Stochastic Routing in Ad Hoc Wireless Networks

Christopher Lott
and
Demosthenis Teneketzis

Department of Electrical Engineering and Computer Science
University of Michigan
1301 Beal Ave
Ann Arbor, MI 48109-2122

clott@eecs.umich.edu
teneketzis@eecs.umich.edu

Control Group Report CGR 01-01
February 9, 2001

This research was supported in part by ARO Grant DAAH04-96-1-0377, AFOSR Grants F49620-96-1-0028 and F49620-98-1-0370, and NSF Grant ECS-9979347.

Abstract

We investigate a time-invariant network routing problem where a probabilistic local broadcast model for wireless transmission is used. We discuss this model's key features. We present results showing that an index policy is optimal for the time-invariant routing problem. We extend the time-invariant model to allow for control of transmission type, and prove that the index nature of the optimal routing policy remains unchanged. We then allow time-varying system parameters in the original model, and discover conditions under which a time-varying index routing policy or a time-varying priority routing policy is optimal. Finally, we present three distributed implementations of an optimal routing policy for the time-invariant problem and provide results on their convergence properties.

1 Introduction

As communication networks become ever more prevalent, research to improve their design has become both more encompassing and more application specific. Protocol (policy) issues such as service priority, retransmission schemes, flow control, and routing are now studied for large distributed systems and for a variety of communication channel types. Further, it has long been recognized that integrating various source types, such as voice, video, email, web access, and business transaction data into one seamless transparent network results in both the highest efficiency and ease of use. In such networks the ability to provide appropriate Quality of Service (QoS) to each service type becomes paramount. A further challenge is posed by wireless networks, where these service goals must be achieved with unreliable and time-varying channels, and where new concerns, such as energy consumption and multi-user channel interference, impose additional constraints.

The ubiquitous TCP/IP protocol suite, originally developed for non-realtime data for a relatively fixed network with mostly reliable communication links, is not always easily adapted to satisfy the new conditions. For example, once Quality of Service (QoS) requirements, such as timeliness, are added, major design changes become necessary (e.g. [Diff 99]). Another concern is that unreliable links may cause havoc on routing and retransmission protocols designed without this concern in mind. Hence, much recent research has focussed on changes to the IP protocol to handle these new environments. Also, routing protocols for wireless networks must be able to adapt to relatively rapid changes in the node topology, making the dynamics of route computation an important issue.

The term *ad hoc* is often now applied to networks in which there is no central network controller, each node can itself act as a store-and-forward router, and in which the connectivity topology is time-varying (e.g. see [Haas 99]). Such a network is in contrast to, for example, a cellular network where each cell has a central base station through which all data is transmitted. We think of the network as having a number of packets, each of which is destined for some set of destinations. The general routing problem in an ad hoc network is to define a policy which, given the trajectory histories of all the packets, chooses which nodes should next transmit which packet. It is also desired that this policy be implementable in a distributed fashion in the network, so the transmission decisions can be decided locally without knowledge of other parts of the network.

Many approaches can be taken to network routing optimization. A typical one is maximizing the overall throughput the network achieves. But in many cases for the wireless environment other

considerations are at least as important. For example, often in a wireless network the energy source is locally stored in a battery at each node, and the major design goal is to achieve satisfactory communication while using up as little energy as possible. Or there may be other costs associated with each transmission, such as a penalty for signal interference, that need to be incorporated into the optimization criterion. Also, there may be QoS issues, such as delivery timeliness and packet priority, which should have an effect on the service policy.

This paper explores the abovementioned design issues in network routing algorithms for ad hoc wireless networks, and provides a novel system model which allows for optimal design in a number of realistic situations. It is organized as follows. In the remainder of Section 1 we present modeling and routing algorithm issues in ad hoc wireless networks that need to be taken into account in our problem. We also present briefly a discussion of the literature available on routing in ad hoc wireless networks, and a summary of the contributions of this paper. In Section 2 we present a qualitative description of our model. In Section 3 we define and solve a routing problem with time-invariant parameters. In Section 4 we extend this time-invariant routing problem to include choice of transmission type. In Section 5 we define a routing problem with time-varying parameters, and present two solutions valid only under restricted conditions. In Section 6, we present three distributed algorithms which compute the optimal routing policy of Section 3. Section 7 provides conclusions and ideas for future work.

1.1 Ad Hoc Wireless Network Modelling Issues

For our purposes, the model chosen to reflect the reality of a situation should embody the important pertinent truths while still allowing analytic insight. We list here some of the important modeling issues for an ad hoc wireless network, and also mention some of our modeling choices.

1.1.1 Level of Abstraction

It is important to distinguish between a high-level protocol definition and the detailed description necessary for its implementation. Generally, a high-level description does not make too many assumptions about what communications technology the protocol will be run on. For the routing problem, this means an appropriate link-level protocol is assumed, subject to the routing algorithm requirements. These requirements take the form of a model of link-level behavior assumed by the routing algorithm.

Our goal here is to define such a high-level routing protocol, and hence we specify a model for

how the link-level behaves. We then describe a routing protocol optimal for this system model, and present an algorithm which computes this protocol for any given system. The algorithm is designed to be implementable in a real distributed system. But there are many implementation details, such as how local control and information flow is transmitted, and communication channel allocation to each node, which we do not specify here. These aspects of the design are dependent on the communication technology used (which could be CDMA, frequency hop, FDMA, and so on). It is a topic for further study how such implementation details affect the actual performance of our protocol.

1.1.2 Channel Model

Wireless transmission is always somewhat uncertain. Channel noise over a changing distance, multipath fading, signal blockage or scatter, and interchannel interference can all adversely affect data reception. Routing researchers typically model these channels in a deterministic way, either as some kind of power threshold indicating communication is possible (e.g. [Rodo 99], [Wies 00]), or with a numeric link quality indicator which includes channel quality information, which is then used in a deterministic algorithm (e.g. [Purs 93],[Purs 99]). We take another approach, allowing for a probabilistic structure on transmission success, and allowing for routing decisions to be based on transmission result signaling from potential receivers. We then study how routing performance can be better optimized under this model. In this scheme, the local probabilistic structure must be continuously estimated by each node as the network functions.

1.1.3 Antenna Model

The majority of wireless applications involve mobility in the network. Given the uncertain nature of such a topology, it is standard for the radios to incorporate omnidirectional antennas, which effectively provide a local broadcast to all neighboring nodes. It is widely recognized that the omnidirectional nature of these wireless transmissions is a limiting factor in the performance of the network. This is due to the interference one transmitting mobile generates, affecting reception of other mobiles that are nearby. But it is also important to recognize that an omnidirectional transmission, potentially reaching multiple neighbors at once, can actually be used to advantage even by a unicast routing protocol, given the stochastic nature of wireless transmission. Our model will assume omnidirectional antennas, and use this local broadcast effect in the protocol.

1.1.4 Interchannel Interference

Assuming node motion to be outside the control of the routing protocol, there is only one source of channel disturbance which is actually under routing protocol control. This is the interference transmissions cause each other, and hence these disturbances merit special consideration. There is a rich research literature on the topic of power control for systems like cellular networks where network information and control flow is centralized. The problem is far more difficult in ad hoc networks, where a routing choice in one part of the network can produce a down-the-road interference effect far elsewhere, and where control decisions usually need to be made locally without knowledge of what other nodes are doing.

One approach to this difficulty is to assume that interference from other nodes is a phenomenon not unlike channel noise, in that it cannot be precisely predicted at each node given what the node knows. The idea is that from the point of view of each node, transmission in the network is random, so that the best that can be hoped for is to make some statistical model for the effect locally. Hence, interference effects get folded into the probabilistic transmission structure, which the node then estimates. This is the approach taken in this paper, though it is recognized that the detrimental effect of a transmission on neighbor performance is not adequately reflected in the cost structure being optimized.

1.1.5 Packet Size

The network is packet switched, and the allowed packet sizes is an important part of the system definition. In TCP/IP a wide variety of packet sizes is allowed. To achieve QoS requirements, some protocol designs, such as ATM, restrict packet size to a small fixed value and accept the resulting transmission overhead. It is a network modeling issue whether or not a routing algorithm is faced with sending packets of different sizes.

In our model the packet size will be fixed. This is due to the fact that we restrict attention to the problem of transmitting one packet from its source to its destination, and hence the same packet is transmitted at each time step. In Section 2.5 we justify our focus on this one problem in this paper.

1.1.6 Destination Description

A routing problem can be categorized as unicast, multicast, broadcast, or anycast. In unicast, the destination is one particular node. In multicast, it is a set of nodes, and the packet should be

sent to all of them. In broadcast, the packet should be sent to all nodes in the network. Though unicast and broadcast can be thought of as special cases of multicast, usually quite different routing algorithms exist for each case. Hence, it is convenient to consider each as a separate problem. In anycast, the destination is a set of nodes, and the packet should be sent to *any* node in the set.

In this paper, we concentrate exclusively on an anycast problem. However, we specify our problem a little more generally, in that we allow the reward to vary depending on the destination node. Once some destination node has been reached, a decision can be made whether to continue transmitting to reach a node of higher reward, or to stop and take the current reward. It is important to emphasize that only a single node's reward is ever received.

1.1.7 Quality of Service

The performance of a given routing algorithm may depend on multiple factors. For example, energy expended on getting to the destination might be a concern which competes with QoS requirements, such as how quickly the destination is reached. To handle these situations, we include in our most general system model time-varying parameters which capture message priority and the need for delivery timeliness.

1.2 Ad Hoc Wireless Network Routing Algorithm Issues

A number of issues arise when designing a routing algorithm for ad hoc wireless networks, some of them particular to this problem. Of course, the decisions made in designing the algorithm reflect the system model under consideration. In this section we summarize some of these issues, and briefly discuss our own approach.

1.2.1 Hierarchical vs. Flat

One way to deal with the complexity of an ad hoc network is to divide it into hierarchical groupings (often called clusters) (e.g. [McDo 99]). Communication then always goes up and down the hierarchy through cluster heads. Part of the design problem then becomes to define these clusters based on the known details of the network's information flow.

Another approach is to give each node the same routing responsibility with no such groupings. This is perhaps appropriate when little is known ahead of time of the relations among the nodes. Also, even in the cluster approach the routing of information within clusters can be considered a flat routing problem to the cluster head.

Our work considers optimal routing problems within a flat node hierarchy.

1.2.2 Distributed Implementation

An *ad hoc* network consists of a group of nodes distributed in space, each running its own local controller. As such, it is desirable that the routing protocol not require any centralization of information or control. That is, we desire our protocol to run at each node and only use information locally available, in a sense to be made precise. One approach is to only consider policies that have this character, and find the optimal one within this class. However, it is often convenient to think of the routing problem in the context of fully centralized information and control, to derive the optimal protocol for the centralized problem, and then to see if a distributed implementation can be made of it. We will take the latter point of view in this paper.

1.2.3 On Demand vs. Route Maintenance

In a highly dynamic network where packets are rare compared to motion, it might be wasteful to constantly update the network topology, since most of the time no packets are transmitted. Instead, it may be better for the protocol to compute the route for a packet only when the packet itself appears, discovering the route in the current topology. Such protocols are often called "on demand" or "reactive" (e.g. [Cors 95]).

The other approach is to always maintain a set of routes in the network. Protocols with this feature are described as "route maintenance" or "proactive". In a network with a sufficient quantity of traffic, the overhead to accomplish this might be relatively small. Also, often the routing information can be efficiently piggybacked on data packets.

We develop our algorithm using the route maintenance approach, but it is important to keep in mind the conditions under which this approach makes sense.

1.2.4 Optimization vs. Sufficiency

The criterion used to determine a route affects the algorithm design. Possibly it is desired to optimize some system variable, such as energy expenditure or average route delay, for each route. It might then be worthwhile to spend the effort to find the best route for this variable. Alternatively, for example in a high dynamic and low traffic situation, the effort to find a best route might outweigh the benefit from actually using it. In these situations it is better to just guarantee that *some* route is found, and do so with minimal effort.

In this paper we take the former approach, and focus on algorithms which minimize a system variable, which for simplicity of exposition we take to be energy expenditure. This is consistent with our assumption of sufficient traffic to justify route maintenance, and is appropriate for a number of potential applications, such as battery-powered independent mobile units.

1.2.5 Link State Detection

Routing algorithms must make decisions about the status of different links in the network. For example, in typical on-demand protocols link status, either up or down, throughout the network is used to find a path to the destination. The method for determining the link state plays a key role in how a routing algorithm functions. In many protocols, information from the link layer, or Medium Access Control (MAC) layer, is fed up to the routing protocol for this purpose.

In our model the link state information is incorporated into the transmission probability structure, which suits our stochastic channel model. It might be possible to include MAC information in the estimation of these transmission probabilities. It is interesting to investigate how well other algorithms determine link state in the stochastic environment.

1.2.6 Link State vs. Distance Vector

Optimal routing algorithms can generally be classified as either link-state or distance vector. In link-state, network topology information is transmitted to each node, which can then choose routes accordingly. As a typical example, the TCP/IP routing protocol OSPF (Open Shortest Path First) [Moy 91] is a link-state protocol. In distance vector, each node stores a best next hop and resulting cost for each destination, and neighbors share this information. From the neighbors' values, a node can set its own value for each destination. A typical example of a distance vector protocol is TCP/IP's RIP [Malk 93]

If each node in the network transmits local topology information to all other nodes, this requires $O(N^2)$ packet deliveries ([Kuro 00] p.243). For even a moderate sized network in a dynamic environment, which has a need for constant updating of such information, this control flow is unwieldy. An implementation of a proactive routing algorithm in a flat ad hoc network will need to be more like a distance vector than a link state algorithm.

Though our initial analysis assumes a central controller and hence appears more like a link state algorithm, later we present an implementable distributed distance vector style algorithm. But distance vector algorithms have some well-known limitations, such as convergence rate, the

counting to infinity problem, and packet looping ([Kuro 00] p.252). The extent to which these limitations also exist in the distributed algorithms we present in Section 6 is a topic for future research, as we briefly discuss in Section 7.

1.3 Ad Hoc Wireless Network Routing Literature

A lot of research work on routing for ad hoc wireless networks has been published in recent years. Most of this work includes simulation results, comparing the proposed protocol with a generic link-state or distance vector protocol, or comparing specific algorithms against each other. A minority of the published papers include analytic results, such as proofs of convergence or convergence rates. The work on routing for ad hoc wireless networks can perhaps most conveniently be categorized as either on-demand or route maintenance.

1.3.1 On-Demand Protocols

On-demand protocols are the algorithms which only update routing tables when a new packet arrives. On-demand protocols usually store old routing information and use it as long as packets still get to their destination. When a given path fails, there is some mechanism for probing the network to find a new path, which often involves flooding the network. Usually the path found is not optimal in any sense, and no QoS guarantees are made. Link detection used to decide path failure generally uses information from the wireless Medium Access Control (MAC) layer. The value of an on-demand protocol is that the communication requirement to set up a route is minimized. Hence, these protocols can be valuable when network dynamics are high compared to packet transmission frequency.

A number of on-demand routing protocols have recently been proposed for ad hoc wireless networks. The Temporally-Ordered Routing Algorithm (TORA) ([Cors 95], [Park 97]) is based on work originally in [Gafn 81]. TORA tries to quickly establish routes on demand using a path reversal method which is guaranteed to find some route (this is proved in [Gafn 81]). The resulting route is generally non-optimal. Dynamic Source Routing (DSR) ([John 94], [Malt 99]) uses source routing, where each packet header contains the entire route for the packet. Existing routes are used without update until one fails. When one fails, a Route Discovery is executed, during which Route Request packets flood the network until any route is found. The Ad Hoc On-Demand Distance Vector (AODV) protocol ([Perk 97]) is a kind of hybrid of on-demand and route maintenance protocols. The link status between a node and its neighbors is maintained through periodic

beacon signals between nodes. But routes are only updated when a packet routing fails, and then this link status information is used. The route chosen by AODV is the one with the minimum number of hops. The Zone Routing Protocol (ZRP) ([Pear 99]) is another hybrid of on-demand and route maintenance. The network is grouped into zones, which are groups of nodes. As in route maintenance, each node maintains a continuous knowledge of the topology of all the nodes within its own zone. However, routes to nodes outside a node's zone are determined on-demand in a Route Discovery process.

1.3.2 Route Maintenance Protocols

Given their wide acceptance, the benchmarks for route maintenance protocols are often the TCP/IP standards RIP [Malk 93] and OSPF [Moy 91]. However, as discussed above, the Internet implementations of these algorithms are not well-suited to the wireless ad hoc network environment (e.g. see [Graf 98]). In highly mobile environments, communication overhead to implement a full link-state algorithm is prohibitively costly. Alternatively, much research has been performed to adapt distance vector type algorithms to the mobile environment. In these approaches, it is common to use a Distributed Bellman-Ford (DBF) type algorithm [Bert 92] to update the routes and cost estimates. A common theme in the research, then, is to address the weaknesses of this type of algorithm in a highly mobile and uncertain environment.

An early effort along these lines is [Merl 79], which endeavors to provide loop-free routing in a quasi-static environment. In the Destination-Sequenced Distance Vector (DSDV) protocol ([Perk 94]), routes are tagged with sequence numbers. This allows the most up-to-date route information to be included, as well as providing a quick means for propagating link breakage information. In this way, loop-free routing is achieved. In Least Resistance Routing (LRR) ([Purs 93], [Purs 99]), the link quality between nodes and the current buffer length of the receiving node are used to define a link resistance value. A DBF type algorithm is then used to find a least resistance path for such a network. The Loop-free Path-finding algorithm (LPA) ([Garc 97]) eliminates the counting-to-infinity and packet looping problems with the DBF algorithm by using predecessor information and loop-detection at each router.

A minimum energy approach to routing in ad hoc wireless networks is taken in [Rodo 99]. The most striking difference to our work is in the channel model. In [Rodo 99] communication is peer-to-peer, and the stochastic nature of the channel is modeled as a threshold value for transmission power that ensures reliable communication. This threshold value is computed from

a signal transmission model requiring knowledge of node positions (via global positioning system (GPS) fixes). Once energy requirements to transmit to each node are determined, a DPF algorithm is run to compute minimum energy routes.

There has been growing interest in providing QoS guarantees for transmission in many networks. The QoS problem in wireless ad hoc networks is made especially challenging by their dynamic and uncertain nature. Some recent examples of research along these lines are [Chen 99], [Iwat 99], and [Siva 99].

An interesting approach to the multicast problem is developed in [Wies 98] and [Wies 00]. A deterministic local broadcast model for the wireless link, with a power threshold determining the availability of a link, is used to set up a problem to find the minimum energy to span a given set of destination nodes. An optimal solution to this problem is computationally infeasible, but the satisfactory performance of heuristic algorithms is studied using simulation. This circuit-switched multicast problem with deterministic local broadcast model is quite different from the problems we define in this paper.

In many situations where the relations among nodes are well established (e.g. nodes grouped by affiliation), it makes sense to think in terms of hierarchical groupings. Some recent work along these lines include [McDo 99], which uses motion estimates of node clusters to optimize the routing protocol, and [Joa 99], a zone based system using GPS that is derived from the Zone Routing Protocol ([Pear 99]).

A detailed comparison via simulation of a few different routing protocols for ad hoc wireless networks is described in [Broc 98]. It is clear that to truly understand the tradeoffs involved among the wide variety of available algorithms, more work along these lines is necessary. And of course it would be even better if progress in our conceptual understanding led to better methods for critiquing different techniques.

1.4 Contribution

In this work we present what is, to the best of our knowledge, the first network routing protocol which uses a probabilistic local broadcast model for wireless transmission. This model captures (i) the coupling between network layers, and (ii) the wireless channel's key characteristics, and it allows for routing decisions to be made based on immediate feedback from actual transmissions. In contrast to most of the existing literature, where link-level communication is point-to-point, we fully utilize the result of each local broadcast and use the information provided by the immediate

feedback to construct an optimal routing protocol. Such a protocol is well suited for a distributed implementation, as it is based only on information local to a node. We present a method describing how the protocol is updated when topological changes occur in a distributed mobile network, and this demonstrates the protocol's practicality.

We extend the model to allow for control of transmission type at each node, and show how the fundamental nature of this optimal protocol doesn't change. In the particular case of power control, an optimal protocol effectively resolves the tradeoffs between fewer long hops vs. more short hops. Finally, we analyze a time-varying network, where reward for packet delivery depends on time, and present conditions under which we can specify an optimal protocol. This model captures the important tradeoff between QoS requirements, such as delivery timeliness, and other criteria, such as energy efficiency.

2 Model Description

In this section, we expound on the modeling issues introduced in Section 1.1, and discuss our own modeling choices in more detail. The discussions here are qualitative and descriptive, intended to provide insight into the thinking behind the model we have chosen. We point out that there is no "proof" of correctness of our model, or of any model. A model's value depends on how well it captures important aspects of reality, and how cooperative it is under analysis or simulation. Indeed, finding the right level of model description for a problem is one of the key advances necessary for achieving useful results. Our model should be judged by the utility of the results we obtain with it. A precise mathematical description of our model, Model **(M)**, is given in Section 3.

Throughout this paper, we will refer to the data bits to be transmitted from a network node as a *message*. This is not meant to preclude the possibility that this data is a data packet in a packet network protocol suite. As defined in the ISO standard (see [Kuro 00]), the function of the network layer in such a protocol is to transmit packets from one network node to another. As we discuss below, our model is defined for use at the network layer, though modeling of some link and physical layer behavior is included. Our optimal routing algorithm performs the function of routing at the network layer, and can be implemented to support any transport layer protocol.

2.1 Motivation

Our goal in this work is to find practical routing algorithms well suited to the ad hoc wireless environment. To achieve this, we need a model satisfying two key points:

1. Model essential aspects of the wireless channel and the wireless ad hoc network
   - The simple models used for wired networks can be improved upon, at the cost of additional complexity.
   - The model needs to allow for analysis and simple implementation.
2. Obey appropriate informational constraints
   - An ad hoc network is a flat space of nodes with no central control.
   - Each node exchanges information with its neighbors only.

Point 1. leads us to define a probabilistic local broadcast model for node transmission, which we describe in detail below. We find that indeed a relatively simple stochastic model captures far more than standard deterministic models, yet still allows for analysis and a feasible implementation.

We desire local exchange of information in Point 2. to avoid the need for network flooding, as is used, for example, in OSPF ([Moy 91]). We handle Point 2. as follows: we begin with a centralized model where *full* information is available to a central controller. That is, we assume the central controller knows all events in the system, and at each time instant can use this information to make optimal decisions. We determine an optimal centralized policy, and then show two separate facts about it:

1. The policy can be *implemented* at each node using only information local to that node, satisfying the network's information constraints (Section 3.3.3). Because the policy is optimal for the problem with full information, it is also optimal for any problem with restrictive information constraints.
2. The optimal policy can be *determined* by a distributed algorithm (of distance vector style) which also satisfies the network's information constraints (Section 6).

Thus in the following modeling discussions, we focus on a problem of centralized information, and the term *controller* refers to a decision-maker which has access to all network information.

We desire for our model to capture: i) the coupling among model layers, and ii) the wireless channel's key characteristics. These features are described below.

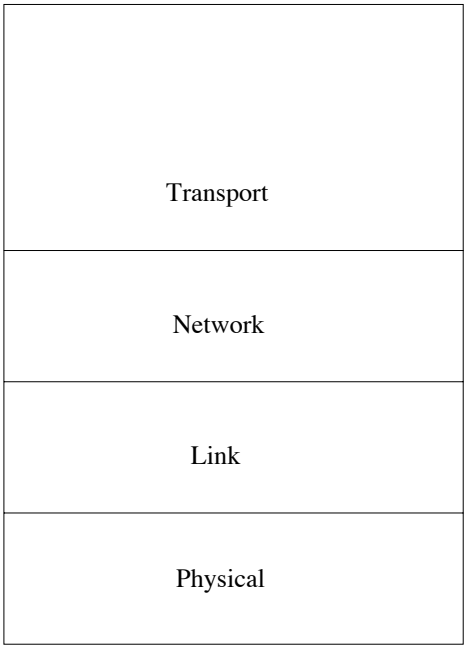

Figure 1: Lower Layers of ISO Network Model

2.2 Coupling Among Model Layers

A novel feature of our modeling approach is the close interplay of what traditionally are separate physical, link, and network layer functions. See Figure 1 as a reference for the lower layers of the standard ISO network model. In typical network routing algorithms, the information used at the network layer to determine best routes is usually very terse. Often, each link between connected nodes is given a number which indicates the point-to-point link cost, and a shortest path routing algorithm is implemented (for example, see discussions of OSPF or RIP in [Kuro 00], in which each hop is given a cost of 1). The nature of the physical channel is completely subsumed in this straightforward approach.

Our goal is to model link behavior in a manner more suited to wireless transmission. The special nature of wireless communication may not fit well into a simple point-to-point link cost model, primarily due to four interacting effects:

1. Due to the omnidirectional nature of mobile antennas, each message transmission can potentially be received by multiple neighbors.

2. Given the uncertain nature of many wireless links, message reception events are best modeled as random events.

3. Each transmission in the wireless system consumes energy, or some other limited resource.

4. The topology of the network, meaning link quality to neighbors, may be a function of choices made at the transmitter, such as power level, bit rate, modulation, and coding.

Points 1. , 2. , and 3. affect how the link layer should be modeled by the routing algorithm at the network layer. Point 4. requires some understanding about how choices at the physical layer affect the probabilistic transmission events. When these choices are available at the transmitter, they can be taken into account by the routing algorithm when making optimal routing choices. A typical example of this situation is transmission power control.

Points 1. and 2. lead to a natural modeling choice as follows. When a node is chosen for transmission, a random event occurs which is the set of neighbors which receive the message from that transmission. Each such event is given its own probability. After the transmission, the resulting event is made known to the controller as information to be used for the next node transmit decision. This point is very important: the routing algorithm uses feedback from actual transmission events to determine the optimal next action. Such a possibility is important only in the context of stochastic transmission events. In effect, we have discarded the point-to-point communication function of the link layer, and replaced it with a link model which 1) reflects mobile radio's local broadcast behavior and 2) allows reaction to the random nature of wireless transmission. As we will show, the optimal control of this more complicated model requires a close coupling between the routing algorithm and the link function.

Note that reception events can be coupled among neighbors. That is, independence of the events of individual neighbors receiving the message is not required. This allows modeling of situations such as where the same interferer affects transmission to all neighbors, or neighbors in physical proximity have the same random blockage.

To capture Point 3., we take a straightforward approach by incorporating into our model the cost of each transmission. This is simply to accrue a cost, specific to that node, for each node transmission. We allow this cost to vary over all the nodes, but at first we assume it is fixed for each node. Later, we modify the model to allow this cost to vary with choice of transmission type, as next discussed.

We model the functionality of Point 4. as follows. Once the controller learns the result of

the last transmission, it chooses a node for the next transmission, and it also chooses a type of transmission for that node. Transmission types might include power, coding, modulation, or even antenna directionality. To each choice of type is associated a cost, and a probability of success for each transmission event.

By this point we have discussed the basic notion of a transmission model where reception events can include multiple neighbors, and where these events are stochastic. Before completing the discussion of our link model, we introduce some well-known wireless transmission impairment types, and then discuss how we can interpret our model in their context.

2.3 Channel Models

Because of its clear importance for improving system design, the detailed character of the wireless transmission channel has been studied for many years, and in a variety of environments. We very briefly summarize here the work most relevant to our problem, focussing on the relation of theoretical and empirical studies to the development of our own model. For a more detailed description of the wireless channel and the typical models used for system analysis and design, see, for example, [Rapp 96], [Samp 97], or [Garg 96], and the references therein.

One approach to channel modeling begins with detailed models of hardware operation, including amplifier and antenna, as well as coding and modulation type. Then as much actual environmental information as possible is incorporated, including the physical layout of the mobiles and terrain, as well as the quality of the signal environment. Mathematical models of channel behavior are then developed, including diffraction, reflection, and scattering models, from which link quality can be determined among the nodes given their arrangement in the environment. Often the link quality is specified in terms such as signal-to-noise ratio or probability of error.

Another approach is, prior to system deployment, to map the signal transmission quality throughout the operational environment by taking extensive measurements. A database containing this data is then available for making network routing decisions, given a current configuration of the nodes in the network.

Perhaps needless to say, each of these approaches is very difficult, if not prohibitive. There are a few reasons for this: 1) the complexity of the computations for determining link quality, including signal propagation effects, 2) the difficulty in gathering the requisite data from the environment, and 3) the time-varying nature of the operational environment. Hence, it is usually necessary to create more tractable channel models by considering the general effects of the environment

on the transmitted signal, and then lumping these effects into just a few descriptive categories. Typically both analytic reasoning and empirical data are used to justify these categories. The relative importance of the different categories is a strong function of the operational environment, which can be as diverse as urban canyon, dense forest, or open field.

The model we now discuss is a typical representative of the approach to wireless channel modeling which does not use specific terrain data. The effect of the environment on the transmitted signal is divided into four categories:

1. Average Path Loss
2. Log-Normal Shadowing
3. Multipath Fading
4. Interference

We discuss each of these in turn.

2.3.1 Average Path Loss

It is observed (e.g. see [Rapp 96] Ch. 3) that the long-term average path loss (PL) between a transmitting and receiving node separated by a distance d roughly follows the equation

$$PL(d) \sim d^h \tag{1}$$

where h is a constant called the *path loss exponent*, whose value depends on environmental conditions. The typical range for h outdoors is $h = 2$ to 5, where $h = 2$ is the theoretical value for free space, and $h = 5$ is at the extreme of a cluttered urban environment. The idea is that obstacles in the environment cause increased signal degradation over the path length, and for a given environment there is an average rate of degradation with distance.

Another approach to average path loss is to take a lot of measurements in a particular environment, and then create curves of average path loss plotted as a function of d and transmitter frequency. A well-known model of this type is described in [Okum 68]. In [Hata 90] some simple formulas are found to fit Okumura's empirical curves.

The average path loss concept abstracts out the local and transient phenomena which affect signal power. In free space, it should give a consistently good estimate of PL as a function of d. In a cluttered environment with a properly determined h, it should give a good estimate of PL for each d, but only in a very long-term averaged sense. At a given instant, the actual power received

might be very different from the average power, perhaps due to blockage, or unusual shadowing or interference. These relatively short-term fluctuations are not captured by the simple equation (1), or any of the other average path loss approaches.

2.3.2 Log-Normal Shadowing

Because the information in the average path loss (1) is very limited, researchers have developed an empirical model of how path loss varies when the transmission distance is fixed at d, but measurements are taken at different times and/or locations in the environment. Through extensive testing it has been determined that this path loss often tends to fit a log-normal distribution reasonably well, and this phenomenon has come to be called *log-normal shadowing*. A typical formulation is ([Rapp 96] p.104)

$$PL(d) = \overline{PL}(d_0) + 10h \log\left(\frac{d}{d_0}\right) + X_\sigma \tag{2}$$

where $PL(d)$ is given in dB, d_0 is a reference distance, $\overline{PL}(d_0)$ is the average path loss at distance d_0, and X_σ is a zero-mean Gaussian random variable (in dB) with standard deviation σ (also in dB). Note that the average path loss model, and its path loss exponent h, is included in this formulation. In this section for convenience, we separate these loss effects into the average path loss and the variation of path loss, and we use the term *log-normal shadowing* for the variation, represented by X_σ in (2). The idea, then, is to decide upon d_0, and then take sufficient field measurements in a particular operating environment to accurately estimate the values h, X_σ, and $\overline{PL}(d_0)$.

The log-normal distribution of (2) at a given d can be interpreted as the probability distribution for path loss taken over all possible paths of a given d in the system. A major limitation of this log-normal shadowing concept is the lack of information on how path loss is correlated in time or place. For example, if we take a power measurement at a certain spot and time, how will this received power change over time at the same spot, or if we move at a fixed rate? Recognition of the importance of this limitation has led to recent field work measuring these correlations. For example, see [Gudm 91], in which a simple exponential correlation model is proposed and fitted to urban data.

2.3.3 Multipath Fading

In many environments, due to reflection, scattering, and doppler effects, the signal at the receiver consists of the sum of many components, each a version of the transmitted signal time delayed by some random amount. These many components arise when there is no one dominant transmission path for the signal, and many paths of comparable path loss are available. The net effect of this phenomenon is that the received signal fluctuates in usable power rather rapidly, as the different signal components add and subtract in various ways. Without serious efforts to incorporate countermeasures in the system design, multipath fading can lead to serious performance degradation.

Fortunately, the effect of multipath fading on the receiver signal is a well-understood phenomenon, and it turns out there are a variety of receiver design techniques which greatly mitigate, though not entirely remove, its deleterious effect. General models for multipath fading require a parameter which in some way specifies the rate of signal amplitude variation, which is related to the correlation over time of this amplitude (for example, see [Rapp 96] Sec. 4.7.2, and the multipath fading simulation parameter $T = \frac{1}{\Delta f}$, which defines the time duration of a fading waveform).

2.3.4 Interference

Through Point 1., we use the local broadcast nature of radio transmission to advantage. But omnidirectionality is a two-edged sword. As is well known, we must also take into account the interference neighboring nodes have on each other's signal reception. In CDMA cellular systems, this effect is in fact the primary limitation to system capacity, giving rise to the fundamental near/far problem, whose solution requires careful power control. This kind of coordinated power control can only be accomplished in systems under centralized control from a base station.

Our goal is to develop a routing protocol for an ad hoc network with no centralized control. It is certainly possible to define a problem model which incorporates full interference effects, but analysis for the ad hoc network is enormously difficult. We aim for minimum-cost routing, not maximal system capacity. Hence, we assume sufficient messages in the system to justify Route Maintenance operation, but not so many that neighbor interference is the dominant channel impairment. Under these conditions, and given the difficulty of more complex interference models, we choose to model neighbor transmission interference as another unpredictable channel impairment, included in the probabilistic transmission structure.

