

2)

Hyphos: A Self-Organizing, Wireless Network

Robert Dunbar Poor

Submitted to the Program in Media Arts and Sciences
School of Architecture and Planning
in partial fulfillment of the requirements for the degree of
Master of Science at the Massachusetts Institute of Technology

June 1997

© 1997 Massachusetts Institute of Technology. All Rights Reserved.

Author

Program in Media Arts & Sciences
May 9, 1997

Certified by

Michael J. Hawley
Assistant Professor of Media Arts & Sciences
Thesis Supervisor

Alex Dreyfoos Jr. (1954) Career Development Professor of Media Arts & Sciences

Accepted by

Stephen A. Benton
Chair

Departmental Committee on Graduate Students
Program in Media Arts and Sciences

MASSACHUSETTS INSTITUTE
OF TECHNOLOGY

DEC 02 1997
LIBRARIES

Hyphos:

A Self-Organizing, Wireless Network

Robert Dunbar Poor

Submitted to the Program in Media Arts and Sciences

School of Architecture and Planning

on May 9, 1997

in partial fulfillment of the requirements for the degree of

Master of Science at the Massachusetts Institute of Technology

Abstract

Everyday objects are beginning to find expression in digital networks, creating a new family of objects that displays personalized behavior and memory. But how can you weave a thousand objects in one room into the Web? How can you connect them together without a tangle of wires, a burden of battery packs, or a Ph.D. in network administration?

The goal of this work is to develop *Hyphos*¹, a wireless network for interconnecting thousands of everyday objects. My thesis is that by using very short-range transceivers and relaying messages among the nodes, a new class of network emerges: a Hyphos network is self-organizing with a low cost per node; transmissions are tightly localized resulting in high bandwidth and low power consumption; fully distributed routing and control assures robust communication in the face of changes to the network topology.

Thesis Advisor:

Michael J. Hawley

Assistant Professor of Media Arts & Sciences

Alex Dreyfoos Jr. (1954) Career Development Professor of Media Arts & Sciences

This research was sponsored by the Things That Think Consortium. The author gratefully thanks the Motorola Fellows program for its support.

1. **hy•phos** \ˈhi-fōs\ n [Greek *hyphos* web]

Hyphos:

A Self-Organizing, Wireless Network

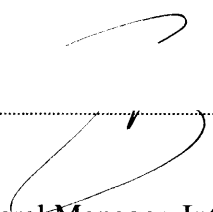
Robert Dunbar Poor

The following people have served as readers for this thesis:

Reader

Andrew Lippman
Associate Director
M.I.T. Media Laboratory

Reader



Vice President and General Manager, Internet Technology Laboratory
Intel Corporation

Steven McGeady
Visiting Fellow
Intel Corporation

Contents

Contents	7
List of Figures	9
List of Tables.....	11
1.0 Introduction	13
1.1 What if network connections were smaller?	
1.2 A thousand connections in one room	
1.3 About this thesis	
1.4 Some application scenarios	
2.0 Context	19
2.1 Scope of this work	
2.2 Design Goals	
2.3 Approach	
2.4 Assumptions	
2.5 Prior Work	
3.0 System Description	31
3.1 System Overview	
3.2 Reference Model	
3.3 Message Format	
3.4 Physical Layer	
3.5 Data Link Layer	
3.6 Medium Access Layer	
3.7 Network Layer	
3.8 Transport Layer	
4.0 Simulation and Results	41
4.1 Components of the Simulator	
4.2 The Tests	
4.3 Simulation Assumptions	
4.4 load: when does Hyphos hit the wall?	
4.5 density: how well does Hyphos scale?	
4.6 motion: how do mobile nodes affect performance?	
4.7 radius: what are the effects of adjusting transmit radius	
4.8 chatter: how do multiple dialogs affect performance?	
4.9 server: how does one server / many clients perform?	
4.10 Performance Summary	

5.0	A revised transmitter/receiver model	69
5.1	Modelling Attenuation and Noise	
5.2	Implications	
5.3	The revised transceiver model	
5.4	The Results	
5.5	load redux: when does the network hit the wall?	
5.6	chatter redux: multiple dialogs	
5.7	server redux: a client/server simulation	
6.0	Conclusions	83
6.1	Summary	
6.2	Contributions	
6.3	Future Work	
6.4	Acknowledgments	
7.0	References	89

List of Figures

FIGURE 2-1	Relaying a message from A to C	21
FIGURE 2-2	Relay costs form a contour	24
FIGURE 3-1	Relaying a message from Node A to Node C	31
FIGURE 4-1	Testing the network with varying loads	47
FIGURE 4-2	Throughput vs. Client Rate	49
FIGURE 4-3	Average and Maximum Latency vs. load	50
FIGURE 4-4	Reliability vs. load	51
FIGURE 4-5	Testing the network with varying node density	52
FIGURE 4-6	Reliability vs. node count	53
FIGURE 4-7	Number of hops vs. node count	54
FIGURE 4-8	Average and maximum latency vs. node count	55
FIGURE 4-9	Testing the network as nodes move	56
FIGURE 4-10	Network reliability vs. mobility	57
FIGURE 4-11	Network reliability vs. mobility, potential boost = 1	58
FIGURE 4-12	Testing the network as the transmitter radius varies	59
FIGURE 4-13	Network reliability vs. transmitter coverage	61
FIGURE 4-14	Latency vs. coverage	61
FIGURE 4-15	Testing the network with multiple “dialogs”	63
FIGURE 4-16	Reliability vs. simultaneous dialogs	64
FIGURE 4-17	Latency vs. simultaneous dialogs	65
FIGURE 4-18	multiple clients talking to one server	66
FIGURE 4-19	Reliability vs. # of clients (1 server)	67
FIGURE 4-20	Latency vs. # of clients (1 server)	67
FIGURE 5-1	Simple vs. Revised Propagation Models	69
FIGURE 5-2	Performance as a function of load (alternate transceiver model)	75
FIGURE 5-3	Throughput vs. offered rate (alternate transceiver model)	76
FIGURE 5-4	Maximum latency vs. offered rate (alternate transceiver model)	77
FIGURE 5-5	Reliability vs. offered rate (alternate transceiver model)	77
FIGURE 5-6	Multiple Dialogs(alternate transceiver model)	78
FIGURE 5-7	Reliability vs. # of dialogs (alternate transceiver model)	79
FIGURE 5-8	Latency vs. # of dialogs (alternate transceiver model)	79
FIGURE 5-9	One server, many clients (alternate transceiver model)	80
FIGURE 5-10	Reliability vs. # of clients (alternate transceiver model)	81
FIGURE 5-11	Latency vs. # of clients (alternate transceiver model)	81
FIGURE 6-1	Artists conception of a Hyphos node	83

List of Tables

Table 3-1:	Format of a Hyphos message.....	30
Table 3-2:	Format of a Routing Table Entry.....	34

1.0 Introduction

1.1 What if network connections were smaller?

Today, connecting to a digital network usually involves a computer. This isn't surprising: most network interfaces are designed and marketed specifically for use with a computer. The smallest interfaces currently available, marketed for lap-top computers, cost on the order of hundreds of dollars and require support from the operating system in order to communicate with the target network.

But what will happen when the cost of connecting to a digital network drops to a few dollars and doesn't require the direct support of a computer? What kinds of new connections will be possible?

- You won't need to set your watch ever again. Your watch will be on-line all the time and will query the nearest network time server to discover the local time. Your appointment book, also on the network, will ask your watch to notify you about upcoming appointments.
- Tomorrow's "radio" will be a network appliance: regardless of when you turn it on, it immediately starts reading the latest headlines custom tailored to your interests.
- The smoke detector in your basement will be networked. If it detects trouble, rather than futilely beeping in isolation, it will enlist the services of an effective software ally to alert the members of the house, and if none are found, to start making telephone calls until someone is found to address the problem.
- A custom-built office chair will be on-line from the moment it's ordered until the time it's delivered to the customer. A networked tag, carrying all of the ordering and shipping information, will literally be built into the chair as it progresses through the production line. At each station, the tag informs the machinery as to the desired fabric, style, and options and logs the progress of the operations. At the loading dock, the tag informs the truck of the customer's shipping address.

- Toy dolls will know how to read, talk and listen. Rather than putting a powerful computer in each toy, it will suffice to network each toy to the local family PC. Connected to the Web, the family PC can download new activities and lessons every day; these toys will always have new surprises in store.

1.2 A thousand connections in one room

Today, the density of network connections rarely exceeds one connection per person: one network connection serves one computer which serves one person. But this ratio will change.

The number of host computers connected to the Internet is estimated to be growing by a factor of ten every three years. This could facetiously be interpreted to mean that you'll have ten computers on your desk where you now have one. More likely, however, networks will diffuse into objects much smaller than today's personal computer, and a typical room will contain dozens or hundreds of devices connected to networks.

What is required to move beyond the one person / one computer / one network model of today? What are the implications of a hundred network connections per room, or even per person?

1.2.1 Wireless

First, such a network must be *wireless*: In a room with hundreds of connections we can afford neither the clutter nor the expense of hundreds of interconnecting cables.

1.2.2 Low power

Second, the individual nodes on the network must be *low power*. Since the nodes are wireless, they must be powered from batteries or from surplus energy scavenged from the environment. As a practical consideration, nodes powered from batteries should run unattended for at least a year.

1.2.3 Self-organizing

Third, the network must be dynamically *self-organizing*. A network may contain hundreds or thousands of nodes, some of which may be mobile or temporary. In such a network, it's impossible to maintain network routing tables manually. Rather, the network must automatically adapt as its topology changes and as nodes arrive at or depart from the network environment.

Some corollaries arise from the above rules:

1.2.4 Limited range

Nodes will communicate only with their nearby neighbors. Since nodes operate with limited power, they have a limited broadcast range. In a turnabout of conventional wisdom, a limited broadcast range has advantages over the larger broadcast areas touted by today's wireless LANs. In particular, when a node turns on its transmitter, it usurps the airspace for its area of coverage. A node with a smaller area of coverage affects fewer nodes, permitting a higher overall network bandwidth.

1.2.5 Fully distributed routing

Since any node communicates only with its immediate neighbors, there can be no central controlling "hub," and the burden of network control falls upon the individual nodes. Each node must play an equal part in routing packets and maintaining network state.

1.3 About this thesis

This thesis describes *Hyphos* (from the Greek word for *web*): a self-organizing, wireless network. A node in a Hyphos network communicates only with other nodes in its immediate vicinity. Neighbors relay messages to their neighbors in turn until the message reaches its destination.

1.3.1 A scene at a banquet

Imagine you are seated at a large banquet table. You need urgently to deliver a message to your host, seated at the head of the table. You scribble a note and seal it in an envelope addressed to your host. You then pass the envelope to your

neighbor with instructions to “pass it on” until it reaches the host. The amount of effort expended by each guest is minimal—conversation is barely interrupted—and your message quickly finds its way into the hands of your host.

The above scenario approximates the basic workings of a Hyphos network. The principal difference is that in Hyphos the communication consists of electronic bits and radio links rather than sheets of paper and hands.

1.3.2 The rest of this document

Hyphos has been designed to create fine-grained, low-cost networks. The last section of this chapter describes several novel applications made possible by this approach.

The essence of Hyphos is captured by an architecture (an interconnected network of wireless nodes) and a set of protocols (the routing and message forwarding algorithms resident in each node). Chapter 2.0, “Context” describes the scope of this thesis, specifying the design goals, underlying assumptions, and prior work in the field.

Chapter 3.0, “System Description,” gives the technical details of Hyphos and how it is implemented. The chapter introduces the layered reference model used to describe the network and the format of messages relayed among Hyphos nodes. It follows with a detailed description of each of layer.

Hyphos’s behavior has been modelled by software simulation. Chapter 4.0, “Simulation and Results” describes the assumptions made in the simulation and presents quantified results of the simulation.

Chapter 5.0, “A revised transmitter/receiver model” investigates a model for wireless communication more precise than the simple model used in Chapter 4.0, and shows its effects on network performance.

Chapter 6.0, “Conclusions” describes the innovations particular to Hyphos, summarizes the results of the Hyphos network and suggests directions for future work.

1.4 Some application scenarios

Imagine a Hyphos network, created simply by virtue of assembling a collection of wireless nodes. The nodes of this network are:

- Compact: Each node is a small disc, 25mm diameter by 10mm thick.
- Inexpensive: Each node costs approximately \$2.
- Fast: Each node provides a 2 M bit / second data link to the network.
- Simple: Each node is self-configuring, adding itself to the network upon being physically placed in an environment with other nodes.

What applications can we build from a collection of these nodes?

1.4.1 Corporate LANs

A corporate office populated with a Hyphos network can dispense with intra-building network wiring. Every workstation, printer, and laptop uses a Hyphos node rather than an Ethernet interface. This approach eliminates the installation costs associated with wall jacks, cabling, and phone closets. Adding new equipment to the network is as simple as powering up a new node. Relocating equipment requires no more work than carrying it to its new location—no cables have to be moved, no routing tables updated.

1.4.2 “Last millimeter” delivery into the home

Many communications carriers are scrambling to get networks to the home. Carriers don't generally provide wiring inside the house; however few homeowners wish to pay for extensive rewiring.

A Hyphos network offers a cost-effective gateway between an external network and devices within the house. Since each node is wireless, cables are eliminated. Nodes are inexpensive, making it practical to connect “everyday objects” to the network. For example, all clocks are connected to centralized time server; the “flashing 12:00” on a VCR becomes a thing of the past. The ubiquitous TV Remote Control brings up today's TV guide from the WEB and instructs the VCR

as to which shows are to be recorded. An appliance automatically calls its home office for self-diagnosis or firmware updates.

1.4.3 Intelligent Warehousing, Routing and Shipping

Every box and pallet in a warehouse carries a Hyphos node. Since nodes can contain state and computing power, database functions are distributed among the objects themselves, rather than maintained in a central database.

An incoming box announces its contents to the warehouse when it arrives. The warehouse assigns that box to a particular truck for the outbound leg, using dynamic routing to minimize cost. When the time comes to send the box, the truck queries the boxes in the warehouse directly: “Which ones of you belong in this truck?” Each box maintains its entire shipping history and can be queried at any time.

Since state and routing information is kept with the shipping boxes themselves, maintaining database consistency is no longer a problem.

2.0 Context

2.1 Scope of this work

This work examines the domain of *ad-hoc*¹ wireless networking and presents a design for *Hyphos*, a self-organizing wireless network. A *Hyphos* network is formed by a collection of individual wireless nodes, whose overall architecture is presented in terms of the algorithms and data structures resident in each node. A software simulator checks the correctness and behavior of the *Hyphos* architecture.

To use terminology from the International Standards Organization's OSI (Open Systems Interconnection) Reference Model, the design of the *Hyphos* algorithms focuses on a *Medium Access Layer* and a *Networking Layer*, which are described in detail in Chapter 3.0, "System Description."

Hyphos does not try to specify what applications or even what high-level protocols are to be run over the network. The contract of *Hyphos* is to deliver valid data packets to a *Transport Layer* upon which higher level protocols (e.g. TCP) can be implemented.

At the low end of the design space, *Hyphos* doesn't specify what sort of physical medium is used to transport the bits from one node to the other. The primary assumption that *Hyphos* makes of the *Data Link Layer* and the *Physical Layer* is a "local multicast" system, that is, when a node transmits data, one or more nearby nodes receive the transmission.

2.2 Design Goals

The goal of this work is to define *Hyphos*, a new model for digital networks. By significantly decreasing the cost and complexity of forming a digital connection, new possibilities arise. Everyday objects, previously excluded from participation

1. The term "ad-hoc" is applied to networks that don't require pre-existing infrastructure and whose topologies are dynamically established.

in the digital realm, can now provide means for much richer communication between people and the digital world.

Towards this goal, the design of Hyphos has the following objectives.

2.2.1 Low Cost

Assertion: The cost per network connection should be sufficiently low that connecting a thermostat or a clock to the network is economically attractive. The cost of a connection should add no more than 10% to the cost of these devices.

Implications: Assuming the cost of finished goods for a clock or thermostat to be \$20, the cost of a Hyphos node will not exceed \$2.

2.2.2 Dense, fine-grained networking

Assertion: The network should be able to interconnect dozens or hundreds of objects in one room or building.

Implications: It's neither practical nor economical to string cables between hundreds of devices in one room. Hyphos nodes will be wireless.

2.2.3 Easy setup and low maintenance

Assertion: Since there are potentially hundreds of nodes on the network, it is impractical to manually update host and routing tables for individual nodes.

Implications: The Hyphos network will be self-organizing, and will adapt automatically as new nodes are introduced into or removed from the network.

2.2.4 Low Power

Assertion: A network node should run unattended for over a year.

Implications: Since a Hyphos node is wireless, it will scavenge power from environmental sources (photocells, stray RF energy), or will run from a battery. Given a small lithium cell battery with a capacity of 1000 mA_H, a Hyphos node will limit its average current consumption to 110 μ A or less.

2.2.5 Mobile

Assertion: Nodes may be carried on people or mobile objects, and may constantly be moving in relation to other nodes.

