flows for a variety of pathloss parameters.

interference constraints. As such we simulate 1000 random topologies for a variety of network loads (number of flows), to study how simultaneous scheduling opportunities can improve network performance. We report our results in Figure 4.9, for a variety of network isolation levels (pathloss values). Note that for low isolation ($\alpha = 2$), NFIC and RR perform roughly equally for all traffic loads. This suggests that simultaneous scheduling–while permitted by the scheduling constraints–is being prevented by the interference constraints. As the pathloss factor increases, opportunities for simultaneous network use increase, and NFIC scheduling begins to outperform RR allocation considerably, with up to five times higher throughput achieved under light traffic loads.

We also compare power used by terminals in the RR and NFIC schemes. Figure 4.10 shows the extra power required by the RR allocation to achieve the same throughput as determined by NFIC allocation, normalized by the NFIC power required and reported as an average per flow. Note that in high pathloss especially, NFIC allocation is much more efficient–this is because the NFIC exploits *multihop advantage* to help frames overcome high pathloss over long distances.

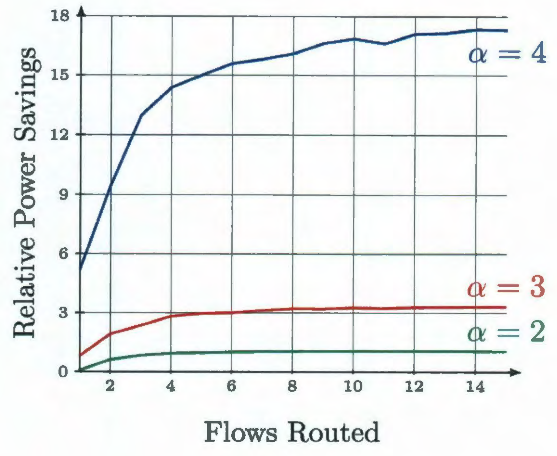


Figure 4.10: Normalized additional power required per flow by RR allocation over NFIC allocation for the same throughput.

4.9.3 Simultaneous Allocation and Intra-Flow Interfence

Simultaneous allocation can occur for frames in the same flow, provided the scheduling constraints remain satisfied. The fully connected nature of our network means that these frames will interfere with each other, an effect we term *intra-flow* interference. This is shown spatially in Figure 4.11, where frames from a single flow follow two different routes to the same destination. Duplexing constraints permit source s to transmit in all timeslots, but multi-hop terminals x and y can only serve one frame for every two timeslots. However, because they are spatially separated, they may be active at the same time: significantly increasing throughput. In this example with moderate isolation ($\alpha = 3$), the route multiplicity increased throughput by 20%.

We study this on a larger scale in Figure 4.12. Here, NFIC scheduling algorithms run for three flows, in a network with a mean of 40 terminals. We count the mean number of routes frames use between source and destination, and plot that against the mean throughput for all flows. Note that as route multiplicity increases, the duplexing penalty is mitigated and interference management becomes easier, hence all frames may be transmitted at higher rates. Data for low number of routes is

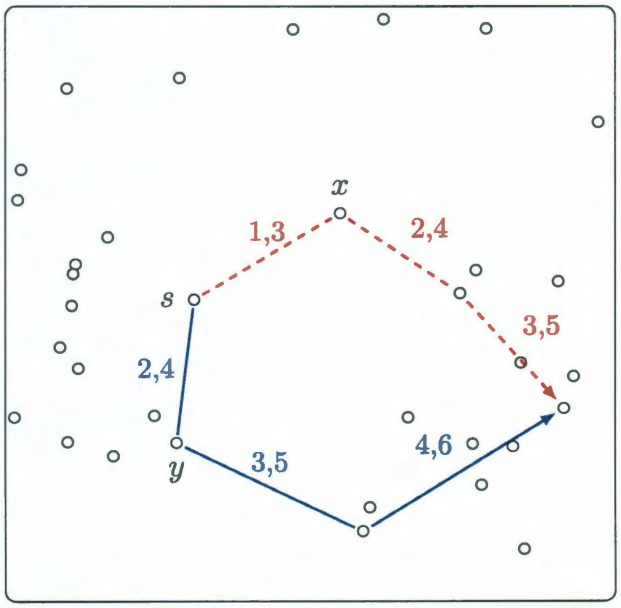


Figure 4.11: Two routes between the source and destination eliminate the half-duplex penalty. In this case, route multiplicity increased throughput 20%. Numerals indicate the timeslots in which each link is active.

sparse because, in networks with an average of 40 terminals, NFIC allocation will almost always identify multiple routes for each flow. This illustrates an important consequence of NFIC allocation: that, in contrast to conventional routing schemes which establish a single route for the frames in a flow, the spatial-temporal network representation in the NFIC allows us exploit more available resources in the network, to achieve a higher throughput. This is further demonstrated in Figure 4.13, where we show how the number of routes a given flow uses increase with resource competition: as more flows require access to the network, NFIC allocation routes frames in different ways to maximize spatial and temporal reuse. In lightly loaded networks with few flows, route diversity is low, but it increases considerably with network load.

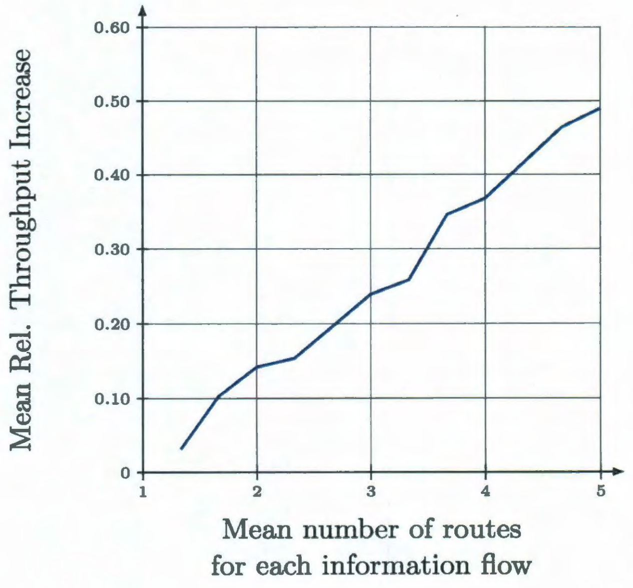


Figure 4.12: Relative throughput gain over single-route allocation for networks with 3 flows of 50 frames each. The number of routes available to each flow is artifically constrained to be in $\{2, 3, 4, 5\}$, and the mean across all three after NFIC allocation is reported here. As frames are permitted to take more routes through the network, mean throughput increases.

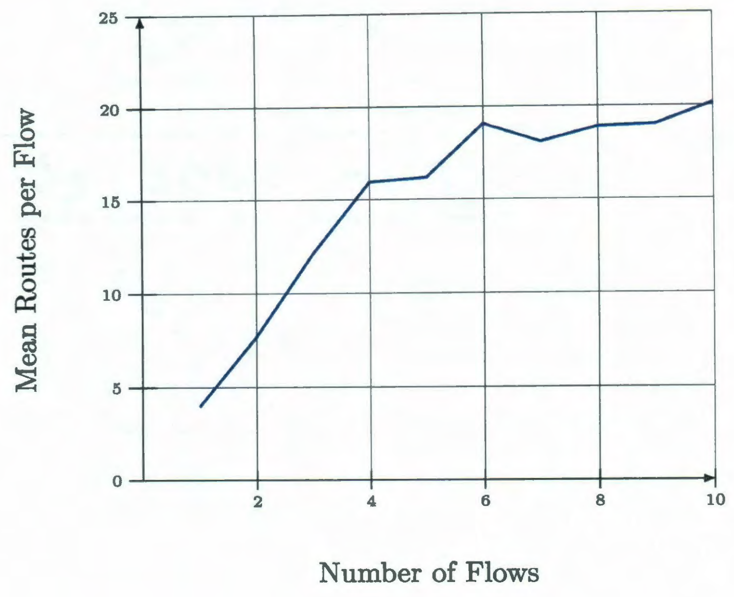


Figure 4.13: Number of routes frames take as a function of flows in the network, where $A = 20$, $\lambda = .1$, and $\alpha = 2$. 50 frames are allocated for each flow. As the number of flows increases and competition for resources increases, NFIC allocation results in frames traveling through the network in an increasing variety of ways.

4.9.4 Permitting Interference

With this transmission power, we update all edges in the NFIC at time t. Note that when R_{Int} is small, (4.17) goes to zero, meaning that allowable power is zero also. This means that in the edge-update step, if a frame is being transmitted at time t, all other edges at time t are set to zero. This is not a result of the integer constraints but rather of the interference constraint. In the following, we will refer to this mode as *Interference-free Multihop* (IFM), because the NFIC scheduler is allowed several hops to schedule frames, but not more than one frame is allowed in the network at a time.

As we increase R_{Int}, correspondingly $P_t(z)$ rises as well, more so for higher pathloss. Similarly, as the distance between the node in question and the receiving node increases, so too does $P_t(z)$: links further away from the active link are made available

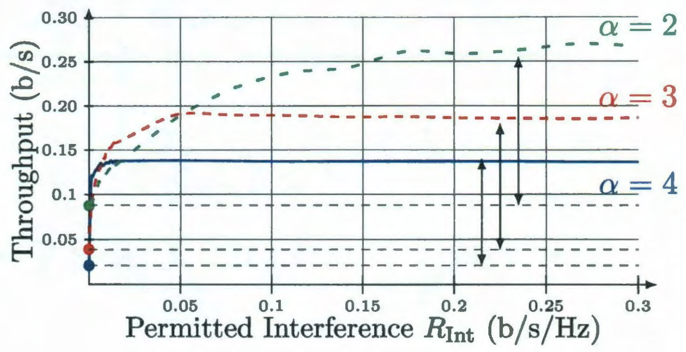


Figure 4.14: Throughputs increase considerably as interference is allowed in the network. Arrows indicate gains from permitting interference; when $R_{\text{Int}} = 0$, there is only one frame at a time in the network.

because the distance mitigates interference. In the case of streaming frames, as a frame makes its way to its destination, another may begin at the source without interfering excessively with those further down the route. We simulate the throughput gains possible with managed interference, and report the results in Figure 4.14. As we raise R_{Int}, the NFIC scheduling algorithm is permitted to exploit spatial reuse as described above. Throughput increases correspondingly, over the IFM throughput by the amount indicated by the arrows. Since P_{Tx} is constant for all pathloss parameters, we see higher throughput for lower pathloss, because data propagates further in freespace. We also note that very little interference must be permitted before the high-pathloss network reaches its best throughput; the reason is as given above: the effect of spatial isolation is much higher for large α, and so the distance multiplier in (4.18) is larger. Practically, this results in more frames simultaneously sharing the network while interfering very little.

CHAPTER 5

Distributed Resource Allocation

The network model and solution techniques developed in the preceeding chapters have all assumed a centralized approach. By this we mean that full information about the network is gathered, a solution is calculated, and the terminals are then given the schedules they implement.

In practice, this requires knowledge of all channels between all terminals, or N^2 channel coefficients gathered across a wide area. This precludes any form of practical deployment. In this chapter, we will study how to update our model and approach to make it amenable to a distributed implementation. Our goal is to establish the allocation without need for a single central authority.

In establishing a distributed allocation, we wish to preserve as much of the NFIC approach as possible. This will require some sense of centralization to execute NFIC algorithms, since the NFIC is–by design–a representation of a network, composed of several terminals. Our methodology will be to define *neighborhoods* of the network, and execute parallel NFIC allocations within each neighborhood. An example of a network partitioned into neighborhoods is drawn in Figure 5.1. Frames needing to travel across neighborhood boundaries will be handled using *gateway* terminals, as we will discuss in detail in this chapter.

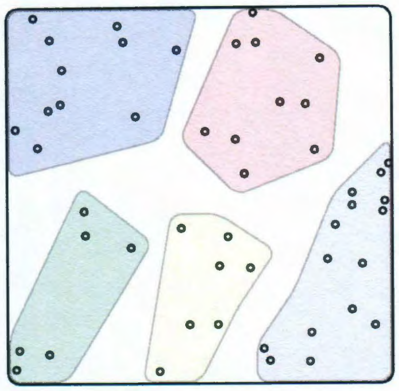


Figure 5.1: Coarsely drawn clusters in a network illustrate how groups of terminals can be considered to exist in a "neighborhood."

To begin, we will introduce a notion of *locality*, allowing us to form small groups of terminals which can collaboratively schedule and route frames using a common NFIC.

5.1 Forming Network Neighborhoods

5.1.1 Locality and Identification

The pathloss-based channel model naturally leads to a geographical notion of locality, based upon proximity of terminals. Terminals situated near one another are considered to be neighbors and are aware of each other's existence based on a simple beacon broadcast.

Definition 9. *For terminal x, the set of* neighbors $\mathcal{N}_x$ *are those which can receive data at a rate no less than $\bar{R}$ from x in the absence of interference.*

This corresponds to a maximum distance $D_{\bar{R}}$ between x and a neighbor y as

$$D_{\bar{R}} = D_{x,y} \le \left[\frac{P_{\text{Tx}}}{N_o \cdot (2^{\bar{R}} - 1)}\right]^{\frac{1}{\alpha}} \tag{5.1}$$

The set of neighbors formed in this way is therefore

$$\mathcal{N}_x = \left\{ y \in \mathcal{N} \;\middle|\; D_{x,y} \leq \left[\frac{P_{\text{Tx}}}{(N_o + \gamma_o) \cdot (2^{\bar{R}} - 1)} \right]^{\frac{1}{\alpha}} \right\} \tag{5.2}$$

which will increase with transmit power, and decrease with increasing rate requirement $\bar{R}$ or pathloss parameter α. Here γ_o is a (possibly zero) interference margin. Note that for a non-neighboring terminal $z \notin \mathcal{N}_x$, a transmission from z at time t will contribute to interference $\gamma_t(x)$ at terminal x.

This value of $\bar{R}$ is a parameter in our model, and defines a critical element of the overall problem: the cluster size. As more and more terminals are included in the cluster, the NFIC solution space increases; hence increasingly efficient allocations are found. By contrast, smaller clusters admit fewer solutions, leading to a suboptimal result.

We assume that each terminal is equipped with a unique identification symbol, which could be the last 4 digits of a MAC address. Critically, we require that the identification symbols possess a partial ordering, so that we can say $a \leq b$ and $b \leq c$ implies that $a \leq c$. Throughout this thesis, we have identified terminals using the letters of the alphabet; the ordering we choose to apply is the sequence in the alphabet, hence $a \leq b$.

5.1.2 Clustering Algorithm

Together with the neighbor $\mathcal{N}_x$ definition above and the terminal ID numbers, we are ready to execute a clustering algorithm over the network to form neighborhoods. Since we seek to form local NFIC's within a neighborhood, we must define these such that they do not overlap–the clustering must perfectly partition the network terminals. This is because collisions could occur if terminals are listed in multiple NFIC's, since one cluster may view a terminal as being available in a particular timeslot while

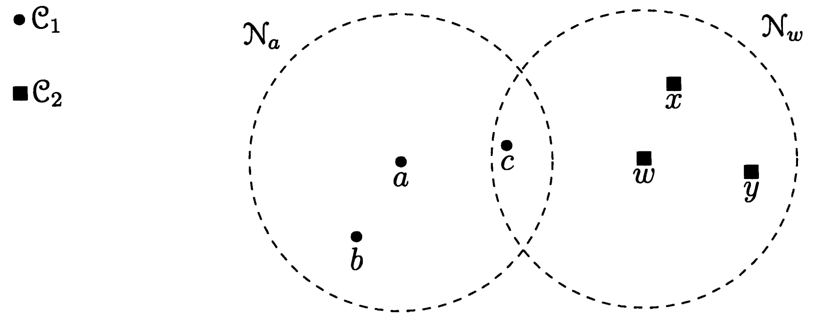


Figure 5.2: An example network and neighbors for terminals a and w. Note that terminal c is a neighbor of both a and w, but will join the cluster of terminal a since $a < w$. The resulting clusters are $\mathcal{C}_1 = \{a, b, c\}$ shown as solid circles and $\mathcal{C}_2 = \{w, x, y\}$, shown as squares.

another cluster has already allocated its use. An example of this appears in Figure 5.2, terminal c neighbors both terminals a and w.

We term the neighborhoods *clusters* $\mathcal{C}_i$, and there will be C of them. We require that a terminal be a member of exactly one cluster: $x \in \mathcal{C}_i \Rightarrow x \notin \mathcal{C}_j, \forall\, j \neq i$.

The development of clustering techniques has been widely studied, see [101] and [109] and references therein. We choose to employ the distributed clustering algorithm developed in [12]. The algorithm is as follows:

1. Using a beaconing technique, all terminals identify their neighbor sets $\mathcal{N}_x$.

2. If a terminal has the lowest ID number in its neighbor set, it sends a broadcast message to all terminals in the neighbor set inviting them to join a cluster.

3. If a terminal receives more than one invitation, it accepts the broadcast invitation from the terminal with the lowest ID. This acts as a tiebreak, such that terminals with many neighbors remain in just one cluster. In Fig. 5.2, both a and w broadcast invitations, and c responds to a after hearing both invitations.

The result is a set of C disjoint clusters of terminals $\mathcal{C}_i$ such that $\bigcup_{i=1}^{C} \mathcal{C}_i = \mathcal{N}$, as shown in Figure 5.2. Within $|\mathcal{C}_i|$ stages of channel sounding, all terminals in the

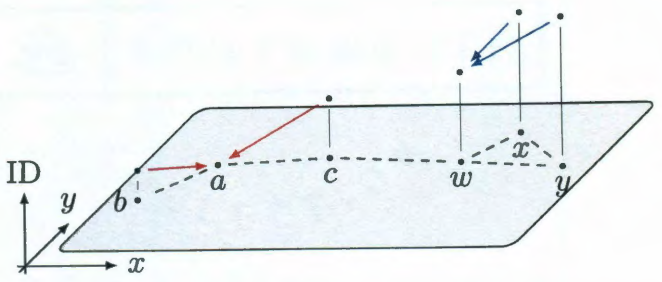


Figure 5.3: The clustering algorithm can be regarded as local attraction, where terminals naturally "fall" into clusters with their neighbors of lowest ID number.

cluster are aware of all channels. Clusters are regions of locally complete topological information, where any terminal can transmit to any other at a rate dictated by (2.13). We can therefore consider a cluster to be a *clique*. Within a cluster, it can be shown that any two terminals are separated by a maximum of two hops at rate $\bar{R}$.

Given that rate $\bar{R}$ defines a radius $D_{\bar{R}}$, the average number of terminals in a cluster will be $\lambda\pi D_{\bar{R}}^2$, a consequence of the Poisson distribution governing the placement of our terminals.