2.4 Model Assumptions

2.4.1 Basic Assumption

Based on our previous discussion, we plan to use a probabilistic model describing, via probabilistic transitions, successful local broadcast communication. As pointed out before, these transitions capture the coupling of the physical, link, and network layers, and they account for signal degradation due to path loss, log-normal shadowing, multipath fading, as well as the interference created by other transmitting nodes. To complete the model we must specify the statistical properties of the aforementioned transition probabilities over time. Many modeling formalisms exist, each with its own salient features and degree of realism. The main model of this paper, Model **(M)** defined in Section 3, is based on the following assumption.

Assumption 2.1 *Transmission events at a given node are iid, and transmission events are independent among all nodes.*

In Section 5 we define a problem which relaxes this assumption, and study how the structure of the optimal policy changes. Our goal here is to clarify the meaning of Assumption 2.1 in terms of the channel model of Section 2.3, and the network model.

2.4.2 Characteristic Times of the Network

Assumption 2.1 is the fundamental assumption we require for most of the analysis in this paper (that is, for all except that of Section 5). We now proceed to clarify what this assumption means in terms of the channel model of Section 2.3 and the overall network model.

We can think of the time evolution of the network in terms of characteristic times of the different on-going processes. By characteristic time we mean a time period, specified as τ, which approximates the length of time the process requires for its transitions. For example, the characteristic time of a node transmission is the time it takes to transmit the message. The characteristic time of network topology change is roughly the amount of time it takes, both through node relative motion and through time-varying disturbances in the environment, for the topology to be altered in a significant way. We do not precisely specify, that is, we do not specify quantitatively, the system characteristic times, as they are only meant to be used for very rough comparisons among themselves. We use them as a means to succinctly summarize the meaning of the iid nature of transmission specified in Assumption 2.1.

We define the following characteristic times of three fundamental network processes.

$$\begin{aligned} \tau_n &:= \text{characteristic time of significant network topology variation} \\ \tau_t &:= \text{characteristic time of message transit from source to destination} \\ \tau_m &:= \text{characteristic time of one node message transmission} \end{aligned}$$

We also define the characteristic times of the different channel effects on the received signal.

$$\begin{aligned} \tau_s &:= \text{characteristic time of average path loss} \\ \tau_l &:= \text{characteristic time of log-normal shadowing} \\ \tau_p &:= \text{characteristic time of multipath fading} \end{aligned}$$

It is immediate that

$$\tau_m \leq \tau_t \tag{3}$$

because at least one transmission is necessary to reach the destination.

The iid assumption of Assumption 2.1 has two requirements: 1) independent, and 2) identically distributed. The latter requirement imposes a time-homogeneity on transmission probabilities, and fundamentally says that the topology of the network is not varying at all. However, we can satisfy this requirement with the less rigid assumption that the time it takes to get the message to its destination is far less than the time it takes for the topology to change in any *significant* way. The optimal routing policy for a particular message need not concern itself with this very slowly time-varying topology, so long as the transition probabilities used accurately reflect the topology at the time the message is to be routed. Under this condition, the assumption of identically distributed transition probabilities is justified. We can state this assumption as

Condition 1

$$\tau_t \ll \tau_n \tag{4}$$

The average path loss is assumed fixed at each point, but it is a time-varying quantity due to node mobility. The rate of its change is thus intimately tied to the rate of change of the node topology, and we have

$$\tau_s \approx \tau_n \tag{5}$$

Perhaps the most uncertain time parameter is τ_l. Log-normal shadowing relates to both mobile motion and the motion of clutter in the environment. Without more detailed models of the correlation of this kind of path loss, both spatial and over time, it is not possible to fully justify

any assumptions we might make. The iid requirement of Assumption 2.1 implies that transmissions are independent over time at a given node, which in turn requires that channel disturbances which affect transition probabilities are either much slower or much faster than a message transmission time. Using this as a guide, we thus simply require that

Condition 2

$$\tau_l \gg \tau_n \quad or \quad \tau_l \ll \tau_m \tag{6}$$

Multipath fading relates to mobile motion and the distribution of clutter in the environment. Though its effects can be alleviated by good system design, including diversity reception and signal design, it still will have some effect on transmission probability. The multipath fading effect on the received signal occurs much more quickly than that due to log-normal fading. We will assume that rate of multipath fading is uniformly faster than a message transmission, and hence we have

Condition 3

$$\tau_p \ll \tau_m \tag{7}$$

As discussed in Section 2.3.4, we model neighbor interference as just another factor affecting a node's transmission probabilities. This is justified in the context of iid transmission and Section 2.3.4, under

Condition 4 *At any node, neighbor transmissions are iid events.*

One very rough way to think about Condition 4 is that, given what a node knows, transmissions at other nodes *appear* iid. It is out of the scope of this work to make this notion more precise.

We note in passing another important meaning of the parameter τ_n. In Section 6 we investigate distributed algorithms which compute an optimal routing policy for the ad hoc network. Such an optimal routing policy will only be valid for a *static* network, where the transition probabilities are fixed. For many real networks, the topology of the network will be slowly varying. The parameter τ_n is a measure of the rate of change of the network topology, and hence gives a rough measure of the speed at which the distributed algorithms must track topological changes. This raises many questions about the convergence *rate* of these distributed algorithms. We will not study convergence rate in this paper, but it is an issue of great interest.

We summarize the requirements of Condition 1-Condition 4 on characteristic times as follows. One of the following two relations must be true.

$$\tau_p, \tau_l \ll \tau_m \leq \tau_t \ll \tau_n, \tau_s \tag{8}$$

$$\tau_p \ll \tau_m \leq \tau_t \ll \tau_l, \tau_n, \tau_s \tag{9}$$

We can interpret (8) and (9) by noting that 1) degradation effects much slower than τ_t can be considered constant over the entire time it takes the message to reach its destination, whereas 2) degradation effects much faster than τ_m look random at the transmitting node. We call type 1 degradations "slow" effects, and type 2 degradations "fast" effects. Relations (8) and (9) reflect the possibilities for the degradation categories we consider. Of course, both categories affect the transmission probabilities. For Assumption 2.1 to be true, we require that all effects degrading the signal transmission be either "slow" or "fast". Under Condition 1-Condition 4, Assumption 2.1 can be justified as a good model for the network.

Based on the proposed model, we proceed to formulate the routing problem we intend to investigate in this paper.

2.5 The Anycast Routing Problem

Our model requires the specification of node transmission event probabilities and costs at each node. For this kind of information to be available, link quality information must be consistently collected in the network. Hence, we consider the Route Maintenance approach to the routing problem (see Section 1.2.3). This means we keep updated routing tables which reflect the current best routing estimates.

We will set up our routing problem as one of routing policy optimization for a stochastic transition model. To accomplish this, we define time-independent rewards for reaching certain destination nodes, and time-independent costs for each node transmission. Our goal will be to maximize the difference between expected reward and expected cost. The costs incurred are generally interpreted as energy consumption, though their meaning can be more general.

Typically a routing problem is classified as either unicast, multicast, broadcast, or anycast. These types are distinguished by the nature of the set of destination nodes. In *unicast* the destination is just one node. In *multicast* the destination is a set of nodes, and the message must be delivered to every node in the destination set. In *broadcast* the message must be delivered to every node in the network. Finally, in *anycast* the destination is a set of nodes, but the message only needs to reach one of the nodes in the destination set.

Because we set up our routing problem as a routing policy optimization, we are able to be somewhat more general than any one of the above four problems. We assign a reward to each destination node, and then let the control policy decide at what point it should stop transmitting and take its reward. When the reward at all destination nodes is the same, this problem reduces

to an anycast problem, so this is the term we use.

We consider a general anycast problem as follows. There is an ad hoc network with multiple nodes. Messages randomly arrive at nodes, each with its own destination set. A controller must allocate communication channels in some optimal way to maximize expected reward.

Careful use of our model allows us to simplify this problem considerably. There are two key points: 1) the system is time-invariant, so delays in transmission do not affect the optimal policy, and 2) interference from neighbors is incorporated in the model. These two properties, which are a consequence of all the modeling assumptions made so far, allow us to approximately compute optimal routing strategies by decoupling the general problem of multiple messages into multiple problems, each of one message.

Even within the context of our modeling assumptions this decoupling is only an approximation, because transmission probabilities are considered fixed for each individual message. But in reality, once an optimal policy is computed for a given message, this impacts the transition probabilities of the other messages through interference. That is, the optimal routing policy for message A must use transition probabilities affected by the optimal policies of the other messages, whose optimal policies can only be computed using transition probabilities affected by the routing of message A. Rather than try to simultaneously solve the optimal policies for all the messages, we simply ignore this coupling. Effectively, we assume that the policies of the other neighbors are fixed, and then optimize the routing policy for one message.

Hence, in this paper we focus on solving only the problem of one message with its set of destinations, and all the routing problems we define in Section 3 are for one message only.

2.6 Transmission Probability Estimation

As discussed in Section 2.3, there are two basic approaches to channel modeling, the analytic and the measurement-based. This same fact also applies to how we might go about estimating the transmission probabilities which our model requires from the network. And as before, it is likely that the pure analytic approach will be much too difficult for most applications. The measurement-based approach has the advantage that it is bottom-line oriented and relatively easy to do. The method itself is generally independent of system details, such as coding and modulation, terrain maps, and types of signal disturbance.

We do not delve deeply here into how to find the best method for transmission probability estimation. It remains, however, an important aspect of our model which deserves further study.

For our purposes, a natural idea is to use a running time average of the proportion of recent transmissions that have been successful to each neighbor node. Such an approach requires the definition of some heuristic averaging window length, which should be related to τ_n, the rate of change of network topology. There is some precedence for time-averaging to estimate the status of a link. For example, in [Rapp 96] Ch. 2, it is stated that this is done in some cellular systems to estimate power in order to decide when a handover should occur.

Note that it might be possible to use some kind of hybrid approach, both an analytic model of signal transmission and empirical measurements of actual transmissions, to improve upon the simple moving average method. Also, information gathered at the link level, for example received power, might be used to improve transmission probability estimation.

Throughout the rest of this paper, we assume that the transmission probabilities are given to us. In Section 6, we discuss distributed algorithms to compute an optimal policy which are able to adapt the policy to slowly-varying topology changes, as reflected in slowly-varying transmission probabilities.

2.7 Discussion

We end this section with some final comments about our model.

2.7.1 Further Clarifications of the Model

First, we list some further clarifications of the proposed model.

- Cost only accrues when a node transmits the message. Of course, in a real system there are also energy costs associated with control signaling, such as neighbor acknowledgements, and message decoding. We can interpret our model in two ways: 1) These other costs are insignificant compared to the transmission cost, or 2) These costs are included as part of the cost for transmission.

- The transmission event is the set of nodes that receive the message when a node transmits. What we really mean by this is the set of nodes that receive the message and successfully acknowledge this fact back to the node that transmitted. This fact will become pertinent when we discuss distributed implementation of an optimal routing policy (Section 3.3.3).

- In our discussion of characteristic times, the message transmission time τ_m was taken as fixed. This is reasonable, because the same message is transmitted each time. It is possible,

however, that these transmission times are not the same, due to different transmission types at different nodes, or even at the same node. For example, we might allow control of the bit rate, slowing it down when transmission quality is poor. Nothing in the above assumptions fundamentally changes in this case: the characteristic time of an impairment is assumed fast compared to whatever transmission rate it is affecting.

- A limitation of our model is that degradation effects of duration comparable to τ_m or τ_t cannot be modeled. In Section 7 we briefly discuss the topic of Markov Channels as a topic for future research. With a Markov Channel model these more general degradation effects can be properly modeled, though the optimal policy grows much more complex.

- Real messages have finite time horizons over which delivery is useful to the system. Our time-invariant reward assumption can be considered a reasonable approximation to the case where the useful life of a message is long compared to the typical trip time in the network, τ_t. In Section 5.2 we consider the case of time-varying system parameters, including rewards.

- Our model can be considered a generalization of the standard deterministic model used in many routing algorithms, such as OSPF or RIP. When each node has a number of transmission choices equal to its number of neighbors, and each choice corresponds to transmission, at a certain cost, to exactly one of its neighbors with probability 1, then our model reverts to the standard one with fixed deterministic link cost. In Section 4.4, we show that for this problem of deterministic channel transitions our main result for the centralized controller (Algorithm 1) reverts to the standard Dijkstra Algorithm (as used, for example, in OSPF).

- Our channel model creates a significant difference, from what is typical, in how the network and link layers must communicate. The typical network layer command to the link layer is: send the message to the following node, and let me know when you're done. Because of the stochastic broadcast nature of transmission in our link model, this simple interface is no longer viable. The link layer must react to stochastic events. Hence, the instructions from the network layer to the link layer need to be more nuanced, involving alternative courses of action depending on these events. We discuss this issue in more detail in Section 3.3.3.

2.7.2 Advantages of our Model

Next, we list what we believe are some of the advantages of our model.

- In an operating environment which matches our model assumptions, the routing policy maximizing expected reward is determined. We expect that in an operating environment close to our assumptions, our routing policy remains close to optimal. In Section 7, we briefly discuss an approach to quantifying some aspect of this problem in terms of a parameter sensitivity analysis.

- The omnidirectional nature of wireless transmission is usually seen as detrimental, as it leads to interference among the nodes. While recognizing interference as a problem, our model also tries to use omnidirectional transmission to some advantage where this is possible.

  - Node transmissions are always local broadcasts, which attempt transmission to all neighbor nodes. This creates the best chance of message forwarding.
  - A related point is that alternate routes are used when available in the routing policy as natural behavior of the optimal routing. If the link to a neighbor suddenly ceases to function, though the optimal routing algorithm will eventually catch this and update itself, in the meantime the transmission to the remaining neighbors continues as before. This provides built-in redundancy and resistance to the damaging effect of total failure at a link.

- Routing decisions are made using immediate feedback from stochastic events, allowing timely reaction to actual system behavior. This feature only makes sense in the context of a stochastic transmission model.

- In many routing algorithms, link state between nodes affects routing only through the determination of whether a connection exists at all between the nodes. That is, some link test is performed, usually at the link layer, and the result is binary: either the link is usable or it is not usable. For wired networks, where IP is typically used, this approach is reasonable, as the key question is whether there is a wire between the nodes at all. For wireless networks, often some kind of threshold for received signal power is used to determine whether a link between two nodes is "up" or "down" (e.g. see [Perk 97], [Cors 95], or [Malt 99]). In many ad hoc wireless network routing algorithms, this link state determination plays a key role. Often when a link state changes, many control messages are sent throughout the network to update routing tables (for example, see [Cors 95] or [Malt 99]). The method for determining the link states is thus a critical design issue for these routing algorithms.

In a wireless network with highly mobile nodes, instantaneous link quality can vary over enormous ranges in a relatively short time, due to the types of signal degradation effects we discussed in Section 2.3. For example, a link might go down very momentarily, perhaps due to a short fade or temporary shadowing effect. For many routing algorithms, it is critical to determine at what point signal loss should trigger rerouting. The detection algorithm used to determine link state must filter these effects in some way, and the relation of the bandwidth of this filter to the functioning of the routing algorithm is a matter which must be given much care in the design.

Our model deals with these kinds of effects in a much different way. As long as these signal degradations occur more rapidly than τ_m, the degradation is incorporated into the transmission probability. Rather than make a binary "up" or "down" detection necessary as the channel quality wavers, the transmission probability acts as a kind of numeric quality measure, and optimal behavior based on this quality measure can then be determined.

- When transmission probability estimates are measurement-based, our model is very bottom-line oriented. That is, routing decisions are based directly on transmission *results*, and we do not need to predict these results from signal *effects* of the operating environment. This means that the enormous complexity of the signal transmission can be subsumed into just the relatively simple transmission probabilities, which are based on actual measured system behavior.

- If it is feasible in the system implementation, our model allows for optimal choice of transmission type at each node. This might include choosing the best power or best data rate at which to transmit.

2.7.3 Limitations of our Model

We end with a brief description of some potential limitations of our model.

- The routing algorithm must run in Route Maintenance mode, which can be expensive in energy. There must be enough messages on average in the system to justify this.

- Estimation of transmission probabilities is a difficult problem. Probability estimation when there are different transmission types and message lengths is an even harder problem. How our optimal routing algorithms behave when the probability estimates have errors is an

important topic for further study (see the brief discussion about sensitivity analysis in Section 7).

- Channel degradations with rate in the range of message transmission to the typical lifetime of a message in the network do not fit the model as stated. As we mention briefly in Section 7, the extent to which the performance of our routing policy degrades as our model assumptions are violated is an interesting question for further study.

- Expected reward, which is closely related to expected total energy cost, may be too simple as a measure of system performance. For example, *where* energy is expended and *when* may be important in some problem instances.

- Our model does not well suit situations where transmission probability is always very near either 1 or 0. In this situation, the local broadcast/stochastic transmission model simplifies to a standard deterministic model, and the optimal algorithm for that problem is somewhat simpler than our optimal algorithm. However, there shouldn't be any harm in implementing our model, either.

3 The Time-Invariant Stochastic Routing Problem

3.1 Notation and Preliminaries

We begin by briefly defining notation and stating definitions for the system model under consideration, which we refer to as Model **(M)**. As discussed in Section 2, in Model **(M)** control is centralized, meaning the controller has access to all information in the network. Also, Model **(M)** is probabilistic, with transitions described by $P(S|i)$, and is based on Assumption 2.1.

N is the number of nodes in the network.

$\Omega = \{1, \ldots, N\}$, the set of all nodes. So $|\Omega| = N$.

$S \subseteq \Omega$ refers to a state of the system, defined as the set of nodes which have received the message. S_t refers to the state at time t.

$R : 2^\Omega \to \mathbb{R}$ is the reward function, and $R_i := R(\{i\})$. Also $R_{max} := \max_{i \in \Omega} R_i$.

π is a Markov policy. We write $\pi(S) = i$ to indicate policy π transmits at node i when in state S.

We write $\pi(S) = r$ to indicate policy π retires and receives reward $R(S)$ when in state S. For convenience we write $\pi(S) = r_i$ as shorthand that policy π retires and receives $R_i, i \in S$. In this case, we say that policy π retires and *receives the reward* of node i.

By $\pi(S) \neq i, r_i$, we mean both $\pi(S) \neq i$ and $\pi(S) \neq r_i$.

By $\pi(S) = \tilde{\pi}(S)$, we mean either $\pi(S) = \tilde{\pi}(S) = i$, or $\pi(S) = \tilde{\pi}(S) = r_i$, for some i.

$V^{\pi}(S)$ is the expected reward when starting in state S under policy π. We often write V_i^{π} for $V^{\pi}(\{i\})$. We use $V^{\pi}(\cdot)$ to indicate the optimal value function.

We write $P^i(S'|S)$ to indicate the probability of reaching state S' from state S when choosing i for transmission, $i \in S$. We write $P^i(S|i)$ as shorthand for $P^i(S|\{i\})$.

We define $P_{ij} := \sum_{S: i,j \in S} P^i(S|i)$.

j is called a *neighbor* of i if $P_{ij} > 0$. $\mathcal{N}(i)$ is the set of all neighbors of i, together with i itself. Note that $P_{ij} \neq P_{ji}$ is permitted.

By $argmax_{i \in S} f(i)$, we mean the *set* of values of i from the finite set S which maximizes $f(i)$.

Definition 3.1 (Increasing Property) *Model* **(M)** *is said to be* increasing *if for any system realization under any policy we have* $S_{t_2} \supseteq S_{t_1}, \ \forall t_1, \forall t_2 > t_1$.

Definition 3.2 (Decoupling Property) *Model* **(M)** *is said to be* decoupled *if transmission success to a set of neighbors from a node at a given time is unaffected by which other nodes already have the message.*

Definition 3.3 *A function* $f : 2^{\Omega} \to \mathbb{R}$ *is an* index function *on* Ω *if* f *satisfies*

$$f(S) = \max_{i \in S} f(\{i\}) \quad \forall S \subseteq \Omega \tag{10}$$

We next formulate the centralized version of the stochastic routing problem with time-invariant parameters.

3.2 Statement of Problem

Problem ($\mathbf{P_1}$)

We consider the transmission of a single message, from a given initial state S_o (i.e. a given set of nodes) to a set of destination states, in a wireless ad hoc network of N nodes described by Model **(M)**. Transmission instances occur at discrete time points. Each transmission from a given

node i incurs a fixed cost $c_i > 0$. According to Model **(M)**: (i) at each transmission instance the transmitting node is chosen by a central controller that always knows the current state of the system (i.e. the set of nodes that have the message); (ii) node transmissions are local broadcasts, that is, multiple neighbor nodes may all simultaneously receive the message; (iii) given the node chosen to transmit, the probability that a given set of nodes receives the message is known and fixed; (iv) The central controller is informed, without any cost, as to which nodes receive the message. In general, control information flow between the nodes and the controller is considered free in energy and instantaneous in time; and (v) each transmission event is assumed independent of those before and after. We assume Model **(M)** is increasing and decoupled. A reward function R is specified, where R is an index function. At any instance, the central controller can terminate the transmission process or choose to continue transmitting. The objective is to choose: (i) the node to transmit at each transmission instance, and (ii) the instance to terminate the transmission process, to maximize

$$E\left\{R(S_f) - \sum_{t=1}^{\tau-1} c_{i(t)}\right\} \tag{11}$$

where τ is the time when the transmission process is terminated, S_f is the state at τ, and $i(t)$ is the node chosen by the transmission policy at time t.

3.3 Analysis of Problem $(\mathbf{P_1})$

We analyze Problem $(\mathbf{P_1})$ and discuss the character of an optimal policy π. The system of Problem $(\mathbf{P_1})$ is a time-homogeneous Markov chain, hence we are faced with a finite-state Markovian Decision Problem with perfect information. We can thus restrict attention to Markov policies on 2^Ω, and we are guaranteed that such an optimal Markov policy exists (cf. [Ross 83] Ch.3 p.51). We seek an optimal Markov policy $\pi : 2^\Omega \to (1, \ldots, N)$ which minimizes (11).

To solve Problem $(\mathbf{P_1})$, we could directly apply stochastic dynamic programming. But since the number of states is 2^N, the complexity of such an approach is at least $O(2^N)$, and generally higher (see [Gare 79]). Instead, we use the special structure of this problem to find a better algorithm.

3.3.1 Structure of an Optimal Policy for Problem $(\mathbf{P_1})$

We begin with some definitions.

Definition 3.4 *A Markov policy π is a* priority *policy if there is a strict priority ordering of the nodes s.t. $\forall i \in \Omega$ we have $\pi(S \cup \{i\}) = \pi(\{i\}) = i$ or $r_i, \forall S \subseteq \Omega_i$, where Ω_i is the set of nodes of priority lower than i.*

Definition 3.5 *For priority policy π, we write $i >_\pi j$ when i has higher priority than j under π.*

Definition 3.6 *A priority policy π is called an* index *policy if $V^\pi(\cdot)$ is an index function on Ω.*

Note that a priority policy need not be an index policy.

Our main goal in this section is to prove that there exists an index policy which is an optimal Markov policy for Problem $(\mathbf{P_1})$. We state this result in the following theorem.

Theorem 3.1 (Index Policy) *There is an optimal Markov policy π for Problem $(\mathbf{P_1})$ which is an index policy.*

We develop a series of lemmas which are used to prove Theorem 3.1. In the first lemma we show that the definition of an index function is equivalent to requiring two properties on f.

Lemma 3.1 *Function f is an index function on Ω if and only if the following two properties (12) and (13) both hold.*

$$f(S \cup \{i\}) \geq f(S), \quad \forall S \subset \Omega, i \in \Omega \tag{12}$$

$$f(S) \neq f(\{i\}) \implies f(S) = f(S - \{i\}), \quad \forall S \subseteq \Omega, i \in S \tag{13}$$

*Proof.*Assume f is an index function on 2^Ω. Then f can be written in the form (10). We have $\forall S \subseteq \Omega, i \in \Omega$

$$f(S \cup \{i\}) = \max_{j \in S \cup \{i\}} f(\{j\}) \geq \max_{j \in S} f(\{j\}) = f(S) \tag{14}$$

and (14) establishes (12). To establish (13), assume we have an $i \in S$ where $f(S) \neq i$. Then

$$f(S) = \max_{j \in S} f(\{j\}) = \max_{j \in S - \{i\}} f(\{j\}) = f(S - \{i\}) \tag{15}$$

Together (14) and (15) establish that if f is an index function, then (12) and (13) both must hold.

Conversely, assume (12) and (13) hold. We proceed by induction on the number of elements in S. When $|S| = 1$, for any S it is clear that $f(\{i\}) = \max_{j \in \{i\}} f(\{j\})$, so the induction base step is established. Now assume $f(S)$ can be written in the form of (10) $\forall S \subset \Omega$ *s.t.* $|S| = K$. Let $S' \subseteq \Omega$ be s.t. $|S'| = K + 1$. We consider two cases.

Case 1 $\quad f(\{i\}) = f(\{j\})\ \forall i, j \in S'$

Assume there is an $i \in S'$ s.t. $f(S') \neq f(\{i\})$. Then by (13) and using the fact that $|S' - \{i\}| = K$ with the inductive hypothesis, we have $f(S') = f(S' - \{i\}) = \max_{j \in S': j \neq i} f(\{j\}) = f(\{i\})$, which is a contradiction. So we must have $f(S') = f(\{i\}) = \max_{j \in S'} f(\{j\})$, and (10) holds for this case.

Case 2 $\quad \exists i, j \in S'$ *s.t.* $f(\{i\}) < f(\{j\})$

By (12), $f(S') \geq f(\{j\}) > f(\{i\})$. Then by (13), and using the fact that $|S' - \{i\}| = K$ with the inductive hypothesis, we have

$$f(S') = f(S' - \{i\}) = \max_{j \in S': j \neq i} f(\{j\}) = \max_{j \in S'} f(\{j\}) \tag{16}$$

and (10) holds for this case.

In both cases f has the form of (10) for S', and this completes the induction step. So by induction f must be an index function on 2^{Ω}. □

Next, we use the decoupling and increasing properties of Problem $(\mathbf{P_1})$ to show that the optimal value function for Problem $(\mathbf{P_1})$ possesses a monotonicity property.

Lemma 3.2 *(Monotonicity) In Problem* $(\mathbf{P_1})$*, let* π *be an optimal Markov policy, and let* $S_1, S_2 \subseteq \Omega$ *and* $S_2 \subseteq S_1$*. Then* $V^{\pi}(S_2) \leq V^{\pi}(S_1)$*.*

*Proof.*Given π and $S_2 \subseteq S_1$, we define a new policy $\hat{\pi}$ acting on state S_1 as follows. Let $i = \pi(S_2)$, and let S_4 be the state resulting if at first π were to choose i. At the first step, $\hat{\pi}$ chooses i from S_1, which is possible since $S_2 \subseteq S_1$. $\hat{\pi}$ learns the result of the transmission, and hence knows the new actual state of the system, which we call S_3. Furthermore, since $\hat{\pi}$ knows which nodes receive the message even if they already have it, it also knows what the new state would be if the previous state were S_2 instead of S_1. By the decoupling property, this new state is S_4. This fact together with the increasing property also imply that $S_4 \subseteq S_3$, since $S_2 \subseteq S_1$.

At the next step $\hat{\pi}$ acts on S_3 by choosing the same node as π would use on S_4; this is possible because $\hat{\pi}$ knows S_4, and $S_4 \subseteq S_3$. The process continues in this way until $\hat{\pi}$ retires at the same time at which π would retire. Policy $\hat{\pi}$ knows π's retirement time because it knows the state π "sees" at each time. Let S_{f1} and S_{f2} be the states at retirement for $\hat{\pi}$ and π, respectively. By the above argument, we know $S_{f2} \subseteq S_{f1}$. At retirement, $\hat{\pi}$ has incurred the same cost as π, since

π and $\hat{\pi}$ use the same nodes to transmit. Because R is an index function, by Lemma 3.1 (12) we have

$$V^{\hat{\pi}}(S_1) - V^{\pi}(S_2) = R(S_{f1}) - R(S_{f2}) \geq 0 \tag{17}$$

Because π is optimal and $\hat{\pi}$ is suboptimal, we conclude from (17) that

$$V^{\pi}(S_1) \geq V^{\hat{\pi}}(S_1) \geq V^{\pi}(S_2) \tag{18}$$

This completes the proof. □

In the next lemma, we use the increasing property of Problem ($\mathbf{P_1}$) to show that a Markov policy which is optimal for all states that are a superset of some $S_1 \subset \Omega$, and that takes an optimal action when in S_1, is also optimal when in state S_1.

Lemma 3.3 *Let $\tilde{\pi}$ be an optimal Markov policy for Problem ($\mathbf{P_1}$). Suppose we are given S_1, and let π be a Markov policy which has the following two properties.*

$$V^{\pi}(S) = V^{\tilde{\pi}}(S), \quad \forall S \supset S_1 \tag{19}$$

$$\pi(S_1) = \tilde{\pi}(S_1) \tag{20}$$

Then

$$V^{\pi}(S_1) = V^{\tilde{\pi}}(S_1) \tag{21}$$

*Proof.*If $\pi(S_1) = \tilde{\pi}(S_1) = r_i$, for some $i \in S_1$, then (21) holds.

Suppose $\pi(S_1) = \tilde{\pi}(S_1) \neq r$. We compare π and $\tilde{\pi}$ when both transmit in state S_1. Let $\tilde{S}_2$ and S_2 be the state after transmitting when in S_1 for $\tilde{\pi}$ and π, respectively. Due to (20), we have

$$\tilde{S}_2 = S_2 \tag{22}$$

Due to the increasing property, we have

$$\tilde{S}_2 = S_2 \supseteq S_1 \tag{23}$$

By (19), we have

$$V^{\tilde{\pi}}(\tilde{S}_2) = V^{\pi}(S_2), \quad \tilde{S}_2 = S_2 \supset S_1 \tag{24}$$

Equations (22)-(24) mean that $\tilde{\pi}$ and π choose the same node from S_1 for transmission, and either reach the same state $S_2 \supset S_1$, which has the same value function for both policies, or both stay in

$S_2 = S_1$, at which point they again both play the same node for transmission. Hence, (21) follows. □

In the next lemma we construct a Markov policy for Problem $(\mathbf{P_1})$ which has many characteristics necessary for an index policy, and then use the lemmas presented above to show that this policy is optimal. The result of this lemma is instrumental in proving Theorem 3.1.

Lemma 3.4 *Let $\tilde{\pi}$ be an optimal Markov policy for Problem $(\mathbf{P_1})$. Then there exists a Markov policy π which has the following three properties.*

1. *For all $S \subseteq \Omega$ where $|S| \geq 2$,*

$$\pi(S) = i \implies \pi(S - \{j\}) = i \quad \forall j \in S, j \neq i \tag{25}$$

$$\pi(S) = r_i \implies \pi(S - \{j\}) = r_i \quad \forall j \in S, j \neq i \tag{26}$$

2. *For all $S \subseteq \Omega$ where $|S| \geq 2$, and $\pi(S)=i$,*

$$V^{\pi}(S - \{j\}) = V^{\pi}(S) = V^{\tilde{\pi}}(S) = V^{\tilde{\pi}}(S - \{j\}) \quad \forall j \in S, j \neq i \tag{27}$$

3. *π is an optimal Markov policy.*

*Proof.*We define π using the following rules:

$$\pi(\Omega) = \tilde{\pi}(\Omega) \tag{28}$$

$$\pi(S - \{j\}) = \pi(S), \qquad \forall S \subseteq \Omega, \forall j : \pi(S) \neq j, r_j \tag{29}$$

$$\pi(S - \{j\}) = \tilde{\pi}(S - \{j\}), \qquad \forall S \subseteq \Omega, j : \pi(S) = j, r_j \tag{30}$$

If $|\Omega| = N = 1$, by (28) the lemma is true. Assume $|\Omega| = N \geq 2$.

It follows directly from (28)-(30) that π satisfies (25) and (26).

We prove (27) by backward induction on the cardinality of S. We know that $\tilde{\pi}(\Omega) = r_i$ for some $i \in argmax_{j \in \Omega} R_j$. By (28) and (29) we have $\pi(\Omega) = \pi(\Omega - \{j\}) = r_i, \ \forall j \in \Omega, j \neq i$. That is, π acting on both Ω and $\Omega - \{j\}$ immediately retires and receives reward R_i. We also have $V^{\tilde{\pi}}(\Omega) = R_i = V^{\tilde{\pi}}(\Omega - \{j\})$, because $\tilde{\pi}$ is optimal and i is available for retirement in $\Omega - \{j\}$. Hence, when $\tilde{\pi}(\Omega) = i$,

$$V^{\tilde{\pi}}(\Omega - \{j\}) = V^{\tilde{\pi}}(\Omega) = V^{\pi}(\Omega) = V^{\pi}(\Omega - \{j\}) = R_i, \ \forall j \in \Omega, j \neq i \tag{31}$$

Equality (31) proves (27) for π when $S = \Omega$, and the basis for induction is established.

If $N = 2$, the argument of (31) completes the proof of (27). We now assume $N > 2$ and prove the induction step. Assume (27) is true for any state S where $|S| \geq L + 1, 2 \leq L < N$. Consider any state S_1 where $|S_1| = L$.

We first prove that

$$V^{\pi}(S_1) = V^{\tilde{\pi}}(S_1) \tag{32}$$

If there exists $j \in \Omega - S_1$ such that $\pi(S_1 \cup \{j\}) \neq j, r_j$, then by (29) we have $\pi(S_1) = \pi(S_1 \cup \{j\})$. By the induction hypothesis, equation (27) is true for $S = S_1 \cup \{j\}$, because $|S_1 \cup \{j\}| = L + 1$. We thus have

$$V^{\pi}(S_1 \cup \{j\} - \{j\}) = V^{\tilde{\pi}}(S_1 \cup \{j\} - \{j\}) \tag{33}$$

Equation (32) follows from (33).