Implications: The routing algorithms of Hyphos will support a network topology that is constantly changing.

2.2.6 Robust

Assertion: The network should remain functional as nodes come on-line or go off-line. Single point failures must not bring down the network.

Implications: Routing and control functions will be fully distributed throughout the network.

2.3 Approach

2.3.1 Multi-hop communication

As has been established, a Hyphos node is wireless and low-power. Taken together, this means that a node has limited transmit range. In order to transmit a message beyond the range of a single transmitter, a Hyphos node calls upon one or more of its neighbors in order to relay the message to its final destination. This technique is commonly called *multi-hop* communication.

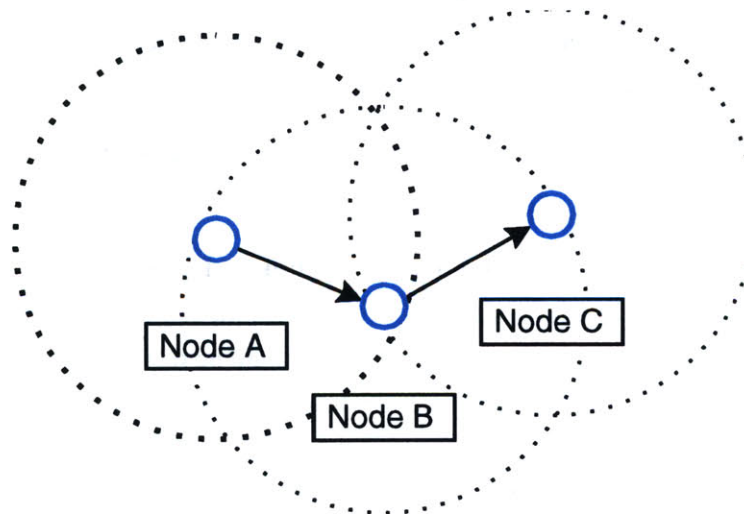


FIGURE 2-1 Relaying a message from A to C

FIGURE 2-1 shows an example of multi-hop communication in a network of three nodes. Each node, indicated by a solid dot, has a limited transmit range, as

indicated by a dotted circle around the dot. Node A can't communicate with Node C directly, so it enlists the services of Node B to relay messages to node C; this is the basis of all multi-hop communication systems.

Different multi-hop systems use different strategies in routing messages through a network. To date, these strategies can be categorized as "link state routing" or "source path routing" algorithms. By contrast, Hyphos uses a new technique called "contour routing." Each approach is described briefly below.

2.3.2 Link State Routing

In link state routing, each node maintains a routing table that, given a particular destination, specifies which one of its neighbor offers the optimal path towards the destination. "Optimal" is generally interpreted to mean the shortest path, but may account for other factors such as load balancing.

When a node in a link state routing system wishes to relay a message, it checks internal routing tables to determine which of its neighbors should relay the message; the address of that neighbor is installed in the message header as the recipient. The node sends the message. Of all the neighbors receiving the broadcast, only the designated recipient acts on the message. That neighbor repeats the process, re-relaying the message as required until the message reaches the ultimate destination.

Internet routers use Link State routing algorithms.

2.3.3 Source Path Routing

In source path routing, the originating node constructs an entire *source route* or "itinerary" of the message in the message header, giving the address of each node through which the message should be relayed in order to reach the destination. If a receiving node isn't the last node named in the message's source route, it simply relays the message to the next node on the source route. This process repeats until the message arrives at its ultimate destination.

Systems that employ Source Path Routing include UUCP mail headers (e.g. barney!bedrock!berkeley) and certain debugging IP packets [Tane96](415).

2.3.4 Contour Routing

Both Link State Routing and Source Path Routing schemes require that each node in the network keep a constant record of neighboring nodes. This requires that each node sends periodic “ping” or “I am here” messages to its neighbors. These periodic transmissions consume power and clutter up the airwaves. These periodic messages pose a problem in certain sensitive applications as it makes it easier for eavesdroppers to locate individual nodes.

By contrast, a node in a Hyphos network doesn't require knowledge of its neighbors in order to relay messages. Instead, the node keeps an estimate of the “cost” required to deliver a message to a particular destination. When it comes time to deliver or relay a message, the sending node writes the estimated cost in the message header and broadcasts the message. Of all the neighboring nodes that receive the broadcast, only those that can relay the message to the destination *at a lower cost* will do so. FIGURE 2-2 helps to visualize this process.



FIGURE 2-2 Relay costs form a contour

FIGURE 2-2 depicts the process of sending a message from node A to node C. Next to each node is a number indicating the estimated cost from that node to Node C. The cost of relaying a message from Node C to itself is 0.0; the cost increases with the number of hops from Node C.

Node A has established the cost of sending a message to C to be 2.0. When Node A sends a message to Node C, it writes information in the header of the message indicating that the message is to be delivered to Node C at a cost of less than 2.0. The neighbors of Node A (Nodes B, D, and E) all receive A's transmission. Since

only Node B can deliver the message at a cost less than 2.0, it relays the message while the other neighbors ignore it.

By this process, only nodes that are “closer to” the destination (in some topological sense) will participate in forwarding the message. Viewing each node’s estimated routing cost as an “altitude,” the process of relaying a message towards its destination is analogous to “rolling a message down a hill.” Given this analogy, this routing technique has been termed *Contour Routing*.

Routing tables are compact and tend to remain small. An entry in a routing table essentially consists of the ID of a target node and the accrued cost of the last message from that target¹. A node creates a routing table entry only when it receives a message originated by a new target. The routing table entry will be pruned if no message is received from that target within a period of time. Over time, the routing table will contain entries only for those targets for whom this node is actively involved in relaying messages. For example, if a node lies along a routing path shared by six different originators, then its routing table will contain only six entries.

The exact mechanics of Contour Routing are described in Chapter 3.

2.3.5 Establishing initial routes

A node estimates the cost to a particular originating node based upon the “accrued cost” of messages it receives from that originating node. But this presumes that the originating node has already sent a message, which would normally require that the originating node already has a cost estimate to its target. This leads to the proverbial chicken and egg quandary: how do initial routes become established?

If an originating node wishes to send a message to another node for which there is no routing information, the originating node will initiate a “flood” message. The flood message carries with it the ID of the intended target, the ID of the originator, and a sequence number keyed to the originator.

1. In truth, a routing table entry contains two additional fields: the sequence number of the last message received from the target and a time stamp used in “pruning” old entries.

Each node in the network receiving a flood message will relay the message exactly once. When a node relays a flood message, it makes a note in its routing table, which includes the originator and sequence number. If the node receives another copy of that flood message, the sequence number in the routing table will indicate that the message is a duplicate, and the node won't relay it a second time. The node will, however, update the cost entry in the routing table if the newer message indicates a lower cost than stored in the routing table.

Barring certain pathologies, a flood message will reach every node in the network¹. After the flood, every node will have a routing table entry "back to" the originator of the flood message, and the designated recipient of the flood message can enter into a dialog with the originator using the routing information established by the act of flooding.

2.4 Assumptions

Hyphos makes a number of assumptions: Some of these assumptions are questioned or addressed in Section 6.3, "Future Work," on page 84.

- Nodes are homogenous. In particular, nodes run the same algorithms and are willing to forwards packets on behalf of other nodes within their own network.
- Hyphos assumes that the wireless links of the network form a single connected graph, that is, there exists at least one path between any pair of transmitters and receivers.
- Transceiver propagation characteristics are assumed to be *reciprocal*: if node A can successfully receive a transmission from node B, then node B can receive a transmission from node A. This assumption will be dropped in Section 5.0, "A revised transmitter/receiver model," on page 69.
- For purposes of Medium Access, Hyphos assumes that the wireless receivers used in the physical link level provide some form of carrier detect that indicate nearby transmitter activity.

1. A flood message can fail to reach a node if the node is out of range of all other nodes, or under situations of extreme network congestion and multiple collisions

- Each node in the network has a unique identifier. In the work presented here, the link-level identifier is also used as the transport level identifier, but just as Ethernet addresses are decoupled from IP addresses, the link level and transport level identifiers may be different in practice.
- Nodes in a Hyphos network may be mobile, but the rate at which nodes change adjacency is small compared to round-trip time of a message.
- Hyphos, like other ad-hoc networks, uses a flat addressing scheme.
- Hyphos offers “best-effort” delivery of messages from one node to another. If an application requires guaranteed delivery, retransmission of messages can be implemented at or above the Transport Layer.
- Hyphos does not offer explicit support security and authentication services. Instead, these may be implemented in a higher level protocol.

2.5 Prior Work

Work in several different disciplines has set the stage for Hyphos.

2.5.1 Decentralized control

In the 1970s, Norman Abramson and his colleagues developed the ALOHA system, described in [Tane96] to provide networking around the Hawaiian islands. One of the earliest packet switched radio systems, ALOHA paved the way for other packet switched systems, such as Ethernet, and for almost every multi-hop wireless system since then.

In 1976, Bob Metcalfe and David Boggs published [Metc76], describing *Ethernet*, one of the first wired networks to use decentralized control.

2.5.2 Ad-Hoc networking

The *Lightweight Mobile Routing* protocol of [Cors95] and its successor, the *Temporally-Ordered Routing Algorithm* (TORA) of [Park97] are forms of link state algorithms. They each describe a mechanism for localizing network update messages to only those parts of the network that have changed. The TORA system

uses a global time base, agreed upon by all nodes, in order to attain timely updates of the network.

A variation of this, *Link Vector Algorithms* (LVA) described in [Garc95], also limits the “damage” messages sent through the network. Consistency is maintained by allowing only the “head node” (the link with no routes going into it) to initiate changes on other links in its chain.

The *Dynamic Source Routing* (DSR) protocol of [John96] uses a form of source path routing. In this system, each message carries with it the sequence of nodes responsible for forwarding the message to its ultimate destination. The *Signal Stability based Adaptive Routing* (SSA), described in [Dube96], is another source path routing system. SSA uses signal strength and location stability as criteria for choosing the best route.

These systems typically require that each node maintain an internal map of the topology of the entire network—this can lead to substantial storage requirements. In addition, the cost to all nodes the nodes in the network must be recomputed (e.g. by applying Dijkstra’s algorithm) whenever the topology changes. By contrast, routing tables in Hyphos are very compact—requiring only one entry per “active” route, and relatively little processing power is needed to maintain the tables.

All four systems described above dedicate some bandwidth to exchanging “ping” or “I am here” messages with their neighbors. By contrast, the Hyphos system is entirely quiescent until a message is presented for transmission.

2.5.3 Room-sized transceivers

[Carv96] describes *BodyLAN*, a system for wirelessly communicating data around a “body-sized” space. Transceivers are optimized for extremely low power, attaining power consumption on the order of 10 nanojoules per bit sent and received.

[Deme94] presents a *Nano-Cellular* system that uses a low carrier frequency (5.3 MHz) and electromagnetic coupling rather than wave propagation to attain sharply localized transmission areas. The paper also describes a modified MACA

(Medium Access with Collision Avoidance) protocol. While appropriate for unicast architectures, the MACA protocol doesn't apply to a multicast system such as Hyphos.

[Bult96] describes a low-power transceiver, integrated onto a CMOS chip along with an impressive assortment of MEMS sensors.

None of these systems are multi-hop systems; they all depend on a centralized hub within transmit range.

2.5.4 Other research

The *Self-organizing Wireless Adaptive Network* (SWAN) protocol described in [Scot95] differs from other protocols described here in that it's designed for circuit-switched connections (for voice, in particular), assumes multiple channel transceivers, and assumes a slotted model with a synchronized clock between the nodes of the network.

Tim Shepard's thesis *Decentralized Channel Management in Scalable Multihop Spread-Spectrum Packet Radio Networks* ([Shep95]) describes a temporally slotted system with power control of the transmitters. Shepard demonstrates, in simulation, collision free transmission of packets without any central control. His system depends on the general noise immunity of spread-spectrum transceivers.

By contrast, Hyphos assumes single-channel transmitters without power control. This design decision is based on economics of cost and power: if "simple" transceivers cost significantly less than spread-spectrum, power controlled transceivers, then it's more economical to fill discontinuities of the network with additional nodes rather than boosting the power of neighboring nodes. Since radiated power falls off by the square of the distance, this approach also consumes less power overall.

} ? node density
↑ x^2 to

2.5.5 Commercial systems

Metricom's *Ricochet* Packet Radio Network connects wireless modems to the Internet infrastructure by means of "poletop" repeaters (so called because they're typically mounted atop street lamp poles). The repeaters form a fixed wireless multihop network to the nearest wireline access point. Since the wireless modems

must transmit to the nearest repeater, power consumption is relatively high, leading to a maximum battery life of four to six hours.

The AT&T *WaveLAN* (a.k.a. the DEC *RoamAbout*) is a wireless LAN system, featuring a 2 Mbit/second link and a range of approximately 50 meters, designed primarily for corporate office settings. Like a miniature cellular telephone system, these LANs require pre-existing infrastructure in the form of one or more wired base stations.

3.0 System Description

3.1 System Overview

This chapter describes the general operation of a Hyphos network and details the operation of an individual node in the network.

3.2 Reference Model

Hyphos adopts a layered network model in which each layer provides a distinct set of services with a clearly defined protocol at each layer. This reference model falls somewhere between the OSI (Open Systems Interconnection) model and the TCP/IP Reference Model as described in [Tane96].

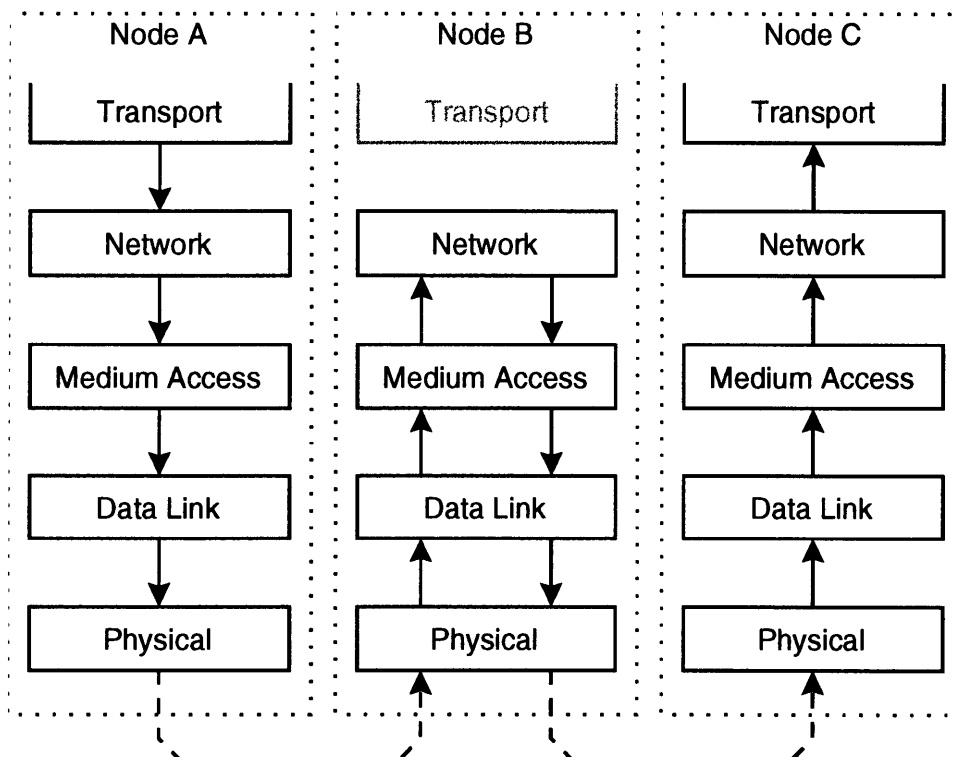


FIGURE 3-1 Relaying a message from Node A to Node C

The above figure shows the levels of a Hyphos network and each level's role in originating, relaying, and receiving a message in a Hyphos network. Each of these levels are discussed in detail in subsequent sections.

3.3 Message Format

In a Hyphos network, all routing information is embedded in the messages exchanged between nodes. It's helpful to examine the components of a Hyphos datagram as these components will be referred to in subsequent sections.

Field	Reference Level	Description
TargetID	Transport	Network address of the recipient of this message
NetMask	Transport	Used to determine if the message is part of "this" network
OriginatorID	Network	Network address of the originator of this message
Sequence#	Network	In conjunction with OriginatorID, used to identify messages that have already been received
IsDebut	Network	Set to TRUE if this is a route discovery message
AccruedCost	Network	Accumulated cost of relaying the message since it was originated. Initialized to zero by the originating node; each relaying node adds its relaying cost to this field before rebroadcasting the message
RemainingCost	Network	Remaining "credit" available to this message before it expires. Initialized to the originating node's cost estimate to the target, each relaying node subtracts its relaying cost from this field before rebroadcasting the message. If the remaining cost ever reaches zero, the message is dropped
PayloadSize	Transport	Number of bytes in the subsequent payload section of the message
CRC	Data Link	Cyclic Redundancy Check on all the above fields (not on the Payload itself)

Table 3-1: Format of a Hyphos message

Field	Reference Level	Description
Payload	Transport	Data bearing part of the message (variable length)

Table 3-1: Format of a Hyphos message

Not shown in Table 3-1 are any framing bits that might be required of the physical (transmitter) level.