One can think of the clustering algorithm as a process of local attraction, where terminals are attracted to their neighbor with the lowest ID number. This is illustrated in three dimensions in Figure 5.3, where the height of each terminal above the plane corresponds to its ID: letters towards the end of the alphabet are higher than those near the beginning, and the downward arrows to the clusterheads show the local attraction. From this intuition it is clear that for a given topology of terminal ID numbers, the algorithm will always return the same cluster set. However, if the terminal locations remain constant but their ID numbers change, the resulting cluster definitions will be quite different. An example is shown in Figure 5.4, where the terminals in the same physical network have been randomly reassigned their ID numbers. The resulting cluster assignments are quite different.

This has profound implications on the corresponding resource allocation: in the

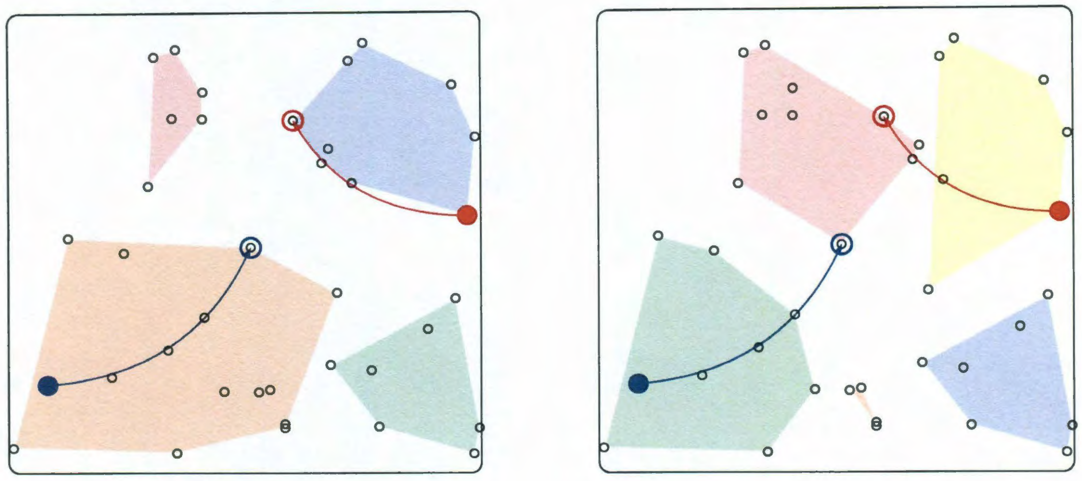


Figure 5.4: Different clustering outcomes as a result of changing terminal ID values. In this example, the clustering is a critical element in the performance of resulting allocation; in the clustering on the left, each flow is handled entirely within a cluster, so the NFIC will handle the entire allocation from source to destination. In the clustering on the left, each frame will be handled by more than one NFIC.

case of the clustering on the left, each flow is contained entirely within one cluster, so that a single NFIC handles allocation for each flow from the source to the destination. In the clustering on the right, each flow must cross cluster boundaries, requiring it to pass through a gateway terminal. This will result in a clearly less efficient solution than the clustering on the left.

Since we seek a general technique regardless of flow source and destination, we do not study how to optimize the clustering technique for a particular network topology. Doing so would require global knowledge of at least endpoint terminals, which we seek to avoid gathering.

5.1.3 Gateway Terminals

As mentioned above and illustrated in Fig. 5.2, some terminals will neighbor terminals in other clusters. These terminals, being locally aware of the neighbors in other clusters, will be able to act as *gateway terminals.* They will transmit and receive frames across cluster boundaries at a rate of at least $\bar{R}$, but do not have knowledge of the topology beyond their gateway partner in the neighboring cluster.

We define only one gateway link between each pair of clusters, so each cluster will have at most $C-1$ gateway terminals. They will be labeled $g_{i,j}$ if they are in cluster i but can transmit to the corresponding gateway in cluster j. As such terminals $g_{i,j}$ and $g_{j,i}$ are endpoints of the same gateway link. In selecting gateways from several possible pairs, those with the best channel are chosen. By this we mean that among possible gateway link candidates, we select the gateway link which can support the highest rate. This is illustrated in Figure 5.5, showing the clusters arising from the network in Figure 5.2–terminals c and w are the gateways $g_{1,2}$ and $g_{2,1}$. While not occurring in this example, it may be that a terminal acts as a gateway to more than one cluster.

We consider only one gateway link between clusters in this work, although there are situations where more than one may exist. In continuing work, we will study how to establish multiple gateway tunnels between neighboring clusters.

An interesting question to consider is what happens when the network becomes *disconnected*, i.e. when the network graph after applying the neighboring rule from (5.1) has portions which are not at all connected. In this case, there may be flows with sources and destinations which cannot be reached using links supporting at least $\bar{R}$. In this situation, we may declare a routing failure and abandon the allocation, since the network knowledge was "too distributed" for the distance-vector-routing approach (discussed in the following section) to completely fill in the routing tables at each cluster. In this work, we do not take this approach, instead gradually lowering $\bar{R}$ until the network connects. However, we must note that in doing so, we *do not* recompute cluster boundaries. This may be viewed simply as a relaxation of gateway link rate requirements until the network becomes connected.

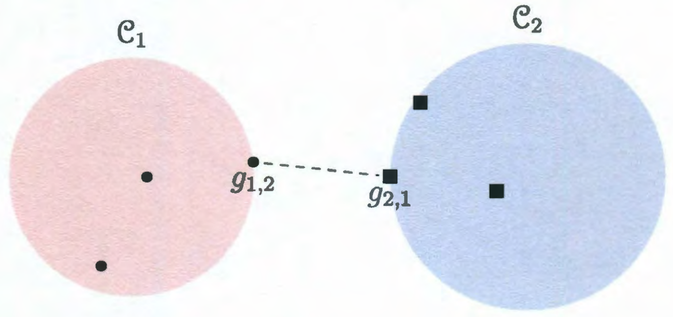


Figure 5.5: The result of clustering the network in Figure 5.2. Terminals c and w become gateways between the clusters.

5.2 Trans-Network Frame Routing

5.2.1 Distance-Vector Routing

To handle inter-cluster frames, those with source and sink in different clusters, we employ a distance-vector routing scheme [87, 19] similar to AODV, so that clusters know where to forward their frames into the broader network. Each cluster maintains a table of destination terminals and corresponding forwarding terminals. Initially it is populated with terminals within the cluster, but after a round of table-exchange between neighboring clusters, each cluster adds to its list the terminals in immediately neighboring clusters with the corresponding gateway as the forwarding terminal. Clusters again exchange tables, now adding terminals in other clusters two hops away. In the worst case of clusters arranged linearly, $C-1$ rounds of exchange are required for all clusters to fully populate their tables.

In the network of Figure 5.5, one round of exchange is sufficient. $\mathcal{C}_1$ will receive a list of terminals within $\mathcal{C}_2$, which will be entered into the table at $\mathcal{C}_1$ with $g_{1,2}$ as the forwarding terminal, and vice versa for $\mathcal{C}_2$. Suppose a frame originating at terminal a is bound for y: the routing table specifies that from a, $g_{1,2} = c$ is the intermittent gateway point. Two independent NFIC's handle allocation with the clusters, while the gateway terminals negotiate frames transiting between clusters.

5.2.2 Measures

In this work, we consider the very simplest metric in the routing vector: cluster count. Forwarding clusters are chosen based on this value, which may not (indeed, rarely is) optimal given the geographical topology. Other measures in the vector may be considered, including but not limited to:

- **Congestion.** A measure of how congested the next-hop cluster might be, perhaps as a function of the number of constituent terminals or number of flow endpoints.
- **Delay.** A measure of how many timeslots frames require to cross the next cluster, which may be a function of the size of the cluster.
- **Connectivity.** A measure of how "connected" the next-hop cluster is, possibly denominated by the number of neighboring clusters.

5.3 Mathematical Formulation: Distributed NFIC

The previous discussion has established the notion of a set of clusters which are able to do resource allocation on a local scale, and which are able to pass frames between each other while they are en route to their final destinations. In this section, we will detail the constraints binding the scheduling, routing, and power allocation decisions made within each cluster.

We continue to use the link-activation variable $I_t^k(x, y)$ which is 1 if frame k is being transmitted from terminal x to terminal y in timeslot t. Without loss of generality, suppose that frame k must transit through a sequence of clusters $\{\mathcal{C}_1, \ldots, \mathcal{C}_p\}$. Each of those clusters will locally allocate resources to frame k, determining a route and schedule between its local arrival terminal (which is either $s(k)$ or a gateway) and its local departure (either the $d(k)$ or a gateway). Notationally, frame k arrives at cluster i at timeslot $t_i^a(k)$ at terminal $s_i(k)$, and departs it from terminal $d_i(k)$ at timeslot $t_i^d(k)$. Referring to the frame traveling from a to y in Figure 5.5, $s_1(k) = a$ and $d_1(k) = g_{1,2}$.

5.3.1 Constraints

Within each cluster, we have a set of locally enforced constraints, which are derived from those outlined in Chapter 2. These require the frame to be accounted for at all times (5.3), the half-duplexing of terminals (5.4), that the frame start at its local source (5.5) and reach its local destination (5.6). We also require continuity of route, (5.7).

$$\sum_{x,y \in \mathcal{C}_i} I_t^k(x,y) > 0 \quad \forall \quad t,k,i \tag{5.3}$$

$$\sum_{x \in \mathcal{C}_i \setminus y} I_t^k(x,y) + \sum_{z \in \mathcal{C}_i \setminus y} I_t^k(y,z) \leq 1 \quad \forall \quad t,k,i \tag{5.4}$$

$$\sum_{y \in \mathcal{C}_i} I_{t_i^s(k)}^k(s_i(k),y) = 1 \quad \forall \quad k,i \tag{5.5}$$

$$\sum_{x \in \mathcal{C}_i} I_{t_i^d(k)}^k(x,d_i(k)) = 1 \quad \forall \quad k,i \tag{5.6}$$

$$\sum_{x \in \mathcal{C}_i} I_t^k(x,y) = \sum_{z \in \mathcal{C}_i} I_{t+1}^k(y,z) \quad \forall \quad t,k,i \tag{5.7}$$

For frames transiting across gateway links, we must enforce the same duplexing and one-frame-per-timeslot constraints:

$$\sum_k \sum_{x \in \mathcal{C}_i \cup g_{j,i}} I_t^k(x, g_{i,j}) \leq 1 \quad \forall \quad t, j \neq i \tag{5.8}$$

$$\sum_k \sum_{y \in \mathcal{C}_i \cup g_{j,i}} I_t^k(g_{i,j}, y) \leq 1 \quad \forall \quad t, j \neq i \tag{5.9}$$

$$\sum_k I_t^k(g_{i,j}, g_{j,i}) \leq 1 \quad \forall \quad i, j, t \tag{5.10}$$

Since the size of a frame should remain constant throughout its journey, we require that for each frame k, the rate of each link in its route is the same:

$$\log_2\left(1 + \frac{P_t(x)D_{x,y}^{-\alpha}}{\gamma_t(y) + N_o}\right) = \log_2\left(1 + \frac{P_{t+1}(y)D_{y,z}^{-\alpha}}{\gamma_{t+1}(z) + N_o}\right) \tag{5.11}$$

where $\{x, y, z\}$ are in route L^k. To meet these constraints, we will solve for I_t^k and P_t. Determining I is an integer problem, while the interference terms γ in (5.11) result in a non-convex problem.

5.3.2 Distributed NFIC

We construct the NFIC as detailed in Chapter 3, though with consideration of terminals only within the same cluster. The sets $\mathcal{V}$, $\mathcal{E}$ are defined as before, using Shannon's equation for edge weights.

The locality of information is represented in the NFIC as an absence of edges. If terminals do not know of each other, an edge does not exist between them. Each cluster is fully connected, and so all nodes in the cluster are fully connected in the NFIC. The NFIC for the clustered network of Figure 5.5 is shown in Figure 5.6. Only the gateway terminals $g_{1,2}$ and $g_{2,1}$ have edges (shown in dark, broken arrows) crossing cluster borders. Using this data structure, we will solve for I_t^k and P_t such that the constraints above are met. Allocation will happen on a per-frame-per-cluster basis,

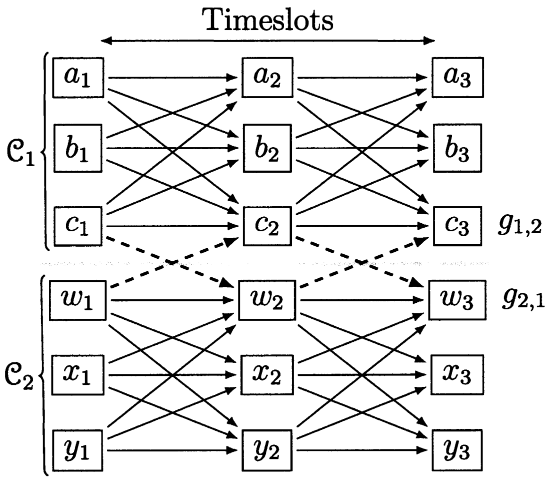


Figure 5.6: An NFIC for the two clusters of Figure 5.5: all terminals in the network are shown as nodes in the NFIC, but only gateway terminals c and w are able to communicate with each other.

such that each cluster allocates resources in parallel for one frame at a time. In this way, we construct the master allocation. We now detail this procedure.

5.4 NFIC Solution Technique

Each cluster uses the NFIC to execute a dynamic programming algorithm detailed in [73] and in Chapter 4, allocating one frame at a time in $O(|\mathcal{C}_i|^3)$ time. Each cluster will allocate resources for one frame, then the gateway terminals between clusters will negotiate frame transfer as required.

Consider cluster $\mathcal{C}_i$. A frame is chosen for allocation in a prioritized manner, in this order: 1) frames with sources inside cluster i, 2) frames with destinations inside cluster i, 3) frames in transit across cluster i. Suppose frame k is being allocated, with source terminal $s_i(k)$ and destination $d_i(k)$. If the frame originates within cluster i, i.e. $s_i(k) = s(k)$, then the NFIC node corresponding to $s(k)$ is assigned a weight of ∞ in timeslot 1, reflecting that the source would like to send as many bits as soon as possible. If $s_i(k)$ is a gateway terminal, then the corresponding node in the NFIC is assigned a weight corresponding to the frame's size in the timeslot the frame arrived at the gateway, $t_i^a(k)$.

The algorithm then executes as follows, where we summarize the details presented in Chapter 4: NFIC edges are updated with the minimum of either the source node weight or the edge weight, since a channel cannot carry more bits than capacity or more than exist at the source. NFIC nodes are updated with the weight of the largest incoming edge, since maximum data should propagate through the cluster. This repeats across timeslots, until all the node weights have stabilized, indicating that the data-maximizing routes between the source and all terminals in the cluster are known. We then look at the node corresponding to $d_i(k)$, and identify the maximum node value. Tracing the path back to the source gives us the throughput-optimal route and schedule through cluster i, which is appended to the frame's overall route L_k. This procedure ensures that (5.5), (5.6) and (5.7) are met for k.

This algorithm is executed in each cluster in parallel, for just one frame as determined using the priority sequence discussed earlier. At most there are $|C|$ allocations happening simultaneously.

5.4.1 Resource Update

Upon allocating frame k, cluster i updates its NFIC to reflect the resources consumed. For all terminals in the path L^k, NFIC edges into and out of the corresponding nodes are assigned value 0, because they cannot accommodate data without violating the duplexing constraint. To guarantee that frames allocated *after* frame k do not cause excessive interference on frame k, we must update the edges in the NFIC. We solve for the amount of interference a terminal receiving frame k can tolerate while not losing more than R_{Int} of rate:

$$\hat{\gamma}_t(y) = \left[\frac{P_t(x)}{\left[2^{-R^k \cdot R_{\text{Int}}} \left(1 + \frac{P_t(x)}{N_o + \gamma_t(y)}\right) - 1\right]} \cdot D_{x,y}^{\alpha} - N_o \right]^+ \tag{5.12}$$

from which we update the maximum power for all other terminals as

$$\hat{P}_t(z) \leq \hat{\gamma}_t(y) \cdot D_{z,y}^{\alpha} \quad \forall \quad z \in \mathcal{N} \tag{5.13}$$

which are then used to set edge weights for the next iteration of allocation. Note that in the distributed case, the calculation of interference

$$\gamma_t(y) = \sum_{x \in \mathcal{C}_i} P_t(x) D_{x,y}^{-\alpha} \tag{5.14}$$

is limited to elements only within the cluster $\mathcal{C}_i$, which is *not* necessarily the entire network.

5.4.2 Gateway Transfers

After a round of resource allocation in each cluster, frames for which $d_i(k) \neq d(k)$ are ready to be transfered across a gateway pair into their next cluster. The protocol is simple: if a gateway needs to transfer a frame, the receiving gateway agrees to receive at its earliest unoccupied timeslot. The gateways receive and transmit only when they are not otherwise occupied, ensuring that constraints (5.8)-(5.10) are met. The power expended in the transmission is determined such that the rate of the frame k is maintained, or maximum power is used and the rate of frame k is decreased. In determining when the gateway transfer takes place, each gateway terminal checks 1) that it is not being accessed locally and 2) that the transmitting gateway will not cause undue interference (specified by R_{Int} above) in its cluster. The earliest timeslot in which these two conditions are simultaneously met is then selected.

5.4.3 Parallel Allocation

Figure 5.7 shows the parallel allocation of two frames in the two cluster network of Figure 5.5 using our approach. The source for one frame is the destination for the other, so this is a case of a bidirectional flow. In the first round of allocation, each NFIC simultaneously allocates the frame to the gateway. This is shown in pane a), where the small arrow indicates the destination of the frame *for that NFIC.*

In pane b), we show the result of the gateway terminals negotiating the transfer. Note that for frames crossing in opposing directions, the half-duplex penalty delays one of them–in this case the orange frame. The allocation is completed in pane c), when NFIC allocation executes for a second time, simultaneously scheduling the frames into their final destinations.

This example shows only two clusters involved in the allocation and only one gateway terminal. However, the parallel nature of our technique means that any number of cluster can be doing such a simultaneous allocation, and then coordinate frame transfers across any number of gateway terminals. By constrast, the full-information solution would have a fully connected NFIC, and would have allocated the green frame from source to destination and then the orange frame also from source to destination. This solution may have allow the two frames considered here to reach their destinations in fewer hops and at a higher rate.