If no such j exists, then by (30) we have $\pi(S_1) = \tilde{\pi}(S_1)$. Because $\pi(S_1) = \tilde{\pi}(S_1)$, and $V^{\pi}(S) = V^{\tilde{\pi}}(S), \forall S \supset S_1$ (because of the induction hypothesis), the conditions (19) and (20) of Lemma 3.3 are satisfied by π and $\tilde{\pi}$ for S_1. Hence

$$V^{\pi}(S_1) = V^{\tilde{\pi}}(S_1) \tag{34}$$

We have shown that (32) holds for any S_1 where $|S_1| = L$. We use (32) to show that (27) holds for all S_1 where $|S_1| = L$. For the remainder of the proof, assume that either $\pi(S_1) = i$ or $\pi(S_1) = r_i$, and let $j \in S_1, j \neq i$.

Consider first the case where $\pi(S_1) = r_i$. By (29), $\pi(S_1 - \{j\}) = r_i$, so that

$$V^{\pi}(S_1 - \{j\}) = V^{\pi}(S_1) = R_i \tag{35}$$

By Lemma 3.2 and the optimality of $\tilde{\pi}$, we have

$$V^{\tilde{\pi}}(S_1 - \{j\}) \leq V^{\tilde{\pi}}(S_1) \tag{36}$$

By (32),

$$V^{\tilde{\pi}}(S_1) = V^{\pi}(S_1) = R_i \tag{37}$$

But $i \in S_1 - \{j\}$, and $\tilde{\pi}$ is an optimal policy, so

$$V^{\tilde{\pi}}(S_1 - \{j\}) \geq R_i \tag{38}$$

Relations (36), (37), and (38) together imply that

$$V^{\tilde{\pi}}(S_1 - \{j\}) = V^{\tilde{\pi}}(S_1) = R_i \tag{39}$$

Equations (32), (35), and (39) imply that

$$V^{\tilde{\pi}}(S_1 - \{j\}) = V^{\tilde{\pi}}(S_1) = V^{\pi}(S_1) = V^{\pi}(S_1 - \{j\}) = V^{\tilde{\pi}}(S_1 - \{j\}) \tag{40}$$

and the induction step for (27) is proved when $\pi(S_1) = r_i$.

Now consider the case where

$$\pi(S_1) = i \tag{41}$$

We claim that

$$\pi(S) \neq j, r_j \quad \forall S \supseteq S_1 - \{j\} \tag{42}$$

We prove (42) as follows. Let S be any state where $S \supseteq S_1 - \{j\}$. If $j \notin S$, then $\pi(S) \neq j, r_j$, and (42) follows. Assume $j \in S$. Then $S \supseteq S_1$ and $|S| \geq L$. If $|S| = L$, then $S = S_1$ and $\pi(S) = i \neq j, r_j$. Assume $|S| > L$. If $\pi(S) = j$, then by successively removing nodes $k \in \Omega - S_1$ and using (29) we obtain $\pi(S_1) = j$, which contradicts (41). If $\pi(S) = r_j$, then by successively removing nodes $k \in \Omega - S_1$ and using (29), we obtain $\pi(S_1) = r_j$, which contradicts (41). Hence (42) is true in all cases.

By the decoupling property and (42), we have $V^{\pi}(S_1 - \{j\}) = V^{\pi}(S_1)$. Using $V^{\pi}(S_1 - \{j\}) = V^{\pi}(S_1)$ with (32), Lemma 3.2, and noting the optimality of $\tilde{\pi}$, we have

$$V^{\pi}(S_1 - \{j\}) = V^{\pi}(S_1) = V^{\tilde{\pi}}(S_1) \geq V^{\tilde{\pi}}(S_1 - \{j\}) \geq V^{\pi}(S_1 - \{j\}) \tag{43}$$

Relation (43) proves the induction step for (27) for $\pi(S_1) = i$.

This completes the induction step for (27). By induction, we have proved that (27) is true for π for all $S \subseteq \Omega$ where $|S| \geq 2$.

Finally, we prove that π is an optimal policy. First, note that by (27),

$$V^{\pi}(S) = V^{\tilde{\pi}}(S), \quad \forall S, |S| \geq 2 \tag{44}$$

Relation (27) also implies that

$$V^{\pi}(\{j\}) = V^{\tilde{\pi}}(\{j\}), \quad \forall j \in \Omega \text{ s.t. } \pi(\{i\} \cup \{j\}) = j, \text{ for some } i \in \Omega \tag{45}$$

We are left to consider $V^{\pi}(\{j\})$ when no such i as in (45) exists. By (30) we have $\pi(\{j\}) = \tilde{\pi}(\{j\})$ when $\pi(\{j\} \cup \{i\}) = i, \forall i \in \Omega$. Under this condition and (27), and identifying $S_1 = \{j\}$, π and $\tilde{\pi}$ satisfy the requirements of Lemma 3.3 (19) and (20). Lemma 3.3 (21) then implies that

$$V^{\pi}(\{j\}) = V^{\tilde{\pi}}(\{j\}), \quad \forall j \in \Omega \text{ s.t. } \pi(\{j\} \cup \{i\}) = i, \forall i \in \Omega \tag{46}$$

Equations (44), (45), and (46) prove the optimality of π. The proof of the lemma is complete. $\Box$

Besides Lemma 3.4, we need one more result to prove Theorem 3.1, which is that the value function for Problem $(\mathbf{P_1})$ is always an index function. We present this lemma next.

Lemma 3.5 *For any optimal Markov policy $\tilde{\pi}$, $V^{\tilde{\pi}}$ is an index function on Ω.*

*Proof.*First note that Lemma 3.2 implies Lemma 3.1 (12) is satisfied for $V^{\tilde{\pi}}$ on Ω.

Next, let π be the Markov policy satisfying (25) and (26) as constructed in Lemma 3.4. Consider any state S and $i \in S$ s.t. $V^{\pi}(S) \neq V^{\pi}(\{i\})$. If $\pi(S) = i$, then by removing all nodes except i from S via repeated application of Lemma 3.4 we would get $V^{\pi}(S) = V^{\pi}(\{i\})$, a contradiction. Hence, $\pi(S) \neq i$. So by Lemma 3.4 (27), we have $V^{\tilde{\pi}}(S) = V^{\tilde{\pi}}(S - \{i\})$. Thus the requirement (13) of Lemma 3.1 is satisfied for $V^{\tilde{\pi}}$ on Ω.

Since both requirements of Lemma 3.1 are satisfied, we have shown that $V^{\tilde{\pi}}$ is an index function on Ω. $\Box$

We now use Lemma 3.1-Lemma 3.5 to prove Theorem 3.1.

Proof of Theorem 3.1

Let π be the Markov policy satisfying (25) and (26) as constructed in Lemma 3.4. By Lemma 3.4 (3.), π is an optimal Markov policy. Hence, Lemma 3.5 indicates that V^{π} is an index function.

We next show that π is a priority policy. By (25) and (26) we have

$$\pi(S) = i \implies \pi(S') = i, \ \forall S' \subseteq S, \ i \in S' \tag{47}$$

$$\pi(S) = r_i \implies \pi(S') = r_i, \ \forall S' \subseteq S, \ i \in S' \tag{48}$$

Properties (47) and (48) show that π is a is a priority policy (cf. Definition 3.4), with node priority as follows. For any S where $\pi(S) = i$ or $\pi(S) = r_i$, i has priority higher than all other nodes of S.

We have shown that π is a priority policy with index function V^{π}. Hence π satisfies Definition 3.6, and is an index policy. $\Box$

Note that (47) and (48) imply that $\pi(S_1) = i$ and $\pi(S_2) = r_i$ cannot both occur for a given π for any S_1, S_2. That is, for a given system for Problem $(\mathbf{P_1})$ and an optimal index policy π, if there is a state where π transmits from node i, then there is no state where π retires and receives the

reward from i. Similarly, if there is a state where π retires and receives the reward from node i, then there is no state where π transmits at i. We henceforth use this fact when we write the node priority list for an index policy π, where exactly one of i and r_i is listed for each $i \in \Omega$. When r_i occurs in the list, this means that for states which include i and no nodes of higher priority than i, the optimal action is is to retire. The reward received at retirement will be R_i.

It is interesting to note that the total number of stationary Markov policies for Problem $(\mathbf{P_1})$ with N nodes is $N^{(2^N)}$, whereas the total number of priority policies is $N!$. Based on the result of Theorem 3.1, we develop an algorithm which is able to compute an optimal index policy for Problem $(\mathbf{P_1})$ with computational complexity of only $O(N^2)$.

3.3.2 Description of Centralized Algorithm

We present an algorithm which computes the optimal index policy. As stated in Section 3.1, we use the notation $V_i^\pi := V^\pi(\{i\})$.

Algorithm 1 *(A Dijkstra-Type Algorithm for an Index Policy)*

Define the sets $\mathcal{A}$ and $\mathcal{X}$ as follows.

Initially: *$\mathcal{A}$ contains the nodes of highest reward in arbitrary order (there must be at least one such node); the action taken by the optimal index policy π on these nodes is r_i. $\mathcal{X}$ is the unordered complement (w.r.t. Ω) of the set of nodes of highest reward.*

During the construction of optimal policy π: *$\mathcal{A}$ contains a priority list of a set S of nodes, $S \subset \Omega$, together with the action specified by π on each node in S. $\mathcal{X}$ is the unordered complement (w.r.t. Ω) of $\mathcal{A}$.*

The algorithm proceeds as follows.

1. *For each $i \in \mathcal{X}$, let π_i be an index policy with the same priority list as π for the nodes of $\mathcal{A}$, with i as the next highest priority node after $\mathcal{A}$, and with the priority of the nodes $\mathcal{X} - \{i\}$ arbitrary, but lower than i. Compute $V_i^{\pi_i}$ from*

$$V_i^{\pi_i} = \max\left\{ \frac{-c_i + \sum_{S \supset \{i\}: \pi_i(S) \neq i} P^i(S|i) V_{\pi_i(S)}^{\pi_i}}{\sum_{S \supset \{i\}: \pi_i(S) \neq i} P^i(S|i)}, R_i \right\} \tag{49}$$

2. *Choose $i \in \mathcal{X}$ with the highest value of $V_i^{\pi_i}$, with ties broken arbitrarily. Append this node to the list $\mathcal{A}$ as the next priority node, together with the action specified by (49). Remove i from $\mathcal{X}$.*

3. If $\mathcal{X}$ is empty, stop. If not, go to step 1.

Remark: In Step 1 the right-hand-side of (49) computes the best expected reward for node i, assuming i is the node of next highest index in π. This computation is feasible because π is a priority policy.

We now establish a relation between (49) and the optimality equation for Problem $(\mathbf{P_1})$. Such a relation allows us to prove that Algorithm 1 indeed computes an optimal index policy. Theorem 3.1 states that there is an optimal index policy for Problem $(\mathbf{P_1})$. Hence the Dynamic Programming equation for Problem $(\mathbf{P_1})$ can be written

$$V_i^\pi = \max\left\{\max_{\tilde{\pi}}\left\{-c_i + \sum_{S\supseteq\{i\}} P^i(S|i)V_{\tilde{\pi}(S)}^{\tilde{\pi}}\right\}, R_i\right\}, \quad \forall i \in \Omega \tag{50}$$

where the inner maximum is taken over all index policies $\tilde{\pi}$. In the following lemma we use the existence of optimal index policy π to put the computation of V_i^π into a more convenient form.

Lemma 3.6 *Assume π is an optimal index policy for Problem $(\mathbf{P_1})$. Then*

$$V_i^\pi = \max\left\{\max_{\tilde{\pi}}\left\{\frac{-c_i + \sum_{\mathcal{N}(i)\supseteq S\supset\{i\}:\tilde{\pi}(S)\neq i} P^i(S|i)V_{\tilde{\pi}(S)}^{\tilde{\pi}}}{\sum_{\mathcal{N}(i)\supseteq S\supset\{i\}:\tilde{\pi}(S)\neq i} P^i(S|i)}\right\}, R_i\right\}, \quad \forall i \in \Omega \tag{51}$$

where the inner maximum is taken over all index policies $\tilde{\pi}$.

*Proof.*Let $i \in \Omega$, and let $\tilde{\pi}$ be any index policy with the property that $\tilde{\pi}$ transmits when in state $\{i\}$. We have for $\tilde{\pi}$

$$\begin{aligned} V_i^{\tilde{\pi}} &= -c_i + \sum_{S\supseteq\{i\}} P^i(S|i)V_{\tilde{\pi}(S)}^{\tilde{\pi}} \\ &= -c_i + \sum_{S\supset\{i\}:\tilde{\pi}(S)\neq i} P^i(S|i)V_{\tilde{\pi}(S)}^{\tilde{\pi}} + \sum_{S\supseteq\{i\}:\tilde{\pi}(S)=i} P^i(S|i)V_{\tilde{\pi}(S)}^{\tilde{\pi}} \end{aligned} \tag{52}$$

Solving (52) for $V_i^{\tilde{\pi}}$ we obtain

$$V_i^{\tilde{\pi}} = \frac{-c_i + \sum_{S\supset\{i\}:\tilde{\pi}(S)\neq i} P^i(S|i)V_{\tilde{\pi}(S)}^{\tilde{\pi}}}{\sum_{S\supset\{i\}:\tilde{\pi}(S)\neq i} P^i(S|i)} \tag{53}$$

Consequently, for optimal index policy π we have

$$V_i^\pi = \max\left\{\max_{\tilde{\pi}}\left\{\frac{-c_i + \sum_{S\supset\{i\}:\tilde{\pi}(S)\neq i} P^i(S|i)V_{\tilde{\pi}(S)}^{\tilde{\pi}}}{\sum_{S\supset\{i\}:\tilde{\pi}(S)\neq i} P^i(S|i)}\right\}, R_i\right\} \tag{54}$$

Because (54) is true for any $i \in \Omega$, this proves (51). $\square$

In the following corollary we show an important property of the update (49) and its relation to (50).

Corollary 3.1 *Assume π is an optimal index policy for Problem $(\mathbf{P_1})$. Let π_i be as defined in Algorithm 1, that is, π_i is the same as π for the set of highest priority nodes $\mathcal{A}$ of π, and node i is the node of highest priority in $\mathcal{X}$ according to π_i. Assume i is also the node of highest priority in $\mathcal{X}$ according to π. If $V_i^{\pi_i}$ is computed as in (49), then*

$$V_i^{\pi_i} = V_i^{\pi} \tag{55}$$

*Proof.*Corollary 3.1 follows from the fact that (49) computes (51) with the policy $\tilde{\pi} = \pi_i$, which is optimal for all nodes of priority i or higher. $\square$

Algorithm 1 also resembles Klimov's algorithm [Klim 74] and has the following feature.

Theorem 3.2 *For Problem $(\mathbf{P_1})$, Algorithm 1 produces an optimal index policy.*

*Proof.*We prove the theorem by induction on the number of nodes in the set $\mathcal{A}$ defined in the description of Algorithm 1. Recall that for a Markov decision problem, the optimal policy maximizes the value function for each state.

Let π be an optimal index policy. Suppose Algorithm 1 has run to the point that $|\mathcal{A}| = L$. Let $i \in \mathcal{X}$ be the node with the actual $(L+1)$'th highest priority according to π, whether retiring or not. Let $j \in \mathcal{X}, j \neq i$. Let π_j denote the priority policy that has the same priority as π in the first L nodes, gives j the $(L+1)$'th node priority, retiring or not as optimal, and arbitrarily gives priority lower than $L+1$ to the remaining nodes.

Then we claim that

$$V_i^{\pi_i} \;=\; V_i^{\pi} \;\geq\; V_j^{\pi} \;\geq\; V_j^{\pi_j} \tag{56}$$

The equality of (56) follows from Corollary 3.1, because i is assumed to be the actual $(L+1)$'th priority node of π. The first inequality of (56) follows because by assumption i is higher priority than j in π. The second inequality follows because π is an optimal policy.

Relation (56) implies that any node $j \in \mathcal{X}$ maximizing $V_j^{\pi_j}$ may optimally be made the $(L+1)$'th priority node. Note that this j is not necessarily unique. This is the procedure used to find the node of next highest priority in Step 2 of Algorithm 1.

This completes the proof of the induction step. Hence, by induction Algorithm 1 produces an optimal index policy. □

3.3.3 Remarks

1. The index policy result of Theorem 3.1 is very general. It says that no matter what the actual values of $(\mathbf{P}, c, R)_i$ are at each node, an index policy is always one possible optimal policy *structure*. There might be errors in knowledge of $(\mathbf{P}, c, R)_i$, and this affects the indices of the computed optimal index policy. But the fact that *some* index policy is optimal is always true.

2. We have shown that Algorithm 1 computes an optimal index policy for Problem $(\mathbf{P_1})$. It is important to understand that Algorithm 1 uses all of the network parameters, meaning the $(\mathbf{P}, c, R)_i$ values at each node. To run this algorithm, all of this information must be available at the same location. In this sense, Algorithm 1 is a *centralized* algorithm. It shares this property with, for example, the standard Dijkstra algorithm used in OSPF (see [Kuro 00] Ch. 4), which assumes that, through flooding, full network topology information has been transmitted to *every* network node. Each node can then run its own centralized Dijkstra algorithm.

3. Assuming the necessary network topology information is available in one location, the computational complexity of Algorithm 1 is $O(N^2)$. This follows directly from the similarity in procedure of Algorithm 1 to the standard Dijkstra algorithm.

4. As discussed in Section 2.7.1, in most network models the network layer requests point-to-point communication from the link layer. However, the stochastic local broadcast nature of our link model requires a more nuanced control, one describing alternative courses of action depending on feedback from random transmission events. In light of Theorem 3.1, we can now state precisely what this means for Problem $(\mathbf{P_1})$. The instructions for transmission at a node consist of a priority list of neighbors and the transmitting node itself. The node transmits until a node of higher priority successfully receives the message. This is the link layer function. Note that the network layer does not dictate the route, and in fact there is no *one* route. The actual route a message takes between source and destination is sample path dependent.

3.3.4 Distributed Implementation of an Optimal Index Policy

We note an interesting feature of an index policy used for Problem ($\mathbf{P_1}$). The indices of the network nodes are fixed, and at each transmission a node of highest index which has the message is chosen to transmit. This leads to the following property.

Property 3.1 *In an index policy for Problem* ($\mathbf{P_1}$)*, the only nodes of index higher than the transmitting node which can receive the message are neighbors of the transmitting node.*

Property 3.1 allows for a natural disributed *implementation* of the index policy, as follows. Imagine there is a token associated with the message that begins with the message at the node of origin. The token indicates which node is to transmit next. After a node transmits, it passes the token to a neighbor (that depends on the outcome of the transmission), or keeps the token for retransmission. By Property 3.1, an optimal index policy can be implemented in this way, as the optimal next node is always a neighbor. Note that there is no central control of this token passing mechanism. All decisions are made locally, and involve only neighboring nodes.

It is important to distinguish distributed optimal routing policy implementation from distributed *computation* of an optimal index policy itself. For policy implementation, it is assumed that the index policy has already been determined. We consider the problem of distributed index policy computation in Section 6.

4 The Time-Invariant Transmission Control Problem

In this section we extend the model of Problem ($\mathbf{P_1}$) to allow for control of transmission type at each node. That is, at each time step the controller may choose a node for transmission, and also a type of transmission at that node. Transmission type may be used to model various physical layer features, such as multiple transmission power levels, modulation/coding scheme, antenna directionality, and destination addressing. We begin with some notation and definitions.

4.1 Notation and Definitions

Definition 4.1 *W_i refers to the number of transmission types available at node i.*

Definition 4.2 *We write $\pi(S) = (i, k)$ to mean that when in state S, policy π chooses node i and transmission type k, $i \in \Omega, k \in 1 \ldots W_i$. The expression $\pi(S) = (i, *)$ means policy π chooses i at some unspecified transmission type. The notation for retirement $\pi(S) = r_i$ is retained unchanged.*

Definition 4.3 *When in state S for which $\pi(S) = (i,k)$, a cost $c_{(i,k)}$ is incurred, and the transition probabilities are written $P^{(i,k)}(S'|S)$.*

4.2 Statement of Problem

Problem $(\mathbf{P_2})$

We consider Problem $(\mathbf{P_1})$, with the following addition. At each time step the central controller chooses a node for transmission, from among the nodes with the message, and a transmission *type* from among the allowable types for that node. To each node and transmission type is associated a transmission cost and a probability that a given set of nodes receives the message. We seek a policy which maximizes (11) under the conditions of Problem $(\mathbf{P_1})$ and the above addition.

4.3 Analysis of Problem $(\mathbf{P_2})$

The time-homogenous Markov nature of Problem $(\mathbf{P_1})$ is not altered by adding in a choice of transmission type. We find that with an appropriate mapping to a new larger space (each transmission type becomes a node), we can apply the result of Problem $(\mathbf{P_1})$ directly. To demonstrate this, we define a new problem, Problem $(\mathbf{P_3})$, show its relation to Problem $(\mathbf{P_2})$, and then show that it is equivalent to Problem $(\mathbf{P_1})$.

4.3.1 Notation and Definitions for Problem $(\mathbf{P_3})$

We define a new space of nodes Ω_P as follows. As in Definition 4.2, we list possible control choices for Problem $(\mathbf{P_2})$ as (i,k), where $i \in \Omega$ and $k \in 1 \dots W_i$. Every node of Ω_P corresponds to exactly one such control choice of Problem $(\mathbf{P_2})$. Defining N_P to be the cardinality of Ω_P, we then have

$$N_P := |\Omega_P| = \sum_{i=1}^{N} W_i \tag{57}$$

where as before $N := |\Omega|$. We write $\mathrm{j} \in \Omega_P$ to refer to a node from Ω_P. We can also refer to the (i,k) pair of a node $\mathrm{j} \in \Omega_P$, which is the associated i and transmission type $k \in W_i$ in the original space Ω. We say that the group of nodes in Ω_P which corresponds to $i \in \Omega$ is the *family* of i, so that the family of i has W_i members. The family of $S \subseteq \Omega$ is the set of all nodes in Ω_P which are in the family of any one of the nodes of S.

The cost to transmit from a node j with (i,k) pair is defined to be $c_{(i,k)}$, the cost for transmission type k at i in Problem $(\mathbf{P_2})$.

Transmissions in the Ω_P space are based on the corresponding events in the Ω space, as follows. Say a node j with (i,k) pair is chosen for transmission. This incurs a cost $c_{(i,k)}$. On Ω, transmission from i with transmission type k leads to a set of nodes, say $S_1 \subseteq \Omega$, receiving the message. Correspondingly, on Ω_P by definition a node $\mathsf{j} \in \Omega_P$ with (l,k) pair receives the message if and only if l receives the message on Ω, $l \in S_1$. That is, nodes in Ω_P in the family of l receive the message precisely when l does. Note that this means that message reception for nodes of Ω_P in the same family are deterministically coupled, in that they all receive or all do not receive it. This strongly restricts the kinds of transitions that can occur in Ω_P.

Each member of the family of $i \in \Omega$ gets the same reward as i. That is, let R_P be the reward function on Ω_P, an index function on Ω_P. If $\mathsf{j} \in \Omega_P$ is in the family of i, then $R_P(\{\mathsf{j}\}) = R(\{i\})$. Because R_P is an index function, this fully defines R_P on Ω_P.

4.3.2 Formulation of Problem $(\mathbf{P_3})$

Problem $(\mathbf{P_3})$

Problem $(\mathbf{P_3})$ is a formulation of Problem $(\mathbf{P_1})$ for the space Ω_P, with the transition probabilities, cost, and reward described above. The starting state is the family of S_o, the initial state for Problem $(\mathbf{P_2})$.

At each time point the controller either chooses a node from Ω_P for transmission, or retires. We seek a policy which maximizes (11) under the conditions of Problem $(\mathbf{P_1})$ defined on Ω_P.

4.3.3 Relation of Problem $(\mathbf{P_3})$ to Problem $(\mathbf{P_2})$

We show the relation of Problem $(\mathbf{P_3})$ to Problem $(\mathbf{P_2})$. There is a one-to-one mapping from the states of Ω-space to the states of Ω_P-space, as follows. Let $S_1 \subseteq \Omega$ be a state of Problem $(\mathbf{P_2})$. The state of Problem $(\mathbf{P_3})$ corresponding to S_1 is the set of all nodes $\mathsf{j} \in \Omega_P$ such that j is in the family of some node $i \in S_1$. There are states of Problem $(\mathbf{P_3})$ which do not correspond to any state in Problem $(\mathbf{P_2})$. To see this, let $\mathsf{j}, \mathsf{k} \in \Omega_P$ both be in the family of $i \in \Omega$. Consider a state $S \subset \Omega_P$ where $\mathsf{j} \in S, \mathsf{k} \notin S$; this state S is not the mapping from any state $S_1 \subseteq \Omega$. The one-to-one mapping from Ω to Ω_P is not in general an *onto* mapping, because $|\Omega_P| > |\Omega|$. Those

states $S \subseteq \Omega_P$ that are not the image of any state $S \subseteq \Omega$ under the above-mentioned mapping are not reachable under the state transition mechanism of Problem $(\mathbf{P_3})$, and play no role in the analysis.

A control action for Problem $(\mathbf{P_3})$ has a a corresponding action in Problem $(\mathbf{P_2})$, as follows. A node $\mathrm{j} \in \Omega_P$, with associated (i,k), chosen for transmission in Problem $(\mathbf{P_3})$ corresponds to transmission at node i of type k in Problem $(\mathbf{P_2})$. The cost $c_{(i,k)}$ incurred for this transmission is the same in both problems, and the state reached by the transmission in Ω maps to the state reached in Ω_P. When retirement is chosen for Problem $(\mathbf{P_3})$, this corresponds to retirement for Problem $(\mathbf{P_2})$, and the same reward is received, because corresponding states have the same reward.

Hence, Problem $(\mathbf{P_2})$ and Problem $(\mathbf{P_3})$ are entirely equivalent, in that to each decision policy for Problem $(\mathbf{P_3})$ there is a corresponding decision policy for Problem $(\mathbf{P_2})$ which results in exactly the same behavior, and hence expected reward, for both systems. We can thus solve Problem $(\mathbf{P_2})$ by finding an optimal policy for Problem $(\mathbf{P_3})$. We proceed to solve Problem $(\mathbf{P_3})$.

4.3.4 Analysis of Problem $(\mathbf{P_3})$

We show that the system of Problem $(\mathbf{P_3})$ satisfies the requirements of Problem $(\mathbf{P_1})$, and hence is a special case of that system. Specifically, we show that the increasing and decoupling properties (cf. Definition 3.1 and Definition 3.2) are satisfied, and that the reward function is an index function.

The increasing property on Ω_P follows directly from the fact that the increasing property holds on the underlying space Ω, together with the way states of Ω map to Ω_P. That is, transmissions only lead to an increasing state in Ω, leading to an increasing state in Ω_P.

The decoupling property holds for Problem $(\mathbf{P_3})$ because it holds for Problem $(\mathbf{P_2})$. The nodes that receive transmissions in Problem $(\mathbf{P_2})$ are unaffected by what other nodes have the message.

Note that transmission events in Ω_P can be highly correlated, in that nodes of Ω_P in the same family either all have the message, or none have it. However, such event correlations are allowed by Model **(M)**.

Finally, R_P satisfies the definition of an index function on Ω_P, because R is an index function on Ω and R_P gives the same reward for the associated states of Ω_P.

Hence, an index policy is optimal for Problem $(\mathbf{P_3})$, and Algorithm 1 can be used to determine such a policy. This policy is also optimal for Problem $(\mathbf{P_2})$. Thus, Algorithm 1 is effectively used

to solve Problem ($\mathbf{P_2}$).

Note that in the resulting priority list of an optimal index policy π, nodes from the same family in Ω_P will appear in some relative order. Since nodes in a given family either all have the message or none have it, it is clear that only one node from each family will ever be used for transmission by the algorithm. Hence, all the nodes that are not of the highest priority within their family can be removed from the priority list for simplicity. In the terms of Problem ($\mathbf{P_2}$), this means that only one transmission type at each node is ever used for transmission.

Algorithm 1 when applied to Problem ($\mathbf{P_2}$) in this manner is $O(N_P^2)$ in complexity, that is, of complexity $O\left(\left(\sum_{i=1}^{N} W_i\right)^2\right)$.

4.4 Reversion of Algorithm 1 to Dijkstra's Algorithm

We demonstrate that Algorithm 1 can be interpreted as a generalization of Dikstra's Algorithm ([Kuro 00] Ch. 4) to the case of stochastic local broadcast transmissions. Assume we have a system for Problem ($\mathbf{P_2}$), and the corresponding system of Problem ($\mathbf{P_3}$) with nodes labeled (i,k). These problems are equivalent, and we use either system as appropriate to simplify notation. We begin with two definitions.

Definition 4.4 *For Problem* ($\mathbf{P_2}$)*, we say the system is* deterministic *if*

$$P^{(i,k)}(S|i) = 0 \qquad or \qquad P^{(i,k)}(S|i) = 1, \quad \forall S \subseteq \mathcal{N}(i), \forall i \in \Omega, k \in 1 \dots W_i \tag{58}$$

Definition 4.5 *For Problem* ($\mathbf{P_2}$)*, we say a deterministic system is* wired *if for each node $i \in \Omega$, to each $k \in 1 \dots W_i$ there corresponds a unique $b_k \in \mathcal{N}(i)$, and*

$$P^{(i,k)}(\{b_k, i\}|i) = 1 \tag{59}$$

$$P^{(i,k)}(S|i) = 0, \forall S \neq \{b_k, i\} \tag{60}$$

We show that the Stochastic Dijkstra Algorithm (Algorithm 1) reverts to the standard Dijkstra Algorithm in a wired system.

Assume we have a wired system with one destination node, node j, of sufficiently large reward that optimally no node retires except j. Suppose Algorithm 1 has run on the wired system formulated as Problem ($\mathbf{P_3}$), and an optimal index policy π has been computed. Consider the nodes of Problem ($\mathbf{P_3}$), each with label (i,k). Algorithm 1 without retirement gives

$$V_{(i,k)}^{\pi} = \frac{-c_{(i,k)} + \sum_{S \supset \{i\}: \pi(S) \neq i} P^{(i,k)}(S|(i,*)) V_{\pi(S)}^{\pi}}{\sum_{S \supset \{i\}: \pi(S) \neq i} P^{(i,k)}(S|(i,*))}, \tag{61}$$

Furthermore, for the wired system (59) and (60) give

$$P^{(i,k)}(\{(b_k,*),(i,k)\}|(i,*)) = 1 \tag{62}$$

Due to the nature of the index policy, when the set of nodes $(b_k,*)$ is reached, the node of highest index in $(b_k,*)$ determines the value function. Using this fact with (62), and letting (b_k,l) be the node of highest value function in $(b_k,*)$, Equation (61) becomes

$$V^{\pi}_{(i,k)} = -c_{(i,k)} + V^{\pi}_{(b_k,l)} \tag{63}$$

Equation (63) can be interpreted as follows. As Algorithm 1 runs on the wired system in the form of Problem ($\mathbf{P_3}$), the largest value, over $k \in W_i$ and $l \in W_{b_k}$, for the RHS of (63) is chosen as the value function for each non-destination node. Because the destination reward is fixed, this is effectively the *same* value function chosen by the deterministic Dijkstra Algorithm, where $c_{(i,k)}$ is the link cost and k represents which link is chosen for transmission.

In this way, with deterministic transitions and choice of which point-to-point link to transmit on, Problem ($\mathbf{P_3}$) (and the equivalent Problem ($\mathbf{P_2}$)) reverts to the standard deterministic shortest path problem, and Algorithm 1 performs the same function as the Dijkstra Algorithm.

5 The Time-Varying Stochastic Routing Problem

In this section we formulate and analyze a time-varying stochastic routing problem. We consider a time-varying version of Model **(M)** and the associated stochastic optimal routing problem. Such a model and problem, formulated precisely in Problem ($\mathbf{P_4}$) below, capture various aspects of Quality of Service (QoS) requirements, such as timeliness and delivery quality, in ad hoc networks.

5.1 Statement of Problem

Problem ($\mathbf{P_4}$)

We consider Problem ($\mathbf{P_1}$), with the following modifications. We allow the parameters in Model **(M)** to be time-varying. That is, at node i the transmission cost is $c_{i,t}$ ($c_{i,t} > 0, \forall t, i$), the transition probability is P^i_t, and the reward function is $R_{i,t}$. The overall reward function is written R_t, and is an index function on Ω at each t. We further assume the existence of a time τ such that $R_{i,t} = 0$, $\forall t \geq \tau, \forall i$. We seek a policy which maximizes (11) under the above conditions.

5.2 Analysis of Problem ($\mathbf{P_4}$)

Without any loss of optimality we restrict attention to Markov policies. Recall that for Problem ($\mathbf{P_4}$) there is a time τ such that $R_{i,t} = 0,\ \forall t \geq \tau, \forall i$. We refer to τ as the *stop time*. Throughout this section we define w as the time-to-go until τ, so that $w := \tau - t$. We notate a time-varying Markov policy defined over a backward time interval $1, 2, \ldots, w$ as $\pi^{\mathbf{w}} := (\pi_1, \pi_2, \ldots, \pi_w)$, where π_w refers to the Markov policy at time-to-go w.

We call $\pi^{\mathbf{w}}$ a *time-varying priority* or *index policy* when π_w is a priority or index policy, respectively, at each w (cf. Definition 3.4 and Definition 3.6). We say $\pi^{\mathbf{w}}$ is of *time-invariant priority* when the node priorities do not change over time according to $\pi^{\mathbf{w}}$. Time-invariant priority includes the case where the node priorities do not change, but the optimal *action* at some nodes changes (e.g. transmits become retires, or vice versa).