3.4 Physical Layer

The Physical Layer specifies the physical means by which data is exchanged between one communication node and the next. In the case of Hyphos, the physical link is assumed to be a wireless link to a subset of other nodes in the network.

The physical link may be implemented as a single frequency RF transceiver, a spread spectrum system, or even an infrared link. As shown by [Shep95], spread spectrum systems offer good noise immunity in an environment with many nodes.

As will be described in the next section, the default implementation of the Medium Access Layer won't attempt transmission while a message is being received. This permits the use of a simple half-duplex transceiver.

3.4.1 Sending Messages

Transmitters are assumed to be very low power. Chapter 5.0 will show that power on the order of one microwatt is sufficient for “room sized” applications with nodes distributed one every 16 square meters on average. But the coverage of the network can be scaled by appropriate choice of transmitter power: if a network density of one node per 16 square kilometers is desired, then transmitter power can simply be increased to one milliwatt.

3.4.2 Receiving Messages

The receiver should have good noise immunity in an electronically noisy environment. Simulations assume a single channel system that can capture a signal with 10dB signal to noise ratio. The Medium Access Layer (q.v.) assumes

that the receiver provides a carrier sense to indicate when a signal is being received. Beyond that, the Hyphos architecture doesn't assume much of the receiver.

3.5 Data Link Layer

In Hyphos, the Data Link Layer handles the framing of the data packets and the generation and checks that the packet is free from transmission errors.

3.5.1 Sending Messages

When Data Link Layer is given a message to be sent, it computes a CRC based on all the fields up to but not including the message Payload and installs the CRC in the message header.

The CRC must be recomputed each time the message is relayed, since the `AccruedCost` and `RemainingCost` fields change. Higher level protocols are responsible for generating and verifying any checksum bits in the `Payload` section of the message.

The Data Link Layer adds any framing and synchronization bits required of the physical transmitter before passing the message to the Physical Layer (the transmitter) for transmission.

3.5.2 Receiving Messages

When the physical layer receives a packet, the Data Link Layer first strips off any framing and synchronization bits and checks the CRC of the incoming message. If the CRC is valid, it passes the message up to the Medium Access Layer for processing.

3.6 Medium Access Layer

The airwaves surrounding a Hyphos node can be thought of as a shared resource; if two or more adjacent transmitters are active simultaneously, then messages being transmitted will be corrupted. The purpose of the Medium Access Layer is to reduce the likelihood of collisions.

3.6.1 Sending Messages

When a node finishes transmitting a message, the end of transmission will typically trigger the relaying of the message by its neighbors. Unchecked, these neighbors would start to transmit simultaneously, leading to many collisions.

To avoid collisions, the Medium Access layer uses a simple binary exponential back-off algorithm, not unlike that used in Ethernet [Metc76]. To implement this, the Medium Access Layer maintains a “back-off” counter whose value is zero when the local airwaves are quiescent and increases as the local airwaves becomes more loaded. The back-off counter is used to set a delay before attempting transmission of an outgoing message.

When presented with a message to send, the Medium Access Level immediately installs the message in a “send queue,” where it waits to be relayed. Earlier implementations of Hyphos exhibited excessive latency delays under high load conditions. It was observed that the send queue would contain multiple messages from one originator destined for one target (albeit with differing sequence numbers). It was also observed that a given message in the send queue had often already been relayed by a neighbor. The latest versions of Hyphos now “sluff” messages, discarding all but the most recent message from a one originator to one target. Although this can somewhat reduce the overall reliability of the network (since messages are dropped at this point), it reduces the maximum latencies by orders of magnitudes under high load conditions.

After queueing a message to be sent, the Medium Access Layer runs a separate process as described below.

- [0] Wait until the Data Link Layer indicates that the transmitter is idle. Go to step [1].
- [1] If the queue is empty, set the back-off counter to 0. Wait until there is one or more messages in the queue to send. Go to step [2].
- [2] Choose a random value D distributed evenly between $backoff-0.5$ and $backoff+0.5$, delay for $K \times 2^D$ before going to step [3]. K is chosen to be the time

required to send the minimum sized message. When backoff = 0, this has the effect of, on average, delaying for the duration of a short message.

- [3] If the Data Link layer reports that the local airwaves are busy (carrier detect is true), increment the backoff counter by setting it to $\text{MIN}(\text{maxbackoff}, \text{backoff}+1)$ and go to step [1]. If the airwaves are not busy, go to step [4].
- [4] Remove the message from the queue, pass it to the Data Link Layer to be transmitted. Decrement the backoff counter by setting it to $\text{MAX}(0, \text{backoff}-1)$ and go to step [0].

3.6.2 Receiving Messages

When the Medium Access layer receives a complete message from the Data Link Layer, it simply passes the message to the Network Layer.

3.7 Network Layer

The Network Layer addresses the issue of how to route a message through the network from an originating node to its ultimate destination. Due to the “local broadcast” nature of wireless communication, a node will always receive messages from its neighbor, regardless of the ultimate destination of the packet, so the Network Layer must decide if and how to act on any message it receives. When a message is received at the Network Layer, one of three conditions must be true:

- The packet is destined for the node itself, in which case it is passed up to the next higher level (the Transport Layer).
- The packet needs to be forwarded, in which case the Network Layer increments the `AccruedCost` field and decrements the `RemainingCost` field of the of the message and passes the message to the Data Link layer for transmission.
- The packet is not to be forwarded, in which case it is discarded.

3.7.1 Format of the Routing Table

The Network Layer creates a routing table entry for each message it receives from the Medium Access layer. Routing entries are time stamped and short lived: if a

the Network Layer hasn't received a valid message from a particular target after a period of time, its entry is deleted from the routing table.

Field Name	Description
TargetID	Network address of the originator of the message, copied from the OriginatorID of an incoming message
Sequence#	Highest sequence number received in a message from the Target, copied from the Sequence# of an incoming message
Cost	Accumulated relay cost of message as received from the Target, copied from the Cost field in the message
UpdatedAt	Local time at which this entry was created or updated

Table 3-2: Format of a Routing Table Entry

3.7.2 Sending Messages

The Network Layer performs the following operations with each outgoing message passed to it from the next higher layer (the Transport Layer):

- The Network Layer fills in its part of the message header. The `OriginatorID` is set to the network ID of this originating node, the `AccruedCost` is set to zero, the internal sequence number generator is incremented and its value is assigned to the `Sequence#` field.
- The Network Layer's routing table is consulted to discover the estimated cost to the Target node. If there is no known route to the Target, the Network Layer initiates a "flood" message by setting the `IsDebut` flag in the message header to `TRUE`.
- If there is a known cost associated with the Target, use it plus any "Potential Boost" to initialize the `RemainingCost` field.
- Finally, the message is passed to the Medium Access layer to be transmitted.

3.7.3 Receiving Messages

An incoming message received by the Network Layer goes through several steps.

- If this node's network ID ANDed with the NetMask of the message differs from the targetID of the message ANDed with the NetMask, then this message is not considered to be part this logical network and it is dropped.
- The AccruedCost field in the incoming message represents the accumulated relaying cost from the OriginatorID of the message. If this cost is less than or equal to the Cost entry in the routing table for that Originator, update the routing table entry to reflect the reduced cost and update the UpdatedAt field. Continue with the next steps.
- If the Transport Layer has already processed another copy of the same message (as indicated by the Sequence# field), discard the message.
- If the message is targeted for this node, pass it to the Transport Layer.
- If the message's RemainingCost has hit zero, discard the message.
- If this is a Debut message (IsDebut is turned on in the message header), relay the message.
- The cost to the Target node is estimated by looking up the routing table entry for the target. If there is no entry (the route is unknown), the message is dropped.
- If the estimated cost to the Target node plus the current forwarding cost for this node exceeds the RemainingCost in the message, then the message is dropped on the assumption that it will expire before reaching the destination.
- If all other conditions are met, the Network Layer will relay the message. Before passing it to the Medium Access layer, the AccruedCost field in the message is incremented by the current forwarding cost and the RemainingCost field is decremented by the current forwarding cost.

3.8 Transport Layer

Hyphos does not officially define a Transport Layer. For purposes of testing and simulation, we created a Transport Layer with simple protocols which are discussed in the section on Simulation and Results.

3.8.1 Sending Messages

The “toy” Transport Layer defined for purposes of simulation simply originates new messages at a predetermined rate. To prevent the network from becoming locked into resonant states, the onset time of each messages is randomized slightly. For purposes of simulation only, messages carry an “origination time” field, which is used to calculate latencies. When a message is originated at the Transport Layer, the simulator fills in the origination time field.

3.8.2 Receiving Messages

At and above the Transport Layer, Hyphos’s work is done. For purposes of simulation and characterizing network performance, the “toy” Transport Layer computes the latency and other statistics for messages received.

4.0 Simulation and Results

The behavior of Hyphos networks has been tested, tweaked, and measured using a software simulator. In its current incarnation, the simulator consists of approximately 8000 lines of Objective C code (an object-oriented dialect of the C programming language) distributed among 45 files. The simulation was run under the NEXTSTEP operating system on a 200 MHz Intel Pentium Pro workstation.

The key purposes of the simulator are to provide controls for various operational parameters and to measure the performance of the Hyphos network as those parameters are changed. To be useful, these parameters and performance measurements must correspond to physical reality. The simulation has been written carefully to model physical units whenever possible; time is measured in seconds, distance is measured in meters, transmitter power in watts.

This chapter describes the basic elements of the simulator, details the individual components and the controls they offer, and quantifies the network performance as these controls are changed.

4.1 Components of the Simulator

4.1.1 Scheduler

At the heart of the simulator is a `Scheduler` object, which is responsible for all temporal aspects of the simulation. The scheduler maintains a queue of `Events`. Each event is associated with a time and an action.

The scheduler processes a single event by removing the “soonest” event from the queue, setting the simulated time to that of the event, and calling the action associated with that event. Running the simulation, therefore, is simply a matter of continually processing events. In fact, the simulator will stop if it runs out of events to process; in practice, the processing of an event typically results in one or more new events being scheduled and the simulation continues.

Any object in the simulator can post a request with the scheduler to invoke a method at some absolute or relative time in the future. This is the mechanism the

Hyphos simulator uses to model temporal behavior: an instance of the `Mover` class (q.v.) updates its x and y coordinates and posts a request to be called again in the near future. A `Node` object, upon discovering that the local airwaves are busy, computes a backoff time and posts a request to be called again at the end of that backoff time.

4.1.2 Node

Hyphos nodes are simulated by instances of the `Node` class. A `Node` is implemented as a collection of sub-objects which correspond to the networking reference levels.

4.1.3 Phys

The `Phys` class handles the mechanics of physical radio transmission.

4.1.4 Link

The `Link` class emulates a virtual data link.

4.1.5 MACA

The `MacA` class implements the Medium Access / Collision Avoidance technique described in “Medium Access Layer” on page 34, and manages access to the `Link` level for sending messages.

4.1.6 NetL

The `NetL` class implements the network layer described in “Network Layer” on page 36. For each message received, it updates its routing table and decides on the disposition of the incoming message: relay it, drop it, or pass it up to the Transport Layer.

4.1.7 Trans

The `Trans` class implements a simple Transport Layer, sufficient for originating packets of data and verifying their receipt. A “client” node can be configured to fire off messages of a given size at some regular interval to a “server.” When the `Trans` class receives any message that it didn’t originate, it sends a reply back to the sender (in this case, a client.).

4.1.8 Mover

The `Mover` class maintains the position, bearing, and velocity of each `Node`. During simulation, if the `Node` runs into one of the walls of the `Arena` (q.v.), it reflects off the wall but maintains its velocity.

4.1.9 NodeView

A `NodeView` class provides a graphical representation of the nodes. The `NodeView` displays the position and basic status of each node (transmitting, receiving, colliding) and also provides the user interface functions for running the simulation in manual mode.

4.1.10 Arena

The single instance of the `Arena` class defines the virtual space in which each of the `Node` objects reside. The `Arena` provides user interface controls for the number of nodes, their initial configuration, what percentage of the nodes are mobile, and the number of “clients” and “servers.”

4.1.11 Other Classes

Rounding out the suite of classes that comprise the simulator is `TweakFace`, to manage manual adjustment of the simulator’s tunable parameters, `Metrics`, to monitor and report on the network performance, and `Script`, to import and export parameter sets and run the simulator automatically.

4.2 The Tests

4.2.1 The Questions

A suite of tests were developed to gain insights into the performance of the Hyphos network and its algorithms. Questions for which we wanted answers include:

- When does Hyphos “hit the wall,” that is, at what point does increasing the offered load result in a decrease in the throughput?
- What are the expected latencies in the network?

- How well does the network handle mobile nodes? What percentage of motion can it tolerate and still give good performance?
- What happens when there are multiple, unrelated dialogs taking place between pairs of nodes?
- What happens when multiple nodes try to communicate with one “server” node?
- How well does the network scale? How does performance change as the network grows?

4.2.2 The Testing Methodology

Individual test scripts were created to answer the questions above. Each script was designed to measure network performance as one or more parameters were changed. A simple scripting language and reporting mechanism reduced the chance of “operator errors,” and provided a written record of the tests that were run.

In general, Nodes were distributed randomly in the simulator’s testing arena. (The exception to this is that the “client” and “server” nodes were sometimes assigned to fixed locations.) Since minor variations in network topology can influence the network’s performance, the simulator was run ten times for each data point, randomly redistributing the nodes between each run. Each run of the simulation was set for 30 seconds of “virtual” time.

The load test (q.v.), for example, took twelve measurements as a loading factor was varied. Each measurement was run ten times over, for a total of 120 runs. Each run lasted for 30 virtual seconds; over 3600 of virtual simulation time went into the load test. The purpose of such liberal application of computer cycles was to guard against misinterpreting the behavior of a particular network topology as representing the general case.

The results of each test were gleaned from the log files and transcribed into files formatted as input for Mathematica [Wolf88]. Mathematica, in turn, was used to further crunch the numbers and display the results.

4.3 Simulation Assumptions

The simulator provides many tunable parameters. For the tests described here, some default parameter values were chosen. These default values, along with other assumptions made by the simulation, are detailed here.

4.3.1 Physical

The simulated Hyphos world is a 40×40 meter two-dimensional “arena” within which all nodes are located. Mobile nodes move at a constant velocity of one meter per second, approximating a leisurely stroll around the office. When a mobile node collides with the wall of the arena, it reflects but doesn’t otherwise change its velocity.

Unless otherwise specified, there are 100 nodes randomly distributed in the arena. The nodes may be fixed (indicated by a full circle in the simulation’s NodeView) or mobile (indicated by a semi-circle).

4.3.2 Transceivers

Transceivers have a constant bit rate of 2 M bits/second. In addition to the time required to transmit header and payload bits specified by a message, the transceiver models an additional ten bit periods for “trailer” bits at the end of the message.

Each transmitter has a fixed radius—within this radius, any receiver will properly receive messages from the transmitter unless another overlapping transmitter starts transmitting while the original message is in progress.¹

Unless otherwise specified, the transmit radius is adjusted so that its area, on average, covers ten neighboring nodes.²

4.3.3 Medium Access

Under unloaded conditions (backoff counter is zero), the average wait before attempting transmission is set to 256 bit times. This value doubles each time the

1. A less simplistic approach is presented in Section 5.0, “A revised transmitter/receiver model,” on page 69.
2. As will be seen in section 4.7, a coverage of ten nodes is parsimonious—sixteen would probably be better.

backoff counter is incremented. The backoff counter is limited to a maximum of 5; the average wait under these conditions is $256 * 2^5 = 8192$ bit times. At a transmitter rate of 2 Mbits/sec, this works out to be about 41 milliseconds.

4.3.4 Network Layer

Debut messages, and by extension, all other messages, are limited to a maximum hop limit of 20. Theoretically, a limit of 20 may be too parsimonious for very large networks, but in practice, poses no problem to the simulation.

Routing table entries that haven't been referenced in over 1.5 seconds are purged. In the context of an originating node, this means that if it doesn't get a reply from a target node within 1.5 seconds of originating a message, it "forgets" the cost to the target node, and the next originated message will be sent as a Debut (flood) message. For a relaying node, if a route isn't exercised for over 1.5 seconds, it will be dropped from the node's routing table. Future requests to relay to that target will be ignored, unless the request is in the form of a Debut message.

4.3.5 Transport Layer

Messages, whether used as Debut messages or as data bearing packets, are 64 bytes long consisting of 32 bytes of header information and 32 bytes of payload. To change the loading on the network, only the rate at which new messages are originated is modified. For example, a rate of 120 messages per second generates 7680 bytes of traffic per second, or 61440 bits per second. With a transmitter rate of 2 Mbits / second, this is about 3% of the theoretical capacity of the transmitter.

Whenever a client node originates a message, it expects a reply back from the target of the message. Because of this "call and response" nature of the Transport layer, a client request rate of 120 messages per second will result in an overall traffic load of 240 messages per second, or 6% "offered load" in a perfectly functioning network.