Note that in the simultaneous allocation, there is not an actual *transfer of frame data* happening; this is the computation of the allocation. Once the allocation has been computed, frames are formed and the network implements the schedule. Each cluster is aware only of the behavior of its constituent terminals for each timeslot, and is aware of when to expect frames arriving at its gateways and when its gateways

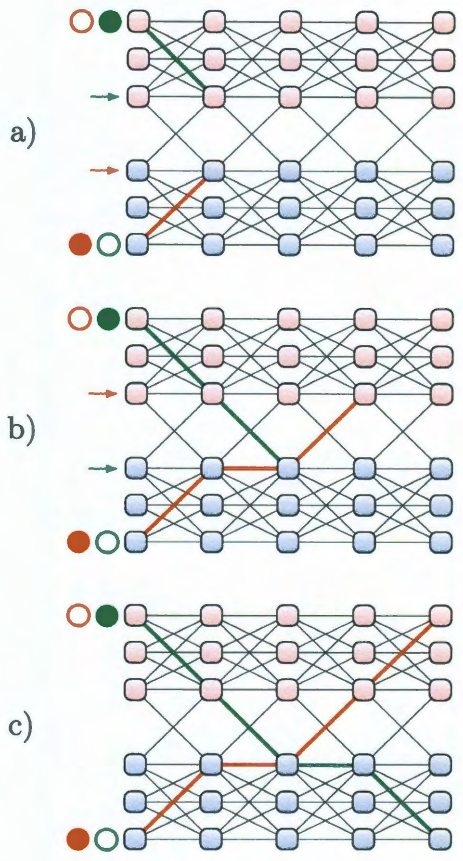


Figure 5.7: Allocation of two frames in two clusters using distributed NFIC techniques. Pane a) shows the allocation of each frame locally, towards its gateway link. Pane b) shows the result of the gateway negotiation, and pane c) shows the allocation of the frame to its destination.

should transfer frames to other clusters.

5.5 Results

The techniques discussed above address the distributed resource allocation problem. We now discuss how the optimality of the allocation declines as topology knowledge decreases. The baseline case we study is that of a single cluster, where all channels are known and one fully-connected NFIC is used to calculate the resource allocation.

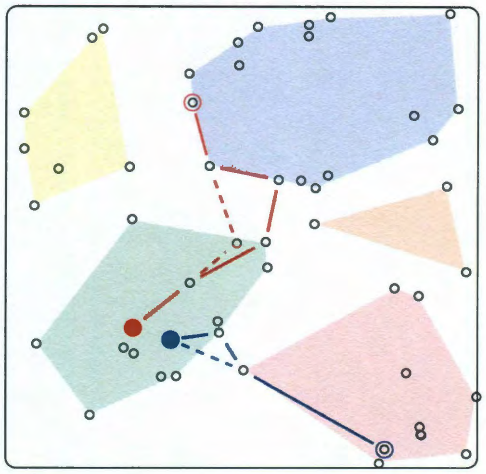


Figure 5.8: An ad-hoc network partitioned into five clusters with two information flows. Frame sources are solid circles, the sinks are open circles. The distributed solution is shown in solid lines, the full information solution in dashed.

5.5.1 Example

An example of local versus global allocation appears in Figure 5.8. Here $A = 20$ and $\lambda = 0.1$, and the network contains 53 terminals. There are two flows, with frames traveling from the solid circles to the open circles. Localizing the allocation resulted in five clusters, shown by the colored regions.

The distributed allocation is shown in solid lines, while the full-information solution is shown in dashed lines. In the distributed case, the frames take a less direct route to their destinations as a result of needing to cross cluster boundaries, using gateway links. Indirect routing lowers the performance of the network; in the full-information case, the throughput is 0.1 and 0.2 bits/sec/Hz for each flow, while the distributed case sees only 0.08 and 0.10 bits/sec/Hz. By way of comparison, 0.10 b/s/Hz corresponds to approximately 200 bits per frame under IEEE 802.11 network parameters.

In the case of the blue and red flows, the full-information solution is both more direct geographically and also shorter, since there are no extra hops through gateways.

However, the shorter routes require considerably more information to establish and schedule. This is a consequence of the clustering result in this network, which now study in more depth.

5.5.2 Effect of Clustering

In Figure 5.9, we show the different allocations resulting from clustering using different ID terminals. The locations of the terminals is held constant, but the ID values are randomly reassigned. This leads to a different clustering result, which in turn leads to a different allocation solution. In the clustering on the left, the throughput for the blue flow is 0.0521 b/s/Hz, and for the red flow, it is 0.0420 b/s/Hz. Here the bottleneck terminal is the gateway entry to the upper-left cluster, through which both flows must pass.

In the clustering on the right, the throughputs are lower: 0.0321 b/s/Hz for the blue flow and 0.0362 b/s/Hz for the red flow. The blue flow's throughput declines because it is using a longer route, but in the case of the red flow, it declines because there are longer hops in the route–in particular those active at timeslots 2 and 5. The bottleneck terminal here is squarely in the center of the upper-right cluster, which becomes the bottleneck because the red flow must avoid the blue flow as it enters the cluster on the southern end. The shortest-path through that cluster for the red flow is more southerly, but the red flow would experience delays because the blue flow is consuming resources there.

We also calculated the mean throughputs over different clusterings for this network: the mean throughputs over 100 different clustering realizations (topology fixed, ID numbers changed) are 0.0382 b/s/Hz and 0.0330 b/s/Hz for the blue and red flows respectively. From this, we can draw the conclusion that the clustering on the left is advantageous to both flows, while that on the right is suboptimal for each.

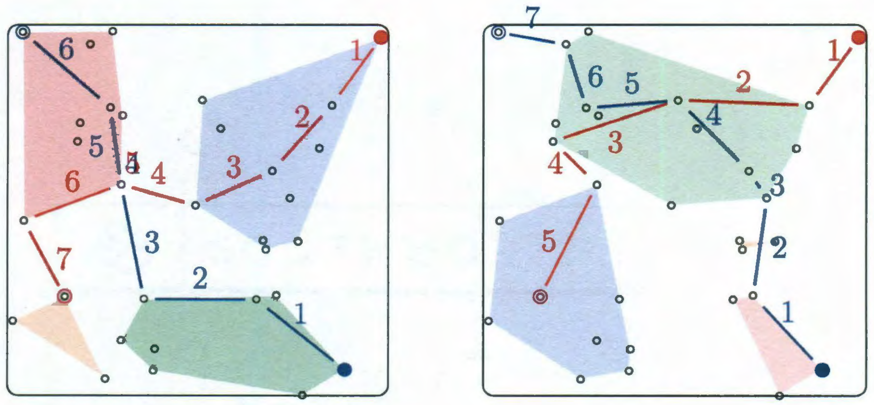


Figure 5.9: Here we show allocation for the same two flows under alternative clusterings. On the left, throughputs are 0.0521 and 0.0420 b/s/Hz, while on the right, they are 0.0321 and 0.0362 b/s/Hz.

5.5.3 Gains in Goodput

For the purposes of the following discussion, we must define how much *overhead* is expended to learn the channels and establish the allocation. We use a very general overhead model, in which we simply count the number of bits transmitted in this effort and ignore the beacons required to perform the clustering. We regard overhead as

$$B_{\text{OH}} = B_{\text{Channels}} + B_{\text{Schedule}} \tag{5.15}$$

which is the sum of the number of bits spent learning channels between terminals and the number of bits spent distributing the schedule. Recall that we have C clusters and within each cluster, where cluster i contains $N_{\mathfrak{C}_i}$ terminals. As such, we have

$$B_{\text{Channels}} = \sum_{i=1}^{C} N_{\mathfrak{C}_i}^2 \tag{5.16}$$

since we must learn all of the channels between all terminals in the cluster.

To distribute the schedule, we must send a table of T timeslots and N_{C_i} entries to each terminal. For simplicity, we consider the entries of the table to be a binary quantity specifying transmission state, akin to $I(\cdot,\cdot)$. This gives the total overhead as

$$B_{\mathrm{OH}} = \sum_{i=1}^{C} N_{\mathfrak{C}_i}^2 + \sum_{i=1}^{C} T \cdot N_{\mathfrak{C}_i}^2 \tag{5.17}$$

The number of data bits transmitted is a function of frame size. Following the execution of resource allocation, frames for flow f are of size R^f. If K frames are transmitted according to the computed allocation, the total transmitted data is

$$B_{\mathrm{data}} = \sum_{f=1}^{F} K \cdot R^f \tag{5.18}$$

We define *goodput* as the fraction of transmitted bits which carry user data rather than overhead bits. Formally,

$$G = \frac{B_{\mathrm{data}}}{B_{\mathrm{data}} + B_{\mathrm{OH}}} = \frac{B_{\mathrm{data}}}{B_{\mathrm{Tot}}} \tag{5.19}$$

This value is defined for both distributed and full-information systems. Overhead bits are those required to learn channels, exchange routing tables (in the distributed case), and send out allocations. Data bits are exchanged in frames once the allocation has been determined, where the number of such frames is set by the *network lifetime*: the time before the network topology changes, necessitating a recalculation of the allocation.

The gains in goodput are the consequence of local decision making; as $\bar{R}$ grows and therefore the size of clusters shrinks, the overhead required to establish the allocation also declines. Conversely, larger clusters require overhead converging to the full-information solution, which can be viewed as a single-cluster network. This is shown in Figure 5.10. For short network lifetimes (L), we see good performance with small

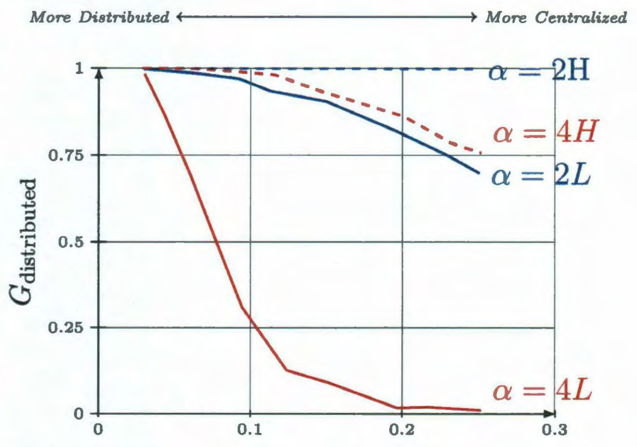


Figure 5.10: Goodput for the distributed case as a function of cluster size, for either 2000 (L) or 10^5 (H) frames. As network lifetime increases, goodput increases as overhead costs become negligible. With fixed transmission power, high-pathloss environments experience lower goodput, since both overhead and data transmissions are relatively more expensive

clusters and low pathloss, both of which lower the cost of overhead. As network lifetime increases (H), goodput is high in both pathloss regimes, even as cluster size grows. This results from one-time overhead cost becoming a diminishing term in the denominator of G. We now look at how the higher cost of full-information affects overall throughput performance.

With full information, the allocation algorithm has access to the complete solution space, and so is able to calculate schedules and routes to most optimally manage interference. As a result, the allocation computed from the full information will necessarily outperform the distributed result in a throughput-sense. However, the price of the overhead required to establish that solution makes the distributed alternative more attractive for short network-coherence times, as shown in Figure 5.11. Here, the total number of data bits is plotted against the number of data frames sent, such that *slope* of each curve is a measure of throughput: data bits per frame. Here it can be seen that for shorter coherence times, the distributed solution offers significantly

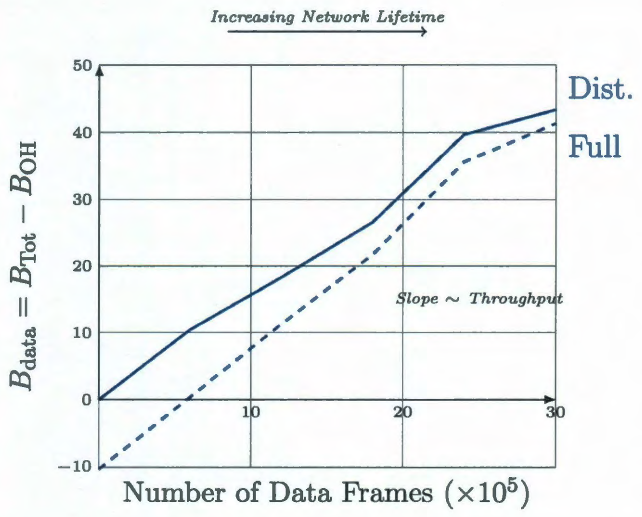


Figure 5.11: Transmitted bits minus overhead bits, is reported as a function of data frames for both full (single-cluster) and distributed (each cluster contains 20% of network terminals on average) allocations. Networks have a mean of 40 terminals. If the network lifetime is long, the optimality of the full information solution makes the overhead cost worthwhile.

better performance.

Of interesting note here is the large difference in required overhead, which appears at the shortest network lifetime. This is because in the full information case, there are a mean of 1600 channels which must be learnt, versus 320 in the distributed case. This is the order-of-magnitude difference borne out on the plot, highlighting the overhead savings in deploying a decentralized allocation. While not emphasized in this chapter, such a deployment also results in less energy being expended on channel discovery, preserving the battery lives of the devices in the network.

5.6 Remarks

We have shown how a centralized resource allocation scheme, which jointly computes schedules, routes, and power allocations, can be deployed in a distributed network. This allowed us to study the relative suboptimality of the distributed solution as a function of overhead. We saw that savings in overhead can be useful over short network lifetimes and in high pathloss, but can result in throughput-suboptimal allocations when the number of data bits is larger than the one-time overhead cost.

The techniques developed here are particular to the NFIC implementation, and should not be regarded as having any claim to optimality in decentralized resource allocation. Rather, we took the focus of how to manage the interference-scheduling tradeoff in environments in which full interference and scheduling knowledge is not known, and to study the extent to which such a lack of knowledge hinders overall network performance. This will become critically important as we move into our final area of study, how to deploy cooperative links in the large network. In this domain, interference is beneficial to one link (the cooperative link) while harmful to all others.

Chapter 6

The Cooperative Paradigm

We now turn to our final paradigm of study in large-network resource allocation: how to address multi-terminal links in wide-area networks. Until now, we have considered a point-to-point architecture on the physical layer, where the performance of that link is a function of power allocation $\mathbf{p}$ and the active transmitters as specified by $I(\cdot, \cdot)$.

In this chapter, we broaden our focus to include the *relay channel,* where a third terminal assists a point-to-point link by retransmitting a version of the original message. The relay therefore introduces interference into the overall network, which requires management at both the physical and MAC layers. Put concisely, this requires a combination of *scheduling* and *power control*: when to coordinate simultaneous relay transmissions, and at what power level so as to minimize interference at other network terminals.

Since the relay channel has only been briefly mentioned in the Introduction, we will devote the beginning of this chapter to a discussion of this architecture before showing how the NFIC framework can be used to allocate resources when cooperation is available.

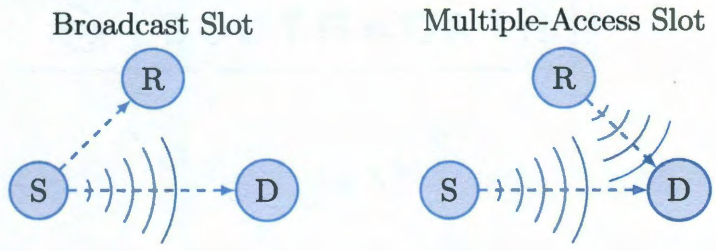


Figure 6.1: The classical three terminal relay network operating in isolation. In the broadcast (BC) slot, the source's transmission is received at R and D. The source and relay then transmit together to the destination in the multiple-access (MA) slot.

6.1 Background

The layered OSI network model for wireless communication has resulted in the focused study of fundamental signal-level interactions in small networks, while at the same time efficient medium-access techniques have been developed to manage time and space in large groups of terminals while abstracting away the physical layer. The need for increased network performance is necessitating a synergy between localized, advanced physical layer techniques–such as cooperative communications–and the medium access control (MAC) technologies charged with managing the broader network.

We will see that our NFIC approach is general to any information-theoretically characterizable multiterminal scheme, but we will restrict our attention in this chapter to the *relay channel* [98, 14], shown in Figure 6.1. Throughout, we will describe the relay system in terms of its three terminals and its two phases. The three terminals are uniquely the source, relay, and destination–we note the uniqueness because the achievable rate of the cooperative link will change if the terminals change roles. The first of the two phases is the *broadcast* (BC) phase, in which the source alone transmits, but both the relay and destination receive. Note here that two atomic, point-to-point links are active, one between the source and relay, and another between the source and destination. The second phase is the *multiple access* (MA) phase, in which the source and relay transmit simultaneously to the destination. The destination, because it receives three frames over two timeslots, is then able to receive more information from the source than if the source alone had transmitted. We will regard this coordinated action as a single physical layer unit, which can be used by the MAC layer to aid data transfer in the larger network.

The information-theoretic characterization of the three-terminal cooperative network has shown that such a structure offers considerable improvement in link performance [14, 91], confirmed by recent hardware deployments [78, 80]. Incorporating cooperative links into a larger network with multiple flows requires fundamentally rethinking the MAC scheme: how does one route frames and schedule their transmissions so that cooperation benefits performance, while simultaneously managing the increased interference caused by cooperative links? As we've seen, the joint scheduling-routing-power allocation problem is NP-hard even in the point-to-point case. Adding cooperation serves only to increase the complexity of the problem.

Until now, scheduling and routing algorithms for wireless networks have treated point-to-point links as the fundamental unit of allocation: routes are composed of them and a schedule specifies which of them are active at a given time. By definition, cooperative links are comprised of several point-to-point links operating in a highly coordinated manner *at the signal-level* over two timeslots. Schedulers must now account for temporal dependency of the links, and routing algorithms must be able to distinguish the increased throughput of a cooperative link from those of point-to-point links. In calculating routes and schedules, the algorithms must manage interference–by managing terminals' transmission power–such that cooperative links

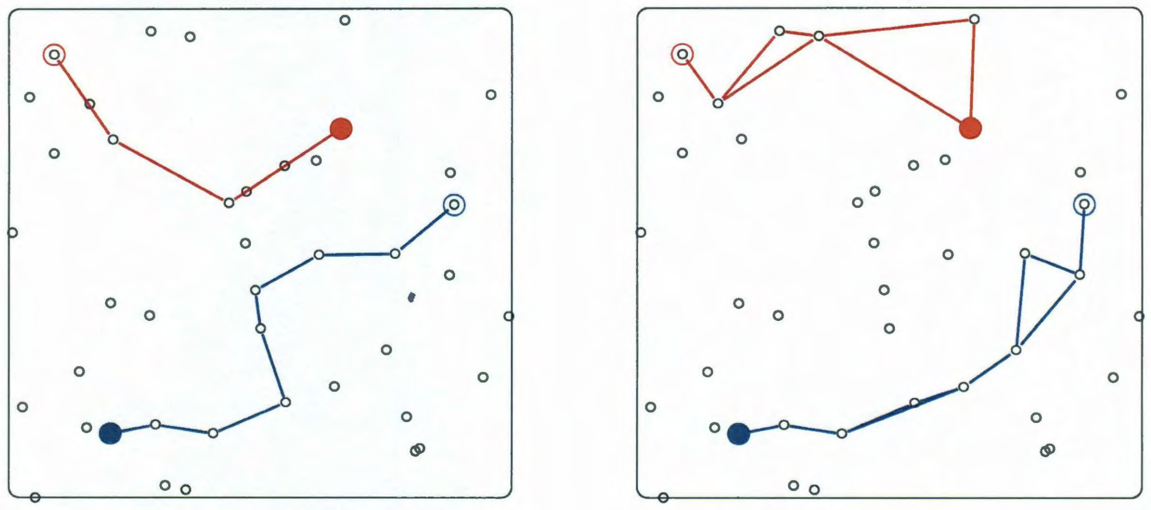


Figure 6.2: Permitting cooperation can affect link-layer routing decisions. Here, two flows (red and blue) traverse from their sources (solid circles) to their destinations (open circles) in the same topology under both point-to-point and cooperative paradigms. Both routes change, since signal-level cooperation helps overcome pathloss to allow longer bottleneck links. Here, cooperation increases throughput for the blue flow from 0.80 to 0.86, and for the red flow from 0.64 to 0.71 bits/sec.

are used only when and where they don't negatively impact the network.