There are instances of Problem ($\mathbf{P_4}$) for which no time-invariant priority policy is optimal. For example, high priority sources that are multiple hops from a reward node must lose priority as time τ approaches. More interestingly, there are instances of Problem ($\mathbf{P_4}$) for which no time-varying index or priority policy is optimal. In this section we provide conditions under which a time-varying index or time-varying priority policy is optimal for Problem ($\mathbf{P_4}$). We begin by investigating conditions under which a time-varying index policy is optimal.

5.2.1 Conditions for Time-Varying Index Policy Optimality

We find a necessary and sufficient condition for optimality of a time-varying index policy for Problem ($\mathbf{P_4}$).

To model the idle action, we add a node Q_0 to Ω, for which $c_{0,t} = 0, R_{0,t} = 0, P_t^0(S|S) = 1\ \forall S \subseteq \Omega, \forall t$. Also, we always assume that $Q_0 \in S_o$, so that the idle action is always available by choosing Q_0. In the subsequent analysis, we need only include this node in the system to include the idling action.

Note that in general the value functions are functions of time. Let $V^{\pi^{\mathbf{w}}}(S)$ be the value function when in state S at time $\tau - w$ under $\pi^{\mathbf{w}}$.

Definition 5.1 *We write $\pi_i^{\mathbf{w}}$ to denote a policy which chooses $i \in \Omega$ for transmission at w, and then is identical to $\pi^{\mathbf{w-1}}$ thereafter.*

We present an algorithm that computes an optimal time-varying index policy for Problem ($\mathbf{P_4}$) whenever such a policy exists.

Algorithm 2 *The algorithm is defined inductively on the time-to-go w.*

When $w = 1$, the optimal policy π_1 is to retire and obtain the reward of the node which has received the message and has the largest $R_{i,\tau-1}$. This π_1 is an index policy.

Assume that for some arbitrary $w - 1 < \tau$ we have defined $\pi^{\mathbf{w-1}}$, an optimal time-varying index policy. We show how an optimal index policy π_w is computed, assuming a certain system condition (specified below) is satisfied at each node computation. If the condition is not satisfied, the algorithm halts, and an optimal time-varying index policy does not exist.

To compute the policy at w, we start with two sets $\mathcal{A}$ and $\mathcal{X}$ that are defined in precisely the same way as in Algorithm 1.

The algorithm proceeds in three steps.

1. *Let $R^{\mathcal{X}} := \max_{i\in\mathcal{X}} R_{i,\tau-w}$. Let $\pi_i^{\mathbf{w}}$ be as in Definition 5.1. Compute $V^{\pi_i^{\mathbf{w}}}(\mathcal{X})$ for each node $i \in \mathcal{X}$ using*

$$V^{\pi_i^{\mathbf{w}}}(\mathcal{X}) = -c_{i,\tau-w} + \sum_{S\supseteq\mathcal{X}} P^i_{\tau-w}(S|\mathcal{X}) V^{\pi^{\mathbf{w-1}}}(S) \tag{64}$$

   Define $H = \max_{i\in\mathcal{X}} V^{\pi_i^{\mathbf{w}}}(\mathcal{X})$.

2. *Consider two cases.*

Case 1: $R^{\mathcal{X}} < H$

Define $D = \{i \in \mathcal{X} : V^{\pi_i^{\mathbf{w}}}(\mathcal{X}) = H\}$. Suppose there exists $i \in D$ that satisfies either Condition 1 or Condition 2 stated below. Set $\pi_w(S) = i$, $\forall S \subseteq \mathcal{X}$ such that $i \in S$, append i to $\mathcal{A}$ and remove it from $\mathcal{X}$.

If there exists no $i \in D$ that satisfies either Condition 1 or Condition 2, halt. A time-varying optimal policy does not exist.

Condition 1 and Condition 2 are the following:

Let $j = \pi_{w-1}(\mathcal{X})$.

Condition 1 *$i = j$*

Condition 2 *$i \neq j$ and $\exists B \subseteq \Omega$ such that both of the following relations hold.*

$$V^{\pi^{\mathbf{w-1}}}(\{k\}) \geq V^{\pi^{\mathbf{w-1}}}(\{j\}) \ \ \forall k \in B \tag{65}$$

$$\sum_{S\supseteq\{i\}:B\cap S\neq\emptyset} P^i_{\tau-w}(S|i) = 1 \tag{66}$$

Case 2: $R^{\mathcal{X}} \geq H$

Set $\pi_w(S) = r_i$, $\forall S \subseteq \mathcal{X}$ *such that* $i \in S$, *where* i *is any node such that* $R^{\mathcal{X}} = R_{i,\tau-w}$; *append* r_i *to* $\mathcal{A}$ *(indicating retirement for node* i*) and remove* i *from* $\mathcal{X}$.

3. *If* $\mathcal{X}$ *is empty, an optimal index policy* $\pi^{\mathbf{w}}$ *for time* w *has been completely specified. Otherwise, go to Step 1.*

Relations (65) and (66) mean the following: when node i transmits at w (i.e. with w time units to go), a node that is at least as good as j at $w-1$ is reached with probability 1.

The following theorem summarizes the main result about how Algorithm 2 relates to Problem $(\mathbf{P_4})$.

Theorem 5.1 *A time-varying index policy is optimal for Problem* $(\mathbf{P_4})$ *if and only if Algorithm 2 terminates with* $w = \tau$.

Proof.Necessity. First, assume there is a $\mathcal{X}$ and $w < \tau$ where Algorithm 2 halts. We show that no time-varying index policy is optimal. Because Algorithm 2 halts before τ, this means that at $\mathcal{X}$: 1) there is no retirement, and 2) for any $i \in D$ we have $i \neq j$, and there is no $B \subseteq \Omega$ satisfying (65) and (66). Hence for each such $i \in D$ there exists $B \subseteq \Omega$ and $k \in B$ ($k = i$ is possible), where $\pi_{w-1}(B) = k$, and

$$V^{\pi^{\mathbf{w-1}}}(B) = V^{\pi^{\mathbf{w-1}}}(\{k\}) < V^{\pi^{\mathbf{w-1}}}(\{j\}) \tag{67}$$

$$P^i_{\tau-w}(B|i) > 0 \tag{68}$$

where the equality in (67) follows since $\pi^{\mathbf{w-1}}$ is an index policy at $w-1$. In words, (67) and (68) mean that there is a positive probability when transmitting at i at w to end up in a state whose best node is strictly worse than j under $\pi^{\mathbf{w-1}}$. Since $\pi^{\mathbf{w-1}}$ is an index policy at $w-1$ with j as the highest priority node in $\mathcal{X}$, recalling that $\pi_w(\mathcal{X}) = i$ and using (67) and (68) it then follows from (64) that

$$V^{\pi^{\mathbf{w}}}(\{i\}) < V^{\pi^{\mathbf{w}}}(\mathcal{X}) \tag{69}$$

Since any optimal time-varying index policy must play a node $i \in D, i \neq j$ at time w, and because (69) follows for each $i \in D, i \neq j$, no time-varying index policy is optimal.

Sufficiency. Now assume that Algorithm 2 runs to completion until $w = \tau$. We will show that at each step of the algorithm, as long as the algorithm does not halt, the node i added to $\mathcal{A}$

dominates the remaining nodes in $\mathcal{X}$ in the appropriate sense for an index policy. Assume that the algorithm has successfully run up to w, at which time we have currently defined $\mathcal{A}$ and $\mathcal{X}$, where $|\mathcal{X}| \geq 2$. Let $\pi_k^{\mathbf{w}}$ be defined as in Algorithm 2.

First, assume Case 1 is satisfied. Assume i has been chosen satisfying Condition 1, so that $i = j$. Let $S \subseteq \mathcal{X},\ j, k \in S$. We then have

$$\begin{aligned} V^{\pi_j^{\mathbf{w}}}(S) &= V^{\pi_j^{\mathbf{w}}}(\mathcal{X}) && (70)\\ &\geq V^{\pi_k^{\mathbf{w}}}(\mathcal{X}) && (71)\\ &= V^{\pi_k^{\mathbf{w}}}(S) && (72) \end{aligned}$$

Equation (70) follows from the decoupling property, and the facts that $V^{\pi^{\mathbf{w-1}}}$ is an index function, and that $\pi_{w-1}(\mathcal{X}) = j,\ j \in S \subseteq \mathcal{X}$. Inequality (71) follows because $j \in D$. Equation (72) follows from the decoupling property, and the facts that $V^{\pi^{\mathbf{w-1}}}$ is an index function, that $\pi_{w-1}(\mathcal{X}) = j,\ j \in S \subseteq \mathcal{X}$, and that $k \in S$. Because retirement is not chosen in Case 1, we also have

$$V^{\pi_j^{\mathbf{w}}}(\mathcal{X}) = V^{\pi_j^{\mathbf{w}}}(S) = V^{\pi_j^{\mathbf{w}}}(\{j\}) \geq R^{\mathcal{X}} \tag{73}$$

Hence $\pi_j^{\mathbf{w}}$ is as good as all other policies for all $S \subseteq \mathcal{X}, j \in S$, and j can thus be appended to $\mathcal{A}$.

Now assume Condition 2 is satisfied, so that $i \neq j$, but there is a $B \subseteq \Omega$ satisfying (65) and (66). Let $S \subseteq \mathcal{X},\ i, k \in S$. We then have

$$\begin{aligned} V^{\pi_i^{\mathbf{w}}}(S) &= V^{\pi_i^{\mathbf{w}}}(\mathcal{X}) && (74)\\ &\geq V^{\pi_k^{\mathbf{w}}}(\mathcal{X}) && (75)\\ &= V^{\pi_k^{\mathbf{w}}}(S) && (76) \end{aligned}$$

Equation (74) follows from the decoupling property, (65) and (66), and the fact that $V^{\pi^{\mathbf{w-1}}}$ is an index function. (75) follows because $i \in D$. (76) is by the decoupling property, $S \subseteq \mathcal{X}$, and the fact that $V^{\pi^{\mathbf{w-1}}}$ is an index function. Because retirement is not chosen in Case 1, we also have

$$V^{\pi_i^{\mathbf{w}}}(S) = V^{\pi_i^{\mathbf{w}}}(\mathcal{X}) \geq R^{\mathcal{X}} \tag{77}$$

Hence $\pi_i^{\mathbf{w}}$ is as good as all other policies for all $S \subseteq \mathcal{X}, i \in S$, and i can thus be appended to $\mathcal{A}$.

Finally, assume Case 2 is satisfied, so that retirement is chosen, receiving reward $R_{i,\tau-w}$, where possibly $i = j$. Let $S \subseteq \mathcal{X},\ i, k \in S$. By (70) and (74) and the choice of retirement in Step 1, we then have

$$R_{i,\tau-w} \geq V^{\pi_i^{\mathbf{w}}}(\mathcal{X}) = V^{\pi_i^{\mathbf{w}}}(S) \geq V^{\pi_k^{\mathbf{w}}}(S) \tag{78}$$

Hence, retirement receiving reward $R_{i,\tau-w}$ is optimal for all $S \subseteq \mathcal{X}, i \in S$, and r_i can thus be appended to $\mathcal{A}$. $\square$

Remarks

1. As mentioned, relations (65) and (66) in Step 2 of Algorithm 2 mean that when node i transmits with w time units to go, a node that is at least as good as j at $w-1$ is reached with probability 1. That this condition is both necessary and sufficient when $i \neq j$ follows from the observation that for a time-varying index policy which plays i at w to be optimal, the question of whether or not node $j \neq i, j \in \mathcal{X}$ has the message can have no effect on $V^{\pi_i^{\mathbf{w}}}$. Hence, if j is a best node in $\mathcal{X}$ at $w-1$, one of the best nodes of $w-1$ must be reached with probability 1. Otherwise, whether or not j has the message at w affects the value function $V^{\pi_i^{\mathbf{w}}}$.

2. We allow idling as a control action because in a time-varying system to idle might in fact be the optimal action. This was not true for Problem **(P)**, due to its time-homogeneous nature. Idling might be optimal if, for example, transition probabilities are improving with time, which could occur due to temporary network congestion. Another possibility is that $R(t)$ is non-monotonic in time, due to, say, congestion at the destination making delayed delivery more desirable. Inclusion of the idling option allows for optimal behavior in these types of situations.

3. Time-varying transition probabilities can be thought of as communication channels varying in time in a known way. For example, there might be kind of model for each channel, where the state determines probability of successful transmission. For the results of this section, we assume that the time evolution of the transition probabilities are entirely known when the optimal routing policy is computed. In Section 7, we briefly mention a model for time-varying channels, a Markov channel model, where future evolution of the channels is described with a stochastic model and imperfect channel state observations. This is a topic for future research.

4. In general, the expected reward for a priority policy can be a different value for each state. This means there are 2^N different values (each corresponding to a different state in the state space) that must be computed at each time. When Algorithm 2 terminates at $w = \tau$, only N

V's have been computed at each time. This reduced complexity is a major benefit in those systems where Condition 1 or Condition 2 is always satisfied. In Section 5.2.2 we consider more general conditions under which we can compute an optimal time-varying priority policy without computing all 2^N possible V's at each time.

5. We conjecture that if we remove Condition 2 from Algorithm 2, require Condition 1 at each step that $R^{\mathcal{X}} < H$, and keep all other steps the same, we create an algorithm which computes the time-invariant index policy if and only if one exists.

We illustrate the result of Theorem 5.1 with the following examples. Example 1 presents a system for which a time-invariant index policy is optimal. Using the same node layout as in Example 1, but with different transmission probabilities, Example 2 presents a system for which a time-varying index policy, but not a time-invariant index policy, is optimal.

Example 1 *Index Policy with Time-Invariant Priority*

Consider the system of Figure 2, where $\Omega = \{1, 2, 3, 4, 5\}$. We assume transition success is independent among all receiving nodes, with probability given by the time-independent values notated on each arc (i.e. P_1, P_2, P_3, P_4, or 1). Assume each node has time-invariant transmission cost $c_i = 1, \forall i \in \Omega$. Rewards are zero over all time, except for that of Q_5. The reward of Q_5 is time-varying, defined as $R_{5,t} = 100, t < \tau, \;\; R_{5,t} = 0, t \geq \tau$. Before specifying particular values, we present some useful relations in terms of general P_1, P_2, P_3, and P_4. We assume that

$$P_4 \geq \frac{1}{100} \tag{79}$$

We define $\pi^{\mathbf{w}}$ for this system as follows. π_1 and π_2 are index policies, with priority lists $\pi_1 = \{r_5, r_4, r_3, r_2, r_1\}$, and $\pi_2 = \{r_5, 4, 3, r_2, r_1\}$. We compute for $w = 1$ that

$$V_i^{\pi^{\mathbf{1}}} = 0, \;\; \forall i \in \Omega, i \neq 5 \tag{80}$$

$$V_5^{\pi^{\mathbf{1}}} = R_{5,\tau-1} = 100 \tag{81}$$

For a set of transition probabilities P_1, P_2, P_3, P_4, and under (79), the time-invariant index policy $\pi^{\mathbf{2}} = (\pi_1, \pi_2)$ for $\tau = 3$ is optimal. To see this, consider that with only one transmission left, nodes 1 and 2 just retire, as node 5 cannot be reached. Node 5 always retires. Node 4 takes

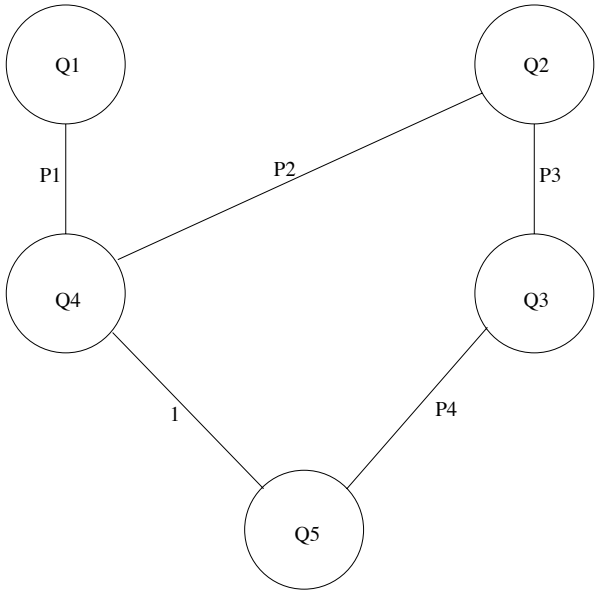

Figure 2: System for Time-Varying Examples

precedence over node 3 because $P_4 \leq 1$. To demonstrate that neither node 3 nor node 4 retires, we compute for $w = 2$ that

$$V_i^{\pi^{\mathbf{2}}} = 0, \quad i \in \{1, 2\} \tag{82}$$

$$V_5^{\pi^{\mathbf{2}}} = R_{5,\tau-2} = 100 \tag{83}$$

For nodes 3 and 4, we use (64) and (83) to get

$$V_4^{\pi^{\mathbf{2}}} = -1 + V_5^{\pi^{\mathbf{1}}} = 99 \tag{84}$$

$$V_3^{\pi^{\mathbf{2}}} = -1 + P_4 \, V_5^{\pi^{\mathbf{1}}}$$

$$= 100P_4 - 1 \tag{85}$$

Under (79) we have using (85) that

$$V_3^{\pi^{\mathbf{2}}} \geq 0 \tag{86}$$

By (84) and (86), transmission for nodes 3 and 4 is optimal. Hence $\pi^{\mathbf{2}}$ is optimal.

Now consider $\tau = 4$, and let $P_1 = P_2 = P_3 = P_4 = 1$. Let $\pi_3 = \{r_5, 4, 3, 2, 1\}$. Then the index policy with time-invariant priority $\pi^{\mathbf{3}} = (\pi_1, \pi_2, \pi_3)$ is optimal.

Example 2 *Time-Varying Index Policy*

We consider the same system as Example 1, but now let $P_1 = P_4 = .1$, $P_2 = .9, P_3 = 1$. Let $\pi_3 = \{r_5, 4, 2, 3, 1\}$. We show that $\pi^{\mathbf{3}}$ is an optimal time-varying index policy.

Using (85) and $P_4 = .1$, we obtain

$$V_3^{\pi^{\mathbf{2}}} = -1 + .1 \cdot 100 = 9 \tag{87}$$

We show that the conditions of Algorithm 2 are satisfied by π_3. When $w = 3$, the algorithm starts with $\mathcal{A} = \{r_5\}$, $\mathcal{X} = \{1, 2, 3, 4\}$, and $V_5^{\pi^{\mathbf{3}}} = R_{5,\tau-3} = 100$.

We list the result of each algorithm iteration.

Iteration 1 $\mathcal{A} = \{r_5\}, \mathcal{X} = \{1, 2, 3, 4\}$

We have

$$H = V^{\pi_4^{\mathbf{3}}}(\{1, 2, 3, 4\}) = -1 + V_5^{\pi^{\mathbf{2}}} = 99 \tag{88}$$

Because $H > R^{\mathcal{X}} = 0$, we use Case 1. Then Condition 1 is satisfied, because $i = 4 = \pi_2(\mathcal{X})$. Hence, 4 is appended to $\mathcal{A}$.

Iteration 2 $\mathcal{A} = \{r_5, 4\}, \mathcal{X} = \{1, 2, 3\}$

Using (87) and (84) we have

$$\begin{aligned} H = V^{\pi_2^{\mathbf{3}}}(\{1, 2, 3\}) &= -1 + .9\, V_4^{\pi^{\mathbf{2}}} + .1\, V_3^{\pi^{\mathbf{2}}} \\ &= -1 + .9 \cdot 99 + .1 \cdot 9 = 89 \end{aligned} \tag{89}$$

Because $H > R^{\mathcal{X}} = 0$, we use Case 1. Condition 1 is not satisfied, because $i = 2 \neq \pi_2(\mathcal{X}) = 3$. But letting $B = \{3\}$, Condition 2 is satisfied. Hence, 2 is appended to $\mathcal{A}$.

Iteration 3 $\mathcal{A} = \{r_5, 4, 2\}, \mathcal{X} = \{1, 3\}$

We have

$$\begin{aligned} H = V^{\pi_3^{\mathbf{3}}}(\{1, 3\}) &= -1 + .1\, V_5^{\pi^{\mathbf{2}}} + .9\, V_3^{\pi^{\mathbf{2}}} \\ &= -1 + .1 \cdot 100 + .9 \cdot 9 = 17.1 \end{aligned} \tag{90}$$

Because $H > R^{\mathcal{X}} = 0$, we use Case 1. Then Condition 1 is satisfied, because $i = 3 = \pi_2(\mathcal{X})$. Hence, 3 is appended to $\mathcal{A}$.

Iteration 4 $\mathcal{A} = \{r_5, 4, 2, 3\}, \mathcal{X} = \{1\}$

We have

$$\begin{aligned} H = V^{\pi_1^{\mathbf{3}}}(\{1\}) &= -1 + .1\, V_4^{\pi^{\mathbf{2}}} + .9\, V_1^{\pi^{\mathbf{2}}} \\ &= -1 + .1 \cdot 100 + .9 \cdot 0 = 9 \end{aligned} \tag{91}$$

Because $H > R^{\mathcal{X}} = 0$, we use Case 1. Then Condition 1 is satisfied, because $i = 1 = \pi_2(\mathcal{X})$. Hence, 1 is appended to $\mathcal{A}$.

At this point, Algorithm 2 terminates at $w = 3$ with $\mathcal{X}$ empty. Hence, from the specification of π_1 and π_2, the fact that $\pi^{\mathbf{2}} = (\pi_1, \pi_2)$ is optimal for $\tau = 3$, and Theorem 5.1, $\pi^{\mathbf{3}}$ is an optimal time-varying index policy. □

5.2.2 Conditions for Time-Varying Priority Policy Optimality

We now determine a condition sufficient to guarantee the optimality of a time-varying priority policy for Problem ($\mathbf{P_4}$). We develop an algorithm that uses this sufficient condition to compute an optimal time-varying priority policy for Problem ($\mathbf{P_4}$).

Definition 5.2 *We write Ω_i^π to indicate the set of nodes of equal or less priority than i in priority policy π.*

Algorithm 3 *In the following discussion, L_i^w and $U_i^w(\mathcal{X})$ are variables specific to this algorithm, updated by (92) and (93), which are used to determine sufficiency conditions below.*

The algorithm is defined inductively on the time-to-go w. When $w = 1$, the optimal policy π_1 is to retire and receive the reward of the node with the largest $R_{i,\tau-1}$ which has received the message. This π_1 is a priority policy. Note that $V^{\pi^{\mathbf{1}}}(\{i\}) = V^{\pi^{\mathbf{1}}}(\Omega_i^{\pi_1}) = R_{i,\tau-1}$, $\forall i \in \Omega$. We define $L_i^1 = U_i^1 := R_{i,\tau-1}$, $\forall i \in \Omega$. This completes the induction basis.

Now assume that for some arbitrary $w - 1 < \tau$ we have defined $\pi^{\mathbf{w-1}}$, an optimal time-varying priority policy, and assume we have determined $V^{\pi^{\mathbf{w-1}}}(\{i\})$ and $V^{\pi^{\mathbf{w-1}}}(\Omega_i^{\pi_{w-1}})$, $\forall i \in \Omega$. We show how priority policy π_w is computed, assuming a certain system condition, specified below, is satisfied at each node computation. If the condition is not satisfied, the algorithm halts; a time-varying priority policy still may or may not exist, as the required condition is only sufficient.

Compute L_i^w for each node $i \in \Omega$ using

$$L_i^w := -c_{i,\tau-w} + \sum_{j \in \Omega} \sum_{S \supseteq \{i\}: \pi_{w-1}(S) = j} P_{\tau-w}^i(S|\{i\}) L_j^{w-1} \tag{92}$$

To compute the policy at w, we start with two sets $\mathcal{A}$ and $\mathcal{X}$ that are defined in precisely the same way as in Algorithm 1.

The algorithm proceeds in three steps.

1. Let $R^{\mathcal{X}} := \max_{i\in\mathcal{X}} R_{i,\tau-w}$. Let $\pi_i^{\mathbf{w}}$ be as in Definition 5.1.

Compute $U_i^w(\mathcal{X})$ for each node $i \in \mathcal{X}$ using

$$U_i^w(\mathcal{X}) := -c_{i,\tau-w} + \sum_{j\in\Omega} \sum_{S\supseteq\mathcal{X}:\pi_{w-1}(S)=j} P^i_{\tau-w}(S|\mathcal{X}) U_j^{w-1}(\Omega_j^{\pi_{w-1}}) \tag{93}$$

Define $L := \max_{i\in\mathcal{X}} L_i^w$.

Define $U := \max_{i\in\mathcal{X}} U_i^w(\mathcal{X})$.

2. Consider three cases.

Case 1: $R^{\mathcal{X}} \leq L$

Define $D = \{i \in \mathcal{X} : L_i^w = L\}$. Suppose there exists $i \in D$ that satisfies Condition 1 stated below. Set $\pi_w(S) = i$ for all $S \subseteq \mathcal{X}$ such that $i \in S$; append i to $\mathcal{A}$ and remove it from $\mathcal{X}$.

If there exists no $i \in D$ that satisfies Condition 1, halt. The algorithm fails to determine an optimal priority policy.

Condition 1 is the following:

Condition 1 $L_i^w \geq \max_{j\in\mathcal{X}-\{i\}} U_j^w(\mathcal{X})$

Case 2: $R^{\mathcal{X}} \geq U$

Set $\pi_w(S) = r_i$, $\forall S \subseteq \mathcal{X}$ such that $i \in S$, where i is any node such that $R^{\mathcal{X}} = R_{i,\tau-w}$; append r_i to $\mathcal{A}$ and remove i from $\mathcal{X}$.

Case 3: $L < R^{\mathcal{X}} < U$

Halt. The algorithm fails to determine an optimal priority policy.

3. If $\mathcal{X}$ is empty, an optimal priority policy $\pi^{\mathbf{w}}$ for time w has been completely specified. Otherwise, go to Step 1.

This completes the induction step for algorithm definition.

We prove in Lemma 5.2 below that the variable L_i^w is a lower bound on the expected reward for transmitting at node i when in state $\{i\}$. Similarly, we show $U_i^w(S)$ is an upper bound on the expected reward for transmitting at node i when in state $S \subseteq \Omega_i^{\pi_w}$.

We may interpret the variables L_i^w and $U_i^w(S)$ as follows. The expression for L_i^w in (92) uses the previously computed lower bounds L_j^{w-1} and computes the expected reward when transmitting at node i, and assuming that no other node has the message. From a monotonicity property of the value function, which we develop in Lemma 5.1 below, the case when no node other than i has the message is, in value function terms, a worst case. Thus, L_i^w is a lower bound on the value function of state $\{i\}$ at w.

Similarly, the expression for $U_i^w(\mathcal{X})$ in (93) uses the previously computed upper bounds, at their maximum value $U_j^{w-1}(\Omega_j^{\pi_{w-1}})$ for each node j, and computes the expected reward when transmitting at node i, assuming that all other nodes of $\mathcal{X}$ have the message. As before, from the monotonicity property of the value function, the case when all other nodes in $\mathcal{X}$ have the message is, in value function terms, a best case. Thus, $U_i^w(\mathcal{X})$ is an upper bound on the value function of state $\mathcal{X}$ at w.

We may now interpret Condition 1 of Algorithm 3 as follows. Condition 1 is true when there is a node $i \in \mathcal{X}$ such that the value function of state $\{i\}$, lower bounded by L_i^w, is larger than the upper bound on the value function of state $\mathcal{X}$ when a node other than i transmits. When Condition 1 is true, then transmission at i is optimal for all subsets of state $\mathcal{X}$. Hence, node i takes priority over all other nodes of $\mathcal{X}$.

We make these ideas concrete in the proof of Theorem 5.2 below. First, we present a series of lemmas needed for the proof of Theorem 5.2. We begin with Lemma 5.1.

Lemma 5.1 *Let $\pi^{\mathbf{w}}$ be an optimal policy for Problem $(\mathbf{P_4})$. Then*

$$V^{\pi^{\mathbf{w}}}(S_1) \geq V^{\pi^{\mathbf{w}}}(S_2) \qquad \forall S_2 \subseteq S_1 \subseteq \Omega \tag{94}$$

*Proof.*We use induction on w. Equation (94) clearly holds for $w = 1$. Now assume (94) holds for $w-1$. Let $S_1, S_2 \subseteq \Omega$ be such that $S_2 \subseteq S_1$. Let $i = \pi_w(S_2)$. We claim that

$$V^{\pi^{\mathbf{w}}}(S_1) \geq V^{\pi_i^{\mathbf{w}}}(S_1) \geq V^{\pi^{\mathbf{w}}}(S_2) \tag{95}$$

The first inequality in (95) follows from the optimality of $\pi^{\mathbf{w}}$. The second inequality follows from the decoupling and increasing properties, and the induction hypothesis.

Since (95) is true for any such S_1 and S_2, this completes the induction step. □

Corollary 5.1 *Let $\pi^{\mathbf{w-1}}$ be an optimal policy for Problem $(\mathbf{P_4})$. Let $\pi_i^{\mathbf{w}}$ be as in Definition 5.1. Then*

$$V^{\pi_i^{\mathbf{w}}}(S_1) \geq V^{\pi_i^{\mathbf{w}}}(S_2) \qquad \forall S_2 \subseteq S_1 \subseteq \Omega \tag{96}$$

*Proof.*Inequality (96) follows immediately from the decoupling and increasing properties and Lemma 5.1. □

Lemma 5.2 *If Algorithm 3 halts with $w = \tau$, then $\forall w, 1 \leq w < \tau$ and $\forall i \in \Omega$ we have*

$$\begin{aligned} V^{\pi_i^{\mathbf{w}}}(\{i\}) &\geq L_i^w & (97)\\ V^{\pi_k^{\mathbf{w}}}(\Omega_i^{\pi_w}) &\leq U_k^w(\Omega_i^{\pi_w}), \quad \forall k \in \Omega_i^{\pi_w} & (98) \end{aligned}$$

*Proof.*We proceed by induction. When $w = 1$, we have $V^{\pi^{\mathbf{1}}}(\{i\}) = V^{\pi^{\mathbf{1}}}(\Omega_i^{\pi_1}) = R_{i,\tau-1}$, $\forall i \in \Omega$ and $L_i^1 = U_i^1 := R_{i,\tau-1}$, $\forall i \in \Omega$. Hence, (97) and (98) follow.

Now assume the lemma is true for $w-1$, so that $\forall i \in \Omega$

$$\begin{aligned} V^{\pi_i^{\mathbf{w-1}}}(\{i\}) &\geq L_i^{w-1} & (99)\\ V^{\pi_k^{\mathbf{w-1}}}(\Omega_i^{\pi_{w-1}}) &\leq U_k^{w-1}(\Omega_i^{\pi_{w-1}}), \quad \forall k \in \Omega_i^{\pi_{w-1}} & (100) \end{aligned}$$

Consider $i \in \Omega$. Then we have

$$\begin{aligned} V^{\pi_i^{\mathbf{w}}}(\{i\}) &= -c_{i,\tau-w} + \sum_{S \supseteq \{i\}} P^i_{\tau-w}(S|\{i\}) V^{\pi^{\mathbf{w-1}}}(S) & (101)\\ &= -c_{i,\tau-w} + \sum_{j \in \Omega} \sum_{S \supseteq \{i\}: \pi_{w-1}(S)=j} P^i_{\tau-w}(S|\{i\}) V^{\pi^{\mathbf{w-1}}}(S) & (102)\\ &\geq -c_{i,\tau-w} + \sum_{j \in \Omega} \sum_{S \supseteq \{i\}: \pi_{w-1}(S)=j} P^i_{\tau-w}(S|\{i\}) V^{\pi^{\mathbf{w-1}}}(\{j\}) & (103)\\ &\geq -c_{i,\tau-w} + \sum_{j \in \Omega} \sum_{S \supseteq \{i\}: \pi_{w-1}(S)=j} P^i_{\tau-w}(S|\{i\}) L_j^{w-1} & (104)\\ &= L_i^w & (105) \end{aligned}$$

Equation (101) is a special case of (64). In (102) the sum is written in a different way. Application of Lemma 5.1 then leads to (103). Inequality (104) follows from (99) (the induction hypothesis). And (105) is direct from (92).

We also have $\forall k \in \Omega_i^{\pi_w}$

$$
\begin{aligned}
V^{\pi_k^{\mathbf{w}}}(\Omega_i^{\pi_w}) = & -c_{k,\tau-w} + \sum_{S \supseteq \Omega_i^{\pi_w}} P_{\tau-w}^k(S|\Omega_i^{\pi_w}) V^{\pi^{\mathbf{w-1}}}(S) && (106)\\
= & -c_{k,\tau-w} + \sum_{j \in \Omega} \sum_{S \supseteq \Omega_i^{\pi_w} : \pi_{w-1}(S)=j} P_{\tau-w}^k(S|\Omega_i^{\pi_w}) V^{\pi^{\mathbf{w-1}}}(S) && (107)\\
\leq & -c_{k,\tau-w} + \sum_{j \in \Omega} \sum_{S \supseteq \Omega_i^{\pi_w} : \pi_{w-1}(S)=j} P_{\tau-w}^k(S|\Omega_i^{\pi_w}) V^{\pi^{\mathbf{w-1}}}(\Omega_j^{\pi_{w-1}}) && (108)\\
\leq & -c_{k,\tau-w} + \sum_{j \in \Omega} \sum_{S \supseteq \Omega_i^{\pi_w} : \pi_{w-1}(S)=j} P_{\tau-w}^k(S|\Omega_i^{\pi_w}) U_j^{w-1}(\Omega_j^{\pi_{w-1}}) && (109)\\
= & U_k^w(\Omega_i^{\pi_w}) && (110)
\end{aligned}
$$

Equation (106) is a special case of (64). In (107) the sum is written in a different way. Application of Lemma 5.1 then leads to (108), because $\pi_{w-1}(S) = j$ means that $S \subseteq \Omega_j^{\pi_{w-1}}$. Inequality (109) follows from (100) (the induction hypothesis), with $i, k = j$. And (110) is direct from (93).