This leads to a subtlety in the interpretation of the offered load statistics: If a client message is lost before it reaches its target, the offered load will decline since the target doesn't originate a reply.

4.4 load: when does Hyphos hit the wall?

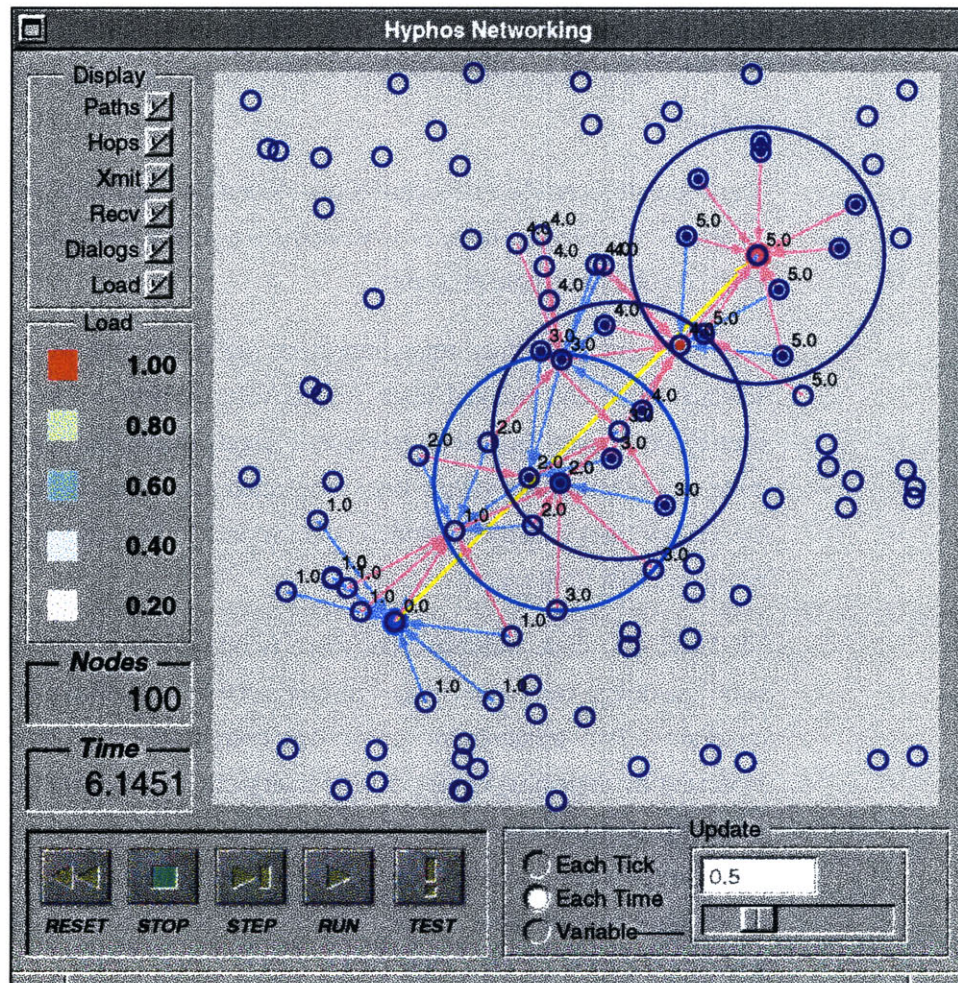


FIGURE 4-1 Testing the network with varying loads

4.4.1 Interpreting the simulator's display

FIGURE 4-1 shows the Hyphos simulator running the load test. The large rectangle represents the “arena,” 40 meters on each side. Within the arena, each of the 100 small blue circles represents one Hyphos node. The large dark blue circles are centered around nodes that are actively transmitting, the large pale blue circles

indicate nodes waiting to transmit. A blue dot in the middle of a node indicates a message being successfully received, a red dot indicates a collision.

A yellow line connects a “client” and a “server.” At the Transport layer of the simulation, a client initiates messages destined for its server at a predetermined rate. The server’s contract is to send a reply to the client for each valid message it receives.

The simulator’s `NodeView` allows one node to be “selected” and one node to be “marked.” A selected node is indicated by a pale blue aura surrounding it, a marked node has a pale pink aura.

The blue and pink lines interconnecting the nodes indicate paths towards the selected node and marked node respectively. A path is “known” if there is a routing table entry in a node towards a target. In this particular case, the client node (at the tail of the yellow line) is selected and the server node (at the head of the yellow line) is marked. Numbers above and to the right of nodes indicate the estimated cost to the selected node.

FIGURE 4-1, then, indicates that there is one client (selected node, lower left hand quadrant) communicating with one server (marked node, upper right hand quadrant). Two transmitters are active, each transmitter is covering the area indicated by their blue circles. At the intersection of the two circles, there is one node experiencing a collision of messages (red dot). One transmitter is waiting to transmit (pale blue circle). Only the nodes along a path between the client and the server have routing table entries (indicated by the blue and pink lines); the other nodes in the network aren’t involved.

4.4.2 The load test

The purpose of the load test is to observe the throughput of the Hyphos network as the offered rate varies. In this test, a single client node originates messages destined for a single server node. When the server node receives a message, it sends a reply to the client.

In this test, we define *throughput* as the rate at which valid messages are received (either at the server or at the client) as a fraction of the bit rate of a transceiver, and

client rate as the rate at which the client node originates new messages. In a perfectly functioning network, each packet originated by the client would be matched by an acknowledgment from the server, resulting in a throughput exactly twice the client rate.

As previously described, each message is 64 bytes long with a data rate of 2 Mbits per second or an “air time” of 256 μ sec per message. The test varies the client load from 40 messages per second to 1250 messages per second, corresponding to a channel capacity of 1% to 32% of the channel capacity. Note, however, that each message that the client sends is potentially matched by an acknowledgment message from the server, so the theoretical rates vary from 2% to 64% of channel capacity.

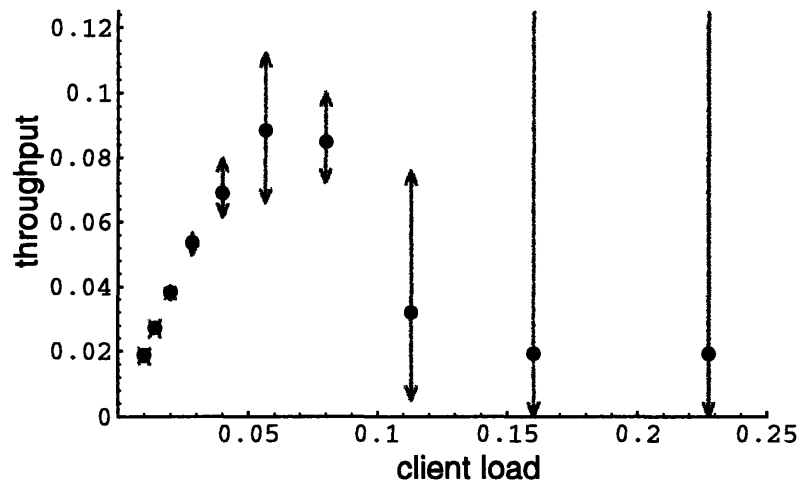


FIGURE 4-2 Throughput vs. Client Rate

FIGURE 4-2 shows the throughput as a function of client rate. At client loads up to 6%, throughput grows linearly with client load, indicating that about 84% of all originated messages are correctly received. Above 6%, throughput starts to drop. Error bars on the graph show the minimum and maximum values of these figures over a sequence of ten runs per data point—the error range is small at low client rates, but are less predictable as the client rate increases.

4.4.3 Latency

For a lightly loaded network, the time required to relay a message from the originating node to the destination is dominated by the time required simply to store and forward the message from one node to the next, and scales linearly with the number of hops. But as the network becomes loaded, messages spend more time queued at the Maca level waiting for the airwaves to become available.

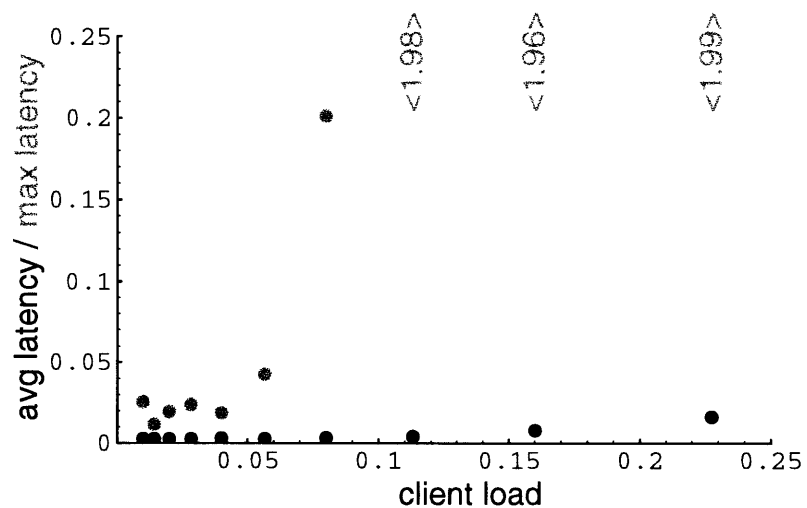


FIGURE 4-3 Average and Maximum Latency vs. load

In FIGURE 4-3, the average latency is plotted with black dots, the maximum latency (averaged over ten samples for each client rate) is plotted with gray dots. This graph confirms our suspicions that the network is in trouble at client loads greater than about 8%. At a client rate of 8% and higher, the latencies grow quickly as the network becomes congested—occasional delays of approximately 2 seconds are observed.

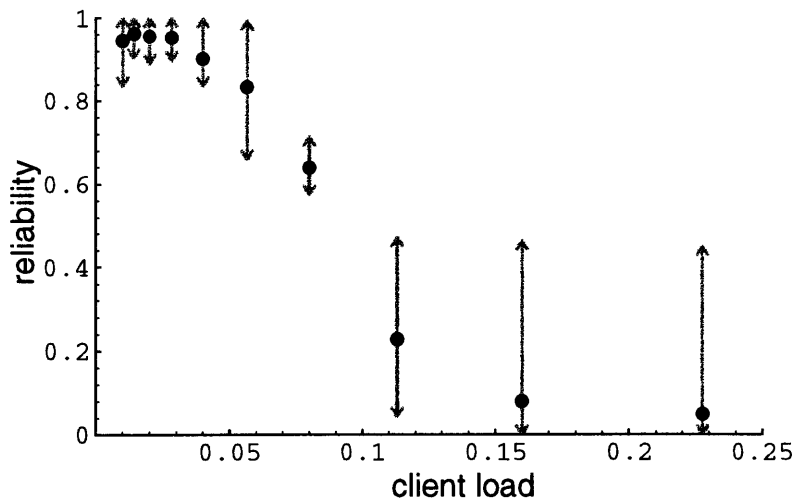


FIGURE 4-4 Reliability vs. load

FIGURE 4-4 introduces the concept of *reliability*, defined as the ratio of the number of messages correctly received to the number of messages originated. As can be predicted from the throughput graph of FIGURE 4-2, reliability remains above 84% for client loads up to 6% and falls off quickly thereafter.

4.5 density: how well does Hyphos scale?

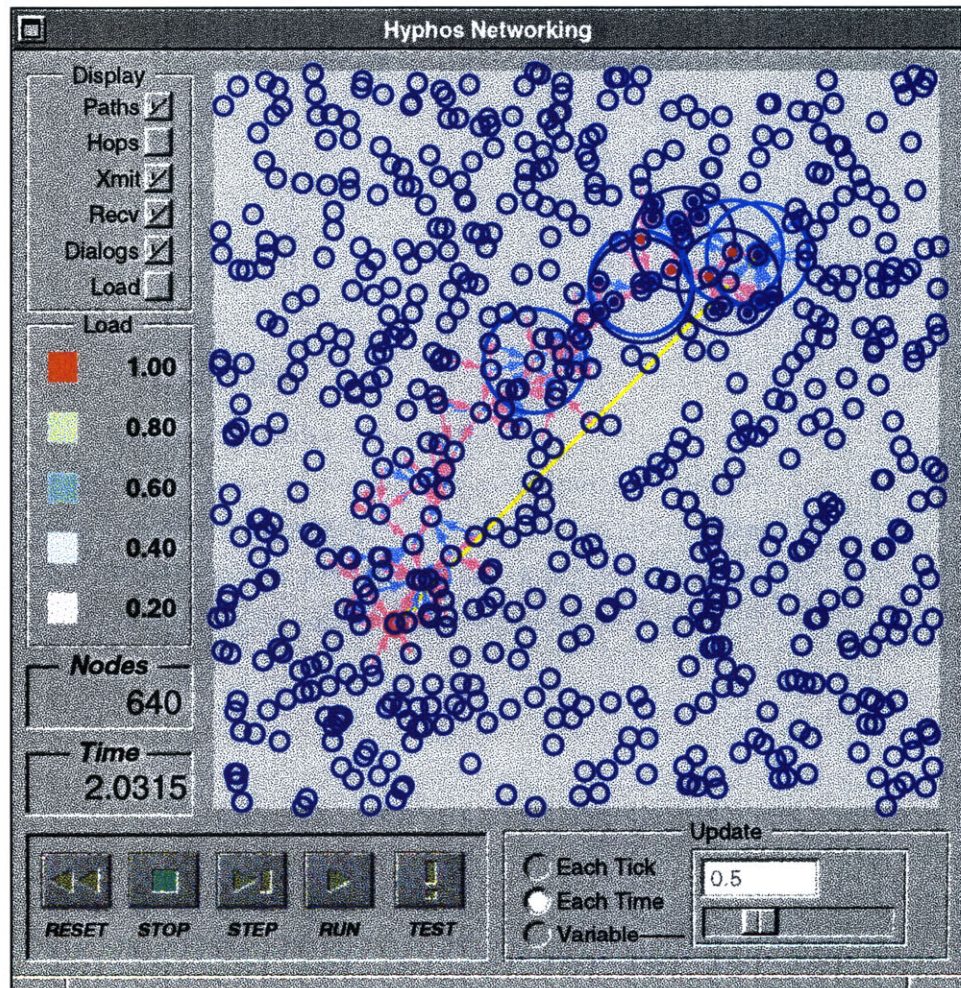


FIGURE 4-5 Testing the network with varying node density

How does network performance change as the total number of nodes increases? [Shep95] makes the point that as the total number of nodes increases on a two-dimensional plane with constant average density, the average path length between two nodes increases as the square root. This means that routes between two randomly chosen nodes will, on average, become more congested as the number of nodes increases.

The density test examines the reliability of the network as the number of nodes changes. In this test, the number of nodes varies from 10 to 640, doubling each time. Transmitter range is adjusted so that, on average, one transmitter reaches ten nodes. Note that this is analogous to increasing the area of the Arena while varying the number of nodes while holding the transmitter radius and average node density constant.

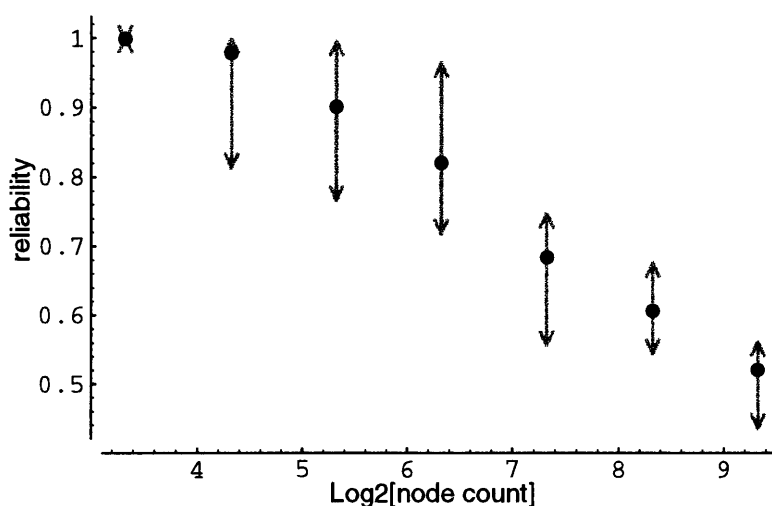


FIGURE 4-6 Reliability vs. node count

As before, *reliability* of the network is defined to be the ratio of the number of messages correctly received to the number of messages originated. A perfectly functioning network will have a reliability of 1.0. As can be seen in FIGURE 4-6, the reliability falls gradually from nearly 100% for a sparse network of 10 nodes to 52% for a dense network of 640 nodes.

The density test uses one client and one server. The client originates messages at 6.14% of the channel bandwidth (240 messages per second). The aggregate throughput for a network with reliability of 1.0 would be 12.28% of the channel bandwidth; the actual throughput can be computed as 12.28% pro-rated by the reliability.