These qualities of cooperative links lead to a tight connection between physical-layer cooperation and power control decisions and medium-access routing and scheduling decisions. An example of this is shown in Figure 6.2. For this network in a pathloss-dominated environment, the availability of cooperative links significantly changes the throughput-optimal route for each of the two flows. As a result, scheduling and routing decisions should not be divorced from the selection of cooperative links.

We will now address the problem of wide-area resource allocation in the N-terminal network where physical-layer cooperation is possible. Namely, we will show how our network model and NFIC can be extended to subsume cooperation as a transmission technology, without significantly increasing the complexity of the solution algorithms. In doing so, we will present a simple, effective technique for selecting cooperative units from the combinatorial number of possibilities in the network, and we will solve a nonlinear optimization problem to perform power-management.

6.2 Cooperative Model & Constraints

The model introduced in Chapter 2 is the foundation for our study here, though some elements must be updated to reflect the availability of cooperative communication at the physical layer. We now detail these modifications.

6.2.1 Network Model

The network model–data flows composed of variably sized frames–remains unchanged; the details given in Chapter 2 still apply. An important point here is that terminals are still capable of only half-duplex communication, which is why the BC and MA cooperative modes must exist in disjoint timeslots.

Our goal is to incorporate signal-level cooperation into the larger network. Cooperation, like all advanced physical layer techniques–is best characterized using information theory, which provides achievable rates (and in some cases capacity) in terms of transmit power, interference, and noise. We retain the information model of the physical layer, where

$$\log_2\left(1+\frac{P_t(x)D_{x,y}^{-\alpha}}{\gamma_t(y)+N_o}\right) \tag{6.1}$$

upper-bounds the rate of a frame on a point-to-point link. As before, we normalize timeslots to be 1 second long and using a bandwidth of 1 Hertz, such that this expression specifies exactly the size in bits of a frame transmitted between x and y. As in the other sections of this thesis, we may also regard this as the spectral efficiency of the transmission, measured in b/s/Hz.

The information theoretic characterization of the relay channel is incomplete, i.e. the capacity of the relay link is not known exactly. However, under the timeslotted assumptions we employ here, achievable rates have been determined. We will assume an asychronous *decode-and-forward* scheme, in which the relay terminal detects the frame it receives in the broadcast timeslot and forwards it to the destination in the multiple-access timeslot. In this case, the achievable rate for the relay channel is given by [39]:

$$\min\begin{cases}\alpha\log_2\left(1+P_B(s)h_B^{sr}\right)+(1-\alpha)\log_2\left(1+(1-\beta)P_M(s)h_M^{sd}\right)\\ \alpha\log_2\left(1+P_B(s)h_B^{sd}\right)+(1-\alpha)\log_2\left(1+P_M(s)h_M^{sd}+P_M(r)h_M^{rd}\right.\\ \qquad\left.+2\sqrt{\beta P_M(s)h_M^{sd}P_M(r)h_M^{rd}}\right)\end{cases} \tag{6.2}$$

Here we use the notation B and M in the timeslot subscript to indicate the broadcast and multiple-access timeslots, respectively. We use a consolidated channel variable to represent all the relevant parameters as:

$$h_B^{sr} = \frac{D_{s,r}^{-\alpha}}{\gamma_B(r) + N_o} \tag{6.3}$$

and similar for the other channels and the other timeslot. Since time is split equally across timeslots, we have that $\alpha = \frac{1}{2}$. Further, we assume no correlation between the relay transmission and the source transmission in the multiple access timeslot. Hence $\beta = 0$. With these values so determined, we can reduce the expression somewhat:

$$\min \begin{cases} \frac{1}{2}\log_2\left(1 + P_B(s)h_B^{sr}\right) + \frac{1}{2}\log_2\left(1 + P_M(s)h_M^{sd}\right) \\ \frac{1}{2}\log_2\left(1 + P_B(s)h_B^{sd}\right) + \frac{1}{2}\log_2\left(1 + P_M(s)h_M^{sd} + P_M(r)h_M^{rd}\right) \end{cases} \tag{6.4}$$

As in the non-cooperative case, normalizing timeslots and bandwidth means that (6.4) specifies the size in bits of a frame transmitted over the cooperative link. For the example network shown on the left side of Figure 6.3, equation (6.4) specifies the data rate from w to z with y acting as a relay.

6.2.2 Scheduling and Routing Constraints

The scheduling and routing constraints (2.1) - (2.7) described in Chapter 2 prevent cooperation in the form given there. In particular, constraint (2.1) prohibits more than one link from carrying a frame at a given time, which puts both modes of cooperation outside the feasible solution space.

As such, we must modify these constraints to permit multiterminal technology in our network. To do so, let us first define two new binary variables:

Definition 10. *Let the binary variable B_t^k be '1' if frame k is being transmitted in*

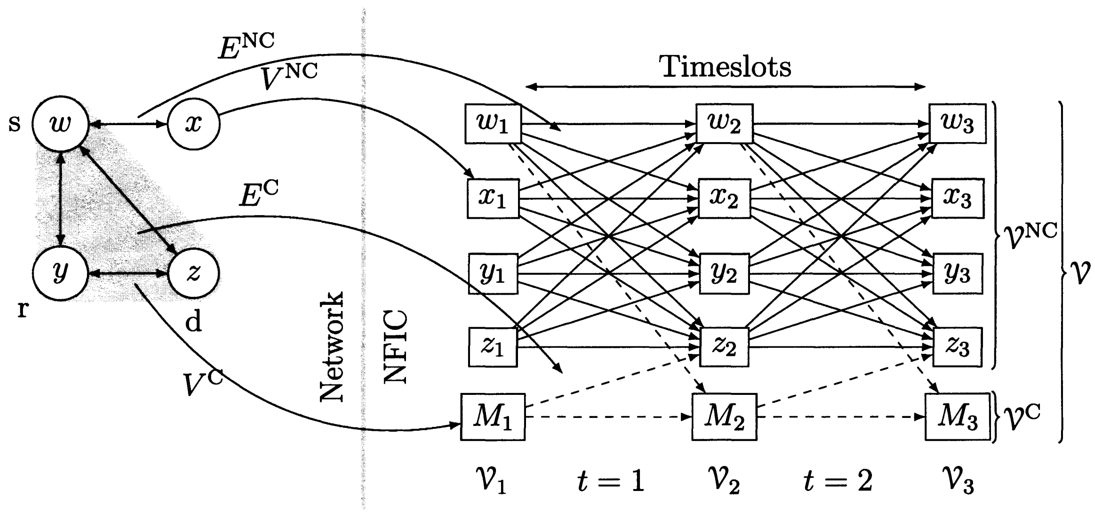


Figure 6.3: An example network (left) and the corresponding NFIC (right). The mappings E^{NC} and V^{NC} define nodes and edges corresponding to point-to-point links, while E^{C} and V^{C} define those corresponding to cooperative links. In this example, the metanode M represents the cooperative triple $\{w, y, z\}$.

the broadcast mode of cooperation in time t, '0' otherwise. Similarly, let the binary variable M_t^k be '1' if frame k is being transmitted in the multiple-access mode of cooperation in time t, '0' otherwise.

Since these two modes may not be active concurrently, we restrict them as such:

$$B_t^k + M_t^k \leq 1 \quad \forall \quad t, k \tag{6.5}$$

The cooperative broadcast mode must happen prior to the multiple-access mode:

$$\sum_{i=1}^{t} B_i^k > \sum_{i=1}^{t-1} M_i^k \quad \forall \quad t, k \tag{6.6}$$

Last, if a broadcast mode is used, a multiple access mode must be used in a subsequent (though not necessarily consecutive) timeslot:

$$B_t^k = M_{t+\Delta}^k \quad \forall \quad t, k \tag{6.7}$$

where Δ is a non-negative integer time-shift parameter, possibly greater than one. As we will see later, we assume that $\Delta = 1$ in this thesis to limit complexity. These two variables will enable cooperation in the integer program, in the highly coordinated manner we described earlier. We may now modify our constraints accordingly. Constraint (2.1), is updated to become

$$\sum_{x,y \in \mathcal{N}} I_t^k(x,y) = 1 + B_t^k + M_t^k \quad \forall \quad k, \quad t \geq 1 \tag{6.8}$$

The constraints binding a frame to its source and destinations are also updated. Note that the source can be the beginning of a cooperative BC mode, and the destination can receive in a cooperative MA slot:

$$\sum_{y \in \mathcal{N}} I_1^k(s(k),y) \;=\; 1 + B_t^k \quad \forall \quad k \tag{6.9}$$

$$\sum_{x \in \mathcal{N}} I_{T_k}^k(x,d(k)) \;=\; 1 + M_t^k \quad \forall \quad k,\ T_k > 1 \tag{6.10}$$

The constraint preventing circular routing and forcing continuity of routing remains unchanged:

$$\sum_{x \in \mathcal{N}} I_t^k(x,y) = \sum_{z \in \mathcal{N}} I_{t+1}^k(y,z) \quad \forall \quad k,\ y \in \mathcal{N},\ t \geq 1 \tag{6.11}$$

In particular, if a terminal receives a frame in the BC mode, it can store it ($I_t^k(x,x) = 1$) until the MA mode. We must modify the duplexing and multicast

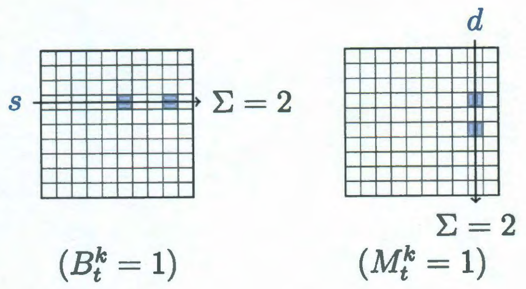


Figure 6.4: On the left, the BC mode is active, while on the right, the MA mode is active. Note that in each case, the blue frame appears on two links at once.

constraints, as follows:

$$\sum_k \left[\underbrace{\sum_{z \in \mathcal{N} \setminus (x,y)} \left(I_t^k(x,z) + I_t^k(z,x)\right)}_{\text{Tx must be half-duplex}} + \underbrace{\sum_{z \in \mathcal{N} \setminus (x,y)} \left(I_t^k(y,z) + I_t^k(z,y)\right)}_{\text{Rx must be half-duplex}} \right] \leq 1 + B_t^k + M_t^k \tag{6.12}$$

which hold for all $(x, y) \in \mathcal{N}$. In Figure 2.2 we illustrated how these binary variables behave in the point-to-point case. To make clear the constraints above, Figure 6.4 provides an illustration similar to that we showed for point-to-point constraints.

6.3 Representing Cooperation on the NFIC

We must capture the temporal nature of the network in order to manage the BC and MA timeslots of the cooperative links. The NFIC developed in Chapter 3 is well-suited to this task, since it concisely represents all the parameters of the network: time, space, and power. It also captures the interference associated with wireless transmissions, which is critical to properly manage the use of cooperative links.

We recall the definitions of the NFIC mappings in Chapter 3, and will add to them here.

The driving notion behind the NFIC definition and the following node and edge definitions is the need to represent the network in both space and time. The phases of cooperative links are temporally coupled, necessitating a network graph which exposes the temporal dimension. The x axis of the NFIC is exactly this dimension: as it extends to the right, the replication of the nodes on the y axis indicate the availability and capability of terminals at specific timeslots in network operation. The edges of the NFIC will be weighted to represent the third dimension, power.

The sets $\mathcal{V}^{\mathrm{NC}}$ and $\mathcal{E}^{\mathrm{NC}}$ are the NFIC nodes and edges corresponding to point-to-point links in the network, which will fully define the NFIC if cooperative links are not used. We will now show how to extend the NFIC to represent such multiterminal links, by defining a special type of node called a *metanode*. These nodes, while single elements on the NFIC, represent the coordinated action of more than one terminal in the network. This allows resource allocation algorithms–which act on the NFIC rather than the network graph itself–to regard cooperative links no differently from point-to-point ones, although potentially offering higher throughput.

Definition 11. *Define the* metanode mapping $V^{\mathrm{C}} : \mathcal{N}^3 \times T \to \mathcal{V}^{\mathrm{C}} \times T$*, which maps triples of terminals in the physical network to single nodes in the NFIC. The details of how triples of terminals are chosen are made precise in Section 6.5.*

Definition 12. *Define the* cooperative-rate mapping $E^{\mathrm{C}} : \mathcal{N}^3 \times \mathcal{P}^3 \times T \to \mathcal{E}^{C} \times T$*, mapping power availability at cooperative terminals to edge weights in the NFIC. The edges are given weights according to*

$$e_t^{x,M} = \min \begin{cases} \frac{1}{2}\log_2\left(1 + P_B(s)h_B^{sr}\right) + \frac{1}{2}\log_2\left(1 + P_M(s)h_M^{sd}\right) \\ \frac{1}{2}\log_2\left(1 + P_B(s)h_B^{sd}\right) + \frac{1}{2}\log_2\left(1 + P_M(s)h_M^{sd} + P_M(r)h_M^{rd}\right) \end{cases} \tag{6.13}$$

This edge-weighting definition implicitly requires that $\Delta = 1$, or that the BC and MA slots are consecutive: $B = t$ and $M = t + 1$. We recognize that this limits the solution space somewhat. In order to decouple these cooperative modes, we would require the NFIC to be a hypergraph, with several edges between the cooperative source and the metanode, one for each combination of B and M.

Figure 6.3 illustrates how V^{C} maps the group of cooperating terminals $\{w, y, z\}$ to the node M in the NFIC. The rate achievable across that cooperative link is mapped by E^{C} to the edges connecting M to the other nodes in the NFIC, shown with broken lines.

The metanodes defined by V^{C} are appended to NFIC below those created by V^{NC}, such that there are now more nodes in the NFIC than terminals in the network: the relationship between $\mathcal{V}$ and $\mathcal{N}$ is no longer one-to-one. The edges created by E^{C} are not fully connecting within a timeslot as they are in the point-to-point case. Specifically, each metanode is the termination of only two edges: one from the source in the cooperative triple, and one from the metanode itself in the previous timeslot. The metanode is the origination for two edges only: one to the destination node, and one to the metanode itself in the next timeslot. As such, data passes from the source, through the metanode, to the destination, with the possibility of being stored at the metanode. In this way, the broadcast and multiple access timeslots need not be consecutive, but must occur in that sequence.

The final NFIC is therefore comprised of the nodes $\mathcal{V} = \mathcal{V}^{\mathrm{NC}} \cup \mathcal{V}^{\mathrm{C}}$ and edges $\mathcal{E} = \mathcal{E}^{\mathrm{NC}} \cup \mathcal{E}^{\mathrm{C}}$. The baseline case we study is when cooperation is not possible in the network, such that $\mathcal{V} = \mathcal{V}^{\mathrm{NC}}$ and $\mathcal{E} = \mathcal{E}^{\mathrm{NC}}$. In Figure 6.3, the reference case would map the network to an NFIC consisting of the upper four rows of nodes, while the cooperative case expands to include the metanode M. As before, the initial state of metanode edges in the NFIC is calculated with $P_t(x) = P_{\max}$ and $\gamma_t(x) = 0$.

Example 1. *Consider a network of four terminals $\{w, x, y, z\}$, as shown in Figure 6.5. The NFIC for two timeslots is constructed using the four mappings defined above; V^{NC} leads to four of the nodes in each time slot, representing the terminals of the network acting as transmitters for point-to-point links. The edge mapping E^{NC} connects all of these nodes together in each timeslot, since all point-to-point links exist. For this network the cooperative mapping V^{C} has defined two cooperative units: the metanode M^1 represents the cooperative triple in which w is the source, z is the destination, and y is the relay, as shown in the upper pane of the figure. M^2 also represents data from w bound for z, but using x as the relay instead of y. In the NFIC, the metanodes appear at all timeslots in the same way as the point-to-point nodes, because they may be used in any slot. The metanode edges specify how data uses the cooperative system: M^1 receives an edge from w to represent the BC phase of w transmitting with both y and z receiving. The edge out of M^1 represents the MA phase, when both w and y transmit to z.*

6.3.1 Frame Resource Allocation Algorithm

We briefly review the allocation technique presented in detail in Chapter 4. We decompose the joint allocation problem across frames, such that we compute a schedule, route, and power allocation for frame k alone, given the set of network resources available after allocating frames $\{1, \ldots, k-1\}$. Our approach to allocating resources will involve three steps: first, we execute a dynamic programming algorithm across the NFIC to find the optimal sequence of terminals and timeslots for frame k, fixing the schedule and route, which may involve cooperative links. Second, interference-management is addressed when the power used by each of the terminals in the route is optimized. Third, the edge weights of the NFIC will be updated to reflect the resources consumed by frame k. Some edges will be removed, because they will be

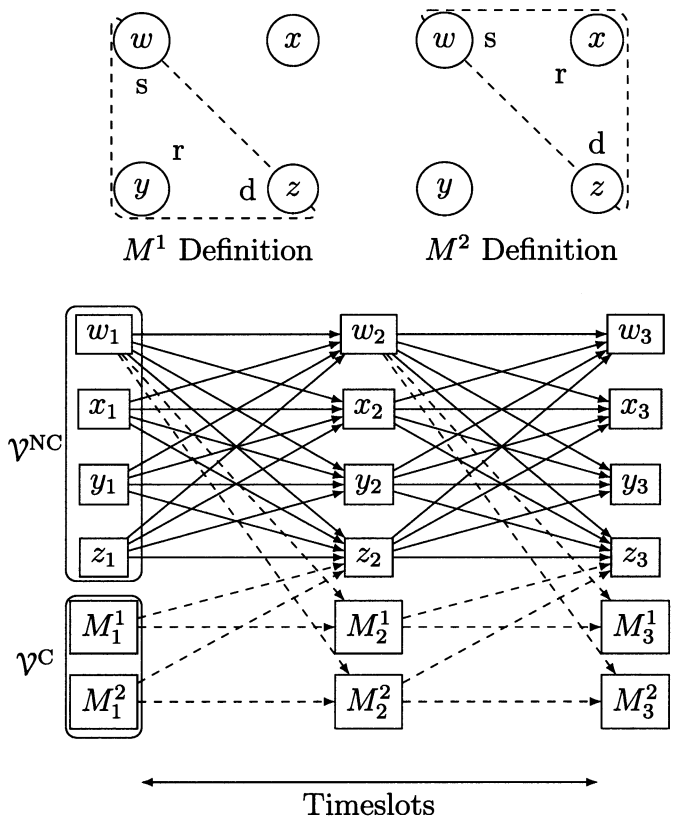


Figure 6.5: Top: A network for four nodes $\{w, x, y, z\}$, from which two metanodes are defined. Bottom: NFIC corresponding to the network. The nodes corresponding to the single terminals $\mathcal{V}^{\text{NC}}$ are all fully connected in each timeslot, since any node can communicate with any other. The nodes corresponding to cooperation $\mathcal{V}^{\text{C}}$ are connected in a manner specific to the roles of the terminals involved in the cooperation.

connected to terminals in use handling frame k and cannot be used by a future frame. We repeat for frame $k+1$. Note that the allocation and optimization is performed entirely on the NFIC, which fully represents network conditions as a function of time.