Inequalities (105) and (110) complete the induction step. □

The following theorem summarizes the main result about how Algorithm 3 relates to Problem $(\mathbf{P_4})$.

Theorem 5.2 *If Algorithm 3 halts with $w = \tau$, then the time-varying priority policy $\pi^{\mathbf{w}}$ which Algorithm 3 has determined is optimal for Problem* $(\mathbf{P_4})$.

*Proof.*By construction $\pi^{\mathbf{w}}$ is a priority policy. We show that $\pi^{\mathbf{w}}$ is an optimal priority policy. Assume Algorithm 3 halts with $w = \tau$.

Now let w be any value $1 \leq w < \tau$. Consider any node $i \in \Omega$. Consider two cases.

Case 1: $\pi_w(S) = i, \ \forall S \subseteq \Omega_i^{\pi_w}$

Because π_w transmits at node i, it must be that Case 1 in Step 2. was chosen in Algorithm 3 for i. Hence

$$V^{\pi^{\mathbf{w}}}(S) \geq R(S), \quad \forall S \subseteq \Omega_i^{\pi_w} \tag{111}$$

We also have $\forall S \subseteq \Omega_i^{\pi_w}$

$$
\begin{aligned}
V^{\pi^{\mathbf{w}}}(S) = V^{\pi_i^{\mathbf{w}}}(S) \ & \geq \ V^{\pi_i^{\mathbf{w}}}(\{i\}) && (112)\\
& \geq \ L_i^w && (113)
\end{aligned}
$$

$$\geq U_k^w(\Omega_i^{\pi_w}) \qquad \forall k \in S, k \neq i \tag{114}$$

$$\geq V^{\pi_k^{\mathbf{w}}}(\Omega_i^{\pi_w}) \tag{115}$$

$$\geq V^{\pi_k^{\mathbf{w}}}(S) \tag{116}$$

Inequality (112) follows from Corollary 5.1. From Lemma 5.2 (97) we obtain inequality (113). Inequality (114) follows because of Condition 1 in Algorithm 3, which is satisfied for all $\Omega_i^{\pi_w}$ when i is added to the $\mathcal{A}$ list for transmission. Inequality (115) follows from Lemma 5.2 (98). Finally, inequality (116) follows from Corollary 5.1.

Inequalities (111) and (116) show that for Case 1, choosing i for transmission is optimal when in state $S, \forall S \subseteq \Omega_i^{\pi_w}$.

Case 2: $\pi_w(S) = r_i, \ \ \forall S \subseteq \Omega_i^{\pi_w}$

Because π_w retires, it must be that Case 2 in Step 2. was chosen in Algorithm 3 for i. Hence we have $\forall S \subseteq \Omega_i^{\pi_w}$

$$R_i \geq U_k^w(\Omega_i^{\pi_w}) \qquad \forall k \in S \tag{117}$$

$$\geq V^{\pi_k^{\mathbf{w}}}(\Omega_i^{\pi_w}) \tag{118}$$

$$\geq V^{\pi_k^{\mathbf{w}}}(S) \tag{119}$$

Inequality (117) follows because of Case 2 in Algorithm 3, satisfied when i is added to the $\mathcal{A}$ list for retiring. As before, inequality (118) follows from Lemma 5.2 (98). Finally, inequality (119) follows from Corollary 5.1.

In both Case 1 and Case 2, π_w produces the optimal action $\forall S \subseteq \Omega_i^{\pi_w}$. Because both $i \in \Omega$ and w are arbitrary, $\pi^{\mathbf{w}}$ is an optimal priority policy $\forall 1 \leq w < \tau$. □

Discussion

In general, a priority policy can have 2^N different values (each corresponding to a different state in the state space) at each time. The complexity to compute all of these value functions is high. Instead, we have used monotonicity to retain upper and lower bounds for V at each node. This has allowed us to determine sufficient conditions to guarantee the optimality of a time-varying priority policy, without explicitly computing the value function for each of the 2^N states.

Algorithm 3 has some additional interesting properties which are presented below.

Lemma 5.3 *If $\pi^{\mathbf{w}}$ is an optimal time-varying index policy for Problem* $(\mathbf{P_4})$*, then in Algorithm 3 we have*

$$L_i^w = U_i^w(\Omega_i^{\pi_w}) = V_i^{\pi^{\mathbf{w}}}, \quad \forall i \in \Omega \tag{120}$$

*Proof.*We proceed by induction. For the induction base step, note that (120) is true when $w = 1$, by the definition of L_i^1 and U_i^1 in Algorithm 3.

Now assume (120) is true for $w - 1$. We have

$$\begin{aligned}
L_i^w &= -c_{i,\tau-w} + \sum_{j\in\Omega} \sum_{S\supseteq\{i\}:\pi_{w-1}(S)=j} P_{\tau-w}^i(S|\{i\}) L_j^{w-1} && (121)\\
&= -c_{i,\tau-w} + \sum_{j\in\Omega} \sum_{S\supseteq\{i\}:\pi_{w-1}(S)=j} P_{\tau-w}^i(S|\{i\}) V_j^{\pi^{\mathbf{w-1}}} && (122)\\
&= -c_{i,\tau-w} + \sum_{S\supseteq\{i\}} P_{\tau-w}^i(S|\{i\}) V^{\pi^{\mathbf{w-1}}}(S) && (123)\\
&= V_i^{\pi^{\mathbf{w}}} && (124)
\end{aligned}$$

Equation (121) is a restatement of (92). Equation (122) follows from the induction hypothesis. Equation (123) follows from the decoupling property and the fact that $\pi^{\mathbf{w}}$ is an optimal time-varying index policy. And (124) is the dynamic programming equation for $V_i^{\pi^{\mathbf{w}}}$.

Also, we have

$$\begin{aligned}
U_i^w(\Omega_i^{\pi_w}) &= -c_{i,\tau-w} + \sum_{j\in\Omega} \sum_{S\supseteq\Omega_i^{\pi_w}:\pi_{w-1}(S)=j} P_{\tau-w}^i(S|\Omega_i^{\pi_w}) U_j^{w-1}(\Omega_j^{\pi_{w-1}}) && (125)\\
&= -c_{i,\tau-w} + \sum_{j\in\Omega} \sum_{S\supseteq\Omega_i^{\pi_w}:\pi_{w-1}(S)=j} P_{\tau-w}^i(S|\Omega_i^{\pi_w}) V^{\pi^{\mathbf{w-1}}}(\Omega_j^{\pi_{w-1}}) && (126)\\
&= -c_{i,\tau-w} + \sum_{S\supseteq\{i\}} P_{\tau-w}^i(S|\{i\}) V^{\pi^{\mathbf{w-1}}}(S) && (127)\\
&= V_i^{\pi^{\mathbf{w}}} && (128)
\end{aligned}$$

Equation (125) is a restatement of (93). Equation (126) follows from the induction hypothesis. Equation (127) follows from the decoupling property and the fact that $\pi^{\mathbf{w}}$ is an optimal time-varying index policy. And (128) is the dynamic programming equation for $V_i^{\pi^{\mathbf{w}}}$.

Equations (124) and (128) together prove the lemma. □

Corollary 5.2 *If there exists an optimal time-varying index policy for Problem* $(\mathbf{P_4})$*, then Algorithm 3 will determine an optimal time-varying priority policy, which will in fact be an optimal time-varying index policy.*

*Proof.*This follows directly from Lemma 5.3 and the conditions of Algorithm 3. □

Corollary 5.2 shows that Algorithm 3 finds an optimal time-varying index policy when one exists, just as Algorithm 2 does. Of course, Algorithm 3 achieves this with substantially more computational burden. However, there are instances where Algorithm 3 finds an optimal time-varying priority policy, for which Algorithm 2 only terminates indicating failure to find an optimal time-varying index policy. We present such an instance via the following example.

Example 3 *Time-Varying Priority Policy*

We consider the system of Example 1, and again define $\pi^{\mathbf{w}}$ for this system as $\pi_1 = \{r_5, r_4, r_3, r_2, r_1\}$, and $\pi_2 = \{r_5, 4, 3, r_2, r_1\}$.

Let $P_1 = P_4 = .1$, $P_2 = P_3 = .9$, $\pi_3 = \{r_5, 4, 2, 3, 1\}$. We show that $\pi^{\mathbf{3}}$ is an optimal time-varying priority policy, but not a time-varying index policy.

First, we show that the conditions of Algorithm 2 are not satisfied by π_3. As before, when $w = 3$, the algorithm starts with $\mathcal{A} = \{r_5\}$, $\mathcal{X} = \{1, 2, 3, 4\}$, and $V_5^{\pi^{\mathbf{3}}} = R_{5,\tau-3} = 100$.

Iteration 1 $\mathcal{A} = \{r_5\}, \mathcal{X} = \{1, 2, 3, 4\}$

This iteration is the same as for Example 2, again giving (88).

Iteration 2 $\mathcal{A} = \{r_5, 4\}, \mathcal{X} = \{1, 2, 3\}$

The H computation is precisely the same as that of (89). Because $H > R^{\mathcal{X}} = 0$, we use Case 1. Condition 1 is not satisfied, because $i = 2 \neq \pi_2(\mathcal{X}) = 3$. Also, there is no B satisfying Condition 2. The difference between Example 2 and Example 3 is that the probability of reaching 3 from 2 is no longer 1, so whether or not 3 has the message matters when transmitting at 2 when $w = 3$.

We have shown that the conditions of Algorithm 2 are not satisfied in Example 3. Hence by Theorem 5.1 no optimal time-varying index policy exists for Example 2.

We now use Algorithm 3 to show that an optimal time-varying priority policy exists for Example 3. We show that the conditions of Algorithm 3 are satisfied by π_3. When $w = 3$, the algorithm starts with $\mathcal{A} = \{r_5\}$, $\mathcal{X} = \{1, 2, 3, 4\}$, and $V_5^{\pi^{\mathbf{3}}} = R_{5,\tau-3} = 100$.

By Lemma 5.3 and the fact that $\pi^{\mathbf{2}}$ is an optimal time-varying index policy, we have

$$L_i^2 = U_i^2(\Omega_i^{\pi_2}) = V_i^{\pi^{\mathbf{2}}}, \quad \forall i \in \Omega \tag{129}$$

We list the result of each algorithm iteration.

Iteration 1 $\mathcal{A} = \{r_5\}, \mathcal{X} = \{1,2,3,4\}$

We have

$$L = U = L_4^3 = U_4^3(\mathcal{X}) = V^{\pi_4^{\mathbf{3}}}(\{1,2,3,4\}) = -1 + V_5^{\pi^{\mathbf{2}}} = 99 \tag{130}$$

Because $L > R^{\mathcal{X}} = 0$, we use Case 1. Then Condition 1 is satisfied, because $L = U$. Hence, 4 is appended to $\mathcal{A}$.

Iteration 2 $\mathcal{A} = \{r_5, 4\}, \mathcal{X} = \{1,2,3\}$

$$\begin{aligned}
L = L_2^3 &= -1 + \sum_{j\in\Omega} \sum_{S\supseteq\{2\}:\pi_2(S)=j} P^2(S|\{2\}) L_j^2 && (131)\\
&= -1 + \sum_{j\in\Omega} \sum_{S\supseteq\{2\}:\pi_2(S)=j} P^2(S|\{2\}) V_j^{\pi^{\mathbf{2}}} && (132)\\
&= -1 + .9\, V_4^{\pi^{\mathbf{2}}} + (1-.9).9\, V_3^{\pi^{\mathbf{2}}} + 1 - (1-.9).9\, V_2^{\pi^{\mathbf{2}}} && (133)\\
&= -1 + .9 \cdot 99 + (1-.9).9 \cdot 9 + 1 - (1-.9).9 \cdot 0 = 88.91 && (134)
\end{aligned}$$

Equation (131) is from (92). Equation (132) follows from (129). Equation (133) follows from the specification of transition probabilities in Example 3, and (134) follows from (82), (87), and (84).

$$\begin{aligned}
\max_{j\in\{1,3\}} U_j^3(\{1,2,3\}) = U_3^3(\{1,2,3\}) &= U_3^3(\{3\})\\
&= -1 + \sum_{j\in\Omega} \sum_{S\supseteq\{3\}:\pi_2(S)=j} P^3(S|\{3\}) U_j^2(\Omega_j^2) && (135)\\
&= -1 + \sum_{j\in\Omega} \sum_{S\supseteq\{3\}:\pi_2(S)=j} P^3(S|\{3\}) V_j^{\pi^{\mathbf{2}}} && (136)\\
&= -1 + .1\, V_5^{\pi^{\mathbf{2}}} + .9\, V_3 \pi^{\mathbf{2}} && (137)\\
&= -1 + .1 \cdot 100 + .9 \cdot 9 = 17.1 && (138)
\end{aligned}$$

Equation (135) is from (93). Equation (136) follows from (129). Equation (137) follows from the specifications of transition probabilities in Example 3, and (138) follows from (83) and (87).

Because $L > R^{\mathcal{X}} = 0$, we are in Case 1. From (134) and (138), we obtain

$$L > \max_{j \in \{1,3\}} U_j^3(\{1,2,3\}) = U_3^3(\{1,2,3\}) \tag{139}$$

Because of (139), Condition 1 is satisfied. Hence, 2 is appended to $\mathcal{A}$.

Iteration 3 $\mathcal{A} = \{r_5, 4, 2\}, \mathcal{X} = \{1, 3\}$

We have

$$\begin{aligned} L = U = V^{\pi_3^{\mathbf{3}}}(\{1,3\}) &= -1 + .1\, V_5^{\pi^{\mathbf{2}}} + .9\, V_3^{\pi^{\mathbf{2}}} \\ &= -1 + .1 \cdot 100 + .9 \cdot 9 = 17.1 \end{aligned} \tag{140}$$

Because $L > R^{\mathcal{X}} = 0$, we are in Case 1. Condition 1 is satisfied, because $L = U$. Hence, 3 is appended to $\mathcal{A}$.

Iteration 4 $\mathcal{A} = \{r_5, 4, 2, 3\}, \mathcal{X} = \{1\}$

We have

$$\begin{aligned} L = U = V^{\pi_1^{\mathbf{3}}}(\{1\}) &= -1 + .1\, V_4^{\pi^{\mathbf{2}}} + .9\, V_1^{\pi^{\mathbf{2}}} \\ &= -1 + .1 \cdot 100 + .9 \cdot 0 = 9 \end{aligned} \tag{141}$$

Because $L > R^{\mathcal{X}} = 0$, we are in Case 1. Condition 1 is satisfied, because $L = U$. Hence, 3 is appended to $\mathcal{A}$.

At this point, Algorithm 3 terminates with $\mathcal{X}$ empty. Hence, by Theorem 5.2 $\pi^{\mathbf{3}}$ is an optimal time-varying priority policy. □

5.2.3 A System With No Optimal Time-Varying Priority Policy

We provide an example of a system which does not satisfy the conditions of either Algorithm 2 or Algorithm 3, and, in fact, cannot be optimally solved with a policy classified as a time-varying index or priority policy.

Example 4 *Unclassified Policy*

We consider the system of Example 1, and let $P_1 = .9, P_2 = .89, P_3 = 1, P_4 = .5$. Let $\pi_3(\{1,2,3\}) = 1$. But $\pi_3(\{1,2\}) = 2$ is optimal. For this problem no time-varying priority policy is optimal.

The problem in Example 4 is that Q_4 is having an effect on whether to use Q_1 or Q_2 to transmit at $w = 3$. This phenomenon, that a possibly distant node can affect the relative priority of two neighboring nodes, is the essential difficulty with the general time-varying problem, and reveals why the simple time-varying index or priority policy is not always optimal. Similar situations can also be created with time-varying transition probabilities or node transmission costs.

The results we have presented in Theorem 5.1 and Theorem 5.2 do not specify how to determine an optimal policy for Example 4. It is a matter for further investigation to decide how best to handle cases like Example 4.

6 Distributed Algorithms for Problem ($\mathbf{P_1}$)

6.1 Definitions and Notation

A general Markov policy can be written

$$\{\pi_1 \pi_2 \dots \pi_t \dots\} \tag{142}$$

where the subscript of π indexes time. Since Problem ($\mathbf{P_1}$) is a time-homogeneous Markov decision problem, we know there exists an optimal stationary policy of the form

$$\{\pi\pi \dots \pi \dots\} \tag{143}$$

Further, from Theorem 3.1 we know that there exists a stationary optimal policy of the form (143) where π is an index policy.

When describing distributed algorithms to compute an optimal policy for Problem ($\mathbf{P_1}$), we need to consider a more general type of policy.

Definition 6.1 *A* local index ranking *or* local index policy *for node i at t is written π_t^i, and defines a node ordering of $\mathcal{N}(i)$, $i \in \Omega$, with retirement for node i indicated by r_i if desired. A* distributed index policy at t *is written* $\mathbf{\Pi}_t = \{\pi_t^1 \pi_t^2 \dots \pi_t^N\}$, *where π_t^i is a local index policy for node i. A* distributed index policy *is written* $\mathbf{\Pi} = \{\mathbf{\Pi}_1 \mathbf{\Pi}_2 \dots \mathbf{\Pi}_t \dots\}$.

A distributed index policy functions by transmitting at the current node, say i, at t, then using the node ordering π_t^i to choose the next node for transmission.

For local index policy π_t^i, we write $j >_{\pi_t^i} k$ when j has higher priority than k under π_t^i, $j, k \in \mathcal{N}(i)$ (cf. Definition 3.5).

Definition 6.2 *Consider nodes i and j, local index policies π_t^i and π_t^j, and let $k, l \in \mathcal{N}(i) \cap \mathcal{N}(j)$. If either 1) $k >_{\pi_t^i} l$ and $k >_{\pi_t^j} l$ or 2) $k <_{\pi_t^i} l$ and $k <_{\pi_t^j} l$, then we say π_t^i and π_t^j* match *on k and l.*

Definition 6.3 *If π_t^i and π_t^j match on k and l $\forall i, j \in \Omega, \forall k, l \in \mathcal{N}(i) \cap \mathcal{N}(j)$, we say distributed index policy $\mathbf{\Pi}_t$ is* uniform *at t.*

When $\mathbf{\Pi}_t$ is uniform at t, a global order of the nodes is induced. We call the index policy which has this global node ordering the associated index policy *of $\mathbf{\Pi}_t$.*

6.2 A Distributed Algorithm for Problem ($\mathbf{P_1}$)

We present an algorithm (Algorithm 4 described below) which computes the optimal solution for Problem ($\mathbf{P_1}$), and has the characteristic that computations at each node use only information directly from neighbor nodes. This property is critical to the distributed implementation of an optimal policy in an ad hoc wireless network. We claim convergence of the algorithm to the optimal node ordering and value function under the following constraints.

1. Each node i keeps a current estimate, denoted by V_i, of its own optimal expected reward value. Each node i also stores the most recently received estimate of each of its neighbors' optimal expected reward values, denoted $V_{i,j}$, where $j \in \mathcal{N}(i)$.
2. Information transfer among neighboring nodes consists only of the current V_i value of the transmitting node.
3. Each node's V_i information is transmitted asynchronously.
4. A node's V_i update, defined below, is also asynchronous.
5. It is assumed that each node has knowledge of its own $P(S|i)$ update structure. For example, i estimates $\hat{P}(S|i)$ based on all its communications, both control signals and messages, as well as its channel measurement.
6. The energy required to run the algorithm is not included in finding the optimal solution for Problem ($\mathbf{P_1}$).

The algorithm is as follows.

Algorithm 4 *At each event time, any number of the following two events can occur.*

Event 1 *A node i receives V_j from a neighbor j, $j \in \mathcal{N}(i)$.*

Event 2 *A node i recomputes V_i using the current $V_{i,j}$ values, as follows.*

$$V_i^n = \max\left\{\max_{\tilde{\pi}}\left\{\frac{-c_i + \sum_{\mathcal{N}(i)\supseteq S\supset\{i\}:\tilde{\pi}(S)\neq i} P^i(S|i)V_{i,\tilde{\pi}(S)}^n}{\sum_{\mathcal{N}(i)\supseteq S\supset\{i\}:\tilde{\pi}(S)\neq i} P^i(S|i)}\right\}, R_i\right\} \tag{144}$$

The maximization in (144) is over all local priority orderings $\tilde{\pi}$ of i and its neighbors.

It is assumed that events 1 and 2 occur infinitely often at each node i.

Note that we require no *a priori* time ordering on the above events, nor on the nodes where they are occurring. At times when neither of the above events is taking place, the system is in a frozen state, with all system parameters remaining unchanged. An event which occurs at some event time can have no effect on other events at the same time. Hence, we can choose an arbitrary order for all events occurring at a given time without affecting the outcome. In this way, we can talk sensibly about the n'th event in the system since the start. We use this convention hereafter.

The local policy which optimizes (144) for node i at event n is a local index policy at i, $\pi_{l_n}^i$. For convenience we will notate this as π_n^i, where the context prevents ambiguity. The distributed index policy after event n will be denoted by $\mathbf{\Pi}_n$.

Let V_i^n be the expected reward for node i just after event n in the above system, so that V_i^0 is this value at the start of the algorithm, where the allowed range is $0 \leq V_i^0 \leq R_{max}$. Let $V_{i,j}^n$ be the value after event n of the last transmitted V_j^m received at i, where $m < n$ is the event index of this transmission. At the start, the $V_{i,j}^0$ values do not need to match the neighbor's V_j^0 values, we only require that $0 \leq V_{i,j}^0 \leq R_{max}$.

The computation of (144) is based on (49). Given the nature of this update equation, finding the maximum in (144) is easier than it might first appear.

A Method for Finding the Maximum Over $\tilde{\pi}$ in (144)

Method 1 *Let event n be i's computation of V_i^n. Our goal is to compute*

$$H := \max_{\tilde{\pi}}\left\{\frac{-c_i + \sum_{\mathcal{N}(i)\supseteq S\supset\{i\}:\tilde{\pi}(S)\neq i} P^i(S|i)V_{i,\tilde{\pi}(S)}^n}{\sum_{\mathcal{N}(i)\supseteq S\supset\{i\}:\tilde{\pi}(S)\neq i} P^i(S|i)}\right\} \tag{145}$$

When i has no neighbors, we define $H = -\infty$, and we have $V_i^n = R_i$. If node i has at least one neighbor, proceed as follows.

1. *Rank order the values $V_{i,j}^n$ that i has most recently received from its neighbors.*

Let $\xi_1, \xi_2, \ldots, \xi_l$ indicate the subscripts of i's neighbors in rank order, where l is the number of i's neighbors. That is, ξ_j is the subscript of the j'th best neighbor of i according to the ranking of $V^n_{i,j}$. Ties are decided arbitrarily. Denote by k the rank of node i w.r.t. its neighbors. Denote by $\tilde{\pi}_k$ the priority policy which ranks the k highest neighbors of node i above i. Initially set $k = 2$. (k is a dummy variable in this computation).

2. *Compute*
$$W = \frac{-c_i + \sum_{S \subseteq \mathcal{N}(i): \tilde{\pi}_k(S) \neq i} P(S|i) V^n_{i,\tilde{\pi}_k(S)}}{\sum_{S \subseteq \mathcal{N}(i): \tilde{\pi}_k(S) \neq i} P(S|i)} \tag{146}$$

3. *If $V^n_{i,\xi_{k-1}} > W \geq V^n_{i,\xi_k}$, then set $H = W$ and halt (we have found the rank of node i w.r.t. its neighbors).*

4. *If $k \leq l$, modify $\tilde{\pi}_k$ by moving neighbor ξ_k to higher priority than i, leaving all else in $\tilde{\pi}_k$ unchanged. Set $k = k + 1$, and go to 2.*

5. *If $k > l$, then node i is the worst node of all the neighbors. Set $H = W$ and halt.*

The process continues until either the condition of step 3. or step 5. is satisfied. Termination is guaranteed, since the condition of step 5. must eventually be satisfied.

Note that the denominator in (146) is not zero as long as i has at least one neighbor.

Lemma 6.1 *Method 1 finds a local policy $\tilde{\pi}$ which maximizes (145).*

*Proof.*Method 1 can be viewed as an implementation of Algorithm 1 for $\Omega = \mathcal{N}(i)$ where $R_i = -\infty$, and where for all nodes $j, k \in \mathcal{N}(i), j \neq i$, we have

1. $P_{jk} = 0$
2. $R_j = V^n_{i,j}$

Under these conditions, each V computation (given by (144)) for node $j \neq i$ gives $V^n_{i,j}$, and Algorithm 1 reduces to Method 1. Since Algorithm 1 finds the optimal policy for all nodes, this policy maximizes (145). □

Comments

1. Using Method 1 to compute (145) is in the worst case $O(l^2)$ whereas an exhaustive search over all local policies $\tilde{\pi}$ (i.e. local node rankings) would be $O(l!)$.

2. In a practical implementation, there may be ways to speed up Method 1 by changing the order in which the neighbor nodes are put into or removed from the list of better nodes. For example, we might want to always try the order which worked at the last update, and then add or remove nodes sequentially from there.

We proceed with the analysis of Algorithm 4. The following theorem is our main result concerning Algorithm 4, summarizing its convergence properties.

Theorem 6.1 *For Algorithm 4 with any initial state s.t.* $0 \leq V_i^0 \leq R_{max}$ *and* $0 \leq V_{i,j}^0 \leq R_{max}$ *for all* $i, j \in \Omega$*, we have*

1.

$$\lim_{n\to\infty} V_i^n = V_i^\pi, \ \forall i \in \Omega \tag{147}$$

2. There exists an event $n_p < \infty$ *s.t.* $\mathbf{\Pi} = \{\mathbf{\Pi}_{n_p} \ \mathbf{\Pi}_{n_p+1} \ \mathbf{\Pi}_{n_p+2} \ldots\}$ *is an optimal distributed policy*

3. If

$$V_i^\pi \neq V_j^\pi, \forall i \in \Omega, j \in \mathcal{N}(i) \tag{148}$$

then there exists an event $n_p < \infty$ *s.t.*

(a) $V_i^n = V_i^\pi, \forall n \geq n_p, \forall i \in \Omega$

(b) $\mathbf{\Pi} = \{\mathbf{\Pi}_{n_p} \ \mathbf{\Pi}_{n_p+1} \ \mathbf{\Pi}_{n_p+2} \ldots\}$ *has an associated index policy that is optimal*

Before proving Theorem 6.1, we present a series of lemmas which we make use of in the proof. We begin with a simple proof of an important property of the node update procedure which is fundamental for the analysis.

Lemma 6.2 (Update Monotonicity) *Consider two cases of a node recomputation for* i *at event* n *using (144). In case 1, the neighbor values are* $\tilde{V}_{i,j}^n$ *and the updated node value is* $\tilde{V}_i^n$*. In case 2 the neighbor values are* $\hat{V}_{i,j}^n$ *and the updated node value is* $\hat{V}_i^n$*. Assume* $\tilde{V}_{i,j}^n \geq \hat{V}_{i,j}^n, \forall j \in \mathcal{N}(i)$*. Then*

$$\tilde{V}_i^n \geq \hat{V}_i^n \tag{149}$$

*Proof.*For a given $\tilde{\pi}$, (146) is monotonic in each $V_{i,j}^n$. Let $\tilde{\pi}$ be the policy chosen in (145) for case 1, and let $\hat{\pi}$ be the policy chosen in (145) for case 2. Let $\bar{H}$ be the value in (146) when case 2's

policy $\hat{\pi}$ is used with case 1's values $\tilde{V}_{i,j}^n$. Then

$$\hat{H} \leq \bar{H} \leq \tilde{H} \tag{150}$$

where the final inequality follows because $\tilde{\pi}$ is optimal in (145).

Equation (144) and inequality (150) imply (149). □

We say that the *monotonicity property* holds for a node update due to the result of Lemma 6.2.

We now define two random sequences, $\underline{D}_i^n$ and $\overline{D}_i^n$, which we use to provide bounds on node updates. Noting that at the start of Algorithm 4 each node has a V value between 0 and R_{max}, and aiming at a kind of worst case initial state in light of the monotonicity property just demonstrated, we define $\underline{D}_i^n$ and $\overline{D}_i^n$ as follows.

Definition 6.4 *We define* $\forall i \in \Omega$ *such that* $R_i \neq R_{max}$

$$\begin{aligned} \underline{D}_i^n &:= \textit{Computed value for } i \textit{ after event } n \textit{ when } V_i^0 = V_{i,j}^0 = 0 \\ \overline{D}_i^n &:= \textit{Computed value for } i \textit{ after event } n \textit{ when } V_i^0 = V_{i,j}^0 = R_{max} \end{aligned}$$

For i *such that* $R_i = R_{max}$, *define*

$$\underline{D}_i^n = \overline{D}_i^n := R_{max}, \forall n$$

This last definition results from the fact that a node which can retire and receive R_{max} should always do so.

In the following, we assume π is an optimal policy, and that V_i^π is the optimal value function for i.

In the next lemma, we demonstrate that $\underline{D}_i^n$ is a monotonically non-decreasing sequence which lower bounds V_i^n, and in turn is upper bounded by V_i^π. Similarly, we demonstrate that $\overline{D}_i^n$ is a monotonically non-increasing sequence which upper bounds V_i^n, and in turn is lower bounded by V_i^π. Once these facts are proved, it will remain to show that $\underline{D}_i^n$ converges to V_i^π from below, and that $\overline{D}_i^n$ converges to V_i^π from above, to obtain covergence of V_i^n to V_i^π.

Lemma 6.3 *We have*

1. $\underline{D}_i^m \leq \underline{D}_i^n \leq V_i^n \leq \overline{D}_i^n \leq \overline{D}_i^m, \ \forall i, n, \ m \leq n$
2. $\underline{D}_i^n \leq V_i^\pi \leq \overline{D}_i^n, \ \forall i, n$

*Proof.*We proceed by induction on event number. Consider the following set of equations at some event n.

(i). $\underline{D}_i^l \leq \underline{D}_i^m \leq V_i^m \leq \overline{D}_i^m \leq \overline{D}_i^l, \ \forall i, \ l \leq m \leq n$

(ii). $\underline{D}_i^n \leq V_i^\pi \leq \overline{D}_i^n, \ \forall i$

At $n = 0$, we have the starting values $\underline{D}_i^0 = 0$ and $\overline{D}_i^0 = R_{max}$. Hence (ii) is true. Also, the starting value for the actual system must be between these extreme values $0 \leq V_i^0 \leq R_{max}$ for any node i, so (i) is true at $n = 0$ also.

Assume that after event n (i) and (ii) are true. We want to prove that (i) and (ii) are true after event $n+1$. We consider the two possible event types at $n+1$, as follows.

(1). A node transmits its V value. This has no effect on (i) and (ii) above, and they remain valid for $n+1$.

(2). A node (call it i) recomputes its V value. Since all nodes other than i are unaffected, we can focus on i at $n+1$. The last time i's V value was computed, some estimate of each neighbor's V value was used. Consider neighbor j, and let l be the past event at which j computed the V value which was used by i for j at its last V value computation. Similarly, let m be the past event at which j computed its value which is being used by i for the current computation at $n+1$. Then clearly $l \leq m \leq n$, and by (i) and the induction hypothesis we have

$$\underline{D}_j^l \leq \underline{D}_j^m \leq V_j^m \leq \overline{D}_j^m \leq \overline{D}_j^l \tag{151}$$

Since (151) is true for all neighbors of i, by the monotonicity property we have

$$\underline{D}_i^n \leq \underline{D}_i^{n+1} \leq V_i^{n+1} \leq \overline{D}_i^{n+1} \leq \overline{D}_i^n \tag{152}$$

Equation (152) and our assumption of (i) at n together imply that (i) is also valid for i at $n+1$.

Next, by inductive assumption (ii) we have for any neighbor j at event m, where m is as above,

$$\underline{D}_j^m \leq V_j^\pi \leq \overline{D}_j^m \tag{153}$$

By Lemma 3.6, if each neighbor j has its correct optimal value V_j^π, then (144) for i gives V_i^π, the optimal value for i. So again monotonicity (Lemma 6.2) implies that

$$\underline{D}_i^{n+1} \leq V_i^\pi \leq \overline{D}_i^{n+1} \tag{154}$$

and (ii) is also true at $n+1$.

Hence for both types of events (i) and (ii) remain true at $n+1$, and the induction step is complete □

Comment: In Lemma 6.3, the first part follows directly from the monotonicity property, and in fact is true for any function that is computed at each node that has this property. The second part further requires the property that when all neighbors are correct, the correct value function gets computed.

Corollary 6.1 *We have*

1. $\underline{D}_i^n = V_i^\pi \Longrightarrow \underline{D}_i^m = V_i^\pi, \forall m \geq n$
2. $\overline{D}_i^n = V_i^\pi \Longrightarrow \overline{D}_i^m = V_i^\pi, \forall m \geq n$

*Proof.*These two results follow directly from Lemma 6.3. □

In the following lemma, we demonstrate convergence of $\underline{D}_i^n$ to the optimal value V_i^π in a finite number of steps (we remind the reader that π denotes an optimal policy).

Lemma 6.4 *There exists $n_p < \infty$ s.t. $\forall n \geq n_p, \underline{D}_i^n = V_i^\pi, \ \forall i \in \Omega$.*

*Proof.*By Corollary 6.1, we only need to show existence of an event n where $\underline{D}_i$equals $V_i^\pi, \forall i \in \Omega$. By Corollary 6.1, once a node reaches V_i^π, it does not change thereafter. Define G to be the set of nodes at event n for which (i) $\underline{D}_i^n = V_i^\pi, i \in G$, and (ii) each $i \in G$ has successfully transmitted its optimal value V_i^π to all of its neighbors.