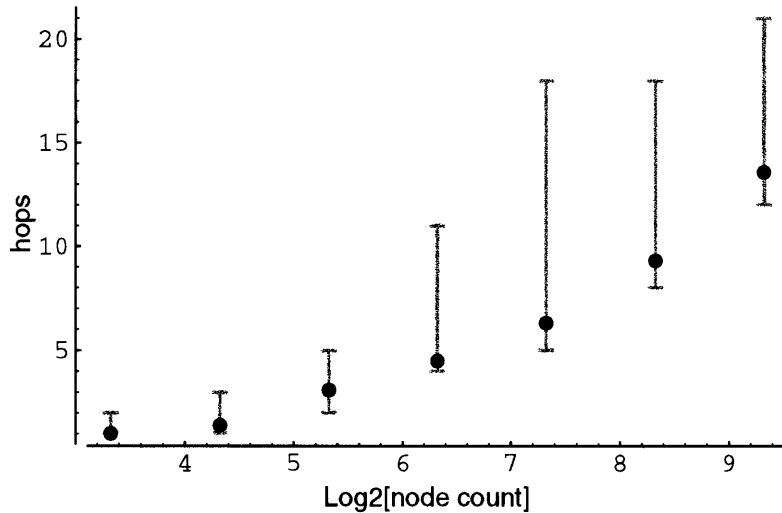


FIGURE 4-7 Number of hops vs. node count

FIGURE 4-7 shows the number of hops as a function of the network density. The transmitter radius is adjusted to cover an average of ten nodes; in a 40×40 meter arena with ten nodes (the minimum shown), the transmitter radius is 24 meters. In the same arena with 640 nodes, the transmitter radius is 2.8 meters. As a result, a message takes more hops in a higher density network than in a low-density network. For historical reasons, the built-in hop limit is set to 20, thus the maximum number of hops never exceeds 21.

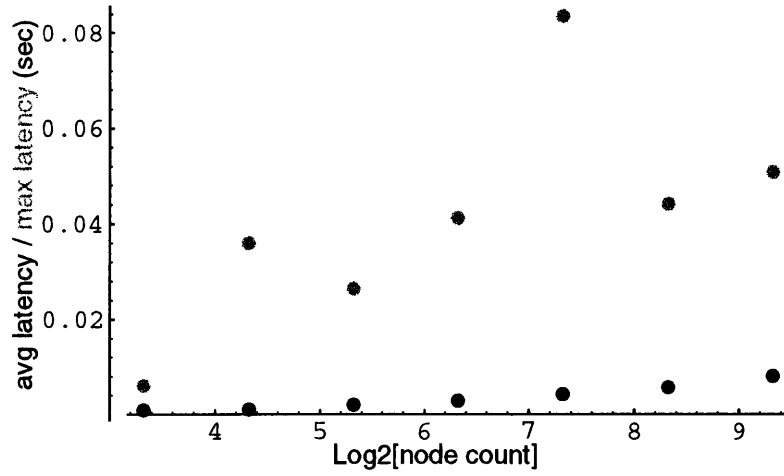


FIGURE 4-8 Average and maximum latency vs. node count

FIGURE 4-8 shows the average and maximum latency as the network density increases. As expected, the average and maximum latencies are roughly proportional to the number of hops observed in FIGURE 4-7.

4.6 motion: how do mobile nodes affect performance?

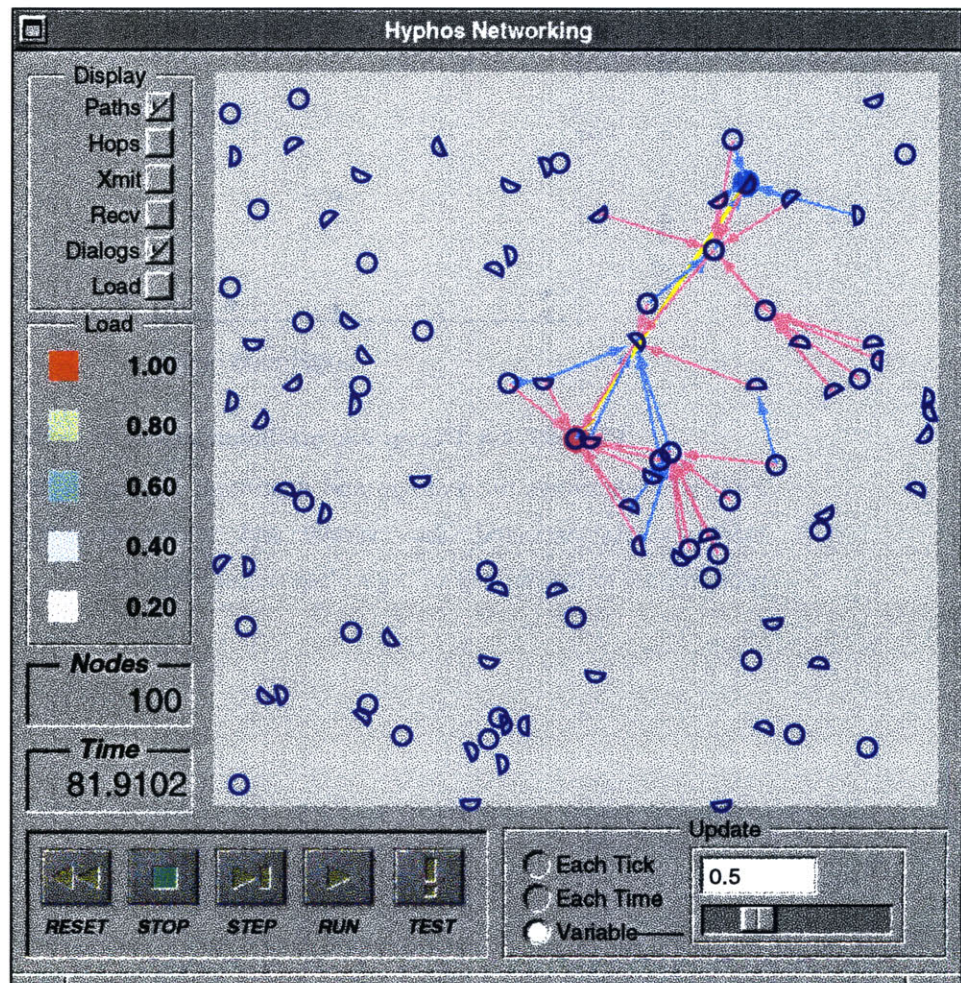


FIGURE 4-9 Testing the network as nodes move

One of the stated goals of the Hyphos design is that the network continue to work well even as nodes move. FIGURE 4-9 shows simulation in which nodes are mobile. A mobile node is depicted as a semi-circle rather than a circle; the flat side faces the direction of travel. In the motion test, when a node moves, it moves with a velocity of 1.0 meters/second, corresponding to a leisurely stroll around an office. The network is mildly loaded; a single client node originates packets at a

rate corresponding to 3.07% of the total network bandwidth; the aggregate bandwidth, taking into account the server's replies, will approach 6.14%.

In the motion test, the server node is stationary at the middle of the arena. The client node is always mobile, and is initialized at each run with a random position and bearing. The other nodes are either fixed or mobile, depending on the *mobility*, which is defined as the ratio of moving nodes to the total number of nodes.

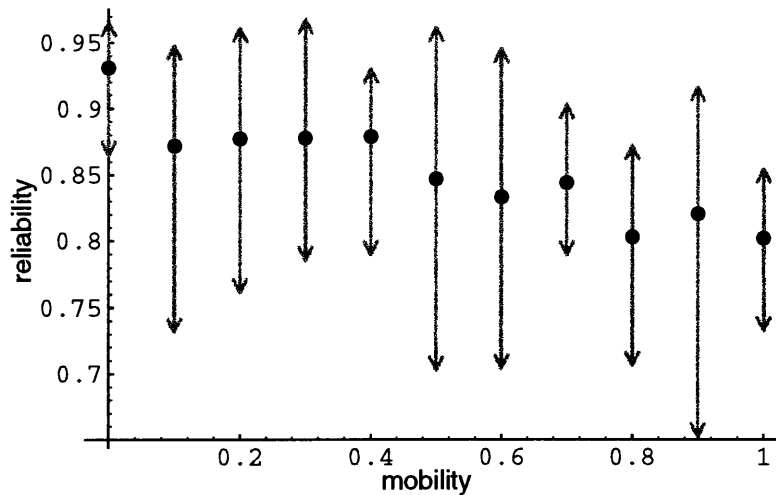


FIGURE 4-10 Network reliability vs. mobility

As before, we define *reliability* as the ratio of ratio of the number of messages correctly received to the total number of messages originated. As can be seen in FIGURE 4-10, the reliability falls gradually from about 93% to about 80% as the mobility increases from 0.0 (no nodes move) to 1.0 (all nodes move).

4.6.1 Increasing reliability in highly mobile networks

How can we attain constant network reliability when a majority of the nodes are in motion?

As described in Section 3.7, “Network Layer,” on page 36, the Network Layer can impose a “potential boost” on a message before sending it. This has the effect of causing the message to spread out through a wider swath of intermediate nodes.

At the price of consuming more network bandwidth, this technique increases the likelihood that the message will reach the destination in the face of changing network topologies.

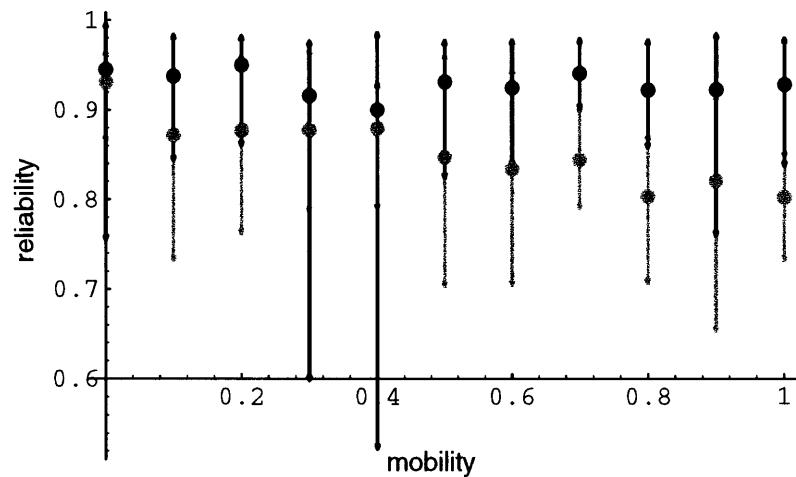


FIGURE 4-11 Network reliability vs. mobility, potential boost = 1

FIGURE 4-11 shows the effect of increased potential boost. The gray dots are values taken from FIGURE 4-10, indicating the reliability of the network with a potential boost of zero. The black dots indicate the reliability of the network with a potential boost of 1.0.

With a potential boost of 1.0, the network reliability effectively remains constant even as the mobility of the nodes increases.

4.7 radius: what are the effects of adjusting transmit radius

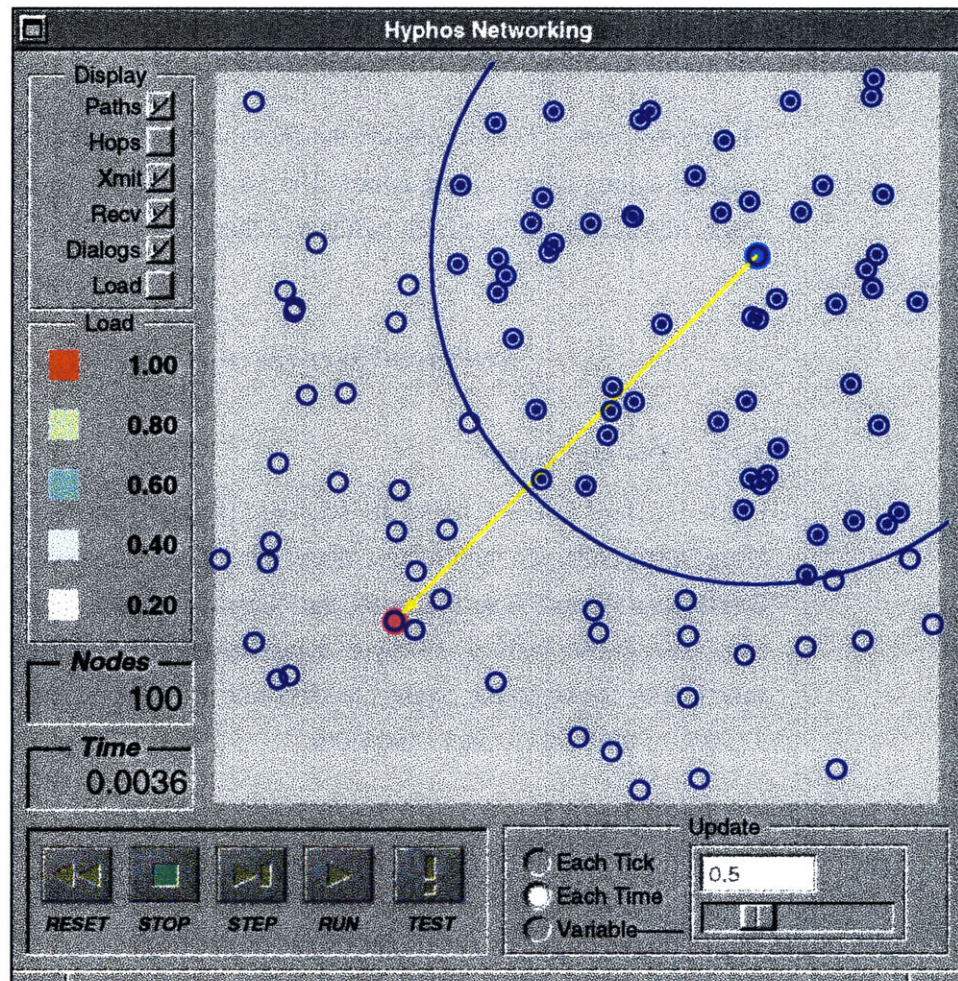


FIGURE 4-12 Testing the network as the transmitter radius varies

Assuming that all transmitters transmit with the same power and that the nodes are randomly scattered throughout the arena, what's the optimal transmit radius? If the transmit radius is too small, there may not any complete paths from an originating node to its receiving node. On the other hand, if the transmit radius is too large, then the effective bandwidth of the network decreases as intermediate nodes contend for the airwaves to forward messages.

4.7.1 Average Coverage

The average number of nodes per square meter is given as:

$$\gamma = \frac{(nodeCount - 1)}{arenaArea} \quad (EQ 1)$$

The $(nodeCount - 1)$ term reflects the fact that the transmitting node itself isn't counted among the receiving nodes. The average number of nodes covered by a single transmitter with transmit radius R is then

$$averageCoverage = \pi R^2 \gamma \quad (EQ 2)$$

4.7.2 The radius test

In the radius test, 100 nodes are scattered randomly throughout a 40x40 arena. None of the nodes are mobile. One node, the "client," originates 64 byte messages 240 times a second. A second node, the "server" replies to each message received from the client with a 64 byte acknowledgment message. In a perfectly functioning network, the aggregate data rate from the client and the server is 12.28% of the total channel bandwidth.

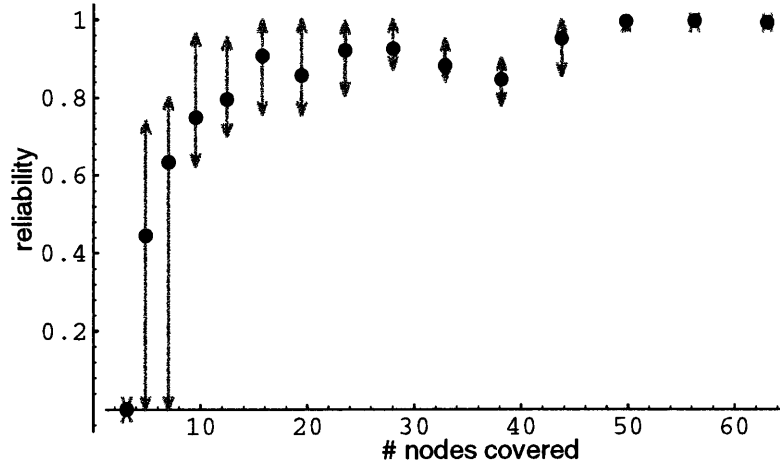


FIGURE 4-13 Network reliability vs. transmitter coverage

FIGURE 4-13 shows the network reliability as a function of transmitter coverage. As anticipated, reliability is poor when the coverage is too small; complete paths between sending and receiving nodes can't always be found. Reliability is essentially level as long as each transmitter covers at least 16 nodes on average.

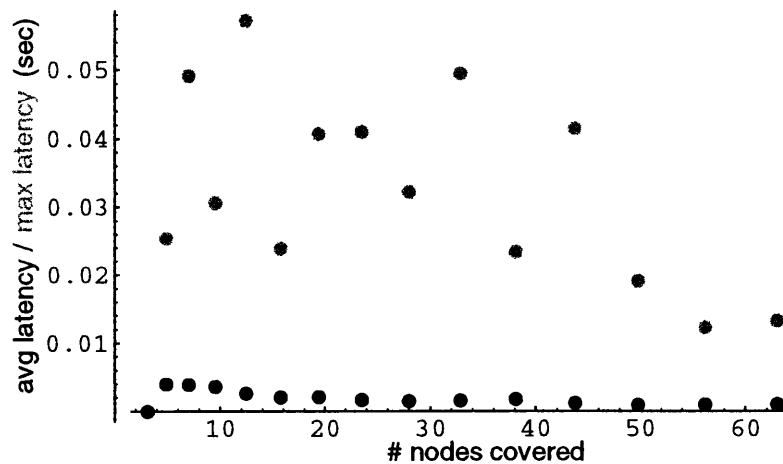


FIGURE 4-14 Latency vs. coverage

FIGURE 4-14 shows the average and maximum latency for delivering messages as the coverage varies. Due to “sluffing” (see Section 3.6, “Medium Access

Layer,” on page 34), maximum latencies remain small even under heavily loaded conditions.