We now detail how resources are allocated for frame k, given an NFIC with edge weights calculated according to E^{NC} and E^{C}. First, the source terminal of frame k is identified, and the corresponding NFIC node is given a weight of ∞ at timeslot $t = 1$. All other nodes receive a weight of 0. The dynamic programming algorithm we executed *over the NFIC* is the same for each timeslot t, starting from slot 1:

1. At timeslot t, update the edges with the minimum of their source weight (data at the terminal) and initialization weight (capacity of the link given available resources).

2. Node weights are updated with the maximum of the incoming edge weights (receive as much data as possible). At each node, store which edge was the maximizer.

3. Repeat until node weights and edge weights remain static across consecutive timeslots.

At this point, the data-maximizing paths between the source and all terminals have been found. The weight of the destination node at each timeslot t represents the greatest amount of data a path of $t-1$ slots can offer. Scaling the destination node weight by the length of the path and then selecting the maximum results in the throughput-optimal path for frame k. To determine the path, we trace from the destination node at maximizing timeslot back through the NFIC to the source at time 1. The active NFIC edge in each timeslot specifies activating the corresponding link in the network, which will be a cooperative link if a metanode is involved.

Areas of high interference in the network, where frames $\{1, \ldots k-1\}$ are being transmitted, are naturally avoided using NFIC algorithms: because of the interference, those edges receive low weight. As a result, the allocation will either route frames around each other, pause in memory to wait for other frames to pass, or employ cooperation to overcome the interference.

Example 2. *In the network of Figure 6.5, suppose frame k is sourced at w and destined for z. If the direct channel is of good quality, then the solution is $w_1 \to z_2$ in timeslot 1. If not, the solution may be $w_1 \to M_2^1 \to z_3$, corresponding to employing terminal y as a relay, using first a cooperative BC mode and then a cooperative MA mode, over two timeslots.*

6.4 Power Control for Interference Management

The preceding section described the NFIC algorithm which, given network conditions, optimizes the schedule and route a frame should use. We now describe how, after determining a schedule and route, the power used by frame k is optimized to limit interference on future frames. Recall that the amount data carried by frame k is determined by the bottleneck link on its route: $R^k = \min_t\{R_t^k\}$. We therefore compute power for all other links in the route so that *all* links operate at rate R^k. This not only conserves power in the network, but also reduces the overall interference temperature to allow for other simultaneous transmissions.

Once the power $P_t(x)$ has been selected for all transmitters in frame k's route (where we use x and t in generality), these values will be used to update the interference seen at all other terminals in the network. That is, we compute $\gamma_t(x)$ so that we may update the edges of NFIC prior to allocating frame $k+1$. In describing the optimal power selection, we consider the non-cooperative and cooperative cases separately.

6.4.1 Non-Cooperative Links

In the case of non-cooperative links, the power allocation method is the same as described in Section 4.4. Suppose the solution calls for the link $x \to y$ to be active at time t. The rate on a non-cooperative link is determined by (6.1), in which the only variable is $P_t(x)$, since fixing the route and timeslot has fixed $D^{\alpha}_{x,y}$ and $\gamma_t(x)$. As such, evaluating

$$P_t(x) = (2^{R^k} - 1)(\gamma_t(x) + N_o)D^{\alpha}_{x,y} \tag{6.14}$$

determines the power x should use such that this link in the route operates at rate R^k.

6.4.2 Cooperative Links

Cooperative links, which are selected if metanodes are part of the NFIC solution, are defined by three different power values: power of the source s in the BC and MA modes, and power of the relay r in the MA mode. We wish to select $P_B(s), P_M(s)$ and $P_M(r)$ such that equation (6.13) equates to R^k.

Minimizing interference in the network requires reducing the footprint of the link, i.e. reducing the power of each transmission as much as possible such that rate R^k is maintained. To that end, we choose to minimize the *sum power* $P_B(s)+P_M(s)+P_M(r)$ which will reduce the footprint both geographically (across the source and relay) and also temporally (in the BC and MA timeslots). The optimization problem becomes

$\min P_B(s) + P_M(s) + P_M(r)$ such that

$$R^k = \min \begin{cases} \frac{1}{2}\log_2\left(1 + P_B(s)h_B^{sr}\right) + \frac{1}{2}\log_2\left(1 + P_M(s)h_M^{sd}\right) \\ \frac{1}{2}\log_2\left(1 + P_B(s)h_B^{sd}\right) + \frac{1}{2}\log_2\left(1 + P_M(s)h_M^{sd} + P_M(r)h_M^{rd}\right) \end{cases} \tag{6.15}$$

The min in the constraint yields a nonlinear system, but we can still apply Lagrangian techniques to each of the components in the min, then comparing the solutions to determine which uses lower sum power. Breaking the problem in two, we first solve

$$\begin{aligned} \min \quad & P_B(s) + P_M(s) + P_M(r) \quad \text{s.t.} \\ R^k = & \frac{1}{2}\log_2\left(1 + P_B(s)h_B^{sr}\right) + \frac{1}{2}\log_2\left(1 + P_M(s)h_M^{sd}\right) \end{aligned} \tag{6.16}$$

for which the corresponding Lagrangian is

$$\begin{aligned} L = & P_B(s) + P_M(s) - \lambda_1 \left[\log_2\left(1 + P_B(s)h_B^{sr}\right) + \right. \\ & \left. \log_2\left(1 + P_M(s)h_M^{sd}\right)\right] - \lambda_2 P_B(s) - \lambda_3 P_M(s) \end{aligned} \tag{6.17}$$

where the the multipliers λ_2 and λ_3 result from the non-negativity of power. Complimentarity requirements at the KKT points necessitate that both of the products $\lambda_2 P_B(s)$ and $\lambda_3 P_M(s)$ evaluate to zero. This can happen if:

- $\lambda_2 = 0, \lambda_3 \neq 0$. This forces $P_M(s) = 0$, so that $P_B(s)$ is found by solving $R^k = \log_2(1 + P_B(s)h^{sr})$
- $\lambda_2 \neq 0, \lambda_3 = 0$. Symmetric to previous case.

If both $\lambda_2 = 0$ and $\lambda_3 = 0$, then we must solve the system for $P_B(s)$ and $P_M(s)$.

Differentiation of the Lagrangian and variable substitution results in $P_B(s)$ being the solution to the quadratic equation:

$$P_B(s)^2(h_B^{sr})^2 + 2h_M^{sd}P_B(s) + 1 - \frac{h_M^{sd}}{h_B^{sr}}2^{R^k} = 0 \tag{6.18}$$

After solving this equation for $P_B(s)$, we determine $P_M(s)$ as

$$P_M(s) = \frac{1}{h_M^{sd}}\left[\frac{2^{R^k}}{1 + P_B(s)h_B^{sr}} - 1\right] \tag{6.19}$$

This establishes the power minimizing solution such that the first part of the constraint is met with equality. If this is the case, we set $P_M(r)$ to zero, and the solution is essentially two transmissions from the source to the destination. This rarely occurs in practice, hence we turn our attention to the second part of the constraint, involving all three power variables. The optimization is posed as

$$\begin{aligned} \min \quad & P_B(s) + P_M(s) + P_M(r) \quad \text{s.t.} \\ R^k &= \frac{1}{2}\log_2\left(1 + P_B(s)h_B^{sd}\right) + \frac{1}{2}\log_2\left(1 + P_M(s)h_M^{sd} + P_M(r)h_M^{rd}\right) \end{aligned} \tag{6.20}$$

While the Lagrangian and corresponding differentials yield a system of four equations in four unknowns, it is underdetermined; the Hessian of the Lagrangian is indefinite. In particular, the variables in the sum term $P_M(s)h_M^{sd} + P_M(r)h_M^{rd}$ never appear separately in distinct constraints, so there is *no unique solution.* Combining these as

$$\delta = P_M(s)\frac{h_M^{sd}}{h_M^{rd}} + P_M(r) \tag{6.21}$$

we are left with

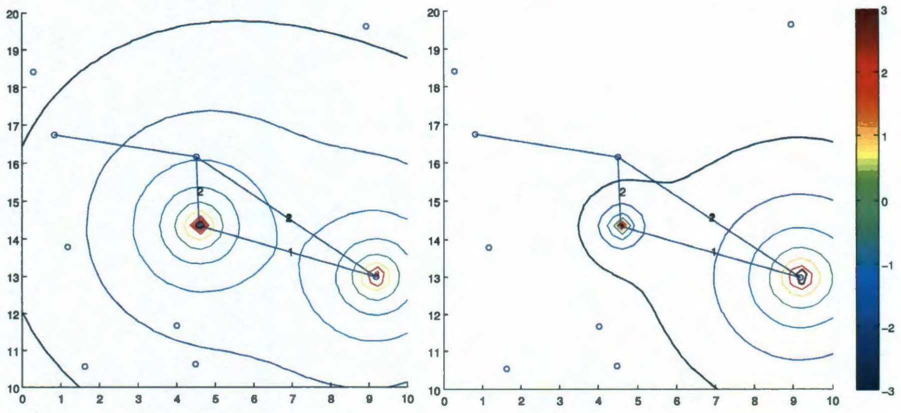


Figure 6.6: Lagrangian interference management in action: the interference contours in the MA timeslot are considerably reduced (right) from the full power case (left) after the rate of the cooperative link is set to match that of the point-to-point link. Scale shown in dBm.

$$\begin{aligned} \min \quad & P_B(s) + P_M(s) + P_M(r) \quad \text{s.t.} \\ & R^k = \frac{1}{2}\log_2\left(1 + P_B(s)h_B^{sd}\right) + \frac{1}{2}\log_2\left(1 + \delta h_M^{rd}\right) \end{aligned} \tag{6.22}$$

which is of exactly the same form as (6.16). Using equations (6.18) and (6.19) but with $P_B(s)$ and δ, we compute each of those values. To allocate $P_M(r)$ and $P_M(s)$ such that (6.21) holds, we split power between the source and relay in an inverse proportion to their channel quality, following the waterfilling solution: $P_M(s)h_M^{sd} = P_M(r)h_M^{rd}$. This gives

$$P_M(r) = \frac{\delta}{2}, \quad P_M(s) = \delta\frac{h_M^{rd}}{2h_M^{sd}} \tag{6.23}$$

We then compare this solution to the solution to (6.16); the one with the smaller sum power is chosen.

Example 3. *Figure 6.6 shows an example of employing this power allocation technique to reduce the power of the cooperative link. In this case, cooperation shifted the bottleneck to the point-to-point link on the left side of the route, meaning that power could be reduced at the cooperative terminals. Doing so reduces the interference temperature in the MA slot.*

Once the power allocation has been computed, network resources become available for other flows. In this example, the terminals in the lower-left corner of the topology can receive frames in the MA slot, since interference has been reduced. This would not be so in the non-cooperative alternative (not shown), because the relay terminal in the center would use higher power in the second timeslot.

6.4.3 NFIC Edge Updates with Optimized Power

We have now described how to limit the amount of power used by the terminals involved in frame k; that is, we have computed $P_t(x)$ for all terminals in the route of frame k, given the interference pattern γ in the network. Before frame $k+1$ can be allocated, the NFIC must be updated to reflect the resources frame k has consumed. This happens in two steps: first, edges connected to active nodes at time t are assigned weight 0, enforcing the half-duplex constraints. Since 0 is the identity element for min in our semiring, these edges will never be selected to be a part of a subsequent frame's allocation.

Second, the maximum transmission power of those terminals not involved in frame k must be limited to prevent them from excessively interfering with frame k. In order to compute $P_t(z)$ for each terminal z not involved in frame k's route, we must determine how much interference frame k can tolerate. As discussed in Chapter 4, we parameterize this as R_{Int} in that we allow other terminals to decrease frame k's rate by the fractional term R_{Int}.[1] If terminal y is receiving frame k at time t, it can tolerate

$$\hat{\gamma}_t(y) = \left[P_t(x) \left[2^{-(R^k \cdot R_{\text{Int}})} \left(1 + \frac{P_t(x)}{N_o + \gamma_t(y)} \right) - 1 \right]^{-1} \cdot D^{\alpha}_{x,y} - N_o \right]^{+} \tag{6.24}$$

amount of interference. We can then determine the maximum power any terminal z can use as

$$\hat{P}_t(z) \leq \hat{\gamma}_t(y) \cdot D^{\alpha}_{z,y} \quad \forall \quad z \in \mathcal{N} \tag{6.25}$$

The edges of the NFIC are updated in time t using (6.1) for point-to-point links and (6.13) for cooperative links, using these new transmission powers. With these updates completed, we repeat to allocate frame $k+1$.

6.5 Metanode Selection

The key to enabling cooperation is the mapping between network terminals and metanodes V^{C}. This mapping alone makes cooperation available to the resource allocation algorithms, so its design is a key part of the success of this approach. Recall that as illustrated in Figure 6.2, the *availability* of cooperation fundamentally changed the routing decisions. Selecting the route first while assuming only point-to-point links and then trying to build in cooperation later would have resulted in a suboptimal solution, as shown by Theorem 1. We must therefore define metanodes *prior* to executing any of our resource allocation algorithms.

In general, for a network with N terminals, there are $\frac{N!}{(N-3)!}$ possible metanode definitions, the permutations of N terminals taken 3 at a time. The permutation

[1]The choice of R_{Int} depends largely on network parameters; see Chapter 4 for details.

is needed because the metanode representing {source, relay, destination} $\{x, y, z\}$ is distinct from $\{x, z, y\}$, as can be seen from (6.4). In a network with 50 terminals, this results in 117600 possible metanodes, or a growth in $|\mathcal{V}|$ by 235000%. Since we seek to leverage cooperation while preserving the low complexity of our techniques, this is untenable. Hence for each terminal $n \in \mathcal{N}$, we will define $N_R \ll N$ metanodes: as such each terminal will be the source in N_R {s, r, d} triples, each represented by a metanode in the NFIC. Thus we will have that $|\mathcal{V}| = N + (N \cdot N_R) = N(1 + N_R)$. This limits the growth of the NFIC, while making a large number of cooperative links available to the scheduling and routing algorithms.

6.5.1 Nearest-Neighbor Metanode Definition

Our first approach is to define, for each terminal, a specific number N_R of cooperative triples. We choose these triples based solely on distance: for terminal x, we identify the nearest N_R terminals (which will become relays) and for each one, choose the nearest terminal *not* nearer to x to act as the destination in the triple. Thus here and in the following algorithms, the number of metanodes is $N \cdot N_R$, so the total number of nodes in the NFIC is $N(1 + N_R)$. This algorithm is formalized as Algorithm 1 and illustrated in Figure 6.7. Advantages of this algorithm are the simplicity of its design, which is amenable to practical implementation. It also has the smallest interference footprint, since sources and relays are in close proximity to each other. However, the algorithm does not consider the direction traffic may need to move, so it may offer cooperation in directions where it is not needed, and not when it is. We assume that the notion of a "nearest neighbor" is delivered using a localization system such as GPS, which we will incorporate in this and the following metanode selection algorithms.

Algorithm 1: Distance-based Metanode Selection Algorithm

```
Create the one-to-one map of terminals to nodes:
$\mathcal{V}^{\mathrm{NC}} \leftarrow \mathcal{N}$
Assign point-to-point edges: $\mathcal{E}^{\mathrm{NC}} = E^{\mathrm{NC}}(\mathcal{P})$
Create an empty set of metanodes: $\mathcal{V}^{\mathrm{C}} \leftarrow \emptyset$
Define metanodes for each terminal:
foreach $n \in \mathcal{N}$ do
    Define the set of possible relays and destinations:
    $\mathcal{N}' = \mathcal{N} \setminus n$
    foreach $i \in \{1, 2, \ldots, N_R\}$ do
        Assign terminal nearest to $n$ in $v_i$ as relay $r_i$, $r_i \in \mathcal{N}'$
        Assign terminal nearest to $r_i$ as destination $d_i$, $d_i \in \mathcal{N}'$
        Define the metanode: $M_i = \{n, r_i, d_i\}$
        $\mathcal{V}^{\mathrm{C}} \leftarrow M_i$
        Add edge to $\mathcal{E}^{\mathrm{C}}$, using eqn. (6.4):
        $e_t^{n,M_i} = E^{\mathrm{C}}(\mathcal{P}) \quad \forall \quad t \leq T$
        $e_t^{M_i,d_i} = E^{\mathrm{C}}(\mathcal{P}) \quad \forall \quad t \leq T$
        $\mathcal{E}^{\mathrm{C}} \leftarrow e_t^{n,M_i}, e_t^{M_i,d_i}$
        Make $r_i$ and $d_i$ unavailable to be chosen again: $\mathcal{N}' = \mathcal{N} \setminus \{r_i, d_i\}$
Set of nodes is union of terminals and metanodes:
$\mathcal{V} \leftarrow \mathcal{V}^{\mathrm{NC}} \cup \mathcal{V}^{\mathrm{C}}$, $\mathcal{E} \leftarrow \mathcal{E}^{\mathrm{NC}} \cup \mathcal{E}^{\mathrm{C}}$
```

6.5.2 Geographically-Equalized Metanode Definition

The geographical nature of our problem will inform the definition of V^{C}, and the basis from which we start is that for any terminal in the network, the NFIC allocation algorithm may wish to direct a frame in any geographical direction. This is because the NFIC routes frames through the network following paths which trade off interference with distance, giving unpredictable traffic patterns.

To the end of allowing cooperation to aid frame transmission in any direction, we will define N_R cooperative triples for each terminal in a geographically-neutral way. By this, we mean that we will define cooperative terminals in an equi-angular manner, using the following algorithm. For each terminal n, we divide the space around it into N_R sectors of equal arc length. Within each sector, we identify the nearest terminal to act as a relay, and the next-nearest to act as the destination. These three terminals:

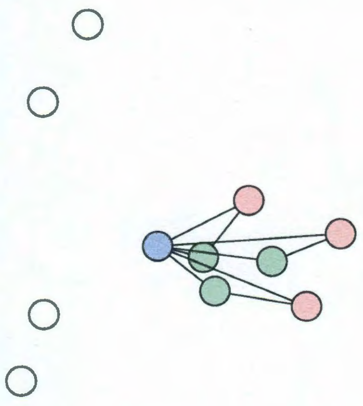


Figure 6.7: Nearest-neighbor algorithm visualized: the center blue terminal has three nearby relays, each with three nearby cooperative destinations.

n, the relay and the destination as identified above, form one of the N_R metanodes anchored at n. This repeats in each sector, as formalized in Algorithm 2.