We proceed by induction. At $n = 0$, the destination nodes of highest reward have the correct V's (i.e. R_{max}). Because transmission occurs infinitely often, there is an event at which these nodes have transmitted their values to their neighbors. This proves the induction basis step.

Now assume that at event $n-1$ set G includes the $g > 0$ best nodes according to π, but not the $(g+1)$'th best node. Let i be the $(g+1)$'th best node according to π. Also assume that i recomputes its V value at event n. Event n exists because node recomputations occur infinitely often. By Lemma 6.3 (2.), the monotonicity property, and the form of the actual optimal value for i given in Lemma 3.6, the above algorithm computing the right-hand-side of (146) will give

V_i^π. This is because neighbors not better than i according to π will have current V's (expressed by the corresponding $\underline{\text{D}}$'s) less than or equal to their optimal values, which are less than i's. Hence computation (146) in Method 1 will produce the optimal value for i. Once i successfully transmits its value after event n to all its neighbors, which occurs in finite time because tranmissions occur infinitely often, node i will enter G.

By induction all nodes $i \in \Omega$ enter G in this way. Let $n_p < \infty$ be the event at which the last node enters G. This n_p satisfies the requirement of the lemma. □

Unfortunately, finite time convergence does not hold in general for $\overline{D}_i^n$, as the following example demonstrates.

Example

Consider the system of Figure 3 with parameters $0 < p < 1$, $0 < q \leq 1$, $c_1 = c_2 = 1$, and $R_1 = R_2 = 0, R_3 = R$. That is, Q_3 is the destination node. Assume that $R > \frac{1}{p}$, so the system has a non-trivial optimal policy. Transmission success from either Q_1 or Q_2 to the other two nodes is independent, with probabilities p and q respectively. Assume Algorithm 4 begins with $V_i^0 = V_{i,j}^0 = R, \forall i, j$.

As Algorithm 4 runs, the node value computations in Q_1 and Q_2 ping-pong back and forth, as the update of one is transmitted to be used in the other's update. Letting V_1^n denote the update value at Q_1 at the nth such update (i.e. the nth update where V_1 actually changes value), it can be shown that

$$\begin{aligned} V_1^n &= \frac{(1-p)^2 q^2}{(p+(1-p)q)^2} \quad V_1^{n-1} + \frac{(pR-1)(p+2(1-p)q)}{(p+(1-p)q)^2} \quad n \geq 1 \\ &:= A \quad V_1^{n-1} + B \end{aligned} \tag{155}$$

where $V_1^0 = R$, and A and B are defined by (155). Since $0 < A < 1$, standard difference equation methods [Wilf 94] yield the closed form solution

$$V_1^n = \left(R - \frac{B}{1-A}\right) A^n + \frac{B}{1-A} \tag{156}$$

A similar equation holds for V_2^n. Since $A < 1$ and $\frac{B}{1-A} = R - \frac{1}{p}$, we obtain $\lim_{n\to\infty} V_1^n = R - \frac{1}{p}$, which is the correct value function. But because $A > 0$, this value is never reached in finite time. □

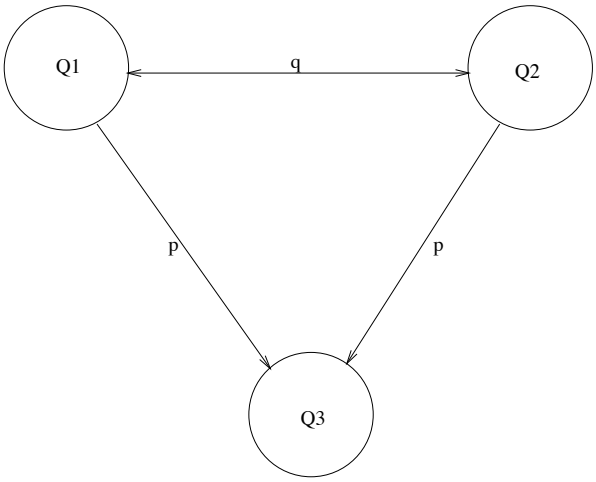

Figure 3: System which takes infinite time to converge

Below we will discuss the sense in which an optimal distributed policy *is* reached in finite time in the above example.

Comment: The key fact in the preceding example is that nodes 1 and 2 have identical value functions. As a result, the V_i^n values of these two nodes converge to the same value, and hence interact in each computation at each event n, preventing convergence in finite time. When limit values are different at each node, a result similar to Lemma 6.4 for the $\overline{D}_i^n$ sequence follows easily, as shown in the following lemma.

Lemma 6.5 *For Algorithm 1,*

1. $\forall i$, *there exists* $W_i \in \mathbb{R}$ *s.t.* $\overline{D}_i^n \downarrow W_i$.

2. *Assume* $W_i \neq W_j, \forall j \neq i$. *Then* $\exists n_p < \infty$ *s.t.* $\forall n \geq n_p, \overline{D}_i^n = V_i^\pi, \forall i$.

*Proof.*By Lemma 6.3 we have that $\overline{D}_i^n$ is monotonically non-increasing and lower bounded by $V_i^\pi, \forall i$. Hence, for each i there is a limit point of the sequence, which we label W_i. This proves 1.

To prove 2., let $\epsilon = \min_{i \neq j}\{|W_i - W_j|\}$. By the definition of sequence convergence, there exists an event m s.t. $\overline{D}_i^n - W_i < \epsilon, \forall n \geq m, \forall i \in \Omega$. This means at every node the node ranking (i.e. their relative priority) is fixed $\forall n \geq m$. Call $\tilde{\pi}$ the policy that uses this node ranking. As the nodes compute their values in the order of $\tilde{\pi}$, and successfully transmit them afterwards, the expected reward value $V_i^{\tilde{\pi}}$ is computed at each node i. This computation of $V_i^{\tilde{\pi}}$ at each node $i \in \Omega$

occurs in finite time because computations and communication occur infinitely often. The value $V_i^{\tilde{\pi}}$ corresponds to the expected reward at node i under $\tilde{\pi}$, a viable, possibly sub-optimal, priority. At a node i once $V_i^{\tilde{\pi}}$ is computed this value is fixed thereafter, so we must have $V_i^{\tilde{\pi}} = W_i$. We thus have

$$V_i^{\tilde{\pi}} = W_i \geq V_i^{\pi}, \ \forall i \tag{157}$$

The inequality in (157) follows from Lemma 6.3, 2. Since π is an optimal policy, $V_i^{\pi} \geq V_i^{\tilde{\pi}}$. Hence $\tilde{\pi}$ is also an optimal policy, which is determined in finite time. □

We are able to prove asymptotic convergence for the $\overline{D}_i^n$ sequence in general. To accomplish this, we define a new system, which we refer to as the *round-robin* (r-r) system. In the r-r system, node update and transmission events follow a fixed pre-defined order. The r-r system is more analytically tractable than the general asynchronous system of Algorithm 4. Using the monotonicity property (Lemma 6.2), we are able to prove that $\overline{D}_i^n$ is bounded above by a corresponding value in the r-r system (specified in Definition 6.6), and then show that they both must converge to V_i^{π}. We begin with the basic definitions.

Definition 6.5 *The* round-robin *(r-r) system is defined as the system of Problem* $(\mathbf{P_1})$ *restricted to the following order of events. For each* $1 \leq i \leq N$*, node* i *computes* V_i*, successfully transmits* V_i *to its neighbors, and then these events are repeated at* $(i+1) mod\ N$*. For consistency in the following, we assume the system starts at* $i = 1$*.*

Definition 6.6 *For the r-r system:*

- *Define* u_n^i *to be the event number of the* n*th computation of the* V *value of* i*.*
- *Let* W_i^k *be the* V *value at* u_n^i *for* i *with initial state* R_{max} *for all nodes in the system.*

For the actual system:

- *Define* v_n^i *to be the event number of the first computation of the* V *value at* i *after each node* j*,* $j < i$ *has completed the* n*th computation of its* V *value, and each node* j*,* $j > i$ *has completed the* $(n-1)$*th computation of its* V *value.*

Note that the u_n^i's and v_n^i's are all finite, since we have assumed that each of these events occurs infinitely often. From the above definitions, we immediately infer that

$$v_n^i \geq u_n^i, \ \ \forall i, n \tag{158}$$

We now show that $W_i^{u_n^i}$ bounds $\overline{D}_i^{v_n^i}$.

Lemma 6.6 *We have*

$$\overline{D}_i^{v_n^i} \leq W_i^{u_n^i}, \ \ \forall i, n \tag{159}$$

That is, the nth computation of the V value at i in the actual system is bounded above by that for the r-r system, when both start all nodes at R_{max}, for all i and number of computations n.

*Proof.*We use induction on i and n. First is induction on n. Since both systems start at the same values, we have $\overline{D}_i^{v_0^i} = W_i^{u_0^i}, \forall i$.

Now assume for a given n we have

$$\overline{D}_i^{v_m^i} \leq W_i^{u_m^i}, \forall m < n, \forall i \tag{160}$$

We now use induction on i. When $W_1^{u_n^1}$ is computed, all neighbor values used are of the form $W_j^{u_{n-1}}, j \neq 1$. When $\overline{D}_1^{v_n^1}$ is computed, all neighbor values used are of the form $\overline{D}_j^k$, $j \neq 1$, where $k \geq v_{n-1}^j$ (i.e. more computations beyond the $(n-1)$'st might have occured in the actual (asynchronous) system). Now Lemma 6.3 and (160) imply that $\overline{D}_j^k \leq \overline{D}_j^{v_{n-1}^j} \leq W_j^{u_{n-1}^j}, \forall k \geq v_{n-1}^j$. Hence, the monotonocity property gives that $\overline{D}_1^{v_n^1} \leq W_1^{u_n^1}$. This is the first induction step on i.

Assume next that for a given i we have

$$\overline{D}_l^{v_n^l} \leq W_l^{u_n^l}, \forall l < i \tag{161}$$

When $W_i^{u_n^i}$ is computed, all neighbor values used are one of the two forms $W_j^{u_n^j}, j < i$ and $W_j^{u_{n-1}^j}, j > i$. When $\overline{D}_i^{v_n^i}$ is computed, all neighbor values used are one of the two forms $\overline{D}_j^k$, $j < i$, where $k \geq v_n^j$, and $\overline{D}_j^k$, $j > i$, where $k \geq v_{n-1}^j$ (again more computations beyond the $(n-1)$'st or n'th might have occured in the asynchronous system). We have by Lemma 6.3 and the inductive assumptions (both (160) and (161))

$$\begin{aligned} \overline{D}_j^k &\leq \overline{D}_j^{v_n^j} \leq W_j^{u_n^j}, \forall k \geq v_n^j, j < i \\ \overline{D}_j^k &\leq \overline{D}_j^{v_{n-1}^j} \leq W_j^{u_{n-1}^j}, \forall k \geq v_{n-1}^j, j > i \end{aligned}$$

So again the monotonicity property gives $\overline{D}_i^{v_n^i} \leq W_i^{u_n^i}$. This completes the induction step for i, so we have that $\overline{D}_i^{v_n^i} \leq W_i^{u_n^i}, \forall i$. And this completes the induction step for n, which completes the proof of the lemma. □

Since Lemma 6.3 is valid for any event order, the results there are also valid for the r-r system. Hence for each i, W_i^k is a non-increasing sequence which is bounded below by V_i^π. Hence it converges down to a limit (see [Rudi 76] Thm 3.14). We will show that this limit must be V_i^π, so that in general $W_i^k \downarrow V_i^\pi, \forall i$.

Definition 6.7 *We write the update equation for i in the r-r system, and define the update function f_i as follows.*

$$W_i^{k+1} = \max\left\{\frac{-c_i + \sum_{S\supset\{i\}:\pi_{k+1}(S)<i} P^i(S|i) W^{k+1}_{\pi_{k+1}(S)} + \sum_{S\supset\{i\}:\pi_{k+1}(S)>i} P^i(S|i) W^k_{\pi_{k+1}(S)}}{\sum_{S\supset\{i\}:\pi_{k+1}(S)\neq i} P^i(S|i)}, R_i\right\}$$

$$:= \max\left\{f_i(W_1^{k+1}, W_2^{k+1}, \ldots, W_{i-1}^{k+1}, W_{i+1}^k, \ldots, W_N^k), R_i\right\} \tag{162}$$

For convenience, we will not notate the k's in (162) when writing the formula for f_i, since they are fixed. Define

$$\begin{aligned}\mathbf{W}^{i-} &:= (W_1, W_2, \ldots, W_{i-1}, W_{i+1}, \ldots, W_N)\\ \mathbf{W}^{ij-} &:= (W_1, W_2, \ldots, W_{j-1}, W_{j+1}, \ldots, W_{i-1}, W_{i+1}, \ldots, W_N)\end{aligned}$$

where we have assumed that $j < i$. Hence we can write (162) as

$$W_i^{k+1} = \max\{f_i(\mathbf{W}^{i-}), R_i\} \tag{163}$$

Note the policy used at each step π_{k+1} is itself varying with time, but is only a function of the neighbor nodes of i at k. Also note that though in general f_i is a function of all nodes in the system other than i, often only some subset of these nodes are actually neighbors of i and affect the update computation.

Lemma 6.7 *For every i, $f_i(\mathbf{W}^{i-})$ is component-wise continuous and piecewise linear, of non-negative monotonically non-decreasing slope.*

*Proof.*The proof of Lemma 6.7 is in Appendix A. □

We pick an arbitrary node from which to observe the update of all the nodes in the r-r system. For notational convenience, and without loss of generality, we consider node N. Let $\mathbf{W}^n$ be the node values after the nth computation and transmission of W_N^n.

Definition 6.8 *We define the mapping* $T : \mathbb{R}^N \to \mathbb{R}^N$ *as*

$$\mathbf{W}^{n+1} = T(\mathbf{W}^n)$$

where

$$\begin{aligned} W_1^{n+1} &= \max\{f_1(W_2^n, \ldots, W_N^n), R_1\} \\ W_2^{n+1} &= \max\{f_2(W_1^{n+1}, W_3^n, \ldots, W_N^n), R_2\} \\ &\vdots \\ W_{N-1}^{n+1} &= \max\{f_{N-1}(W_1^{n+1}, \ldots, W_{N-2}^{n+1}, W_N^n), R_{N-1}\} \\ W_N^{n+1} &= \max\{f_N(W_1^{n+1}, \ldots, W_{N-1}^{n+1}), R_N\} \end{aligned} \tag{164}$$

Note that T is fixed for all n. The fact that this T mapping is fixed is the reason we have introduced this notion of a r-r system. It allows for a fixed-point argument which we now develop. Our procedure is to demonstrate the continuity of the mapping T, show that $\mathbf{V}^\pi$ is a unique fixed point of T, and then use continuity of T to show that the limit point of $\mathbf{W}^n$ is a fixed point of T. We can then conclude that this limit point is the value function.

Lemma 6.8 *The mapping* T *is component-wise continuous for each output.*

*Proof.*The assertion of Lemma 6.8 follows from (164), Lemma 6.7, the fact that $\max\{\cdot, \cdot\}$ is continuous in its arguments, and the fact that composition of continuous functions results in a continuous function. □

Lemma 6.9 $\mathbf{V}^\pi = T(\mathbf{V}^\pi)$*, and there are no other fixed points of* T*.*

*Proof.*By Corollary 6.1, $\mathbf{V}^\pi$ is a fixed point of T.
Now assume there exists $\mathbf{W} \neq \mathbf{V}^\pi$ s.t. $\mathbf{W} = T(\mathbf{W})$. Let

$$i \in argmax_{1 \leq j \leq N}\{W_j : W_j \neq V_j^\pi\} \tag{165}$$

This i may not be unique (there may be a tie), but if so, the following still holds for any one of the maxima. By Lemma 6.3 and Lemma 6.4 we have $W_i \geq V_i^\pi$. This fact and (165) give

$$W_i > V_i^\pi \tag{166}$$

Let $S := \{j \in \Omega : W_j > W_i\}$. Then by (165) we have

$$W_j = V_j^{\pi},\ \forall j \in S \tag{167}$$

This fact, the definition of $T(\cdot)$, and (146) imply that $T(W)$ uses only V_j^{π} values in the computation of $W_i = T(\mathbf{W})|_i$. Let π^i be the local policy for i determined by Method 1, and note that π^i ranks only the nodes of S above i. Then we have by Lemma 3.6 that

$$W_i = T(\mathbf{W})|_i = V_i^{\pi^i} \leq V_i^{\pi} \tag{168}$$

The final inequality of (168) follows because of (167) and the fact that π has an optimal local ordering.

Relation (168) contradicts (166). Hence, no such i can exist, and this completes the proof of the lemma. □

Lemma 6.10 $W_i^* := \lim_{k\to\infty} W_i^k = V_i^{\pi},\ \forall i$

*Proof.*We have

$$\begin{aligned} \mathbf{W}^* &= \lim_{k\to\infty} \mathbf{W}^{k+1} \\ &= \lim_{k\to\infty} T(\mathbf{W}^k) \\ &= T(\lim_{k\to\infty} \mathbf{W}^k) \\ &= T(\mathbf{W}^*) \end{aligned}$$

where the third equality holds because of Lemma 6.7, Lemma 6.8, and Theorem 4.10 of [Rudi 76]. Hence any limit point of $\mathbf{W}^k$ must be a fixed point of T. But by Lemma 6.9, the only fixed point of T is $\mathbf{V}^{\pi}$, and the result follows. □

Based on Lemma 6.2-Lemma 6.10 we prove Theorem 6.1.

Proof of Theorem 6.1

Proof of 1. By Lemma 6.3 and Lemma 6.4, after finite time n_p we have

$$V_i^n \geq V_i^{\pi},\ \forall n \geq n_p, \forall i \in \Omega \tag{169}$$

Combining Lemma 6.6 and Lemma 6.10, we have

$$\lim_{n\to\infty} \overline{D}_i^{v_n^i} \leq \lim_{n\to\infty} W_i^{u_n^i} = \lim_{k\to\infty} W_i^k = V_i^\pi, \ \forall i \in \Omega \tag{170}$$

Lemma 6.3 (1) and (170) ensure that

$$\lim_{n\to\infty} V_i^n \leq \lim_{n\to\infty} \overline{D}_i^{v_n^i} \leq V_i^\pi, \ \forall i \in \Omega \tag{171}$$

Inequalities (169) and (171) imply that $\lim_{n\to\infty} V_i^n = V_i^\pi$.

Proof of 2. Because of (147) there exists n_p after which the node values at each node i and its neighbors $\mathcal{N}(i)$ are in the order of some local index policy that coincides with the order of an optimal index policy. At each event $n \geq n_p$ this local ordering can change, but the new local order coincides with the order of an optimal index policy. Hence, at each node i an optimal action is taken at each event $n \geq n_p$. Consequently, the distributed policy is optimal.

Proof of 3. Because of (147), (148), and Lemma 6.3, there exists n_p after which node values for each node i and its neighbors $\mathcal{N}(i)$ are in the order of the unique optimal local index policy and there are no subsequent changes in this order. After n_p, once a full round of node updates occurs, each node has its correct V_i^π and its correct local index policy, which subsequently do not change. □

We give an example where Algorithm 1 does not converge to an optimal index policy, but still converges to an optimal distributed policy.

Example

Consider the system of Figure 4 with parameters $0 < p < 1$, $0 < q \leq 1$, $c_1 = c_2 = c_4 = 1$, and $R_1 = R_2 = R_4 = 0, R_3 = R$. Assume node recomputations and successful value transmissions occur in the order $[1, 4, 2, 4, 1, 4, 2, 4, \ldots]$. Then node 1 and 2 updates are still represented by (156). The policy at node 1 is fixed at $\pi_t^1 = (3, 2, 1), \forall t$, and the policy at node 2 is fixed at $\pi_t^2 = (3, 1, 2), \forall t$. But at each update of node 4, the policy changes, as the ranking of the values of nodes 1 and 2 alternate. That is, the policy computed at node 4 is

$$\pi_t^4 = \{(1, 2, 4)\ (2, 1, 4)\ (1, 2, 4)\ (2, 1.4)\ (1, 2, 4) \ldots\} \tag{172}$$

Of course, $\pi_t^3 = r_3$. The overall policy is $\mathbf{\Pi}_t = (\pi_t^1\ \pi_t^2\ \pi_t^3\ \pi_t^4)$, which is not a stationary policy, due to π_t^4. It is, however, an optimal distributed policy.

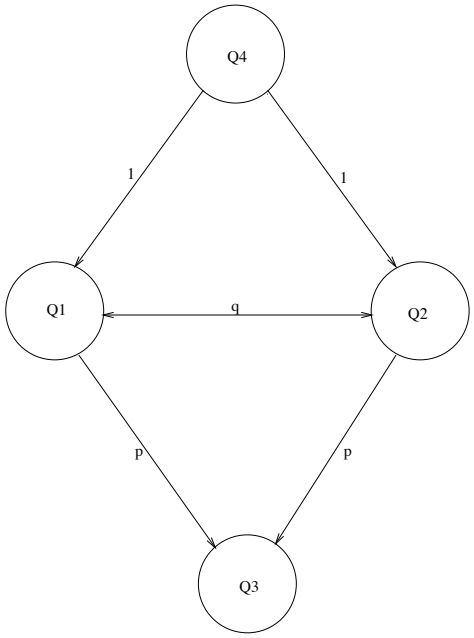

Figure 4: System with optimal distributed policy

6.3 Distributed Dynamic Programming Formulation

We develop a distributed algorithm, different from but related to Algorithm 4, using the methodology of Distributed Dynamic Programming (DDP) [Bert 82]. We first present briefly the model, key assumptions, and key results in [Bert 82].

6.3.1 Summary of Distributed Dynamic Programming [Bert 82]

Distributed Dynamic Programming [Bert 82] is a technique for solving dynamic programming problems using distributed computation. The technique may only be used for problems formulated so that a standard dynamic programming equation applies, and for which a suitable partitioning of the state space among processors ("computation centers") can be made. Two advantages of using distributed computation for solving dynamic programming problems are adduced: 1) Computation of a solution can be accomplished in less time, as work is performed in parallel, and 2) Some problems, such as those involving communication networks, possess a natural decentralization of data. Rather than gather all of this data into one place to perform the computation, it can be more efficient to leave the data distributed throughout the network, and then use a distributed

algorithm like DDP to compute the value function and optimal policy.

In this section we summarize the key results of [Bert 82], using the paper's own notation. The dynamic programming problem is formulated as follows. The state space is S, and the control action space is C. For each $x \in S$ there is a given subset $U(x) \subset C$ which is the control action constraint set at x. F is the set of all extended real-valued functions $J : S \to [-\infty, \infty]$ on S. The following notation is used for any two functions $J_1, J_2 \in F$:

$$J_1 \leq J_2 \qquad \text{if } J_1(x) \leq J_2(x),\ \forall x \in S$$
$$J_1 = J_2 \qquad \text{if } J_1(x) = J_2(x),\ \forall x \in S$$

Let $H : S \times C \times F \to [-\infty, \infty]$ be a mapping which is *monotone* in the sense that for all $x \in S$ and $u \in U(x)$, we have

$$H(x, u, J_1) \leq H(x, u, J_2), \quad \forall J_1, J_2 \in F \text{ with } J_1 \leq J_2 \tag{173}$$

Given a subset $\bar{F} \subset F$ the problem is to find a function $J^* \in \bar{F}$ such that

$$J^*(x) = \inf_{u \in U(x)} H(x, u, J^*), \quad \forall x \in S \tag{174}$$

By considering the mapping $T : F \to F$ defined by

$$T(J)(x) = \inf_{u \in U(x)} H(x, u, J) \tag{175}$$

the problem is alternately stated as one of finding a fixed point of T within $\bar{F}$, i.e., a function $J^* \in \bar{F}$ such that

$$J^* = T(J^*) \tag{176}$$

The DDP algorithm is described as follows. It is assumed there are n computation centers, also called nodes. The state space S is partitioned into n disjoint sets denoted $S_1, \ldots, S_n$. Each node i is assigned the responsibility of computing the values of the solution function J^* at all states x in the corresponding set S_i. A node j is said to be a *neighbor* of node i if $j \neq i$ and there exist a state $x_i \in S_i$ and two functions $J_1, J_2 \in \bar{F}$ such that

$$J_1(x) = J_2(x), \quad \forall x \notin S_j \tag{177}$$
$$T(J_1)(x_i) \neq T(J_2)(x_i) \tag{178}$$

The set of neighbors of i is denoted $N(i)$.

At each time instant, node i can be in one of three possible states - *compute, transmit,* or *idle.* In the compute state node i computes a new estimate of the values of the solution function J^* for all states $x \in S_i$. In the transmit state node i communicates the estimate obtained from the latest compute phase to one or more nodes m for which $i \in N(m)$. In the idle state node i does nothing related to the solution of the problem. No assumption is made on the length, timing, and sequencing of computation and transmission intervals, other than the following.

Assumption 1: *There exists a positive scalar $\mathcal{P}$ such that, for every node i, every time interval of length $\mathcal{P}$ contains at least one computation interval for i and at least one transmission interval from i to each node j with $i \in N(j)$.*

At time t, the last value node i received from node j is denoted $J_{ij}^t, j \in N(i)$. Node i's own value is denoted J_{ii}^t. The rules according to which the functions J_{ij}^t are updated are as follows.

1)If $[t_1, t_2]$ is a transmission interval for node j to node i with $i \in N(j)$, then

$$J_{ij}^{t_2} = J_{jj}^{t_1} \tag{179}$$

2)If $[t_1, t_2]$ is a computation interval for node i, we have

$$J_{ii}^{t_2}(x) = T\left(J_i^{t_1}\right)(x) = \inf_{u \in U(x)} H\left(x, u, J_i^{t_1}\right), \quad \forall x \in S_i \tag{180}$$

One more assumption, Assumption 2, is necessary for the two important propositions of the paper, Propositions 1 and 3, which we state below.

Assumption 2: *There exist two functions $\underline{J}$ and $\bar{J}$ in F such that the set of all functions $J \in F$ with $\underline{J} \leq J \leq \bar{J}$ belongs to $\bar{F}$ and furthermore*

$$\bar{J} \geq T(\bar{J}), \quad T\left(\underline{J}\right) \geq \underline{J} \tag{181}$$

and

$$\lim_{k \to \infty} T^k(\bar{J})(x) = J^*(x), \ \forall x \in S \tag{182}$$

$$\lim_{k \to \infty} T^k(\underline{J})(x) = J^*(x), \ \forall x \in S \tag{183}$$

Three propositions are presented in [Bert 82], of which two are fundamental results, and one is specific to the examples presented in the paper. We state the two fundamental propositions here.

Proposition 1: *Let Assumptions 1 and 2 hold and assume that for all* $i \in 1, \ldots, n$

$$\underline{J}(x) \leq J_{ij}^0(x) \leq \bar{J}(x), \quad \forall x \in S_j, j = i \text{ or } N(i) \tag{184}$$

Then for all $i = 1, \ldots, n$

$$\lim_{t \to \infty} J_{ij}^t(x) = J^*(x), \quad \forall x \in S_j, j = i \text{ or } N(i) \tag{185}$$

The next proposition, Proposition 3, deals with the question of whether at time t control laws $\mu^t : S \to C$ satisfying

$$\mu^t(x) \in U(x), \quad \forall x \in S \tag{186}$$

and

$$H[x, \mu^t(x), J_i^t] = \min_{u \in U(x)} H(x, u, J_i^t), \quad \forall x \in S_i, i = 1, \ldots, n \tag{187}$$

converge in some sense to an optimal control law.

Proposition 3: *Let the assumptions of Proposition 1 hold. Assume also that for every* $x \in S$, $u \in U(x)$ *and sequence* $\{J^k\} \subset \bar{F}$ *for which* $\lim_{k \to \infty} J^k(x) = J^*(x)$ *for all* $x \in S$ *we have*

$$\lim_{k \to \infty} H(x, u, J^k) = H(x, u, J^*) \tag{188}$$

Then for each state $x \in S$ *for which* $U(x)$ *is a finite set there exists* $\bar{t}_x > 0$ *such that for all* $t \geq \bar{t}_x$ *if* $\mu^t(s)$ *satisfies (186), (187) then*

$$H[x, \mu^t(x), J^*] = \min_{u \in U(x)} H(x, u, J^*), \tag{189}$$

Proposition 1 proves asymptotic convergence to the value function under appropriate conditions. Proposition 3 provides conditions under which an optimal policy is reached in finite time.

We now proceed to solve Problem $(\mathbf{P_1})$ using DDP as presented above.

6.3.2 Solution of Problem $(\mathbf{P_1})$ Using DDP

The model of Problem $(\mathbf{P_1})$ is a standard controlled Markov chain with finite state space 2^Ω and action space $i \in \Omega$. Dynamic programming can be directly applied to Problem $(\mathbf{P_1})$ on state space 2^Ω, but this approach is inefficient and does not lead to a direct application of DDP. This is because it is not possible to define useful computation centers as in [Bert 82] satisfying (177)

and (178) through partition of $S = 2^{\Omega}$ (on this state space, to define computation centers in a way that leads to the application of DDP, we would need to know *a priori* the optimal priority list of nodes as dictated by Theorem 3.1). Our approach is to use the index structure of an optimal policy demonstrated in Theorem 3.1 to define a new state space on which DDP can be applied.

To solve Problem $(\mathbf{P_1})$ using DDP, we proceed in three steps.

1. Formulate a new problem, Problem $(\mathbf{F_1})$ below
2. Show that an optimal policy for Problem $(\mathbf{F_1})$ can be mapped to an optimal distributed policy for Problem $(\mathbf{P_1})$
3. Apply DDP to Problem $(\mathbf{F_1})$

Below we present the details of each of the steps above.

Formulation of Problem $(\mathbf{F_1})$

We formulate Problem $(\mathbf{F_1})$.

Problem $(\mathbf{F_1})$

The state space is Ω, the set of nodes. The policy space is the set of all local index policies π^i for all $i \in \Omega$. When transmitting in state i, a cost c_i is incurred and transition to a new state $j \in \Omega$ occurs. Define $P_{ij}^{\pi^i}$ to be the probability of transition from state i to state j, $j \in \mathcal{N}(i)$, under policy π^i. Then,

$$P_{ij}^{\pi^i} = \sum_{S:\pi^i(S)=j, r_j} P^i(S|i) \tag{190}$$

When we retire in state i, the process terminates and a reward R_i is received. The objective is to choose for each state i the local index policy π^i to maximize

$$E\left\{R_{i(\tau)} - \sum_{t=1}^{\tau-1} c_{i(t)}\right\} \tag{191}$$

where τ is the time when the transmission process is terminated, and $i(t)$ is the state at time t.

Mapping of Optimal Policy from Problem $(\mathbf{F_1})$ to Problem $(\mathbf{P_1})$

We define a mapping of policies for Problem $(\mathbf{F_1})$ to policies for Problem $(\mathbf{P_1})$.

Mapping 1 *A general (possibly time-varying) policy for Problem* $(\mathbf{F_1})$ *consists of a sequence of local index policies for each* $i \in \Omega$*. For each* t *we define*

$$\mathbf{\Pi}_t = \{\pi_t^1 \pi_t^2 \dots \pi_t^N\} \tag{192}$$

A policy for Problem $(\mathbf{F_1})$ *is then specified as*

$$\mathbf{\Pi} = \{\mathbf{\Pi}_1 \mathbf{\Pi}_2 \dots \mathbf{\Pi}_t \dots\} \tag{193}$$

We map a policy for Problem $(\mathbf{F_1})$ *represented as (193) into a distributed index policy for Problem* $(\mathbf{P_1})$ *(cf. Definition 6.1) in the obvious way. That is, at each time* t *the local index policy* π_t^i *at each node* $i \in \Omega$ *is the same for Problem* $(\mathbf{P_1})$ *as for Problem* $(\mathbf{F_1})$*.*

We show that mapping an optimal policy of Problem $(\mathbf{F_1})$ by Mapping 1 leads to an optimal policy for Problem $(\mathbf{P_1})$.

Lemma 6.11 *If* $\mathbf{\Pi}$ *is an optimal policy for Problem* $(\mathbf{F_1})$*, then* $\mathbf{\Pi}$ *mapped to a policy for Problem* $(\mathbf{P_1})$*using Mapping 1 is an optimal distributed policy for Problem* $(\mathbf{P_1})$*.*

*Proof.*The assertion of Lemma 6.11 follows from Mapping 1 and the fact that an optimal policy for Problem $(\mathbf{P_1})$ is of the index type. □

A Distributed Dynamic Programming Implementation for Problem $(\mathbf{F_1})$

We formulate a DDP solution of Problem $(\mathbf{F_1})$ by translating the notation of [Bert 82] into our notation. We use our own notation where appropriate. Each node i in our model is a computation center for itself alone, so that $S_i = i, \forall i \in \Omega$. We associate:

$$\begin{aligned} S &= \Omega \\ C &= \{\text{Set of all priority orderings of } \mathcal{N}(i), \forall i \in \Omega\} \\ U(i) &= \{\text{Set of all priority orderings of } \mathcal{N}(i)\} \\ \bar{F} &= F \end{aligned}$$

We also note the notational correspondences $V \leftrightarrow J$, and $V_i^n \leftrightarrow J_i^t$, which are equivalent assuming event n occurs at time t.

A neighbor of a node in the sense of [Bert 82], defined in (177) and (178), corresponds to our notion of neighbor. That is, the neighbors of node i are those nodes with positive marginal probability of receiving the message when i transmits. These are precisely the nodes whose values can affect node i's update. However, the notation for the neighbors of i used in [Bert 82], $N(i)$, does not include i, whereas for our notation $i \in \mathcal{N}(i)$.