4.7.3 Optimal Coverage

FIGURE 4-13 shows that an average coverage of less than 10 isn’t always sufficient to form complete connected paths throughout networks of randomly distributed nodes. Increasing the coverage above 16 nodes has no particular effect other than to increase power consumption.

One can conclude from this that for a network of randomly placed nodes, optimal network performance is attained when each transmitter covers sixteen nodes.

4.8 chatter: how do multiple dialogs affect performance?

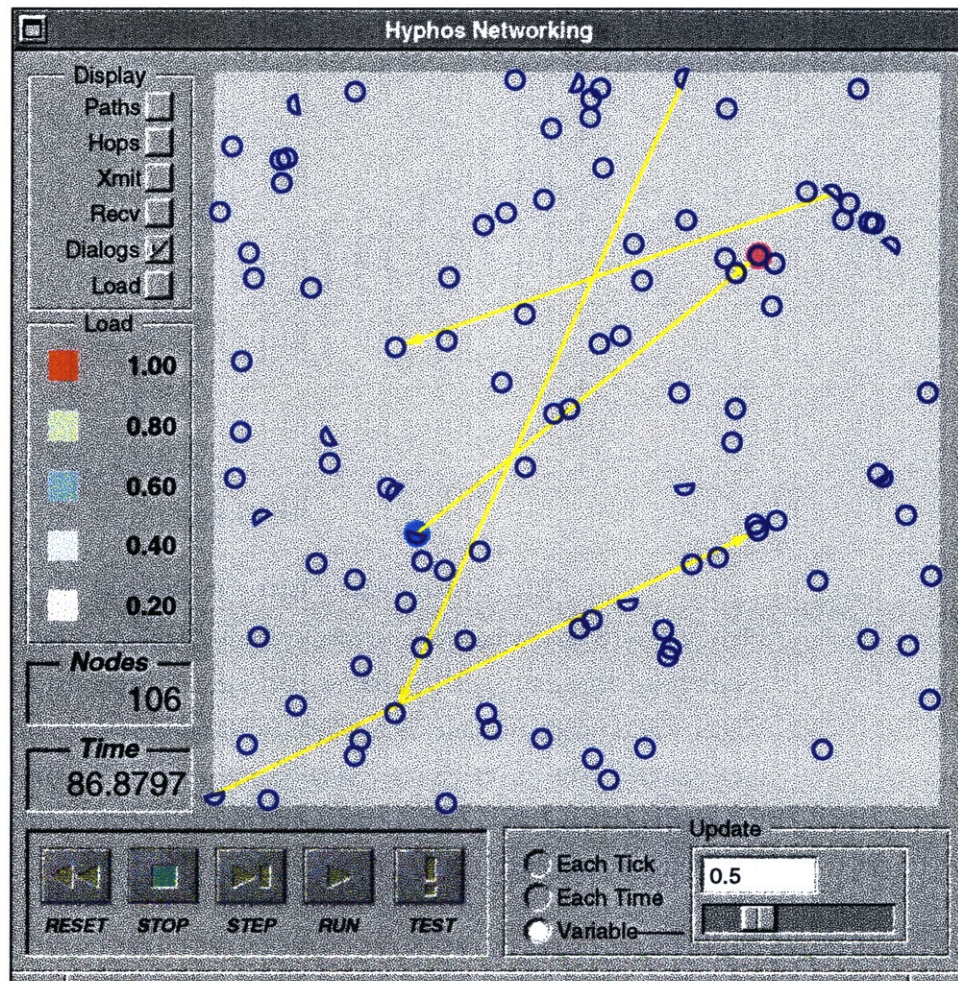


FIGURE 4-15 Testing the network with multiple “dialogs”

So far, all the tests shown have shown network performance with just one “dialog” between a single client and single server. What happens when there are multiple clients in dialog with multiple servers?

The `chatter` test examines network performance as the number of client/server pairs vary. Each client node emits 24 messages a second, contributing a client load of 0.614% to the overall network. The number of client/server pairs varies from 2

to 40; the client load ranges from 1.28% to 24.5%. In the chatter test, clients are always mobile and servers are always stationary. Of the remaining nodes, 10% are mobile; the rest are stationary.

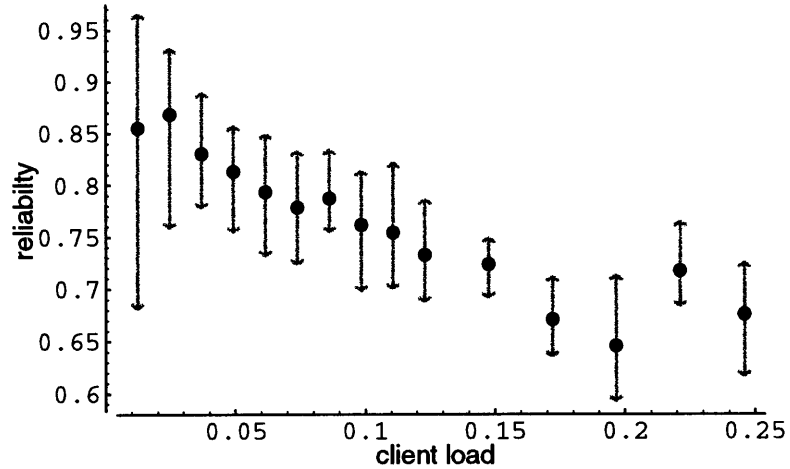


FIGURE 4-16 Reliability vs. simultaneous dialogs

FIGURE 4-16 shows network reliability as a function of the number of client/server pairs. In the single client/server load test of Section 4.4, reliability started higher, but dropped off quickly at client loads above 6%. In this chatter test, the reliability falls off much more slowly, probably because the originated traffic is uncorrelated and follows different paths through the network, leading to more distributed loading and less congestion.

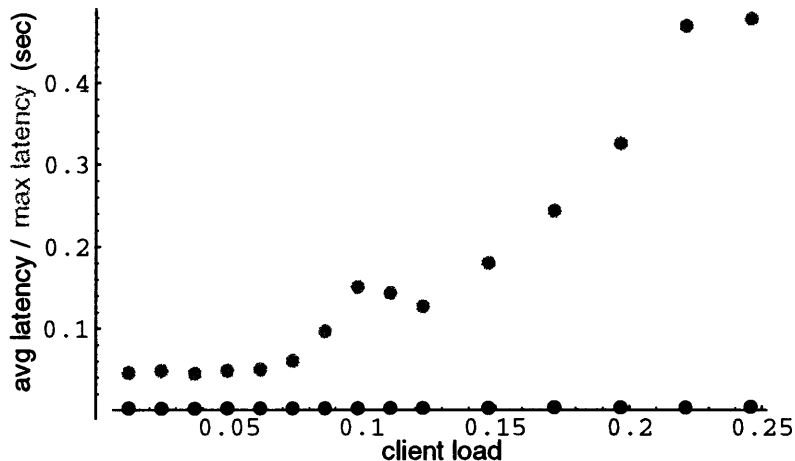


FIGURE 4-17 Latency vs. simultaneous dialogs

The graph of the latencies, FIGURE 4-17, shows an interesting result. As before, the average latency is shown with black dots, the maximum latency (over ten runs per point) is shown with gray dots. The maximum latency is almost constant at client loads up to 6%, and increases linearly thereafter to a maximum of 500 milliseconds with a client load of 24%.

It is encouraging to observe this linear behavior, since it suggests that network performance will degrade gracefully as the load increases.

4.9 server: how does one server / many clients perform?

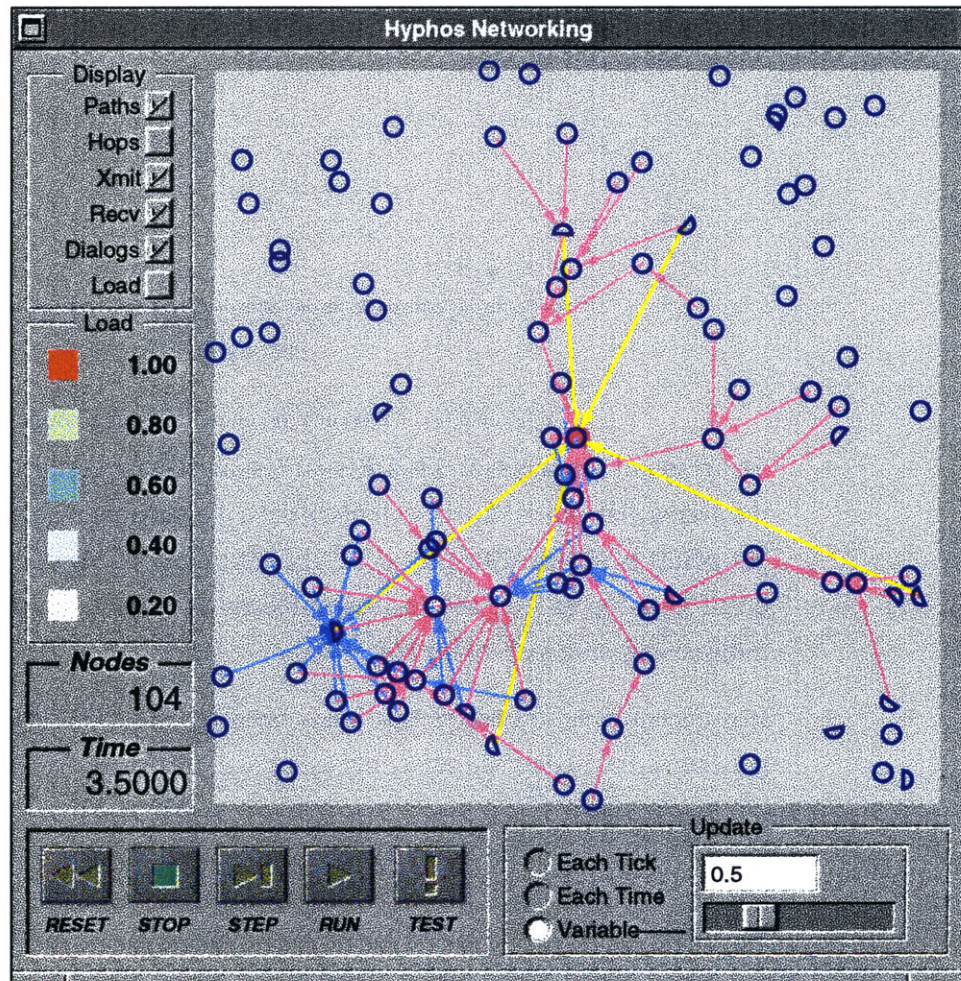


FIGURE 4-18 multiple clients talking to one server

In a typical configuration, one or more nodes may be connected to file servers or computational servers of some sort. What happens when multiple clients all attempt communication with one server?

The server test examines the effect of multiple clients “ganging up” on one server. As in the chatter test, each client node emits 24 messages a second, contributing a client load of 0.614% to the overall network. The number of clients

varies from 2 to 40; the client load ranges from 1.28% to 24.5%. Clients are always mobile, the server is always stationary, and of the remaining nodes, 10% are mobile.

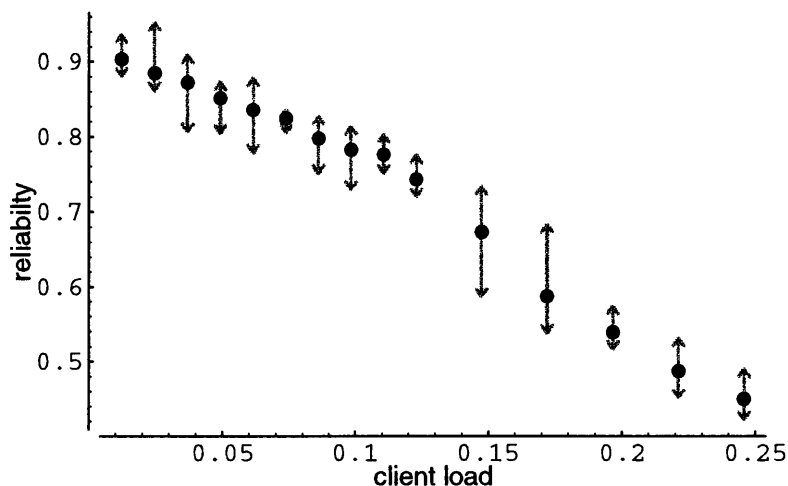


FIGURE 4-19 Reliability vs. # of clients (1 server)

FIGURE 4-19 shows the network reliability to be slightly worse than in the chatter test, but also suggests a graceful degradation of performance as the load increases.

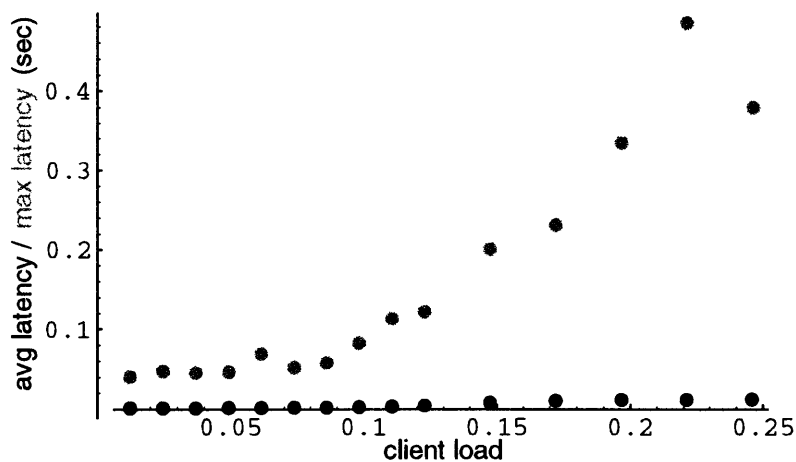


FIGURE 4-20 Latency vs. # of clients (1 server)

The latency graph, FIGURE 4-20, is almost identical to that of the chatter test: the average latency never exceeds 13 milliseconds, the maximum latency varies linearly with the number of clients and never exceeds 500 milliseconds with 40 clients hitting one server.

4.10 Performance Summary

There appear to be two “magic numbers” in Hyphos.

As long as the offered rate stays below 12% of the channel bandwidth, the network performs well under a variety of conditions. For sufficiently dense networks, latencies remain small and the network continues to perform well even when 100% of the nodes are mobile.

Network performance is best when each transmitter reaches, on average, sixteen neighbors. However, this is a “soft” number: as long as there is sufficient coverage to include every node in the network (greater than ten nodes per transmitter on average), reasonable network performance can be expected.

5.0 A revised transmitter/receiver model

In Chapter 4.0, the model used to simulate a transmitter/receiver pair corresponds to descriptions given in [Klei87]: every transmitter has perfect transmission within radius R , falling off completely outside that radius. If two transmissions overlap, both transmissions are considered to be corrupted and are discarded.

This model is both conceptually and computationally simple and is appropriate for many situations. But as [Shep95] points out, the noise floor of a network of transmitters varies dramatically with the number and density of active transmitters. The signal to noise ratio has more of an effect on a receiver's ability to receive a signal correctly than the actual transmitted power—a fact that the simple model fails to take into account.

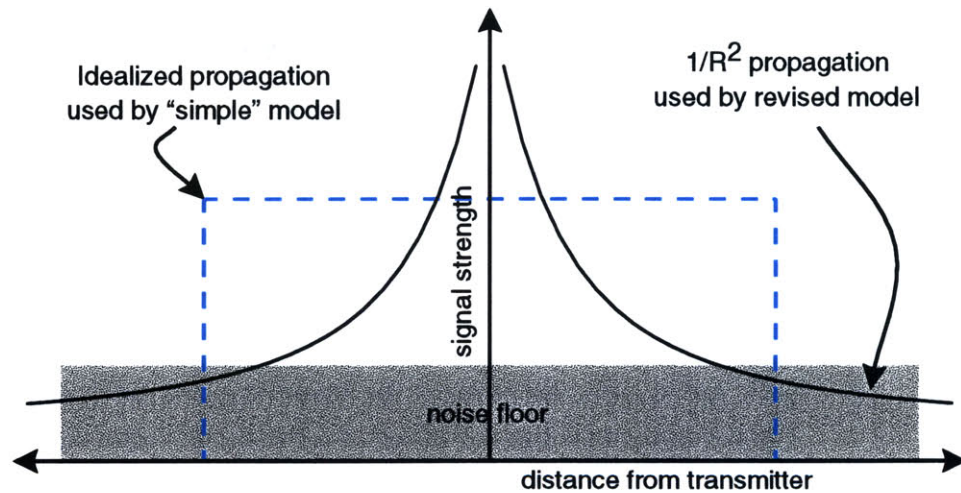


FIGURE 5-1 Simple vs. Revised Propagation Models

The dashed line in Figure 5-1 shows the “simple” model: transmitter field strength is constant within a certain distance from the transmitter, then falls to zero outside of that distance.