We illustrate the algorithm in Figure 6.8. In this depiction, we are defining metanodes with terminal n as the source, and $N_R = 3$. The sectors v_i are bounded by the three dashed lines, and within each sector the nearest terminal is assigned as the relay, and the next nearest as the destination. The three metanodes M_i are then added to the NFIC, such that for frames passing through terminal n, three cooperative links are available in addition to point-to-point links.

6.5.3 Range Extension Metanode Definition

The third mechanism we consider is a hybrid of the previous two, designed to exploit cooperative benefit to allow larger frames to cross the network in fewer hops. Relays are used to aid in the transmission of large frames from the source well beyond nearby intermediate terminals, thereby increasing throughput by reducing a frame's time in transit. For each terminal in the network, we synthesize the geographic partioning approach of Algorithm 2 with the distance-based approach of Algorithm 1, except that instead of choosing the nearest terminal in the sector as the relay, we now select

Algorithm 2: Geographically-Equalized Metanode Selection Algorithm

```
Create the one-to-one map of terminals to nodes:
$\mathcal{V}^{\mathrm{NC}} \leftarrow \mathcal{N}$
Assign point-to-point edges: $\mathcal{E}^{\mathrm{NC}} = E^{\mathrm{NC}}(\mathcal{P})$
Create an empty set of metanodes: $\mathcal{V}_{\mathrm{C}} \leftarrow \emptyset$
Define metanodes for each terminal:
foreach $n \in \mathcal{N}$ do
    Create an equi-angular partition of the area $A$ around $n$ with $N_R$ sectors $v_i$
    such that $\bigcup_{i=1}^{N_r} v_i = A$
    foreach $i \in \{1, 2, \ldots, N_R\}$ do
        Assign terminal nearest to $n$ in $v_i$ as relay $r_i$
        Assign terminal nearest to $r_i$ as destination $d_i$
        Define the metanode: $M_i = \{n, r_i, d_i\}$
        $\mathcal{V}^{\mathrm{C}} \leftarrow M_i$
        Add edge to $\mathcal{E}^{\mathrm{C}}$, using eqn. (6.4):
        $e_t^{n,M_i} = E^{\mathrm{C}}(\mathcal{P}) \quad \forall \quad t \leq T$
        $e_t^{M_i,d_i} = E^{\mathrm{C}}(\mathcal{P}) \quad \forall \quad t \leq T$
        $\mathcal{E}^{\mathrm{C}} \leftarrow e_t^{n,M_i}, e_t^{M_i,d_i}$
Set of nodes is union of terminals and metanodes:
$\mathcal{V} \leftarrow \mathcal{V}^{\mathrm{NC}} \cup \mathcal{V}^{\mathrm{C}}$, $\mathcal{E} \leftarrow \mathcal{E}^{\mathrm{NC}} \cup \mathcal{E}^{\mathrm{C}}$
```

the terminal in the ρ^{th} distance percentile as the destination, and find a terminal near the halfway-point as the relay. In this manner, we define metanodes to exploit the cooperative advantage to move larger frames further, in all directions of the network. This algorithm is formalized in Algorithm 3, and visualized in Figure 6.9 for $\rho = 0.75$.

A plot comparing the throughput differences among these three algorithms is shown in Figure 6.10. Here, "MH" indicates non-cooperative multihop, the benchmark case. In these trials, $N_R = 4$. Range extension performs poorly, because the cooperative gain is not great enough to overcome the higher pathloss over long distances. Geographically-equalized metanode definition performs best, and as a result is the approach we employ throughout the remainder of this thesis.

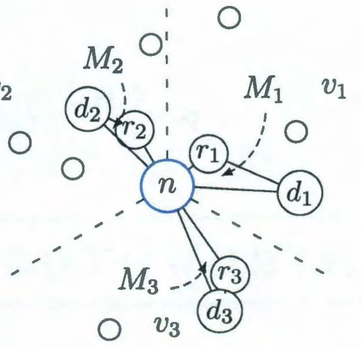


Figure 6.8: Metanode selection algorithm applied to one terminal of the network $n \in \mathcal{N}$. If $N_R = 3$, we choose three relays for n. The region is divided into three sectors $\{v_1, v_2, v_3\}$, and within each, the nearest terminal is selected as the relay. The terminal nearest the relay (but not nearer to n, is selected as the destination. In this way metanodes $\{M_1, M_2, M_3\}$ are chosen.

Algorithm 3: Range-Extension Metanode Selection Algorithm

```
Create the one-to-one map of terminals to nodes:
$\mathcal{V}^{\mathrm{NC}} \leftarrow \mathcal{N}$
Assign point-to-point edges: $\mathcal{E}^{\mathrm{NC}} = E^{\mathrm{NC}}(\mathcal{P})$
Create an empty set of metanodes: $\mathcal{V}^{\mathrm{C}} \leftarrow \emptyset$
Define metanodes for each terminal:
foreach $n \in \mathcal{N}$ do
    Create an equi-angular partition of the area $A$ around $n$ with $N_R$ sectors $v_i$
    such that $\bigcup_{i=1}^{N_r} v_i = A$
    foreach $i \in \{1, 2, \dots, N_R\}$ do
        Sequence the $N_i$ terminals in sector $v_i$ in, starting with those nearest to
        $n$: $\{t_1, t_2, \dots t_{N_i}\}$.
        Select the nearest terminal in the $\rho$ percentile $t_n$, such that $\lfloor \frac{n}{N_i} \rfloor \geq \rho$.
        Assign as destination $d_i$.
        Select the terminal nearest to the midway between $d_i$ and $t$ as the relay
        $r_i$.
        Define the metanode: $M_i = \{n, r_i, d_i\}$
        $\mathcal{V}^{\mathrm{C}} \leftarrow M_i$
        Add edge to $\mathcal{E}^{\mathrm{C}}$, using eqn. (6.4):
        $e_t^{n,M_i} = E^{\mathrm{C}}(\mathcal{P}) \quad \forall \quad t \leq T$
        $e_t^{M_i,d_i} = E^{\mathrm{C}}(\mathcal{P}) \quad \forall \quad t \leq T$
        $\mathcal{E}^{\mathrm{C}} \leftarrow e_t^{n,M_i}, e_t^{M_i,d_i}$
Set of nodes is union of terminals and metanodes:
$\mathcal{V} \leftarrow \mathcal{V}^{\mathrm{NC}} \cup \mathcal{V}^{\mathrm{C}}$, $\mathcal{E} \leftarrow \mathcal{E}^{\mathrm{NC}} \cup \mathcal{E}^{\mathrm{C}}$
```

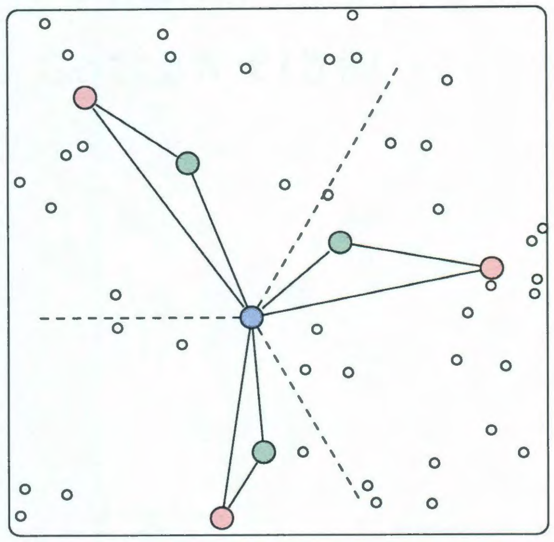


Figure 6.9: Range Extension definition algorithm visualized, where $\rho = 0.75$. Thus, cooperative metanodes are defined with the ability to transmit 75% of the maximal transmission distance from the center terminal.

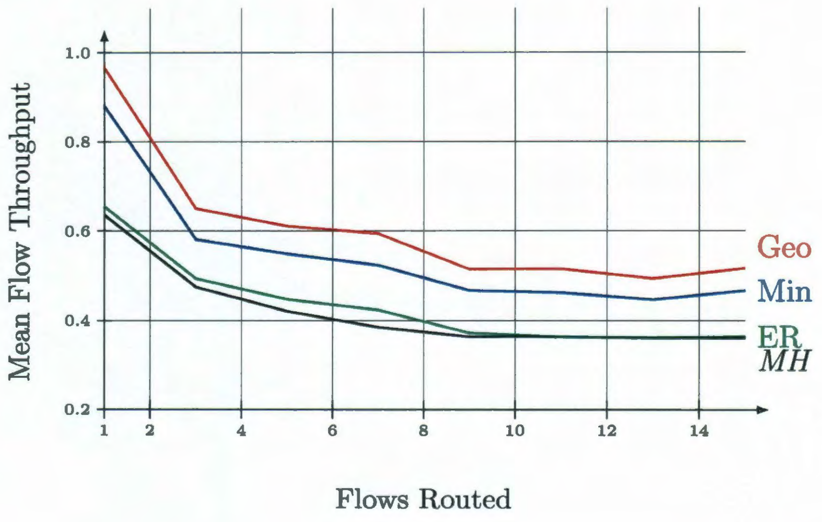


Figure 6.10: Comparison of the three metanode definition algorithms against the multihop benchmark. Geographic metanode definition significantly outperforms the three techniques.

6.5.4 Complexity Considerations

The NFIC allocation is cubic in the number of nodes in the NFIC: at each timeslot, we perform $|\mathcal{V}|^2$ edge updates and $|\mathcal{V}|$ node updates. In the worst case, when all links are used in the route, there are $|\mathcal{V}|$ timeslots for which updates are computed. The product of these three operations gives the cubic time, $O(N^3)$ algorithm. Algorithm 2, for assigning metanodes, is performed only once for each network realization. Its complexity is found as follows: for each terminal, we divide the region into N_R sectors, containing an average of $\frac{N}{N_R}$ terminals. Within each sector, we must execute a sort, determining the nearest terminal (to act as the relay) and the next-nearest to act as the destination. On average, the sort requires $O\left(\frac{N}{N_R}\log\frac{N}{N_R}\right)$ complexity [13], which we execute $N \cdot N_R$ times. This makes Algorithm 2 linear in $O\left(\frac{N^2}{N_R}\log\frac{N}{N_R}\right)$, leaving the overall approach dominated by the resource allocation algorithm runtime of $O(N^3)$.

6.6 Results

We now simulate the previously described network and algorithms. For all simulations, we let the region size $A^2 = 400$ and Poisson intensity $\lambda = 0.1$, resulting in networks with a mean of 40 terminals per realization. We vary the number of flows between 1 and 15, and change the pathloss α between 2 (free space) and 4 (urban).

The first example of an NFIC allocation using cooperative metanodes appeared on the right side of Figure 6.2. In this random network, the allocation for each flow in the left pane is determined without metanodes, i.e. $\mathcal{V} = \mathcal{V}^{\mathrm{NC}}$. On the right, metanodes are defined in the geographically-equalized manner described above with $N_R = 3$. With the availability of cooperative links, the NFIC allocation algorithm was able to move data over longer hops in shorter time, though doing so resulted in a different data path for each flow. Not all links are cooperative; only those which benefit the data flow without causing undue interference. With these parameters, cooperation increased the data rate for the blue flow from 0.80 to 0.86 b/s/Hz, and for the red flow from 0.64 to 0.71 b/s/Hz.

6.6.1 Throughput and Power

We now study the benefits of cooperation from a flow throughput view, reporting our results in Figure 6.11. For these experiments, we randomly realize a network, then compute throughput for all of the flows without defining metanodes, using the NFIC allocation technique. This establishes throughput for the point-to-point scenario. We then define metanodes *for the same topology* according to the algorithm above, with $N_R = 4$. A new allocation is computed, and throughput is calculated for all flows involved. We report the mean increase in the throughput with cooperation employed over the point to point case, as $\overline{T}_{\text{gain}} = \frac{1}{F}\left(T_{\text{Coop}} - T_{\text{NC}}\right)/T_{\text{NC}}$. We observe that cooperation benefits throughput at all levels of network load, particularly when load is high.

When load is light, cooperation benefits only the bottleneck links on the route. However, when load is high and interference is a limiting factor, cooperation helps frames overcome that interference to increase the throughput, up to some point at which the network region is saturated–under these simulation parameters, around 15 flows. The cooperative gain is slightly more pronounced for high pathloss, when network isolation reduces interference and allows cooperation to offer further benefit. This is because in these environments, cooperation can be used to overcome both inter-frame interference *and* high pathloss, whereas interference is the dominant limiting factor in low pathloss environments.

We next turn our attention to the power consumed when the network is operating in a cooperative mode. For the throughput results presented above, we compute the

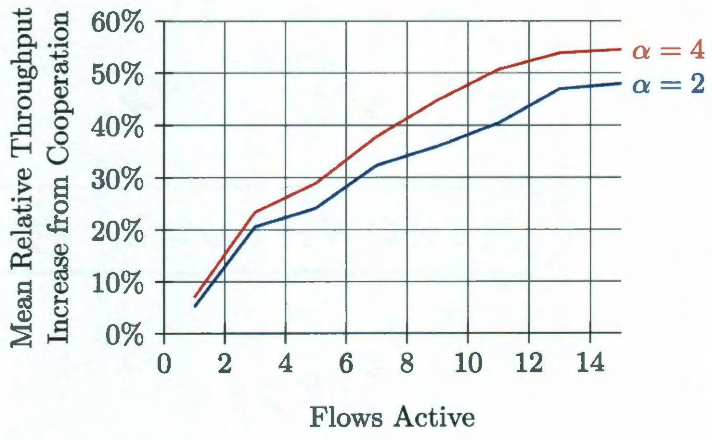


Figure 6.11: Relative increase in flow throughput when cooperation is made available through metanodes. Here $N_R = 4$. Even under high network load, cooperation improves throughput considerably.

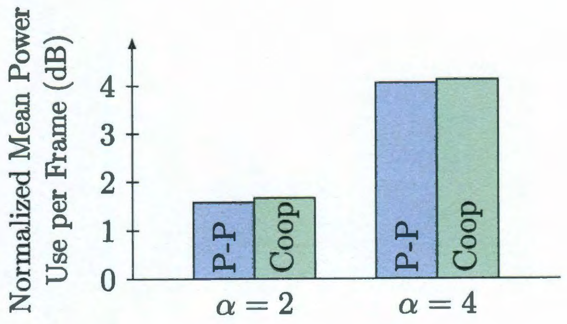


Figure 6.12: Comparison of power use for point-to-point and cooperative networks, define with $N_R = 4$. Isolation in $\alpha = 4$ requires more power to overcome.

average amount of power necessary for one frame to traverse between its source and destination, including relay power if cooperative links are used. Power is optimized as described in Section 6.4. This is normalized against the throughput contribution of that frame, so that we are reporting Joules per bit. We compare these averages in both cooperative and non-cooperative modes, for both $\alpha = 2$ and $\alpha = 4$, reporting our results in Fig. 6.12. We note that in both network environments, the cooperative mode requires slightly more power per bit, since relays are now assisting in data transmission. The power expenditure is higher in the high pathloss case; it is being used to overcome spatial isolation. In the cooperative mode, about the same amount of power per bit of throughput is required in both environments, though the cooperative mode gives substantially higher throughput, as seen in Figure 6.11.

6.6.2 Use of Cooperative Terminals

Throughput gains are a meaningful metric, but more insight results from studying the amount of cooperation required to realize these gains. We next look at what fraction of flows are allocated one or more cooperative links. We simulate 1000 topologies with the given number of flows, and count the number of flows for which cooperation was employed. The results are reported in Figure 6.13. As expected, the higher throughput seen under light load is the result of almost all flows using cooperative links, and because the flows are uniformly distributed geographically, there are always about 20% of them which can exploit a cooperative link. For the same reason, about the same number of flows employ cooperation regardless of the pathloss parameter (network isolation). While the optimal conditions (distances between terminals) for cooperation will change with α, the likelihood of a set of terminals meeting those conditions is independent of it–hence the curves overlap.

We see the pathloss effects when we study the fraction of cooperative links in the

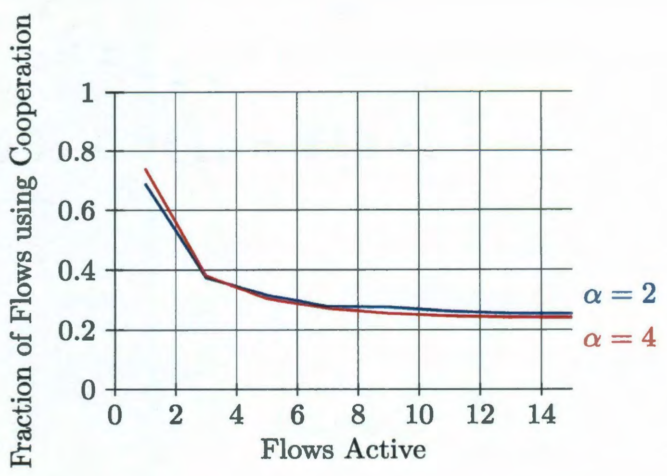


Figure 6.13: Fraction of flows for which one or more cooperative links are part of the allocation.

route, reported in Figure 6.14. Fewer cooperative links occur for $\alpha = 4$, because the cooperative advantage in equation (6.4) is dependent on the source-destination link. The alternative to cooperation is routing through the relay, which–because pathloss is high–is preferred when $\alpha = 4$, unless the relay is in a DF-optimal location; this happens with less likelihood under high pathloss than low pathloss. However, where cooperation is used, it offers large gains. This is why–even though we see fewer cooperative units employed under high pathloss–the high pathloss environment sees larger gains from cooperation.

6.7 Remarks

We have studied the problem of incorporating an advanced physical layer technique–cooperation–into a broader network environment. Cooperation when used in isolation is only advantageous to network performance, but the interference it causes in the larger network can be damaging to frames of other flows passing nearby. As such, cooperation cannot be used to its full benefit unless we account for the interference

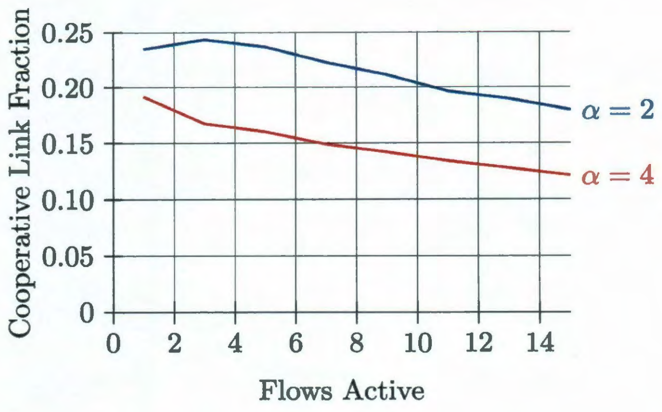


Figure 6.14: For flows employing cooperation, the fraction of links in the path which are cooperative.

it generates, as well as the interference it experiences.