For Problem $(\mathbf{F_1})$ we define the H function of [Bert 82] as

$$H(i, \pi^i, V) = -\max \left\{ -c_i + \sum_{S \subseteq \mathcal{N}(i)} P(S|i) V(\pi^i(S)), \quad R_i \right\} \qquad \forall i \in \Omega, \pi^i \in U(i) \tag{194}$$

The negative sign in the RHS of (194) is used to conform to the convention of [Bert 82] that the goal is to minimize cost. Thus, the dynamic programming equation for Problem $(\mathbf{F_1})$ is

$$V_i^\pi = \min_{\pi^i} H(i, \pi^i, V^\pi) \tag{195}$$

Note that H is monotone in V in the sense of (173).

The update for state $i \in \Omega$ by (180) at event n is

$$V_i^n = \min_{\pi^i} H(i, \pi^i, V^n) \tag{196}$$

We suppose that Assumption 1 of [Bert 82] is true for node updates and transmissions. We show that the functions

$$\underline{V}_i := -R_{max}, \qquad \forall i \in \Omega \tag{197}$$

$$\bar{V}_i := 0 \tag{198}$$

satisfy *Assumption 2* of [Bert 82] as follows. Relation (181) follows from the update definition (194) and (196). Relations (182) and (183) follow from (195) and the standard result on value iteration for a dynamic program. We require that, when running DDP for Problem $(\mathbf{F_1})$, we start with a value between $\underline{V}$ and $\bar{V}$ at each node.

This completes the DDP formulation of Problem $(\mathbf{F_1})$. We now briefly state the results from [Bert 82] which apply to Problem $(\mathbf{F_1})$. Assumption 1, Assumption 2, and (184) are satisfied. Hence, the requirements of Proposition 1 in [Bert 82] are satisfied. From (185) of Proposition 1, translating into our notation, we obtain

$$\lim_{n \to \infty} V_i^n = V_i^\pi, \quad \forall i \in \Omega \tag{199}$$

Because for fixed π^i update (194) is continuous in the components of V , Equation (188) of Proposition 3 is satisfied for H. Hence, the requirements of Proposition 3 of [Bert 82] are also satisfied. Translating to our notation, we obtain from (189) that there exists a $\bar{n}_i > 0$ such that for all $n \geq \bar{n}_i$, if a local index policy π_n^i satisfies

$$H(i, \pi_n^i, V_i^n) = \min_{\pi^i} H(i, \pi^i, V_i^n), \quad \forall i \in \Omega \tag{200}$$

then

$$H(i, \pi_n^i, V^\pi) = \min_{\pi^i} H(i, \pi^i, V^\pi) \tag{201}$$

Equation (201) states that π_n^i takes an optimal action for state i at event n.

Thus, we have shown how Problem ($\mathbf{F_1}$) can be solved with the DDP methodology of [Bert 82] using update (196).

6.3.3 Relation Between DDP Solution to Problem ($\mathbf{F_1}$) and Algorithm 4

Event 1 of Algorithm 4 corresponds precisely to the transmit state, which is update 1) of [Bert 82], described by equation (179). Event 2 of Algorithm 4 is related to update 2) of [Bert 82], described by equation (180). However, the update equations (144) for Algorithm 4 and (196) for DDP are not the same. For an update at node $i \in \Omega$ at event $n+1$, the key difference is that in (196) the value V_i^n affects the updated value V_i^{n+1}, whereas in (144) it does not. Another difference, relatively minor, between the DDP formulation and Algorithm 4 is that where Algorithm 4 assumes Events 1 and 2 occur infinitely often, the DDP formulation of Problem ($\mathbf{F_1}$) requires the somewhat more restrictive *Assumption 1.*

Though Algorithm 4 and the DDP formulation differ in the ways just mentioned, the results proved for each algorithm are quite similar. As remarked in (199), Proposition 1 in [Bert 82] implies that

$$\lim_{n \to \infty} V_i^n = V_i^\pi, \forall i \in \Omega \tag{202}$$

Equation (202) shows asymptotic convergence to the value function for Problem ($\mathbf{F_1}$), and hence for Problem ($\mathbf{P_1}$), and is similar to our result Theorem 6.1, (1.). The result of Proposition 3 (201) is similar to our Theorem 6.1, (2.). No results similar to Theorem 6.1, (3.) for Problem ($\mathbf{P_1}$) under the DDP formulation follow directly from a result of [Bert 82].

As these results show, once Theorem 3.1 is proved for Problem ($\mathbf{P_1}$), application of Distributed Dynamic Programming on an appropriate state space provides a new distributed algorithm for Problem ($\mathbf{P_1}$) with certain properties nearly equivalent to those of Algorithm 4.

6.4 Distributed Rank Method

We define a third distributed algorithm for Problem ($\mathbf{P_1}$), which is similar in spirit to Algorithm 4, except that a different method is used to update the node value function. We define the following

Algorithm 5 *At each event time, any number of the following two events can occur.*

Event 1 *A node i receives V_j from a neighbor j, $j \in \mathcal{N}(i)$.*

Event 2 *A node i recomputes V_i using the current $V_{i,j}$ values, as follows.*

1. *The set of $V_{i,j}$ and V_i values are ranked, high-to-low. A local index policy $\tilde{\pi}$ for i is created which uses this ranking (ties are broken arbitrarily). If $V_i \geq V_{i,j}$ $\forall j \in \mathcal{N}(i)$, then $\tilde{\pi}$ uses* any *ranking of the neighbors, so long as i is given a ranking below at least one neighbor.*
2. *The following is computed.*

$$V_i^n = \max \left\{ \frac{-c_i + \sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \tilde{\pi}(S) \neq i} P^i(S|i) V_{i,\tilde{\pi}(S)}^n}{\sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \tilde{\pi}(S) \neq i} P^i(S|i)}, R_i \right\} \tag{203}$$

Events 1 and 2 occur infinitely often at each node i.

Before we proceed with the analysis of the convergence properties of Algorithm 5, we define certain types of events associated with the execution of Algorithm 5.

Definition 6.9 *We call a* node reset *the event where V_i is equal to or larger than all neighbors in Step (a) of Algorithm 5, at which point the algorithm sets i's rank somewhere lower than the highest neighbor, and the left-hand term using $\tilde{\pi}$ is the larger term in (203) in Step (b).*

When a node reset occurs for node i, the new ranking used for $\tilde{\pi}$ is an implementation decision. All of the following results for Algorithm 5 are valid for any choice of $\tilde{\pi}$ when resetting, so long as node i is not ranked highest. Of course, the choice of $\tilde{\pi}$ affects the convergence properties of Algorithm 5. An example of this choice would be to always make i the node of lowest rank among neighbors when resetting. Another choice would be to always make i the second highest ranked node among the neighbors when resetting.

Definition 6.10 *When a node value recomputation using (203) occurs at node i which is* not *a node reset, we say a* standard update *has occurred.*

Definition 6.11 *When a standard update occurs without the retirement value R_i being the optimal choice, so that the left term in the RHS of (203) is chosen, we say a* rank update *has occurred.*

We state our main results for Algorithm 5 in the following theorem.

Theorem 6.2 *For Algorithm 5 with any initial state s.t. $0 \leq V_i^0 \leq R_{max}$ and $0 \leq V_{i,j}^0 \leq R_{max}$ for all $i, j \in \Omega$, we have*

1. $\lim_{n\to\infty} V_i^n = V_i^\pi,\ \forall i \in \Omega$
2. *There exists an event $n_p < \infty$ s.t. $\mathbf{\Pi} = \{\mathbf{\Pi}_{n_p}\ \mathbf{\Pi}_{n_p+1}\ \mathbf{\Pi}_{n_p+2} \ldots\}$ is an optimal distributed policy*
3. *If*
$$V_i^\pi \neq V_j^\pi, \forall i \in \Omega, j \in \mathcal{N}(i) \tag{204}$$
then there exists an event $n_p < \infty$ s.t.
   (a) $V_i^n = V_i^\pi, \forall n \geq n_p, \forall i \in \Omega$
   (b) $\mathbf{\Pi} = \{\mathbf{\Pi}_{n_p}\ \mathbf{\Pi}_{n_p+1}\ \mathbf{\Pi}_{n_p+2} \ldots\}$ *has an associated index policy that is optimal*

Before proving Theorem 6.2, we present a series of lemmas which will be used in the proof. But first we note that the monotonicity property in the sense of Lemma 6.2 does not hold in general for the update of Algorithm 5. We show this by the following example.

Example

Consider the system of Figure 5. Given R_3, V_3^0, $V_{3,1}^0$, and $V_{3,2}^0$, assume at time 1 there is a node update using (203) for node 3. We consider the following two cases. The values used in the update are: Case 1) R_3, V_3^0, $V_{3,1}^0$, and $\hat{V}_{3,2}^0$, and Case 2) R_3, V_3^0, $V_{3,1}^0$, and $\tilde{V}_{3,2}^0$. Assume $R_3 = 0$ and that

$$\tilde{V}_{3,2}^0 < V_3^0 < \hat{V}_{3,2}^0 < \frac{-c_3}{p} + V_{3,1}^0 \tag{205}$$

Case 1 From (203) we have $\pi_3^1 = (1, 2, 3)$. Then $\hat{V}_3^1 = \frac{-c_3 + pV_{3,1}^0 + (1-p)p\hat{V}_{3,2}^0}{p+(1-p)p}$.

Case 2 From (203) we have $\pi_3^1 = (1, 3, 2)$. Then $\tilde{V}_3^1 = \frac{-c_3 + pV_{3,1}^0}{p}$.

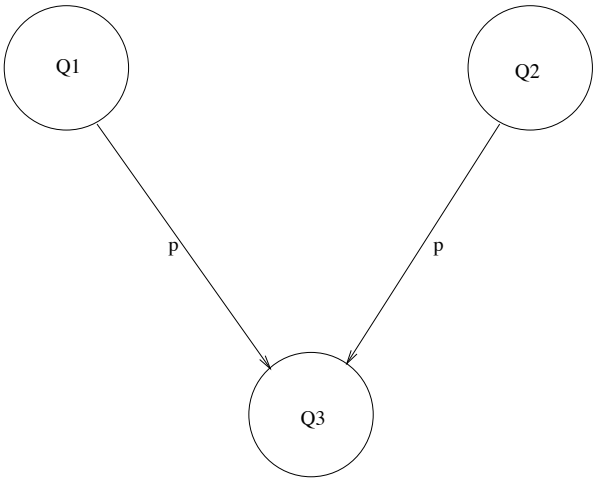

Figure 5: System Demonstrating Violation of Monotonicity for Rank Update

Noting that $\hat{V}^0_{3,2} < \frac{-c_3}{p} + V^0_{3,1}$, we find that $\hat{V}^1_3 < \tilde{V}^1_3$. But since $\hat{V}^0_{3,2} > \tilde{V}^0_{3,2}$ and all else is equal in the two cases, update monotonicity is violated.

To make the example concrete, let $V^0_3 = 5, V_{3,1} = 10, p = .5, c_3 = 1$. Let $\tilde{V}^0_{3,2} = 3$, so $\tilde{V}^1_3 = -2 + 10 = 8$. Let $\hat{V}^0_{3,2} = 6$, so $\hat{V}^1_3 = \frac{-1+(.5)10+.25(6)}{.5(1.5)} = 7\frac{1}{3} < \tilde{V}^1_3$, violating monotonicity. □

Because the update monotonicity property does not hold for Algorithm 5, we must take a very different approach to proving its convergence than what we used for Algorithm 4. In the following lemma we prove what may appear as a technical result of unclear utility, but which proves crucial in the approach we take. It says that when two index policies share the top of a priority list, but differ further down, then their updated values can be usefully related.

Lemma 6.12 *Assume we are given two local non-retiring index policies $\tilde{\pi}$ and $\hat{\pi}$ and a full set of neighbor values $\{V_{i,j}\}$ for node i. Let $A \subset \mathcal{N}(i)$ be the nodes which $\tilde{\pi}$ ranks above node i. Assume that policy $\hat{\pi}$ uses this same ranking for the nodes in A, but that there is also a set $B \subset \mathcal{N}(i) - A$, $B \neq \emptyset$, each member of which $\hat{\pi}$ ranks above node i and below the nodes of A.*

Define $Y_1 := \max_{j \in B} V_{i,j}$, and $Y_2 := \min_{j \in B} V_{i,j}$. Then there exists $0 \leq a < 1$ such that

$$(1-a)V_i^{\tilde{\pi}} + aY_2 \leq V_i^{\hat{\pi}} \leq (1-a)V_i^{\tilde{\pi}} + aY_1 \tag{206}$$

*Proof.*There is no requirement that the node rankings of $\tilde{\pi}$ and $\hat{\pi}$ reflect the actual current ranking of neighbor values $\{V_{i,j}\}$.

The update computation of (203) at node i for local policy $\tilde{\pi}$ is

$$V_i^{\tilde{\pi}} = \frac{-c_i + \sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \tilde{\pi}(S) \neq i} P^i(S|i) V_{i,\tilde{\pi}(S)}}{\sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \tilde{\pi}(S) \neq i} P^i(S|i)} \tag{207}$$

where to ensure (207) is well defined we assume that

$$\sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \tilde{\pi}(S) \neq i} P^i(S|i) \neq 0 \tag{208}$$

A similar computation holds for local policy $\hat{\pi}$.

$$V_i^{\hat{\pi}} = \frac{-c_i + \sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \hat{\pi}(S) \neq i} P^i(S|i) V_{i,\hat{\pi}(S)}}{\sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \hat{\pi}(S) \neq i} P^i(S|i)} \tag{209}$$

Equation (209) is well defined because (208) also ensures its denominator is non-zero.

Write (209) as

$$V_i^{\hat{\pi}} = \frac{-c_i + \sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \hat{\pi}(S) \in A} P^i(S|i) V_{i,\hat{\pi}(S)} + \sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \hat{\pi}(S) \in B} P^i(S|i) V_{i,\hat{\pi}(S)}}{\sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \hat{\pi}(S) \in A} P^i(S|i) + \sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \hat{\pi}(S) \in B} P^i(S|i)} \tag{210}$$

Using the definition of Y_1 and (210) we have

$$V_i^{\hat{\pi}} \leq \frac{-c_i + \sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \hat{\pi}(S) \in A} P^i(S|i) V_{i,\hat{\pi}(S)} + \sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \hat{\pi}(S) \in B} P^i(S|i) Y_1}{\sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \hat{\pi}(S) \in A} P^i(S|i) + \sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \hat{\pi}(S) \in B} P^i(S|i)} \tag{211}$$

Note that since both $\hat{\pi}$ and $\tilde{\pi}$ make A the highest priority nodes with the same ranking, we have

$$\{\mathcal{N}(i) \supseteq S \supset \{i\} : \hat{\pi}(S) \in A\} = \{\mathcal{N}(i) \supseteq S \supset \{i\} : \tilde{\pi}(S) \in A\} \tag{212}$$

and

$$\hat{\pi}(S) = \tilde{\pi}(S), \quad \mathcal{N}(i) \supseteq S \supset \{i\} : \hat{\pi}(S) \in A \tag{213}$$

For simplicity define

$$b_1 := \sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \hat{\pi}(S) \in A} P^i(S|i) = \sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \tilde{\pi}(S) \in A} P^i(S|i) \tag{214}$$

$$b_2 := \sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \hat{\pi}(S) \in B} P^i(S|i) \tag{215}$$

where the equality in (214) follows from (212).

Because of (208), $b_1 > 0$; furthermore

$$0 < b_1 + b_2 \leq 1 \quad \text{and} \quad b_2 \geq 0 \tag{216}$$

Because of (213) and (214), (211) gives

$$V_i^{\hat{\pi}} \leq \frac{-c_i + \sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \hat{\pi}(S) \in A} P^i(S|i) V_{i,\hat{\pi}(S)} + b_2 Y_1}{b_1 + b_2} \tag{217}$$

Write (217) as

$$\begin{aligned} V_i^{\hat{\pi}} &\leq \left(\frac{b_1}{b_1+b_2}\right) \frac{-c_i + \sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \hat{\pi}(S) \in A} P^i(S|i) V_{i,\hat{\pi}(S)}}{b_1} + \left(\frac{b_2}{b_1+b_2}\right) Y_1 \\ &= \left(\frac{b_1}{b_1+b_2}\right) \frac{-c_i + \sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \tilde{\pi}(S) \in A} P^i(S|i) V_{i,\tilde{\pi}(S)}}{b_1} + \left(\frac{b_2}{b_1+b_2}\right) Y_1 \qquad (218) \\ &= \left(\frac{b_1}{b_1+b_2}\right) V_i^{\tilde{\pi}} + \left(\frac{b_2}{b_1+b_2}\right) Y_1 \qquad (219) \end{aligned}$$

Equation (218) follows from (213), and (219) follows from (207).

Define

$$a := \frac{b_2}{b_1 + b_2} \tag{220}$$

Since $b_1 > 0$ and $b_2 \geq 0$, we have

$$0 \leq a < 1 \tag{221}$$

Inequality (219) is then

$$V_i^{\hat{\pi}} \leq (1-a) V_i^{\tilde{\pi}} + a Y_1 \tag{222}$$

A similar argument using Y_2 gives

$$V_i^{\hat{\pi}} \geq (1-a) V_i^{\tilde{\pi}} + a Y_2 \tag{223}$$

□

In the next lemma, we prove a straightforward fact about a rank update, which is that the updated value is upper bounded by the highest neighbor value minus the cost c_i.

Lemma 6.13 *Suppose there is a rank update for node i at event n, with neighbor node values $V_{i,j}^{n-1}, j \in \mathcal{N}(i)$, and neighbor ranking $\tilde{\pi}$. Define $V_{max} = \max_{j \in \mathcal{N}(i): j >_{\tilde{\pi}} i} V_{i,j}^{n-1}$. Then*

$$V_i^n \leq V_{max} - c_i \tag{224}$$

*Proof.*We have

$$\begin{aligned} V_i^n &= \frac{-c_i + \sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \tilde{\pi}(S) \neq i} P^i(S|i) V^n_{i,\tilde{\pi}(S)}}{\sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \tilde{\pi}(S) \neq i} P^i(S|i)} && (225) \\ &\leq \frac{-c_i + \sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \tilde{\pi}(S) \neq i} P^i(S|i) V_{max}}{\sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \tilde{\pi}(S) \neq i} P^i(S|i)} && (226) \\ &= V_{max} - \frac{c_i}{\sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \tilde{\pi}(S) \neq i} P^i(S|i)} && (227) \\ &\leq V_{max} - c_i && (228) \end{aligned}$$

where (228) follows because $\sum_{\mathcal{N}(i) \supseteq S \supset \{i\}: \tilde{\pi}(S) \neq i} P^i(S|i) < 1$ and $c_i > 0$. Inequality (228) completes the the proof of the lemma. □

As before, we use V^π to denote the optimal value function for Problem ($\mathbf{P_1}$).

In the following lemma, we use use a continuity property of the Algorithm 5 update to translate bounds on neighbor values into bounds on the updated value.

Lemma 6.14 *Suppose π_j is an optimal local policy for node j, and let $B := \{k \in \mathcal{N}(j) : k >_{\pi_j} j\}$. Then for any $\epsilon > 0$ there is $\delta > 0$ such that*

$$V_{j,k} \geq V_k^\pi - \delta, \forall k \in B \Longrightarrow V_j^{\pi_j} \geq V_j^\pi - \epsilon \tag{229}$$

Suppose $\tilde{\pi}_j$ is a local policy (possibly suboptimal) for node j, let $\tilde{B} := \{k \in \mathcal{N}(j) : k >_{\tilde{\pi}_j} j\}$, and assume that $\tilde{B} \subset B$. Then for any $\epsilon > 0$ there is $\delta > 0$ such that

$$V_{j,k} \leq V_k^\pi + \delta, \forall k \in \tilde{B} \Longrightarrow V_j^{\tilde{\pi}_j} \leq V_j^\pi + \epsilon \tag{230}$$

*Proof.*Because π_j is optimal, when

$$V_{j,k} = V_k^\pi, \ \ \forall k \in \mathcal{N}(j) \tag{231}$$

by Lemma 3.6 (51) we have that

$$V_j^\pi = \max \left\{ \frac{-c_j + \sum_{\mathcal{N}(j) \supseteq S \supset \{j\}: \pi_j(S) \neq j} P^j(S|j) V_{j,\pi_j(S)}}{\sum_{\mathcal{N}(j) \supseteq S \supset \{j\}: \pi_j(S) \neq j} P^j(S|j)}, R_j \right\} \tag{232}$$

Hence, under (231) the update (203) using π_j computes the value function V_j^π. By [Rudi 76] Section 4.11, every rational function $f : \mathbb{R}^N \to \mathbb{R}$ is continuous. Because: 1) Update (203) is a

composition of a rational function of the values $V_{j,k}, k \in \mathcal{N}(j)$, with the function $\max\{\cdot,\cdot\}$; 2) the function $\max\{\cdot,\cdot\}$ is continuous in its arguments, 3) the composition of continuous functions results in a continuous function, and 4) only node values $V_{j,k}, k \in B$ affect the (203) update using π_j, we conclude that update (203) using π_j is a continuous function mapping $\mathbb{R}^l \to \mathbb{R}$, where $l := |B|$. Since (203) is monotonic in $V_{j,k}$, the result (229) follows.

To obtain (230), we note that under (231) for any local policy $\tilde{\pi}_j$ we have

$$V_j^{\tilde{\pi}_j} \leq V_j^{\pi} \tag{233}$$

By the same argument given above, for fixed $\tilde{\pi}_j$ the update (203) using $\tilde{\pi}_j$ is a continuous function mapping $\mathbb{R}^l \to \mathbb{R}$, where $l := |\tilde{B}|$, and where only node values $V_{j,k}, k \in \tilde{B}$ affect the update. Because (203) is monotonic in $V_{j,k}$, the result (230) follows. □

The fact that Lemma 6.12 guarantees the existence of an $0 \leq a < 1$ satisfying (206) is key to the arguments we develop in this section. We simplify the use of this fact by defining a maximum over all such a's.

Definition 6.12 *For a given node $i \in \Omega$ and a pair of local index policies $\tilde{\pi}$ and $\hat{\pi}$, let $a(i, \tilde{\pi}, \hat{\pi})$ be given by (214), (215), and (220). Define*

$$a_{max} := \max_{i \in \Omega, \tilde{\pi}, \hat{\pi}} a(i, \tilde{\pi}, \hat{\pi}) \tag{234}$$

Note that $0 \leq a_{max} < 1$.

The following lemma is our main substantive result for the rank algorithm, and leads directly to Theorem 6.2.

Lemma 6.15 *For Algorithm 5 we have* $\lim_{n \to \infty} V_i^n = V_i^{\pi}, \ \forall i \in \Omega$

*Proof.*We proceed by induction on the number of nodes. Assume we have a set of nodes $D \subset \Omega, D \neq \emptyset$ with the following properties:

$$\lim_{n \to \infty} V_i^n = V_i^{\pi}, \ \forall i \in D \tag{235}$$

$$V_i^{\pi} > V_j^{\pi}, \ \forall i \in D, j \in \bar{D} \tag{236}$$

where $\bar{D} := \Omega - D$.

Furthermore, because of (235) there exists an event m_1 and $\delta > 0$ such that

$$V_i^n \leq V_i^\pi + \delta, \quad \forall n > m_1, \forall i \in D \tag{237}$$

For the induction basis we let $D = \{i \in \Omega : R_i = R_{max}\}$, and argue as follows. Equation (235) is satisfed because $V_i^n = R_{max}, \forall n, i \in D$; inequality (236) is true because $c_j > 0, \forall j \in \Omega$ and hence $V_j^\pi < R_{max}, j \in \bar{D}$.

We proceed with the induction step. Define

$$E := argmax_{i \in \bar{D}} V_i^\pi \tag{238}$$

Note that E is a non-empty set of nodes. Let node $i \in E$.

Claim 1

$$\limsup_{n \to \infty} V_i^n \leq V_i^\pi \tag{239}$$

Proof of Claim 1: See Appendix B. $\triangleleft$

Claim 2

$$\liminf_{n \to \infty} V_i^n \geq V_i^\pi \tag{240}$$

Proof of Claim 2: See Appendix C. $\triangleleft$

From Claim 1 and Claim 2 it follows that

$$\lim_{n \to \infty} V_i^n = V_i^\pi \tag{241}$$

The arguments leading to (241) can be made for any element $i \in E$. Hence we have shown that

$$\lim_{n \to \infty} V_i^n = V_i^\pi, \quad \forall i \in E \tag{242}$$

We now complete the induction step. Let $D' := D \cup E$ and $\bar{D}' := \Omega - D'$. We show that (235) and (236) are satisfied for D' and $\bar{D}'$.

Relation (242) and the induction assumption together mean that (235) is true for all nodes of D'. The definition of E in (238) with the induction assumption directly implies that (236) is true for D'.

This completes the induction argument, and the lemma is proved. $\square$

We are now ready to present the proof of Theorem 6.2.

Proof of Theorem 6.2

Proof of 1. Relation 1. follows directly from Lemma 6.15.

Proof of 2. Because of 1. there exists n_p after which the node values at each node i and its $\mathcal{N}(i)$ are in the order of some local index policy with an optimal associated index policy. At each event $n \geq n_p$ this local ordering can change, but it always corresponds to some optimal associated index policy. Hence, at each node i an optimal action is taken at each event $n \geq n_p$. Hence, the distributed policy is optimal.

Proof of 3. Because of 1. and Lemma 6.15, there exists n_p after which node values for each node i and its $\mathcal{N}(i)$ are in the order of the unique optimal local index policy. and there are no subsequent changes in this order. Subsequent to n_p, once a full round of node updates occurs, each node has its correct V_i^π and its correct local index policy, which subsequently do not change. □

Discussion

It is interesting to note the relation of Algorithm 5 to the results in [Gafn 81]. Major differences include the on-going nature of the node updates in Algorithm 5, and that node values are interpreted as the expected reward at a given node, whose estimates are constantly being updated by (203). The direction flipping action of [Gafn 81] corresponds to resetting in Algorithm 5. The proof of Theorem 6.2 is more difficult than the analogous result in [Gafn 81], due to the fact that in Algorithm 5 node values can change not just when being reset but also when normally computing updated estimates.

7 Conclusion

7.1 Summary

We have presented a network routing problem which uses a probabilistic local broadcast model for wireless transmission. We then presented results showing that an index policy is optimal for this problem. We extended the model to allow for transmission control, and showed that the index nature of the optimal policy remains unchanged. We then allowed time-varying system parameters in the model, and discussed conditions under which a time-varying index policy and a

time-varying priority policy are optimal. Finally, we presented three distributed implementations of the optimal routing policy for the original problem, and provided results on their convergence properties.

7.2 Future Research Directions

Future work in this area can take many forms. We summarize potential research on routing in ad hoc networks beyond that presented above.

Extensions to Analysis of Problem ($\mathbf{P_5}$), the Time-Varying System

The results achieved in Section 5.2 for Problem ($\mathbf{P_5}$) are partial results and limited to special classes of policies. A fuller accounting of optimal policies for the time-varying problem needs to be made. It would be interesting to consider what classes of policies are worth considering. Also study of special restrictions on the time-varying system (such as only time-varying reward) might lead to useful results.

Estimation of Channel Transmission Probabilities

The models of Section 3 assumed knowledge of the transmission probabilities. To actually implement the algorithms, methods to estimate these probabilities must be investigated. There is a complex interaction between the behavior of these estimation methods and how the resulting probabilities are used in the network. Numerous relevant questions arise, such as the dynamic behavior of the estimation procedure, and to what extent physical modeling of the transmission system should be incorporated into the estimation technique (as opposed to purely measurement-based approaches). Given the complex way the probability estimates affect the behavior of the specific distributed algorithm in use, formulation of criteria for a tractable probability estimation problem is very difficult.

Parameter Sensitivity Analysis

Because channel transmission probabilities must be estimated, there is always some error in the values. Other system parameters may also be only estimates, such as transmission energy cost or message reward value. We know from Theorem 3.1 that the optimal policy for Problem ($\mathbf{P_1}$) is an index policy for any value of these system parameters. But the node indices computed using system parameters in error will be different from the actual optimal indices. It is of great practical interest to determine how sensitive node indices are to errors in these system parameters.

Model Violation

We made a greatly simplifying assumption in Assumption 2.1 by assuming iid transmission. But even in the many cases where this assumption is mostly justified, there can be small violations. Successive channel transmissions can be *almost* independent, or network parameters can be *slightly* time-varying. It would be useful to know just how much the performance of the "optimal" index policy degrades when the iid assumption is violated in a minor way.

Markov Chain Channel

It is interesting to ask what happens when the channel model is no longer a fixed iid transition probability structure, but is modeled instead by a Markov Chain. This allows for correlation in time of transition events at the same node, relaxing the strict constraints of (8) or (9). In this way, more realistic channel models can be accommodated.

The Markov Chain channel model in general leads to a more complicated optimal policy than our Index Policy of Section 3.3. Further research is necessary into the nature of the optimal policy, and into what simplifying assumptions are useful.

General Distributed Algorithms

Algorithm 4, the DDP formulation, and Algorithm 5 are three different ways to compute the optimal policy for Problem ($\mathbf{P_3}$) in a distributed fashion. There exist other ways, each with particular strengths and weaknesses. It would be useful to relate this to a general theory of convergence for algorithms of this type.

Convergence Rate Analysis for Distributed Algorithms

To understand how algorithms like Algorithm 4 and Algorithm 5 behave in dynamic environments, it is necessary to have some understanding of their convergence rates. Analytic work in this area would be beneficial, especially work performed from a statistical viewpoint, and might have general applicability to distributed algorithm research.

Distributed Algorithm Limitations

Distance vector algorithms have some well-known limitations, such as convergence rate, the counting to infinity problem, and packet looping ([Kuro 00] p.252). We would like to determine the extent to which these limitations also exist in the distributed algorithms we presented in Section 6, and what algorithm modifications are possible to mitigate these effects. Distance vector algorithms have undergone a long development to deal with these problems, and we should mine the previous work for ideas where applicable.

Transmission Interference

Research into other methods for handling transmission interference in a distributed wireless network is also important. Our method in this paper has the merit of being amenable to analysis, but counting the penalty of interference as a fixed cost is simplistic. It would be a major achievement to find an analytic approach with a realistic transmission interference model that leads to a distributed optimal policy.

Multidestination

Problem ($\mathbf{P_3}$) defines the destination to be any one node from a set of nodes, the *anycast* problem. Another important problem is the *multicast* problem, which is to send the message to *all* the nodes in a given destination set. It would be interesting to determine the optimal multicast algorithm for our local broadcast model, Model **(M)**.

Random Cost

There may be cases where the message reception energy at each mobile is significant compared to transmission energy. In these cases it is useful to allow local neighbor addressing. But neighbor reception of a message transmission is random, so energy consumption, or cost, is random. To accommodate this, we would like to extend the model to allow for the cost of transmission to be a random variable.

On Demand Systems

We might try to find out if the stochastic local broadcast model we have considered in this paper can be used to any advantage in those ad hoc networks in which messages are rare relative to network topology change. To adequately analyze these networks, the cost of control messages must be included in the network model.

Simulation

It would be of great interest to see how the algorithms discussed here perform in a realistic ad hoc wireless simulation. Of particular interest is the range of system parameters, such as rate of network topology change, rate of message arrival, and operating environment, over which different algorithms perform better than others.