By contrast, a real-world transmitter exhibits a field strength that falls off approximately as $1/distance^2$, indicated by the solid lines in Figure 5-1. In order to

detect a signal, a real-world receiver requires that the field strength exceed the noise floor by some threshold as shown by the gray shading.

The rest of this chapter develops the formulae and constants associated with a revised transceiver model, intended to accurately more represent the behavior of “real-world” transmitters and receivers.

5.1 Modelling Attenuation and Noise

5.1.1 Modelling free-space attenuation

When the distance between a transmitter and receiver is significantly longer than the wavelength, the free space power received at an antenna separated by some distance d from a transmitting antenna is given by the Friis free space equation for far-field radiation [Rapp96](71):

$$P_r(d) = P_t A(d) \quad (\text{EQ 3})$$

where P_t is the transmitted power and $A(d)$ is the attenuation of the signal as a function of distance, which is defined as:

$$A(d) = \frac{G_t G_r \lambda^2}{(4\pi)^2 d^2 L} \quad (\text{EQ 4})$$

G_t and G_r are the gains of the transmitter and receiver antennae, λ is the wavelength of the signal in meters, d is the distance separating the transmitter and receiver, and L is a system loss factor not related to propagation.

5.1.2 Modelling Receiver Input Noise

Johnson Noise is the noise associated with thermal motions of electrons in a resistor; its contribution, measured in watts, at the input of a radio receiver is given by [Rapp96](566):

$$P_{therm} = kT\Delta f \quad (\text{EQ 5})$$

where k is Boltzmann's constant, T is the temperature of the receiver (degrees Kelvin), Δf is the bandwidth of the receiver (in Hertz).

In a practical implementation, there are noise sources in addition to the thermal input noise of the receiver. Environmental noise and semiconductor physics contribute to the equivalent input noise. To account for these other noise sources, we model the receiver input noise as:

$$P_{noise} = P_{therm} + P_{rcvr} \quad (\text{EQ 6})$$

where P_{rcvr} models the additional noise as equivalent input noise at the receiver.

5.1.3 Modelling Signal to Noise

In a Hyphos environment, many transmitters may be active at any given moment. Assume we define *signal* measured at a particular receiver r as the power received from transmitter t and *interference* as the contribution from all other active transmitters. Now we can define the *signal-to-noise* ratio as the signal divided by the sum of the interference plus P_{noise} .

Assume that all transmitters transmit with the same power P_t . Let D_{ij} be the distance between receiver i and transmitter j , and $A[D_{ij}]$ be the free-space attenuation as given in EQ 2. The instantaneous signal-to-noise at receiver i for a signal from transmitter k in an environment with M active transmitters is given as:

$$SNR_{ik} = \frac{P_t A [D_{ik}]}{P_{noise} + \sum_{j=1, j \neq k}^M P_t A [D_{ij}]} \quad (\text{EQ 7})$$

5.1.4 Modelling a practical receiver

Given a carrier frequency of 915 MHz ($\lambda = 0.327639$ M), a standard half-wave dipole antenna for both transmitter and receive antennae ($G_p, G_r = 1.63$), and no system loss ($L = 1.0$), we can re-write EQ 2 as:

$$A(d) = \frac{1.63 \times 1.63 \times 0.328^2}{(4\pi)^2 d^2 \times (1.0)} = \frac{0.00180612}{d^2} \quad (\text{EQ 8})$$

To compute the thermal noise of the receiver, we need to know the temperature and the bandwidth of the receiver. The system is assumed to be at room temperature ($T = 290^\circ \text{K}$). The bandwidth is assumed to be equal to the bit rate of the transmitted data, in this case 2 Mbits/second ($\Delta f = 2 \times 10^6$). The input noise from (EQ 5) will be given as:

$$P_{therm} = (1.38 \times 10^{-23}) (290) (2 \times 10^6) \approx 8.004 \dots \times 10^{-15} \quad (\text{EQ 9})$$

The equivalent noise power, 8 femtowatts, is very small indeed. Our model is much more conservative, assuming that P_{rcvr} (EQ 6) is 100 times that of the thermal noise:

$$P_{noise} \approx 8.00 \times 10^{-13} \quad (\text{EQ 10})$$

A simple FM receiver can lock onto a carrier when the signal to noise ratio exceeds 10 dB [Rapp96](219). For purposes of simulation, we assume that once the receiver has locked onto the carrier, it will remain locked on as long as the signal to noise ratio exceeds 6 dB.

5.1.5 Putting it together

A Hyphos node intended for “room sized” applications will have limited transmit range and will consume little power. To find the signal to noise ratio, we choose a transmitter power of 1 μW and substitute values from (EQ 10) into (EQ 7):

$$SNR_{ik} = \frac{10^{-6} A [D_{ik}]}{8 \times 10^{-13} + \sum_{j=1, j \neq k}^M 10^{-6} A [D_{ij}]} \quad (\text{EQ 11})$$

where $A[d] = (1.8 \times 10^{-3}) / d^2$.

With the given parameters, what is the maximum distance at which a signal will be received correctly from a single transmitter when no other transmitters are active? This can be solved by fixing the signal to noise ratio in (EQ 11) to 10 dB (the capture level of the receiver), setting M to zero (there are no other transmitters), and solving for the distance:

$$SNR_{capture} = \frac{P_t (K/d^2)}{P_{noise}} \quad (\text{EQ 12})$$

$$d^2 = \frac{P_t K}{P_{noise} SNR_{capture}} \quad (\text{EQ 13})$$

$$d = \sqrt{\frac{P_t K}{P_{noise} SNR_{capture}}} \quad (\text{EQ 14})$$

K is defined as the constant factor for free space attenuation, calculated in (EQ 8) as 1.8×10^{-3} . As previously defined, transmitter power $P_t = 1 \mu\text{W}$, receiver input noise equivalent $P_{noise} = 8 \times 10^{-15}$, minimum capture ratio $SNR_{capture} = 10$. With these values, the maximum distance between a transmitter and a receiver, in the absence of any other noise or interference, will be 15 Meters:

$$d = \sqrt{\frac{10^{-6} \times 1.8 \times 10^{-3}}{8 \times 10^{-13} \times 10}} = 15M \quad (\text{EQ 15})$$

5.2 Implications

This alternate transceiver model has two important consequences for a Hyphos network. First, reciprocity can no longer be assumed: if node A can “hear” node B, there’s no guarantee that node B can hear node A, since noise levels will be constantly varying.

Second, transmitter range can no longer be assumed to be constant. Under lightly loaded network conditions (few active transmitters), the range of each transmitter will be considerably greater than in a heavily loaded network (many active transmitters). This may not be bad; it suggests that there may be a built-in “coverage control” that provides greater coverage under light loading conditions and more focussed coverage as the load increases.

5.3 The revised transceiver model

The Hyphos simulator was extended to use the revised model of wireless communication described in the previous section.

Under the revised scheme, whenever a node in the Hyphos network initiates a transmission, the new signal to noise ratio (SNR) is computed at each receiver in the system. At each receiver, if the receiver is locked onto a signal, it will stay locked as long as the new SNR exceeds the “lock” ratio of 6 dB. Otherwise, any message being received is marked as lost. Now, if the SNR exceeds the “capture” ratio of 10 dB, the receiver locks onto the new signal.

A transmitted signal has a simulated duration equal to the number of bits in the message divided by the bit rate of the transmitter. At the conclusion of a transmitted signal, the simulator notifies each receiver that the transmission is ending. Upon this notification, if a receiver is currently locked onto the signal, the signal is marked as correctly received and is passed to the Link level for further processing.

5.4 The Results

The more precise modelling of the behavior of a network of transmitters and receivers described here requires significantly more computing cycles than the simple model described in Chapter 4.0. In the simple model, the onset of a new transmission triggers a computation of order N , proportional to the number of nodes in the network. In this revised model, a new transmission triggers a more complex computation of order N^2 . As an example of the difference, the load test

described in Chapter 4 completes in one hour of compute time on a Pentium Pro computer with 200MHz clock. The same test using the revised transceiver model requires over eight hours.

In the following sections, it's important to bear in mind that only the modelling of the transceivers has changed. The underlying Hyphos algorithms are unchanged.

5.5 load redux: when does the network hit the wall?

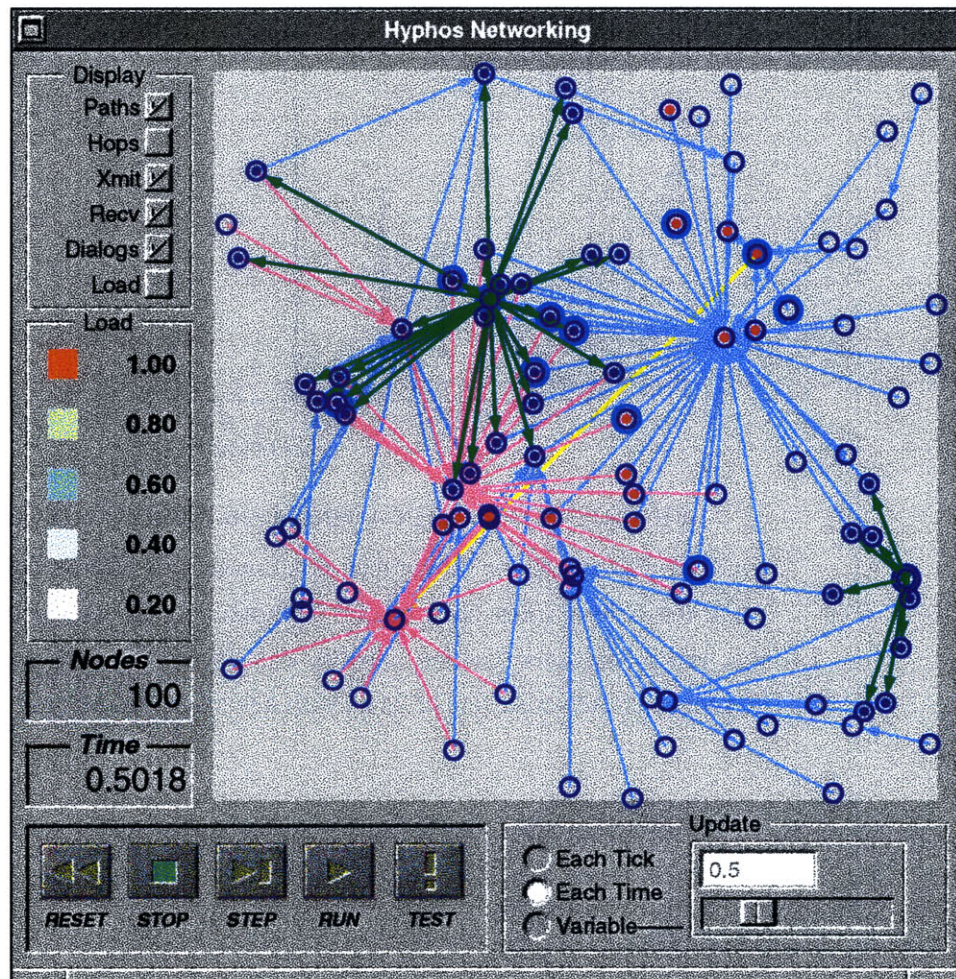


FIGURE 5-2 Performance as a function of load (alternate transceiver model)

Notice in FIGURE 5-2, there are no “transmit rings” indicating the range of the transmitting nodes. Instead, a transmitting node is indicated by a small ring encircling the node. Vectors radiating out from the node connect to those nodes that are actively “locked on” to the transmitter. In the absence of interference from other transmitters, the distance between transmitter and receiver can extend up to fifteen meters [see (EQ 15)]. But as can be seen in the figure above, transmit distance falls off in the presence of two or more active transmitters.

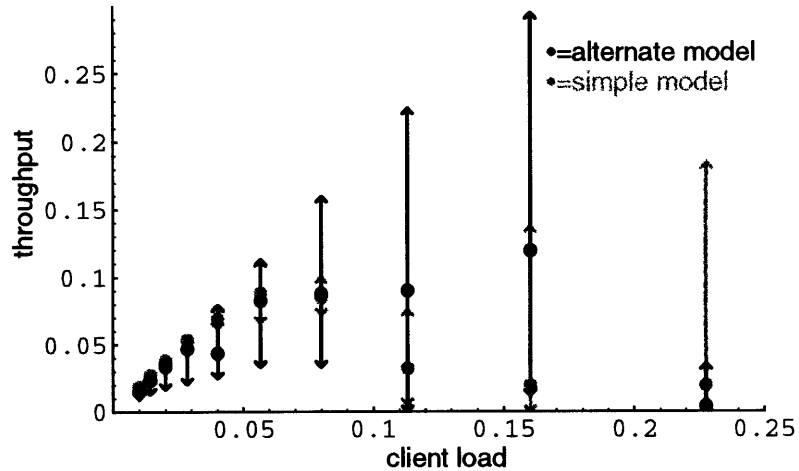


FIGURE 5-3 Throughput vs. offered rate (alternate transceiver model)

FIGURE 5-3 shows the throughput as a function of offered rate for the network using the revised transceiver model. Compared to the simple transceiver model, this graph indicates that the network can sustain a client load of 12% (or a combined client/server rate of 24%) with a resulting throughput of 10%.

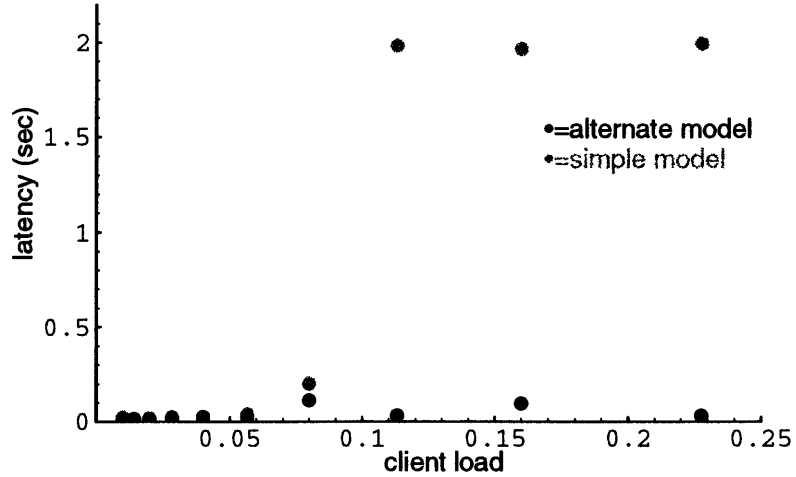


FIGURE 5-4 Maximum latency vs. offered rate (alternate transceiver model)

The latency numbers of FIGURE 5-4 shows how much a change to the transceiver model can affect the predicted results. The large latencies of the “simple” model are gone, and now never exceed 113 milliseconds.

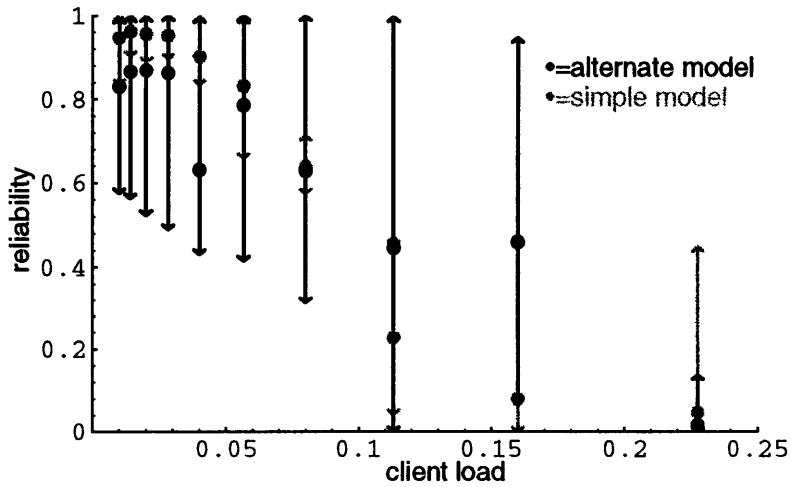


FIGURE 5-5 Reliability vs. offered rate (alternate transceiver model)

Reliability remains acceptable with client loads of 6%.