Topological conditions in conjuction with traffic patterns fully dictate when cooperation should be used. We have presented general tools for attacking a topologically-specific problem, reporting performance improvements as a result. Our framework computes schedules and routes, systematically controlling the rate and power of each cooperative and non-cooperative link. With this representation, we are able to precisely localize the use of cooperation, ensuring that it aids the flow in question without hindering others. Further, because our technique computes the *entire* allocation for a frame, we are able to reduce interference from cooperative links by adjusting the power use at the three terminals involved–all in polynomial time.

The generality of our approach suggests its use when studying other physical layer techniques: cooperation with more than one relay, network coding, etc. For each of these, we look towards distributed implementations of the allocation, executing NFIC-like algorithms on a local view of the network.

Note that nothing about the representation of cooperative links requires full network knowledge. In particular, the algorithms discussed above are scalable, and will return at worst an empty set of metanodes for a given terminal. This means that we can deploy metanodes *in network neighborhoods*, as established in Chapter 5. This allows the cooperative advantage to be delivered on a local scale, as we discuss in the following unifying chapter.

CHAPTER 7

Conclusions

The main result of our work is the development and characterization of an efficient resource allocation framework for large networks. As discussed in the Introduction, we set the goal of determining a resource allocation scheme which could operate in a variety of network environments, for a variety of optimality criteria, and employing a plurality of communication technologies.

The preceding chapters have developed our ideas sequentially. We proposed a general system model, emphasizing spatial and temporal reuse opportunities in large networks. While we consider a single shared bandwidth here, we can imagine the same approach generalizing to a system with several bands by adding edges to the NFIC representing the separate channels. In that case, the NFIC becomes a *hypergraph*, requiring minor changes to the algorithms in order to calculate the allocation.

Using the NFIC, we have been able to perform resource allocation in very large networks in relatively short time: cubic with the number of terminals in the network. After showing how we are able to distribute this approach, we turned our attention to our original goal: an efficient method to allocate resources with multiterminal physical-layer links.

Figure 1.3 depicts how these different information states and physical-layer tech-

	Full	Partial
Point-to-Point	0.42, 0.09	0.05, 0.14
Cooperation	0.87, 0.17	0.05, 0.12

Table 7.1: Throughputs for the two flows shown in Figure 7.1 as a function of network knowledge and physical layer technology.

nologies may be combined using our common approach, though we have not yet shown a single example in which all four combinations are considered. To unify these different elements of the work and demonstrate the flexibility of this approach, we now present the last example of the thesis.

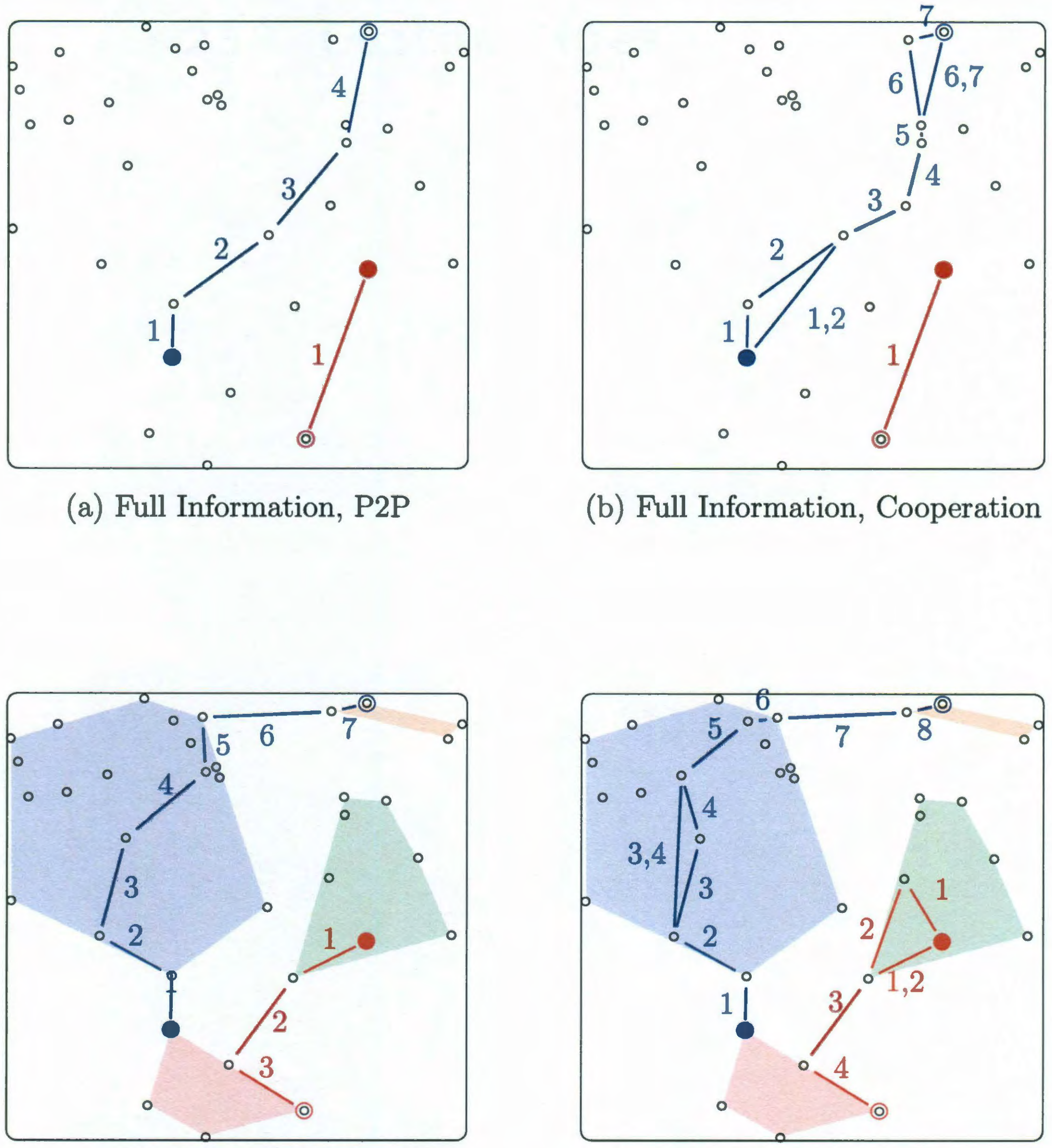
7.1 Unifying Example

In Figure 7.1, we show four panes. Each depicts the same network and the same two flows, the blue flowing "northward" and the red flowing in the other direction. The four cases of our allocation are shown in the four panes, representing the four combinations of network knowledge and physical-layer technologies shown in Figure 1.3.

7.1.1 Full Information with Non-Cooperative Links

Shown in pane (a) is the allocation computed if the NFIC is fully known, and there are only point-to-point links in the network. This is the execution of the algorithm in Chapter 4 with no changes. The result is a more-or-less direct allocation for the blue flow, and a single transmission for the red flow. The red flow, allocated second, has only one alternative to the single hop: the terminal to the south-west of its transmitter. However, with the blue flow receiving a frame at the terminal just north in the second timeslot, the red flow is power-limited in the second timeslot.

As a result, our algorithm selects the single hop for the red flow. The resulting



(a) Full Information, P2P

(b) Full Information, Cooperation

(c) Distributed Information, P2P

(d) Distributed Information, Cooperation

Figure 7.1: Resource allocation under four different paradigms: full information (a), full information with cooperation (b), distributed information (c), distributed information with cooperation (d).

throughput for these flows is 0.42 and 0.09 b/s/Hz respectively.

7.1.2 Full Information with Cooperation

In pane (b) is the result of allocation with cooperation available, as described in Chapter 6. Here, we define three relays per terminal, so that the NFIC is four times larger than it is in pane (a). The red flow is not helped by cooperation, but the blue flow is considerably. Note in particular that the two longest hops in the non-cooperative allocation are now aided by cooperation.

This results in increased throughput for the system, of 0.87 and 0.17 b/s/Hz for the blue and red flows respectively. Note in particular that the red flow experiences nearly twice the rate; this is a direct consequence of the power allocation method detailed in Chapter 6, as it applies to the cooperative link for the blue frame. By using cooperation, the power required by the blue flow in the first timeslot is lower than in the point-to-point case, reducing the interference for the red frame. This results in higher performance.

7.1.3 Partial Information with Non-Cooperative Links

Pane (c) depicts the result of allocation using partial information, clustering, and gateway transfers as discussed in Chapter 5. The blue flow must traverse three clusters, and is routed optimally within each one although this is evident only in blue cluster. Here the blue cluster allocates the frame, unaware that the 6th hop will bottleneck the rate on this frame. The red frame now requires three hops, but this is the minimum given the topology and clustering: one hop to and from the gateway terminals, and then one hop across the gateway link itself.

The throughput here are worse than in the full-information case for the blue flow, 0.05 and 0.14 b/s/Hz. Here, throughput for the red flow actually *increases* over the full information case. This is an interesting scenario; in the full-information case, the red flow is prevented from transmitting in timeslots 2 and 3 due to the presence of the blue flow. However, since its presence is not known with partial information, the red flow is able to use a higher-throughput multi-hop route. This increases its own throughput, but at substantial cost to the performance of the blue flow.

However, it is important to note that the *max-min* throughput declined in as consequence of the distributed allocation. Here the minimum is 0.05 b/s/Hz, while it was 0.09 in the full information case. So while we see an increase in the performance of one flow, overall-fair performance declined.

7.1.4 Partial Information with Cooperation

In pane (d), we synthesize our techniques from this thesis: here we have distributed information, but cooperative links are permitted within each cluster. The results are as we expect: the longest legs on each frame are aided by cooperation where possible: within the green cluster for the red frame and within the blue cluster for the blue frame.

The throughput results here are 0.05 and 0.14 b/s/Hz. Here, throughput for the blue flow is the same as in the partial-information point-to-point case, since the cooperative link does not improve performance over the bottleneck link in timeslot 7. The same is true for cooperation in the red flow; it does not alleviate the subsequent bottleneck link.

This example summarizes our contributions in this thesis: an extensible and flexible framework for resource allocation in a large variety of network environments, allowing for the fine grained study (in both space and time) of how data frames interact with each other and the network charged with their transmission.

7.2 Concluding Remarks

In this work, we have answered the questions and addressed the issues surrounding resource allocation in a multiflow wireless networks. Our results lie in three key areas:

- System Modeling
- Decomposition
- Algorithm Design

In the vein of system modeling, we developed an information-communication model which, under mild conditions, may be used to represent a wide variety of physical-layer technologies. Our model hinges on information-theoretic modeling of links between terminals, which is a function of a single parameter: power available at each terminal.

We have developed a graphical model of network activity capturing both temporal and spatial interactions of frames. We term this model the *Network Flow Interaction Chart* (NFIC), because it captures the interactions of frames and the terminals handling them in both space and time. The model is general and extensible, though in this work we have considered only a fully connected network without any backhauled access-points.

Based on this model, we have developed a decomposition technique for the problem, along with an efficient dynamic programming algorithm to solve each of the sub-problems. This results in a per-frame complexity of $O(N^3)$ time, well below the NP complexity of examining each possible route. Further, because we consider only a single frame at a time, we resolve the convexity issue in the mixed-integer program; the allocation for one frame is strictly convex with high probability, permitting the deployment of fast and efficient dynamic programming techniques.

7.3 Future Directions

We have studied a particular instance of the resource allocation problem in large networks. Our work has been made as intentionally general as possible, so as to apply to a variety of different environments and optimization objectives. In the future, it would be possible to deploy NFIC techniques in the following environments:

- **Generalized Multiterminal Schemes.** The generality of our NFIC, specifically–the nodes in the NFIC–allows us to represent any multiterminal physical layer scheme as an individual routing options, provided it has an information theoretic (or at minimum, single-variable) representation. The study of these techniques in broader networks may be facilitated by our technique.

- **Mesh Networks.** Mesh networks have access points with near-infinite bandwidth between them. These can be modeled as special nodes in the NFIC, with special edges. When designing where to install these access terminals, NFIC methods may shed light on relative optimality of different geographical deployments for a given user model.

- **Bidirectional Communication.** How do frames flowing in both directions interact with other, how does that affect route multiplicity? Here, we have considered unidirectional flows. When frames flow in two directions, they will be directly interfering with each other. NFIC techniques can help design routing protocols and collision management schemes taking advantage of spatial and temporal resource gaps.

- **Queuing Models.** Adding frames queued at terminals changes the weights of edges, and can lead to an implementation of the backpressure algorithm on the NFIC.

This is just a partial list of potential adaptations and extensions of our work in the area of wireless communications. The mixed-integer, non-convex model we study here appears in a wide variety of application settings. We conclude by mentioning some of the problem classes which may be attacked by NFIC methods, provided a good characterization of the problem's structure is available. These examples all assume a temporal axis, although this is by no means a limiting criterion for use of the NFIC.

- **Supply-delivery Scheduling and Routing.** This application is most similar to our model: the integer program is the binary state of a delivery vehicle, the terminals are supply depots, and the continuous program is the amount of a commodity transferred. This is often termed the *Multi-commodity flow problem* in the operational research community, in which the objective is to minimize total time en-route, or total shipping cost.

- **Airline and Crew Scheduling.** Building on the previous model; the integer program is the scheduling (or not) of a particular jetliner to a particular destination in some time period. The continuous program is the number of passengers (or crews) required aboard the plane, which may be generalized using a cost function depending on the type of aircraft, length of the flight, etc. The objective may be to minimize total operating costs, or maximize passenger throughput between some set of origin-destination cities. A similar problem is **air traffic control**, in which several aircraft must be scheduling for landing on any of a number of different runways at an airport. Here, collisions are literally fatal, and we may wish to minimize total fuel burnt on approach or time spent in a holding pattern.

- **Portfolio Management.** Here, the integer program is the holding status of a security, i.e. whether it is included in the portfolio. The continous program is therefore the amount of money invested in the holding. A good (or at least, more reliable than chance) statistical forecast model is required for this scenario, such that the interactions of different portfolio assets can be studied several time periods ahead of the present. Here, the objective may be to either minimize risk exposure (diversification) or to maximize profit by exploiting arbitrage opportunities.

- **Medical Drug Interaction.** Perhaps the most interesting example, here we consider an integer program of drug administration over some time steps (possibly hours). The objective function may be as simple as a patient's temperature, or as complex as the proliferation of a cancer. By modeling how the administration of different medications, at different times, and in different combinations affects a patient's body, our techniques can be used to chart an optimal course of treatment. As in the previous case, this requires a detailed and reliable characterization of the drugs in question and their effect on the patient at various time horizons.

We expect that the insights gained in this thesis will shed light on solution techniques for these types of problems, though with considerable adaptation of our approach required. Such adaptation will need to be specific to the problem at hand, and will require a detailed knowledge of how the different elements of the model interact with each other. In our case, it was through the SINR term of Shannon's logarithmic equation. Once that interaction has been accurately characterized, our techniques offer fast methods of computing near-optimal solutions to a large class of NP-Hard problems.

References

[1] V. Aggarwal, Y. Liu, and A. Sabharwal, "Sum-capacity of interference channels with a local view: Impact of distributed decisions," *submitted to IEEE Transactions on Information Theory*, October 2009.

[2] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows.* Prentice Hall, 1993.

[3] J. G. Andrews, N. Jindal, M. Haenggi, R. Berry, S. Jafar, D. Guo, S. Shakkottai, R. Heath, M. Neely, S. Weber, and A. Yener, "Rethinking information theory for mobile ad-hoc networks," *IEEE Communications Magazine*, vol. 46, no. 12, pp. 94–101, December 2008.

[4] A. Azgin, Y. Altunbasak, and G. Al-Regib, "Cooperative mac and routing protocols for wireless ad hoc networks," in *Proceedings IEEE Global Telecommunications Conference GLOBECOM '05*, December 2005, pp. 2859–2865. 1.1.2

[5] R. Babaee and N. C. Beaulieu, "Cross-layer design for multihop wireless relaying networks," *IEEE Transactions on Wireless Communications*, vol. 9, no. 1, pp. 3522 – 3531, November 2010.

[6] R. Bellman, "On a routing problem," *Quartely Applied Mathematics*, vol. 16, no. 1, pp. 87–90, 1959. 1.1

[7] R. Bhatia and M. Kodialam, "On power efficient communication over multi-hop wireless networks: Joint routing, scheduling and power control," in *Proceedings of the 23rd Annual IEEE INFOCOM*, March 2004, pp. 1457–1466. 1.1.1

[8] V. R. Cadambe and S. A. Jafar, "Interference alignment and spatial degrees of freedom for the k user interference channel," in *IEEE International Conference on Communications*, May 2008, pp. 971–975.

[9] G. Caire, D. Tuninetti, and S. Verdu, "Suboptimality of tdma in the low-power regime," *IEEE Transactions on Information Theory*, vol. 50, no. 4, pp. 608–620, April 2004. 4.8.1, 1

[10] D. Chafekar, V. S. A. Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan, "Approximation algorithms for computing capacity of wireless networks with SINR constraints," in *Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing MOBIHOC*, May 2008, pp. 91–100. 1.1

[11] L. Chen, S. H. Low, M. Chien, and J. C. Doyle, "Cross-layer congestion control, routing and scheduling design in ad hoc wireless networks," in *Proceedings of the 25th Annual IEEE INFOCOM*, April 2006, pp. 1–13. 1.1.1

[12] C.-C. Chian, H.-K. Wu, W. Liu, and M. Gerla, "Routing in clustered multihop, mobile wireless networks with fading channel," in *IEEE SICON '97*, April 1997, pp. 197–211. 5.1.2

[13] T. H. Corman, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms.* McGraw-Hill, 1996. 1.1, 3.5, 6.5.4

[14] T. M. Cover and A. E. Gamal, "Capacity theorems for the relay channel," *IEEE Transactions on Information Theory*, vol. 25, pp. 572–584, September 1979. 1.1.2, 6.1

[15] R. Cruz and A. V. Santhanam, "Optimal routing, link scheduling and power control in multi-hop wireless networks," in *Proceedings of IEEE INFOCOM 2003*, April 2003, pp. 702–711. 1.1.1

[16] M. Dehghan, M. Ghaderi, and D. L. Goeckel, "On the performance of cooperative routing in wireless networks," *IEEE INFOCOM*, vol. 50, no. 4, pp. 1–5, March 2010. 1.1.2

[17] E. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959. 1.1

[18] L. Ding, T. Melodia, S. Batalama, and J. Matyjas, "Distributed routing, relay selection, and spectrum allocation in cognitive and cooperative ad hoc networks," in *Proceedings IEEE Intl. Conf. on Sensor, Mesh and Ad Hoc Communications and Networks SECON*, June 2010, pp. 1 – 9. 1.1.2

[19] R. Draves, J. Padhye, and B. Zill, "Comparison of routing metrics for static multi-hop wireless networks," in *IEEE SIGCOMM '04*, October 2004, pp. 32–48. 5.2.1

[20] M. Ebrahimi, M. A. Maddah-Ali, and A. K. Khandani, "Throughput scaling laws for wireless networks with fading channels," *IEEE Transactions on Information Theory*, vol. 51, no. 11, pp. 4250–4254, November 2007.