References

[Bert 82] D. BERTSEKAS, "Distributed Dynamic Programming", *IEEE Transactions on Automatic Control* **27**:3, p.610-616, June 1982

[Bert 89] D. BERTSEKAS AND J. TSITSIKLIS, *Parallel and Distributed Computation*, Prentice-Hall, 1989

[Bert 92] D. BERTSEKAS AND R. GALLAGER, *Data Networks*, Prentice-Hall, 1992

[Bert 95] D. BERTSEKAS, *Dynamic Programming and Optimal Control*, Athena Scientific, 1995

[Broc 98] J. BROCH, D. MALTZ, D. B. JOHNSON, Y. C. HU, J. JETCHEVA, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols", *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, October 25-30, 1998

[Chen 99] S. CHEN AND K. NAHRSTEDT "Distributed Quality-of-Service Routing in Ad Hoc Networks", *IEEE Journal on Selected Areas in Communications* **17**:8, p.1488-1505, August 1999

[Cors 95] M. CORSON AND A. EPHREMIDES, "A Distributed Routing Algorithm for Mobile Wireless Networks", *Wireless Networks* **1**, p. 61-81, 1995

[Diff 99] The IETF Differentiated Services Working Group homepage, http://www.ietf.org/html.charters/diffserv-charter.html

[Gafn 81] E. GAFNI AND D. BERTSEKAS "Distributed Algorithms for Generating Loop-Free Routes in Networks with Frequently Changing Topology", *IEEE Transactions on Communications* **29**:1, p.11-18, 1981

[Garc 97] J. GARCIA-LUNA-ACEVES AND S. MURTHY "A Path-Finding Algorithm for Loop-Free Routing", *IEEE/ACM Transactions on Networking*, v 5 n 1, p.148-160, Feb 1997

[Gare 79] M. GAREY AND D. JOHNSON *Computers and Intractability : a Guide to the Theory of NP-Completeness*, W. H. Freeman, 1979

[Garg 96] V. Garg and J. Wilkes, *Wireless and Personal Communications Systems*, Prentice-Hall, New Jersey, 1996

[Graf 98] C. Graff, M. Bereschinsky, M. Patel "Application of Mobile IP to Tactical Mobile Internetworking", *Proceedings of the 1998 IEEE Military Communications Conference*, v2, p.409-414, 1998

[Gudm 91] M. Gudmundson "Correlation Model for Shadow Fading in Mobile Radio Systems", *Electronics Letters*, p.2145-6, Nov. 7, 1991

[Haas 99] Z. Haas "Guest Editorial, Wireless Ad Hoc Networks", *IEEE Journal on Selected Areas in Communications* **17**:8, p.1329-1330, August 1999

[Hata 90] M. Hata "Empirical Formula for Propagation Loss in Land Mobile Radio Services", *IEEE Transactions on Vehicular Technology*, Vol. VT-29, No. 3, pp. 317-325, August 1980

[Iwat 99] A. Iwata, C. Chiang, G. Pei, M. Gerla, T. Chen "Scalable Routing Strategies for Ad Hoc Wireless Networks", *IEEE Journal on Selected Areas in Communications* **17**:8, p.1369-1379, August 1999

[Joa 99] M. Joa-Ng and I. Lu "A Peer-to-Peer Zone-Based Two-Level Link State Routing for Mobile Ad Hoc Networks", *IEEE Journal on Selected Areas in Communications* **17**:8, p.1415-1425, August 1999

[John 94] D. B. Johnson, "Routing in Ad Hoc Networks of Mobile Hosts", *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications*, p.158-163, December 1994

[Klim 74] G. Klimov, "Time Sharing Service Systems I", *Theory of Probability and its Applications*, **19**, p.532-551, 1974

[Kuma 86] P. Kumar and P. Varaiya, *Stochastic Systems: Estimation, Identification, and Adaptive Control*, Prentice-Hall, 1986

[Kuro 00] J. Kurose and K. Ross, *Computer Networking*, Addison-Wesley, 2000

[Kush 71] H. Kushner, *Introduction to Stochastic Control*, Holt, Rinehart and Winston, Inc. 1971

[Malk 93] G. S. MALKIN, "RIP Version 2: Carrying Additional Information", RFC 1388 Jan, 1993

[Malt 99] D. MALTZ, J. BROCH, J. JETCHEVA, D. JOHNSON "The Effects of On-Demand Behavior in Routing Protocols for Multihop Wireless Ad Hoc Networks", *IEEE Journal on Selected Areas in Communications* **17**:8, p.1439-1453, August 1999

[McDo 99] A. MCDONALD AND T. ZNATI "A Mobility-Based Framework for Adaptive Clustering in Wireless Ad Hoc Networks", *IEEE Journal on Selected Areas in Communications* **17**:8, p.1466-1487, August 1999

[Merl 79] P. MERLIN AND A. SEGALL "A Failsafe Distributed Routing Protocol", *IEEE Transactions on Communications* **27**:9, p.1280-1287, September 1979

[Moy 91] J. MOY, "OSPF Version 2", RFC 1247 July, 1991

[Okum 68] T. OKUMURA, E. OHMORI, AND K. FUKUDA, "Field Strength and Its Variability in VHF and UHF Land Mobile Service", *Review Electrical Communication Laboratory*, Vol. 16, No. 9-10, pp. 825-873, September-October 1968

[Park 97] V. PARK AND M. CORSON, "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks", *Proceedings of INFOCOM '97*, p.1405-1413, April 1997

[Pear 99] M. PEARLMAN AND Z. HAAS "Determining the Optimal Configuration for the Zone Routing Protocol", *IEEE Journal on Selected Areas in Communications* **17**:8, p.1395-1414, August 1999

[Perk 94] C. PERKINS AND P. BHAGWAT "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers", *Proceedings of the SIGCOMM '94 Conference on Communications Architectures, Protocols and Applications*, p. 234-244, August 1994

[Perk 97] C. PERKINS, E. ROYER, S. DAS, "Ad Hoc On-Demand Distance Vector (AODV) Routing", IETF MANET Working Group, March 2000.

[Purs 93] M. PURSLEY AND H. RUSSELL "Network Protocols for Frequency-Hop Packet Radios with Decoder Side Information", *IEEE Journal on Selected Areas in Communications*, p.612-621, May 1994

[Purs 99] M. PURSLEY, H. RUSSELL, P. STAPLES "Routing for Multimedia Traffic in Wireless Frequency-Hop Communication Networks", *IEEE Journal on Selected Areas in Communications* **17**:5, p.784-792, May 1999

[Rapp 96] S. RAPPAPORT, *Wireless Communications: Principles and Practice*, Prentice-Hall, New Jersey, 1996

[Rodo 99] V. RODOPLU AND T. MENG "Minimum Energy Mobile Wireless Networks", *IEEE Journal on Selected Areas in Communications* **17**:8, p.1333-1344, August 1999

[Ross 83] S. ROSS, *Introduction to Stochastic Dynamic Programming*, Academic Press, 1983

[Rudi 76] W. RUDIN, *Principles of Mathematical Analysis*, McGraw-Hill, 1976

[Samp 97] S. SAMPEI, *Applications of Digital Wireless Technologies to Global Wireless Communications*, Prentice-Hall, New Jersey, 1997

[Siva 99] R. SIVAKUMAR, P. SINHA, V. BHARGHAVAN "CEDAR: A Core-Extraction Distributed Ad Hoc Routing Algorithm", *IEEE Journal on Selected Areas in Communications* **17**:8, p.1454-1465, August 1999

[Wies 98] J. WIESELTHIER, G. NGUYEN, A. EPHREMIDES, "Multicasting in Energy-Limited Ad-Hoc Wireless Networks" *Proceedings of the 1998 IEEE Military Communications Conference*, v.3, p.723-729, 1998

[Wies 00] J. WIESELTHIER, G. NGUYEN, A. EPHREMIDES, "Algorithms for Bandwidth-Limited Energy-Efficient Wireless Broadcasting and Multicasting", *Proceedings of the 2000 IEEE Military Communications Conference*, Los Angeles, October 2000

[Wilf 94] H. WILF, *Generatingfunctionology*, 2nd ed, Academic Press, 1994

Appendix A
Proof of Lemma 6.7

*Proof.*Let j be any neighbor of i. Fix all the components of $\mathbf{W}^{i-}$ except for that of j, where for ease of notation we assume $j < i$. The following arguments with the obvious notation change prove the result for $j > i$ as well. Since the components of $\mathbf{W}^{i-}$ are assumed fixed except for j,

we can think of f_i as being parameterized by $\mathbf{W}^{ij-}$, with only one argument W_j as a free variable. We then write this function as $f_i^{\mathbf{W}^{ij-}}(W_j)$. Note then that we can write (162) as

$$f_i^{\mathbf{W}^{ij-}}(W_j) = A_{\mathbf{W}^{i-}} + B_{\mathbf{W}^{i-}} \cdot W_j \tag{243}$$

where $A_{\mathbf{W}^{i-}}$ and $B_{\mathbf{W}^{i-}}$ are defined as

$$A_{\mathbf{W}^{i-}} := \frac{-c_i + \sum_{S \supset \{i\}: \pi(S) \neq i,j} P^i(S|i) W_{\pi(S)}}{\sum_{S \supset \{i\}: \pi(S) \neq i} P^i(S|i)} \tag{244}$$

$$B_{\mathbf{W}^{i-}} := \frac{\sum_{S \supset \{i\}: \pi(S) = j} P^i(S|i)}{\sum_{S \supset \{i\}: \pi(S) \neq i} P^i(S|i)} \tag{245}$$

Note that implicit in the definitions of $A_{\mathbf{W}^{i-}}$ and $B_{\mathbf{W}^{i-}}$ is the local policy π, which is also a function of $\mathbf{W}^{i-}$(which includes W_j). But over any range of values of W_j for which π doesn't change, $A_{\mathbf{W}^{i-}}$ and $B_{\mathbf{W}^{i-}}$ are constant. Thus, over such a range $f_i^{\mathbf{W}^{ij-}}(W_j)$ is a linear function of W_j. Further, from (245) and the fact that

$$\{S \supset \{i\} : \pi(S) = j\} \subseteq \{S \supset \{i\} : \pi(S) \neq i\}, \tag{246}$$

we have that

$$0 \leq B_{\mathbf{W}^{i-}} \leq 1 \tag{247}$$

As W_j varies at values for which $W_j < f_i^{\mathbf{W}^{ij-}}(W_j)$, node j is given lower priority than i by π, and hence the value of W_j has no effect on $f_i^{\mathbf{W}^{ij-}}(W_j)$. As W_j varies at values for which $W_j \geq f_i^{\mathbf{W}^{ij-}}(W_j)$, policy π can change at only two types of points.

1. $W_j = f_i^{\mathbf{W}^{ij-}}(W_j)$.
2. $W_j = W_k$, for some $k \neq i, j$, and not type 1

A type 1 point is where node j flips priority with node i. Type 2 points are where node j flips priority with node k, but not with node i.

We first examine type 1. Let $W^* = f_i^{\mathbf{W}^{ij-}}(-\infty)$. W^* is the value when π ranks j lower than i. We then have that

$$f_i^{\mathbf{W}^{ij-}}(W_j) = W^* = \frac{-c_i + \sum_{S \supset \{i\}: \pi(S) \neq i,j} P^i(S|i) W_{\pi(S)}}{\sum_{S \supset \{i\}: \pi(S) \neq i,j} P^i(S|i)}, \ \forall W_j \leq W^* \tag{248}$$

and so the function is constant (i.e. the slope is 0).

Consider a point W_j in a linear region, as in (243) above with a fixed π. We then have $f_i^{\mathbf{W}^{ij-}}(W_j) = A_{\mathbf{W}^{i-}} + B_{\mathbf{W}^{i-}} \cdot W_j$. Let U^* be any point on this curve such that $f_i^{\mathbf{W}^{ij-}}(U^*) = U^*$, if such a point exists. We thus have $A_{\mathbf{W}^{i-}} + B_{\mathbf{W}^{i-}} \cdot U^* = U^*$, which simplifies to

$$\begin{aligned} U^* &= \frac{A_{\mathbf{W}^{i-}}}{1 - B_{\mathbf{W}^{i-}}} \\ &= \frac{-c_i + \sum_{S \supset \{i\}: \pi(S) \neq i,j} P^i(S|i) W_{\pi(S)}}{\sum_{S \supset \{i\}: \pi(S) \neq i,j} P^i(S|i)} \\ &= W^* \end{aligned} \tag{249}$$

where the final equality follows from (248). That is, a point of type 1 occurs when $W_j = W^*$, the value function for i when j is of lower priority than i in π. From (248) and (249) we conclude that such a U^* exists and is the unique point where a line of the form (243) intersects the line of slope 0 at W^*. Hence, continuity and piecewise linearity with non-decreasing slope is maintained in this case.

Now we examine a point of type 2. We only need to consider nodes k of priority higher than i under the current π (i.e. $k >_\pi i$) since j is assumed to be at a higher priority than i. Let $A'_{\mathbf{W}^{i-}}$ and $B'_{\mathbf{W}^{i-}}$ be the intercept and slope of (243) for the case where $W_j = W'_j$ is slightly less than W_k, and let π' be the policy in this case. Similarly, let $A''_{\mathbf{W}^{i-}}$ and $B''_{\mathbf{W}^{i-}}$ be these values for the case where $W_j = W''_j$ is slightly greater than W_k, and let π'' be this policy. We need to show that the two lines, described by (243), $A'_{\mathbf{W}^{i-}}$, $B''_{\mathbf{W}^{i-}}$, and (243), $A''_{\mathbf{W}^{i-}}$, $B''_{\mathbf{W}^{i-}}$, intersect at the point where $W_j = W_k$.

Since $j, k >_\pi i$, we have

$$\sum_{S \supset \{i\}: \pi'(S) \neq i} P^i(S|i) = \sum_{S \supset \{i\}: \pi''(S) \neq i} P^i(S|i) := G \tag{250}$$

Relation (250) is valid because π' and π'' differ only in that they switch the respective priorities of j and k. Hence, the set of states above i in priority remain identical.

We can then write (243) as

$$\begin{aligned} f_i^{\mathbf{W}^{ij-}}(W'_j) = \frac{1}{G}(-c_i + & \sum_{S \supset \{i\}: \pi'(S) \neq i,j,k} P^i(S|i) W_{\pi'(S)} + \\ & \sum_{S \supset \{i\}: \pi'(S) = j} P^i(S|i) W'_j + \\ & \sum_{S \supset \{i\}: \pi'(S) = k} P^i(S|i) W_k) \end{aligned} \tag{251}$$

and similarly for W''_j.

From the definitions of π' and π'', we have

$$\sum_{S\supset\{i\}:\pi'(S)\neq i,j,k} P^i(S|i) = \sum_{S\supset\{i\}:\pi''(S)\neq i,j,k} P^i(S|i) \tag{252}$$

Hence,

$$\begin{aligned} f_i^{\mathbf{W}^{ij-}}(W_j'') - f_i^{\mathbf{W}^{ij-}}(W_j') = \\ \tfrac{1}{G}(\textstyle\sum_{S\supset\{i\}:\pi'(S)=j} P^i(S|i)W_j' + \sum_{S\supset\{i\}:\pi'(S)=k} P^i(S|i)W_k - \\ \textstyle\sum_{S\supset\{i\}:\pi''(S)=j} P^i(S|i)W_j'' - \sum_{S\supset\{i\}:\pi''(S)=k} P^i(S|i)W_k) \end{aligned} \tag{253}$$

When $W_j' = W_j'' = W_k$, (253) becomes

$$\begin{aligned} f_i^{\mathbf{W}^{ij-}}(W_j'') - f_i^{\mathbf{W}^{ij-}}(W_j') &= \frac{W_k}{G}\Big(\sum_{S\supset\{i\}:\pi'(S)=j} P^i(S|i) + \sum_{S\supset\{i\}:\pi'(S)=k} P^i(S|i) - \\ &\quad \sum_{S\supset\{i\}:\pi''(S)=j} P^i(S|i) - \sum_{S\supset\{i\}:\pi''(S)=k} P^i(S|i)\Big) \\ &= 0 \end{aligned} \tag{254}$$

where the terms in parenthesis sum to 0 because π' and π'' are identical except for the switching of the adjacent priorities of j and k, so that the sum over all states where either of these two nodes are the highest priority is the same for both π' and π''.

Because the linear functions described by (243), $A'_{\mathbf{W}^{i-}}$, $B''_{\mathbf{W}^{i-}}$, and (243), $A''_{\mathbf{W}^{i-}}$, $B''_{\mathbf{W}^{i-}}$, intersect at $W_j' = W_j'' = W_k$, $f_i^{\mathbf{W}^{ij-}}(W_j)$ is piecewise linear over the region of W_j where π' and π'' are used. Furthermore, $B''_{\mathbf{W}^{i-}} \geq B'_{\mathbf{W}^{i-}}$ because the denominators are the same in both cases, and $B''_{\mathbf{W}^{i-}}$ has a larger numerator than $B'_{\mathbf{W}^{i-}}$, because $j >_{\pi''} k$ and $j <_{\pi'} k$. Hence the slope is monotonically non-decreasing in this region. □

Appendix B

Proof of Claim 1

Proof. To prove (239) it suffices to show that

$$\limsup_{n\to\infty} Z_n \leq V_i^{\pi} \tag{255}$$

where

$$Z_n := \max\left\{\max_{j\in\bar{D}} V_j^n, \max_{j,k\in\bar{D}} V_{j,k}^n\right\} \tag{256}$$

To prove (255) we show that for any $\epsilon > 0$ there is an event m such that

$$Z_n \le V_i^\pi + \epsilon, \ \ \forall n \ge m \tag{257}$$

Consider a node update at event n for any $j \in \bar{D}$. Define

$$A_j^n := \{k \in \mathcal{N}(j) \cap D : V_{j,k}^{n-1} > V_j^{n-1}\}, \quad j \in \bar{D} \tag{258}$$

A_j^n is the set of neighbor nodes in D which are ranked at higher priority than j in the update at n. A_j^n may or may not be empty. Define

$$F_n := \{j \in \bar{D} : A_j^n \neq \emptyset\} \tag{259}$$

$$\bar{F}_n := \{j \in \bar{D} : A_j^n = \emptyset\} \tag{260}$$

We first establish the following fact (recall that m_1 is defined in (237)).

Fact 1 *When $Z_{n-1} \le V_i^\pi + \epsilon_1, 0 < \epsilon_1 < 1$, then at event $n > m_1$,*

$$Z_n \le Z_{n-1} \qquad \textit{if event } n \textit{ is a node transmission} \tag{261}$$

$$V_j^n \le Z_{n-1} \qquad \textit{if event } n \textit{ is an update at node } j,\ j \in \bar{F}_n \tag{262}$$

$$V_j^n \le V_i^\pi + \epsilon_1 \qquad \textit{if event } n \textit{ is an update at node } j,\ j \in F_n \tag{263}$$

Proof of Fact 1: If event n is a node transmission, then (261) follows from the definition of Z_n.

If event n is an update at node j, $j \in \bar{F}_n$, we consider three cases:

1. If the update gives

$$V_j^n = R_i \tag{264}$$

then

$$V_j^n = R_i \le V_j^{n-1} \le Z_{n-1} \tag{265}$$

2. If the update is a reset for $j \in \bar{F}_n$, then

$$V_j^{n-1} \ge V_{j,k}^{n-1}, \ \forall k \in \mathcal{N}(j) \tag{266}$$

Upon resetting, by the specification of Algorithm 5 node j is ranked lower than at least one neighbor, say $k \in \mathcal{N}(j)$. Then by Lemma 6.13 we obtain

$$V_j^n \leq V_{j,k}^{n-1} - c_j \leq V_j^{n-1} - c_j \leq Z_{n-1} - c_j \leq Z_{n-1} \tag{267}$$

3. If the update is a rank update, then from the definitions of F_n, Z_n, and Lemma 6.13 it follows that

$$\begin{aligned} V_j^n &\leq \max_{k \in \bar{D} \cap \mathcal{N}(j)} V_{j,k}^{n-1} - c_j & (268) \\ &\leq Z_{n-1} - c_j \leq Z_n & (269) \end{aligned}$$

From (265), (267) and (269) it follows that (262) is true.

If event n is an update at node j, $j \in F_n$, then $A_j^n \neq \emptyset$ and the update is not a node reset. Let π_j^n be the index ranking local to j which ranks the nodes of A_j^n in the order of their value functions V_k^π, $k \in A_j^n$, and ranks j next. Since $A_j^n \subseteq D$ because of (237), we get

$$V_{j,k}^n \leq V_k^\pi + \delta, \quad \forall n > m_1, \forall k \in A_j^n, \delta > 0 \tag{270}$$

Hence by Lemma 6.14 (inequality (230)) we get

$$V_j^{\pi_j^n} \leq V_j^\pi + \epsilon_1 \leq V_i^\pi + \epsilon_1 \tag{271}$$

We use (271) together with Lemma 6.12 to prove (263). We let $A = A_j^n$, $\tilde{\pi} = \pi_j^n$, $B \subseteq \mathcal{N}(j) \cap \bar{D}$ in Lemma 6.12. Then Lemma 6.12 and (271) give

$$\begin{aligned} V_j^n &\leq (1-a) V_j^{\pi_j^n} + a Y_1 & (272) \\ &\leq (1-a) V_j^{\pi_j^n} + a Z_{n-1} & (273) \\ &\leq (1-a)(V_i^\pi + \epsilon_1) + a Z_{n-1} & (274) \\ &\leq V_i^\pi + \epsilon_1 & (275) \end{aligned}$$

where

$$Y_1 := \max_{k \in B} V_{j,k}^{n-1} \tag{276}$$

and a is defined by relations similar to (214), (215), and (220).

The inequality in (272) follows from Lemma 6.12. The inequality in (273) follows from the definitions of Y_1 and Z_{n-1}. The inequality in (274) follows from (271). The inequality in (275) follows from the assumption $Z_{n-1} \leq V_i^\pi + \epsilon_1$ made in Fact 1.

Inequalities (272)-(275) prove (263). The proof of Fact 1 is now complete. $\triangleleft$

We proceed with the proof of Claim 1 by establishing the following fact.

Fact 2 *When $Z_{n-1} > V_i^\pi + \epsilon_1$, $0 < \epsilon_1 < \epsilon$, then at event $n > m_1$,*

$$\begin{aligned} Z_n &\leq Z_{n-1} && \text{if event } n \text{ is a node transmission} && (277)\\ V_j^n &\leq \max\{Z_{n-1} - c_j, R_j\} && \text{if event } n \text{ is an update at node } j,\ j \in \bar{F}_n && (278)\\ V_j^n &\leq \max\{(1-a_{max})(V_i^\pi + \epsilon_1) + a_{max} Z_{n-1}, R_j\} \\ & && \text{if event } n \text{ is an update at node } j,\ j \in F_n && (279) \end{aligned}$$

Proof of Fact 2: If event n is a node transmission, then (277) follows from the definition of Z_n

If event n is an update at node j, $j \in \bar{F}_n$, we consider three cases:

1. The update gives
$$V_j^n = R_i \tag{280}$$

2. The update is a reset for $j \in \bar{F}_n$. Then by the same arguments as those leading to (267) we obtain
$$V_j^n \leq Z_{n-1} - c_j \tag{281}$$

3. The update is a rank update for node $j \in \bar{F}_n$. Then by arguments that are the same as those leading to (269) we obtain
$$V_j^n \leq Z_{n-1} - c_j \tag{282}$$

From (280)-(282) it follows that

$$V_j^n \leq \max\{Z_{n-1} - c_j, R_j\} \tag{283}$$

which is precisely (278).

If event n is an update at node j, $j \in F_n$, then if the optimal action is not to retire, by arguments that are the same as those leading to (274) we obtain

$$V_j^n \leq (1-a)(V_i^\pi + \epsilon_1) + aZ_{n-1} \tag{284}$$

where a is again defined by relations similar to (214), (215), and (220).

Furthermore,

$$a_{max} := \max_{i \in \Omega, \tilde{\pi}, \hat{\pi}} a(i, \tilde{\pi}, \hat{\pi}) \geq a \tag{285}$$

and

$$Z_{n-1} > V_i^\pi + \epsilon_1 \tag{286}$$

by assumption.

Because of (285) and (286), (284) gives

$$\begin{aligned} V_j^n &\leq (1-a)(V_i^\pi + \epsilon_1) + aZ_{n-1} \\ &\leq (1-a_{max})(V_i^\pi + \epsilon_1) + a_{max}Z_{n-1} \end{aligned} \tag{287}$$

If the optimal action is to retire, then

$$V_j^n = R_i \tag{288}$$

Combining (287) and (288), we obtain

$$V_j^n \leq \max\{(1-a_{max})(V_i^\pi + \epsilon_1) + a_{max}Z_{n-1}, R_j\} \tag{289}$$

which is precisely (279).

This completes the proof of Fact 2. $\triangleleft$

We use Fact 1 and Fact 2 to complete the proof of Claim 1. We consider two cases.

Case 1 There is an event $l > m_1$ where $Z_l \leq V_i^\pi + \epsilon_1$.

Under Case 1, relations (261), (262), and (263) together imply that

$$Z_n \leq V_i^\pi + \epsilon_1 \quad \forall n \geq l \tag{290}$$

Because inequality (257) follows from (290), we have shown that Claim 1 is true under Case 1.

Case 2 $Z_n > V_i^\pi + \epsilon,\ \forall n > m_1$

We claim that under Case 2 the following inequality (291) is true for any event, update or transmission, of any node $j \in \bar{D}$ at any $n > m_1$.

$$V_j^n \leq Z_{n-1},\ \ n > m_1 \tag{291}$$

We prove (291) as follows.

1. Suppose there is an update at $j \in \bar{F}_n$. Then

   (a) If $\max\{Z_{n-1} - c_j, R_j\} = R_j$, then from (278) we have $V_j^n \leq R_j$. Since $V_j^n \geq R_j$ by its definition, it follows that

   $$V_j^n = R_j \tag{292}$$

   And since $R_j \leq Z_{n-1}$, (291) follows.

   (b) If $\max\{Z_{n-1} - c_j, R_j\} = Z_{n-1} - c_j$, then from (278) we have

   $$V_j^n \leq Z_{n-1} - c_j < Z_{n-1} \tag{293}$$

2. Suppose there is an update at $j \in F_n$. Then

   (a) If $\max\{(1 - a_{max})(V_i^\pi + \epsilon_1) + a_{max} Z_{n-1}, R_j\} = R_j$, then from (279) we have $V_j^n \leq R_j$. Since $V_j^n \geq R_j$ by its definition, it follows that

   $$V_j^n = R_j \tag{294}$$

   And since $R_j \leq Z_{n-1}$, (291) follows.

   (b) If $\max\{(1 - a_{max})(V_i^\pi + \epsilon_1) + a_{max} Z_{n-1}, R_j\} = (1 - a_{max})(V_i^\pi + \epsilon_1) + a_{max} Z_{n-1}$, then from (279) we have

   $$\begin{aligned} V_j^n &\leq (1 - a_{max})(V_i^\pi + \epsilon_1) + a_{max} Z_{n-1} \\ &< (1 - a_{max}) Z_{n-1} + a_{max} Z_{n-1} \qquad (295) \\ &= Z_{n-1} \qquad (296) \end{aligned}$$

   Inequality (295) follows because $Z_n > V_i^\pi + \epsilon_1, \forall n > m_1$.

3. Suppose there is a transmission from $j \in \bar{D}$. Then (291) follows from (277). This completes the proof of (291).

From (291) it follows that

$$Z_n \leq Z_{n-1}, \;\; n > m_1 \tag{297}$$

From (297) we also have

$$Z_{n-1} \leq Z_{m_1}, \;\; n > m_1 \tag{298}$$

We complete the proof of Claim 1 under Case 2 by establishing the following two facts.

Fact 3 *For all $n > m_1$ and $j \in \bar{D}$ we have*

$$V_j^n \leq max\left\{(1 - a_{max})(V_i^\pi + \epsilon_1) + a_{max} Z_{m_1},\; Z_{m_1} - c_j\right\} \tag{299}$$

Proof of Fact 3: Using (298) in (279), we have for an update at $j \in F_n$ that

$$V_j^n \leq \max\left\{(1 - a_{max})(V_i^\pi + \epsilon_1) + a_{max} Z_{m_1}, R_j\right\}, \;\; n > m_1 \tag{300}$$

Because $Z_{m_1} > V_i^\pi + \epsilon_1 \geq R_j, \forall j \in \bar{D}$, we can write (300) as

$$V_j^n \leq (1 - a_{max})(V_i^\pi + \epsilon_1) + a_{max} Z_{m_1}, \;\; n > m_1 \tag{301}$$

Using (298) in (278), we obtain for $n > m_1$

$$V_j^n \leq \max\left\{Z_{m_1} - c_j, R_j\right\} \tag{302}$$

Using (301), (302) and the fact that $R_j \leq V_j^n \leq (1 - a_{max})(V_i^\pi + \epsilon_1) + a_{max} Z_{m_1}$, we have for $j \in \bar{D}$ and $n > m_1$ that

$$V_j^n \leq max\left\{(1 - a_{max})(V_i^\pi + \epsilon_1) + a_{max} Z_{m_1},\; Z_{m_1} - c_j\right\} \tag{303}$$

◁

Fact 4 *There is an event $m_2 > m_1$ where*

$$Z_{m_2} \leq (1 - a_{max})(V_i^\pi + \epsilon_1) + a_{max} Z_{m_1}, \tag{304}$$

Proof of Fact 4: We prove (304) by contradiction. Assume (304) is not true, which means that

$$Z_n > (1 - a_{max})(V_i^\pi + \epsilon_1) + a_{max} Z_{m_1}, \ \ \forall n > m_1 \tag{305}$$

By the definition of Z_n, (299), (305) and the fact that node transmissions and updates occur infinitely often (so that for all $j, k \in \bar{D}$, V_j^n satisfying (299) becomes $V_{k,j}^{n'}$ at some $n' > n$) there exists $l > m_1$ such that

$$Z_l \leq Z_{m_1} - \min_{j \in \bar{D}} c_j \tag{306}$$

By repeating the argument leading to (306), we conclude that for large enough n, Z_n can become arbitrarily small, and this contradicts (305). $\triangleleft$

Repeating the argument leading to (304), and noting that $a_{max} < 1$, we conclude that there is an event m_p such that

$$Z_n \leq V_i^\pi + \epsilon_1 + \epsilon_2, \ \ \forall n \geq m_p \tag{307}$$

Relation (307) proves (257) and hence Claim 1 under Case 2.

We have proved Claim 1 under both Case 1 and Case 2, and this completes the proof of Claim 1. $\square$

Appendix C
Proof of Claim 2

*Proof.*To prove (240) we show that for any $\epsilon > 0$ there is an event m such that

$$V_i^n \geq V_i^\pi - \epsilon, \ \ \forall n \geq m \quad (i \in E) \tag{308}$$

If $V_i^\pi = R_i$, then (308) holds. Henceforth, assume $V_i^\pi > R_i$.

Define $A_i := \mathcal{N}(i) \cap D$. A_i is not empty, reasoned as follows. Because $V_i^\pi > R_i$, a rank update with correct neighbor values and optimal local policy gives V_i^π. Lemma 6.13 and the fact that $c_i > 0$ ensure that $V_i^\pi < V_j^\pi$ for some $j \in \mathcal{N}(i)$. Because of (236) and (238), $j \in D$. Hence, $j \in A_i$.

Let π_i be the index policy local to i which ranks the nodes of A_i in the order of their value functions $V_j^\pi, j \in A_i$, and then ranks i next. Note that due to (238), π_i is the optimal local policy for node i. We write $V_i^n|_{\pi_i}$ to indicate the update value computed at node i under local policy π_i.

Let $\epsilon > 0$, and choose any ϵ_1 and ϵ_2 such that $\epsilon_1 > 0, \epsilon_2 > 0$ and $\epsilon = \epsilon_1 + \epsilon_2$.

We establish the following fact.

Fact 5 *There is an event $m_2 > m_1$ such that for $n \geq m_2$ we have*

$$V_i^n \geq (1 - a_n)(V_i^\pi - \epsilon_1) + a_n V_i^{n-1} \tag{309}$$

where $0 \leq a_n \leq a_{max}$, $n \geq m_2$, m_1 is defined in (237), and a_{max} is given by (234).

Proof of Fact 5: By Lemma 6.14 (229) there is a $\delta > 0$ such that when $V_{i,j}^{n-1} \geq V_j^\pi - \delta, \forall j \in A_i$, we have

$$V_i^n|_{\pi_i} \geq V_i^\pi - \epsilon_1 \tag{310}$$

Let $m' > m_1$ be an event such that

$$V_{i,k}^n \geq V_k^\pi - \delta, \quad \forall n \geq m', k \in D \tag{311}$$

Event m' exists because of (235) and because transmissions occur infinitely often. Hence (310) holds for $n \geq m'$.

By Claim 1, (235) and (236), there is an event $m'' > m'$ such that $\forall j \in \bar{D}$ we have

$$V_j^n < V_k^n, \ \ \forall n \geq m'', \forall k \in D \tag{312}$$

Because of (312) and the fact that transmissions occur infinitely often, there is an event $m_2 > m''$ such that $\forall j \in \bar{D}$ we have

$$V_j^n < V_{j,k}^n, \forall n \geq m_2, \forall k \in D \tag{313}$$

Consider a node update at event $n > m_2$ at node i. To apply Lemma 6.12, we note that due to (313) an update at i will rank the nodes of A_i highest, then possibly rank other neighbors above i. We identify $A = A_i$ and let $B = \{j \in \bar{D} : V_{i,j}^{n-1} > V_i^{n-1}\}$. Define $Y_2 := \min_{j \in B} V_{i,j}^{n-1}$. By Lemma 6.12, at each update $n > m_2$ there exists $0 \leq a_n < a_{max}$ such that

$$\begin{aligned} V_i^n &\geq (1 - a_n)V_i^n|_{\pi_i} + a_n Y_2 && (314) \\ &\geq (1 - a_n)V_i^n|_{\pi_i} + a_n V_i^{n-1} && (315) \\ &\geq (1 - a_n)(V_i^\pi - \epsilon_1) + a_n V_i^{n-1} && (316) \end{aligned}$$

Relation (315) follows from and the definitions of B and Y_2. Relation (316) follows from (310). $\triangleleft$

We use Fact 5 to prove Claim 2 by considering two cases.

Case 1 There is an event $l > m_2$ where $V_i^l \geq V_i^\pi - \epsilon_1$.

When at some update event $n > m_2$ we have $V_i^{n-1} \geq V_i^\pi - \epsilon_1$, from (316) we obtain

$$V_i^n \geq V_i^\pi - \epsilon_1 \tag{317}$$

Relation (317) implies that if the Case 1 condition is met at $l > m_2$, it follows by (316) that

$$V_i^n \geq V_i^\pi - \epsilon_1, \ \ \forall n \geq l \tag{318}$$

Relation (318) proves (308). Hence Claim 2 is proved for Case 1.

Case 2 $\quad V_i^n < V_i^\pi - \epsilon_1, \ \forall n > m_1$

Under Case 2, we use the facts that $V_i^{n-1} < V_i^\pi - \epsilon_1, \forall n > m_2 + 1, \quad 0 \leq a_n \leq a_{max}, \forall n > m_2$ and (316) to obtain

$$V_i^n \geq (1 - a_{max})(V_i^\pi - \epsilon_1) + a_{max} V_i^{n-1}, \ \ \forall n > m_2 + 1 \tag{319}$$

Let x_i^n be defined as follows.

$$x_i^{m_2+1} = V_i^{m_2+1} \tag{320}$$

$$x_i^n = (1 - a_{max})(V_i^\pi - \epsilon_1) + a_{max} x_i^{n-1}, \ \ \forall n > m_2 + 1 \tag{321}$$

Using (319), (320), and (321), and the fact that $a_{max} \geq 0$ with an inductive argument, we have that

$$V_i^n \geq x_i^n, \ \ \forall n \geq m_2 + 1 \tag{322}$$

From (321) and the fact that $0 \leq a_{max} < 1$, we have that

$$\lim_{n \to \infty} x_i^n = V_i^\pi - \epsilon_1 \tag{323}$$

From (322) and (323), we conclude that

$$\lim_{n \to \infty} V_i^n \geq V_i^\pi - \epsilon_1 \tag{324}$$

Relation (324) implies that there exists an event $m_3 > m_2 + 1$ such that

$$V_i^n \geq V_i^\pi - \epsilon_1 - \epsilon_2, \quad \forall n \geq m_3 \tag{325}$$

Relation (325) proves (308). Hence Claim 2 is proved for Case 2. □