[21] G. Egeland and P. E. Englestad, "The availability and reliability of wireless multi-hop wireless networks with stochastic link failures," *IEEE JSAC*, vol. 27, no. 7, pp. 1132 – 1146, September 2009.

[22] T. ElBatt and A. Ephremides, "Joint scheduling and power control for wireless ad hoc networks," *IEEE Transactions on Wireless Communications*, vol. 3, no. 1, pp. 74 – 85, January 2004. 1.1.1

[23] L. R. Ford and D. R. Fulkerson, *Flows in Networks.* Princeton University Press, 1962.

[24] M. Franceschetti, O. Dousse, D. N. C. Tse, and P. Thiran, "Closing the gap in the capacity of wireless networks via percolation theory," *IEEE Transactions on Information Theory*, vol. 53, no. 3, March 2007. 1.1

[25] M. Franceschetti, M. D. Migliore, and P. Minero, "The capacity of wireless networks: Information theoretic and physical limits," *IEEE Transactions on Information Theory*, vol. 55, no. 8, August 2009. 1.1

[26] A. E. Gamal and T. M. Cover, "Multiple user information theory," *Proceedings of the IEEE*, vol. 68, no. 12, pp. 1466–1483, December 1980.

[27] A. E. Gamal and S. Zahedi, "Capacity of a class of relay channels with orthogonal components," *IEEE Transactions on Information Theory*, vol. 51, no. 5, pp. 1815–1817, May 2005.

[28] M. Gastpar and M. Vetterli, "On the capacity of large gaussian relay networks," *IEEE Transactions on Information Theory*, vol. 51, no. 3, pp. 765–779, March 2005.

[29] L. Georgiadis, M. J. Neely, and L. Tassiulas, *Resource Allocation and Cross Layer Control in Wireless Networks.* Now Publishers Inc., 2006. 1.1.1

[30] M. Gerla and C. R. Lin, "Adaptive clustering for mobile wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 7, pp. 1265–1275, September 1997.

[31] R. Gummadi, K. Jung, D. Shah, and R. Sreenivas, "Computing the capacity region of a wireless network," in *Proceedings of the 28th Annual IEEE INFOCOM*, April 2009, pp. 1341–1349.

[32] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 388–404, March 2000. 1.1

[33] ——, "Towards an information theory of large networks: An achievable rate region," *IEEE Transactions on Information Theory*, vol. 49, no. 8, pp. 1877–1894, August 2003.

[34] M. Haenggi, "Outage, local throughput, and the capacity of random wireless networks," in *Proceedings of the IEEE International Symposium on Information Theory*, September 2005, pp. 2070–2074.

[35] E. L. Hahne, "Round-robin scheduling for max-min fairness in data networks," *IEEE Journal on Selected Areas in Communication*, vol. 9, no. 7, pp. 1024–1039, September 1991.

[36] B. Hajek and G. Sasaki, "Link scheduling in polynomial time," *IEEE Transactions on Information Theory*, vol. 34, no. 5, pp. 910 – 917, September 1988. 1.1.1

[37] Z. Han and H. V. Poor, "Impact of cooperative transmission over network routing," *Cooperative Wireless Communications*, May 2009. 1.1.2

[38] K. Hong, Y. Hua, and A. Swami, "Distributed and cooperative link scheduling for large-scale multihop wireless networks," *EURASIP Journal on Wireless Communication Networks*, vol. 2007, pp. 41–49, October 2007.

[39] A. Høst-Madsen and J. Zhang, "Capacity bounds and power allocation for wireless relay channels," *IEEE Transactions on Information Information Theory*, vol. 51, no. 6, pp. 2020–2040, December 2005. 6.2.1

[40] C. Hunter, P. Murphy, and A. Sabharwal, "Real-time testbed implementation of a distributed cooperative MAC and PHY," in *Proceedings of Conference on Information Science and Systems CISS*, 2010, pp. 1–6.

[41] A. S. Ibrahim, Z. Han, and K. J. R. Liu, "Distributed energy-efficient cooperative routing in wireless networks," *IEEE Transactions on Wireless Communications*, vol. 7, no. 10, pp. 3930–3941, October 2008. 1.1.2

[42] P. Jacquet, B. Mans, P. Muhlethaler, and G. Rodolakis, "Opportunistic routing in wireless ad hoc networks: Upper bounds on packet propagation speed," *IEEE JSAC*, vol. 27, no. 7, pp. 1192–1202, September 2009.

[43] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu, "Impact of interference on multi-hop wireless networks," *Springer Wireless Networks*, vol. 11, no. 4, pp. 471–487, July 2005. 1.1.1

[44] R. K. Jain, D.-M. W. Chiu, and W. R. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer systems," DEC Technical Research Report TR-301, Tech. Rep., September 1984.

[45] I. R. John M. McQuillan and E. C. Rosen, "The new routing algorithm for the ARPANet," *IEEE Transactions on Communications*, vol. 28, no. 5, pp. 711–719, April 1980.

[46] C. Joo and N. B. Shroff, "Performance of random access scheduling schemes in multihop wireless networks," in *Proceedings of the 26th Annual IEEE INFOCOM*, June 2007, pp. 19–27. 1.1.1

[47] A. Jovicic, P. Viswanath, and S. R. Kulkarni, "Upper bounds to transport capacity of wireless networks," *IEEE Transactions on Information Theory*, vol. 50, no. 11, pp. 2555–2565, November 2004. 1.1

[48] V. Kawadia and P. R. Kumar, "A cautionary perspective on cross layer design," *IEEE Wireless Communication Magazine*, vol. 12, no. 1, pp. 3–11, February 2005.

[49] A. E. Khandani, "Cooperative routing in wireless networks," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, 2004.

[50] A. Khandani, J. Abounadi, E. Modiano, and L. Zheng, "Cooperative routing in static wireless networks," *IEEE Transactions on Communications*, vol. 55, no. 7, pp. 2185 – 2192, November 2007. 1.1.2

[51] M. A. Khojastepour, "Distributed cooperative communications in wireless networks," Ph.D. dissertation, Rice University, Houston, TX, 2005.

[52] J. Kleinberg, Y. Rabini, and E. Tardos, "Fairness in routing and load balancing," October 1999, pp. 27–38.

[53] M. Kodialam, T. V. Lakshman, and S. Sengupta, "Capacity of multi-hop wireless networks with incomplete traffic specification," in *Proceedings of the 28th Annual IEEE INFOCOM*, April 2009, pp. 2536–2540.

[54] M. Kodialam and T. Nandagopal, "Characterizing achievable rates in multihop wireless networks: The joint routing and scheduling problem," in *Proceedings of ACM Mobicom 2003*, 2003, pp. 42–54. 1.1.1

[55] R. Koetter, M. Effros, and M. Medard, "On a theory of network equivalence," in *IEEE Information Theory Workshop*, June 2009, pp. 326–330.

[56] S. Kompella, J. E. Wieselthier, and A. Ephremides, "Multihop routing and scheduling in wireless networks subject to sinr constraints," in *46th IEEE Conference on Decision and Control*, December 2007, pp. 5690–5695. 1.1.1

[57] G. Kramer, M. Gastpar, and P. Gupta, "Cooperative strategies and capacity theorems for relay networks," *IEEE Transactions on Information Theory*, vol. 51, no. 9, pp. 3037–3063, September 2005.

[58] S. R. Kulkarni and P. Viswanath, "A deterministic approach to throughput scaling in wireless networks," *IEEE Transactions on Information Theory*, vol. 50, no. 6, pp. 2555–2565, June 2004.

[59] A. Kumar and J. Kleinberg, "Fairness measures for resource allocation," *SIAM Journal on Computing*, vol. 36, no. 3, pp. 657–680, August 2006.

[60] M. Kurth, A. Zubow, and J.-P. Relich, "Cooperative opportunistic routing using transmit diversity in wireless mesh networks," in *Proceedings of 27th IEEE INFOCOM*, April 2008, pp. 1310 – 1318. 1.1.2

[61] H. J. Kushner and P. A. Whiting, "Convergence of proportional-fair sharing algorithms under general conditions," *IEEE Transactions on Wireless Communications*, vol. 3, no. 4, pp. 1250–1259, July 2004.

[62] P. Kyasunar and N. H. Vaidya, "Capacity of multichannel wireless networks under the protocol model," *IEEE/ACM Transactions on Networking*, vol. 17, no. 2, April 2009.

[63] J. N. Laneman, D. N. C. Tse, and G. W. Wornell, "Cooperative diversity in wireless networks: Efficient protocols and outage behavior," *IEEE Transactions on Information Theory*, vol. 50, no. 12, pp. 3062 – 3080, December 2004.

[64] O. Leveque and I. E. Telatar, "Information-theoretic upper bounds on the capacity of large extended ad hoc wireless networks," *IEEE Transactions on Information Theory*, vol. 46, no. 3, March 2005. 1.1

[65] S. Li, Y. Liu, and X.-Y. Li, "Capacity of large-scale wireless networks under the gaussian channel model," in *Proceedings of the 14th ACM international conference on Mobile computing and networking MOBICOM*, 2008, pp. 140–151.

[66] Y. Li and A. Ephremides, "A joint scheduling, power control, and routing algorithm for ad hoc wireless networks," *Ad Hoc Networks*, vol. 5, no. 7, pp. 959 – 973, September 2007. 1.1.1

[67] 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, August 2006. 1.1.1

[68] Y.-H. Lin, T. Javidi, R. L. Cruz, and L. B. Milstein, "Distributed link scheduling, power control and routing for multi-hop wireless mimo networks," *Proceeding of 2006 Asilomar Conference on Signals and Systems*, pp. 122–126, November 2006.

[69] Lindo global optimization solver. LINDO Systems, INC. Chicago, IL. [Online]. Available: http://www.lindo.com 3.2

[70] R. Madan and D. Shah, "Capacity-delay scaling in arbitrary wireless networks," in *Proceedings of the Allerton Conference on Communication, Control and Computing*, September 2005.

[71] R. Madan, D. Shah, and O. Leveque, "Product multicommodity flow in wireless networks," *IEEE Transactions on Information Theory*, vol. 54, no. 4, pp. 1460–1476, April 2008. 1.1.1

[72] X. S. Mehrdad Dianait and S. Naik, "A new fairness index for radio resource allocation in wireless networks," in *IEEE Wireless Communications and Networking Conference*, vol. 2, March 2005, pp. 712–717.

[73] G. B. Middleton, B. Aazhang, and J. Lilleberg, "A flexible framework for polynomial-time resource allocation in multiflow wireless networks," in *Proceedings of the 47th Allerton Conference on Communication, Control and Computing*, September 2009, pp. 1126 – 1133. 5.4

[74] ——, "Efficient resource allocation and interference management for streaming multiflow wireless networks," in *IEEE International Conference on Communications ICC*, May 2010, pp. 1 – 5.

[75] ——, "Efficient resource allocation and interference management for streaming multiflow wireless networks," in *Proceedings IEEE Intl. Conf. on Communications*, May 2010, pp. 1 – 5.

[76] E. Modiano, D. Shah, and G. Zussman, "Maximizing throughput in wireless networks via gossiping," *ACM SIGMETRICS Performance Evaluation Review*, vol. 34, no. 1, pp. 27–38, June 2006. 1.1.1

[77] T. Moscibroda and R. Wattenhofer, "The complexity of connectivity in wireless networks," in *Proceedings of the 25th Annual IEEE INFOCOM*, April 2006, pp. 1–13. 1.1

[78] P. Murphy, C. Hunter, and A. Sabharwal, "Design of a cooperative ofdm transceiver," in *Proceeding of 2009 Asilomar Conference on Signals and Systems*, November 2009, pp. 1058–6393. 6.1

[79] P. Murphy, A. Sabharwal, and B. Aazhang, "On building a cooperative communication system: Testbed implementation and first results," *EURASIP Journal on Wireless Communications and Networking*, vol. 2009, 2009.

[80] ——, "On building a cooperative communication system: Testbed implementation and first results," *EURASIP Journal on Wireless Communications and Networking*, vol. 2009, June 2009. 6.1

[81] U. Niesen, P. Gupta, and D. Shah, "On capacity scaling in arbitrary wireless networks," *IEEE Transactions on Information Theory*, vol. 55, no. 9, pp. 3959–3982, September 2009.

[82] A. E. Ozdaglar and D. P. Bertsekas, "Optimal solution of integer multicommodity flow problems with application in optical networks," *Proc of Symposium on Global Optimization*, 2003.

[83] A. Ozgur, R. Johari, D. N. C. Tse, and O. Leveque, "Information theoretic operating regimes of large wireless networks," in *Proceedings of the IEEE International Symposium on Information Theory*, July 2008, pp. 186–190.

[84] A. Ozgur, O. Leveque, and D. N. C. Tse, "Hierarchical cooperation achieves optimal capacity scaling in ad hoc networks," *IEEE Transactions on Information Theory*, vol. 53, no. 10, pp. 3549–3572, October 2007. 1.1

[85] P. M. Pardalos, D. W. Hearn, and W. W. Hager, *Network Optimization.* Springer, 1997.

[86] B. Peis, M. Skutella, and A. Wiese, "Packet routing: Complexity and algorithms," *Approximation and Online Algorithms*, vol. 5893, pp. 217–228, 2010. 3.1

[87] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," *IEEE Workshop on Mobile Computing Systems and Applications*, pp. 90–100, February 1999. 5.2.1

[88] A. Puri and S. Tripakis, "Algorithms for routing with multiple constraints," *AIPS Workshop on Planning and Scheduling using Multiple Criteria*, 2002.

[89] P. Raghavan and C. D. Thompson, "Randomized rounding: A technique for provably good algorithms and algorithmic proofs," *Combinatorica*, vol. 7, no. 4, pp. 365–374, April 1987. 1.1.1

[90] C. Scheideler, A. W. Richa, and P. Santi, "An o(log n) dominating set protocol for wireless ad-hoc networks under the physical interference model," in *Proceedings of the 26th Annual IEEE INFOCOM*, June 2007, pp. 19–27. 1.1.1

[91] A. Sendonaris, E. Erkip, and B. Aazhang, "User cooperation diversity. part I. System description," *IEEE Transactions on Communications*, vol. 51, no. 11, pp. 1927 – 1938, November 2003. 6.1

[92] ——, "User cooperation diversity. Part II. Implementation aspects and performance analysis," *IEEE Transactions on Communications*, vol. 51, no. 11, pp. 1939 – 1948, November 2003.

[93] C. E. Shannon, "A mathematical theory of communication," *Bell Systems Technical Journal*, vol. 27, 1948. 2.2

[94] M. Shreedhar and G. Varghese, "Efficient fair queueing using deficit round robin," *ACM SIGCOMM Computer Communication Review*, vol. 25, no. 4, October 1995.

[95] C. Sorrel, "Reports: AT&T stops some iphone sales in NYC," *Wired Magazine*, December 2009. [Online]. Available: http://www.wired.com/gadgetlab/2009/12/att-stops-iphone-sales-in-nyc/ 1

[96] V. Srivastava and M. Motani, "Cross-layer design: a survey and the road ahead," *IEEE Communications Magazine*, vol. 43, no. 12, pp. 112–119, December 2005.

[97] L. Tassiulas and S. Sarkar, "Maxmin fair scheduling in wireless networks," in *Proceedings of the 21st Annual IEEE INFOCOM*, November 2002, pp. 763–772. 1.1.1

[98] E. C. Van der Meulen, "Three terminal communication channels," *Advances in Applied Probability*, vol. 3, no. 1, pp. 120–154, 1971. 1.1.2, 6.1

[99] E. J. Weisstein. Square line picking. Mathworld - A Wolfram Web Resource. [Online]. Available: http://mathworld.wolfram.com/SquareLinePicking.html

[100] A. Wittneben, I. Hammerstrom, and M. Kuhn, "Joint cooperative diversity and scheduling in low mobility wireless networks," *Proceeding of 2004 IEEE GLOBECOM*, vol. 2, no. 1, pp. 780–784, November 2004.

[101] J. Wu, M. Gao, and I. Stojmenovic, "On calculating power-aware connected dominating sets for efficient routing in ad hoc wireless networks," September 2001, pp. 346–354. 5.1.2

[102] L. Xiao, M. Johansson, and S. P. Boyd, "Simultaneous routing and resource allocation via dual decomposition," *IEEE Transactions on Communications*, vol. 52, no. 4, pp. 1136–1144, July 2004. 1.1.1

[103] L.-L. Xie and P. R. Kumar, "A network information theory for wireless communication: Scaling laws and optimal operation," *IEEE Transactions on Information Theory*, vol. 50, no. 5, pp. 748–767, May 2004. 1.1

[104] F. Xue, L.-L. Xie, and P. R. Kumar, "The transport capacity of wireless networks over fading channels," *IEEE Transactions on Information Theory*, vol. 51, no. 3, pp. 834–847, March 2004. 1.1

[105] Y. Xue, B. Li, and K. Nahrstedt, "Optimal resource allocation in wireless ad hoc networks: A price-based approach," *IEEE Transactions on Mobile Computing*, vol. 5, no. 4, pp. 347–364, April 2006. 1.1.1

[106] Z. Yang and A. Høst-Madsen, "Routing and power allocation in asynchronous gaussian multiple-relay channels," *EURASIP Journal on Wireless Communication and Networking*, vol. 2006, no. 2, April 2006.

[107] E. M. Yeh and R. A. Berry, "Throughput optimal control of cooperative relay networks," *IEEE Transactions on Information Theory*, vol. 53, no. 10, pp. 3827 – 3833, October 2007. 1.1.2

[108] Y. Yi, G. de Viciana, and S. Shakkottai, "On optimal MAC scheduling with physical interference," in *Proceedings of the 26th Annual IEEE INFOCOM*, June 2007, pp. 294 – 302. 1.1.1

[109] J. Y. Yu and P. H. J. Chong, “A survey of clustering schemes for mobile ad hoc networks,” *IEEE Communications Surveys and Tutorials*, vol. 7, no. 1, pp. 32–48, July 2005. 5.1.2

[110] J. Zhang and Q. Zhang, “Cooperative routing in multi-source multi-destination multi-hop wireless networks,” in *Proceedings of 27th IEEE INFOCOM*, April 2008, pp. 2369–2377. 1.1.2

[111] ——, “Contention-aware cooperative routing in wireless mesh networks,” *IEEE International Conference on Communications (ICC)*, pp. 1–5, June 2009. 1.1.2