5.6 chatter redux: multiple dialogs

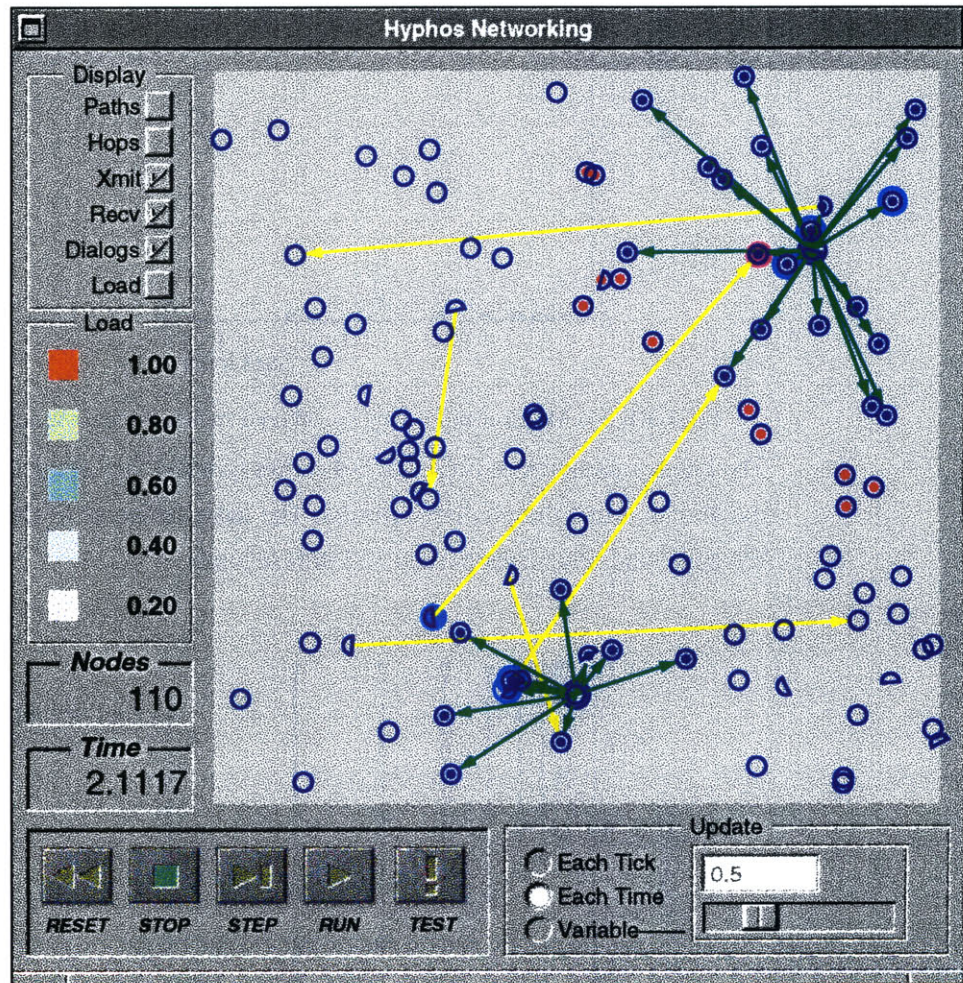


FIGURE 5-6 Multiple Dialogs(alternate transceiver model)

The chatter test was re-run using the alternate transceiver model. It was expected that increased collisions and the dropping of the reciprocity assumption (cited in Section 2.4, “Assumptions,” on page 26) would degrade performance considerably.

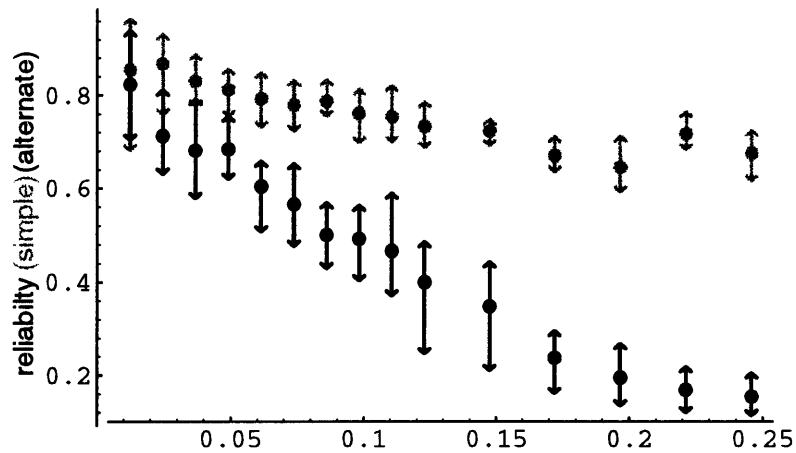


FIGURE 5-7 Reliability vs. # of dialogs (alternate transceiver model)

FIGURE 5-7 shows the reliability of the network with multiple dialogs. Except for the alternate transceiver model, all of the other parameters are identical to the chatter test in Section 4.8 on page 63: the total client load is constant 3.07% of the channel bandwidth. Indeed, the reliability has degraded considerably, although the reliability degrades smoothly.

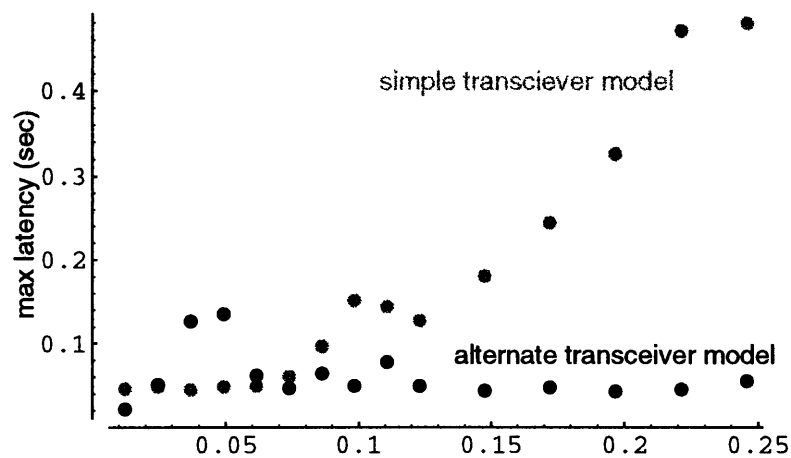


FIGURE 5-8 Latency vs. # of dialogs (alternate transceiver model)

Latencies using the alternate transceiver are comparable to the chatter test of in Section 4.8. Maximum latencies remain under 500 milliseconds.

5.7 server redux: a client/server simulation

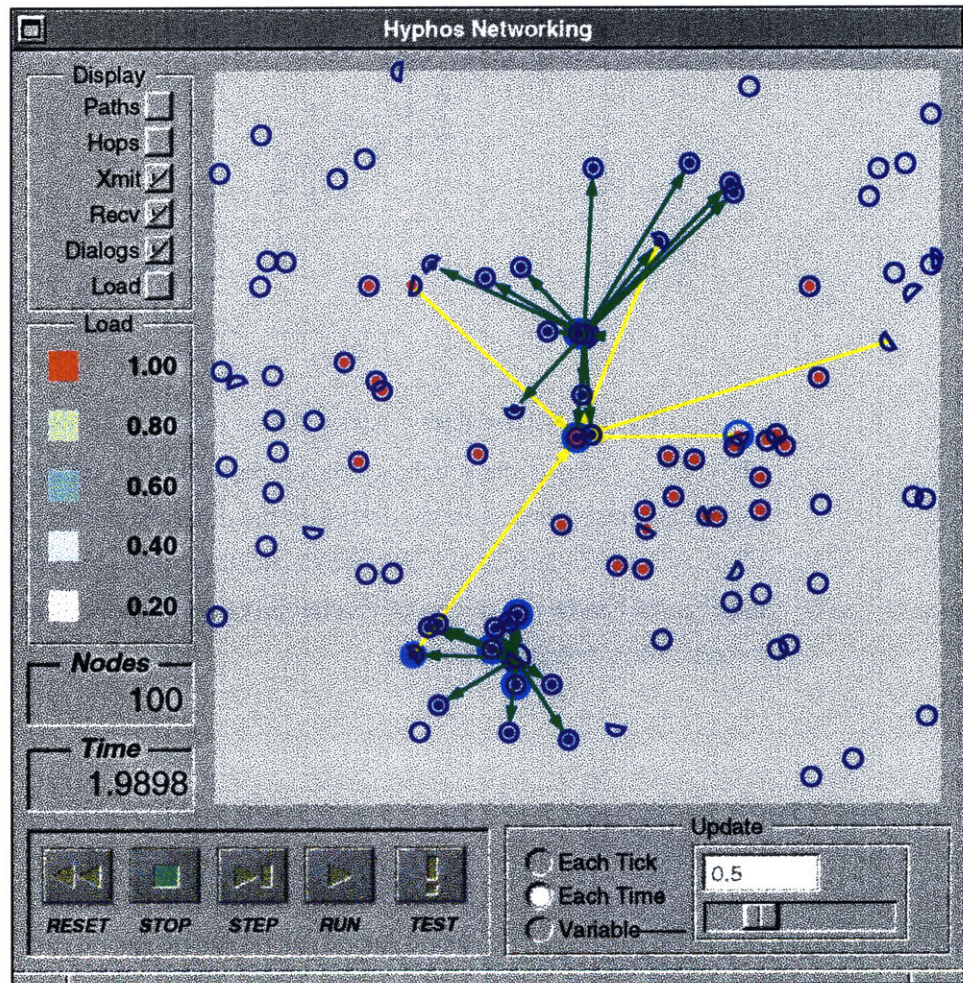


FIGURE 5-9 One server, many clients (alternate transceiver model)

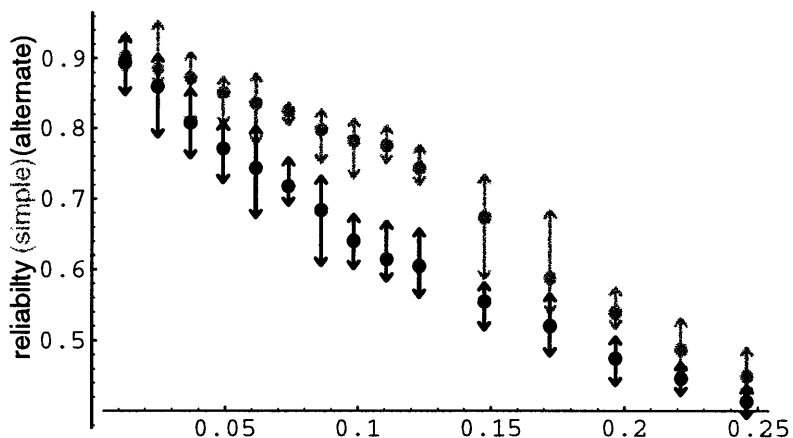


FIGURE 5-10 Reliability vs. # of clients (alternate transceiver model)

The server test using the alternate transceiver model shows performance similar to the same test using the simple transceiver model. As in the chatter test, reliability is somewhat worse than with the simple transceiver model.

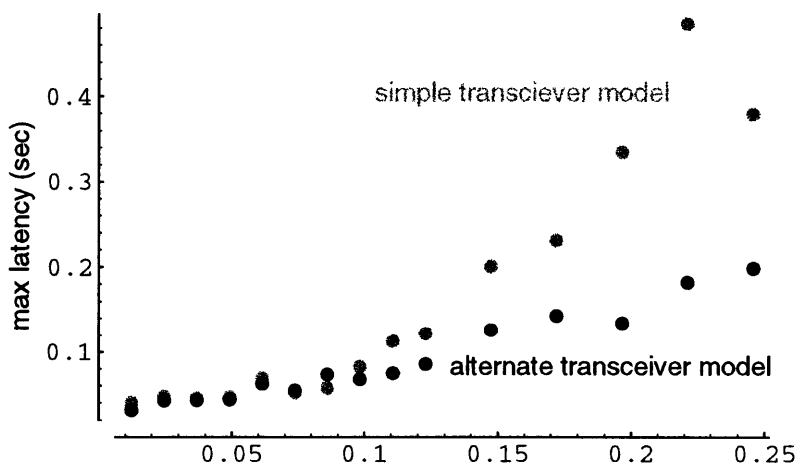


FIGURE 5-11 Latency vs. # of clients (alternate transceiver model)

The absolute maximum latency observed for client loads up to 25% is under 500 milliseconds.

6.0 Conclusions

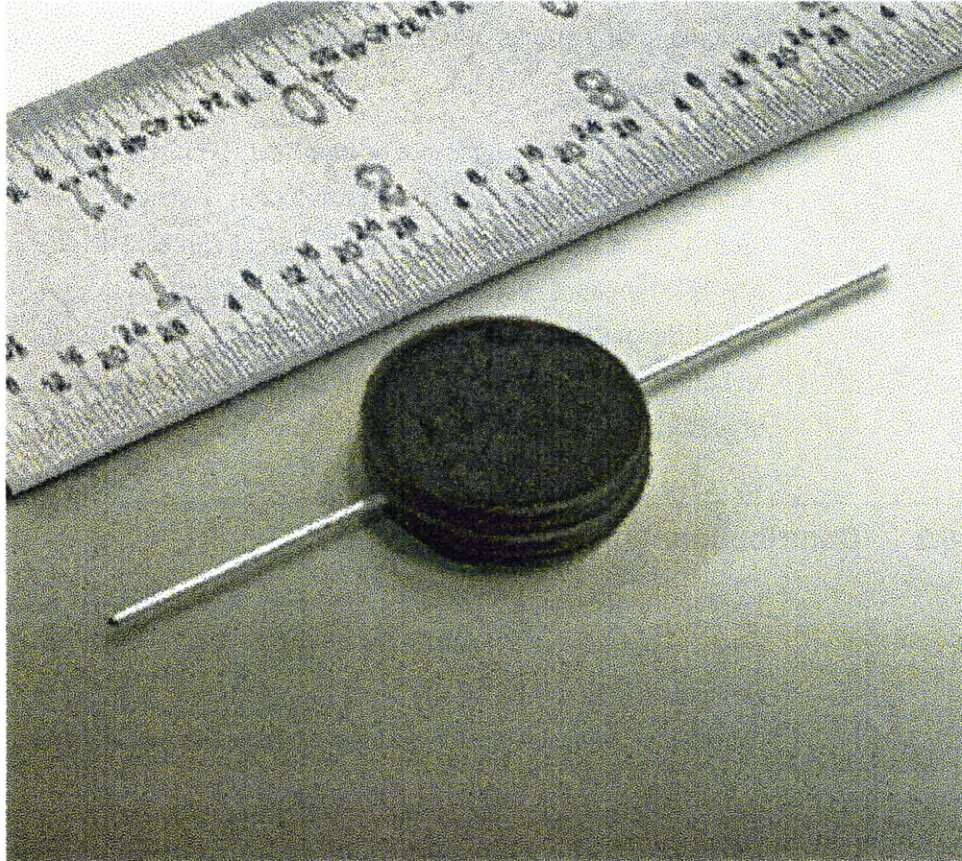


FIGURE 6-1 Artist's conception of a Hyphos node

The above figure depicts a proposed package for a Hyphos node. The wires extending on either side form a 915 MHz quarter wave dipole antenna, 80 millimeters from one end to the other. The bottom half of the package contains a small battery, the top half contains the logic circuitry. Stuck on a wall or a door frame, these nodes can be deployed as required to attain adequate coverage.

6.1 Summary

This work has introduced Hyphos, self-organizing, wireless network. The network is designed with an eye towards very low cost per node. By assuming multi-hop,

wireless communication, we can create nodes that consume very little power. By designing routing algorithms that require very little state and computational power to maintain, we can fabricate nodes on a single integrated circuit. By using self-organizing or “ad-hoc” routing strategies, we create networks that are easy to set up and maintain.

Routing and Medium Access algorithms for Hyphos have been designed and their behavior has been characterized in simulation. The network will support data rates up to 12% of the channel bandwidth without appreciable degradation. A Hyphos network functions well even when all of the nodes in the system are mobile—overall reliability drops only by 10% compared to a fully stationary system, and a technique called “Potential Boost” is shown which eliminates any remaining drop in performance.

6.2 Contributions

Hyphos defines a new model of networking: individual nodes are physically small, consume little power, and organize themselves into sensible networks. Devices that have historically been too small, too mobile, or too price sensitive can now be connected: clocks, light switches, thermostats, shipping pallets, and toys can all find expression on digital networks.

Hyphos introduces *Contour Routing*, a novel approach to routing in a multi-hop wireless network. Since Contour Routing doesn’t require knowledge of particular neighboring nodes, a Hyphos network can dispense with periodic “ping” messages. In addition, Contour Routing shows good reliability in networks where nodes are mobile; the act of passing messages is sufficient to maintain consistent routing information.

6.3 Future Work

The ideas presented here have only scratched the surface and have themselves generated a new list of questions that warrant further investigation..

6.3.1 Physical Layer

The largest unresolved issue is attaining a deep understanding of how low-power wireless communication systems really perform in dense networks. The differences in the test results between Chapter 4 (simple model) and Chapter 5 (alternate model) point out the sensitivity of the Hyphos network to the characteristics of the wireless link.

Rather than putting time into more simulation, we have already planned a “Transceiver Network Test Bed” project, which will build dozens of wireless transceivers with just enough of a controller on each transceiver to send and receive data packets and to collect statistics on the performance of each transceiver. The statistics gathered at this level will be incorporated to the Hyphos simulator for further work.

[Shep95] shows that spread-spectrum systems with power control can produce good results. If such transceivers can be built economically, then they would be good candidates for the Physical Layer of Hyphos.

The algorithms shown in this work assume a single-channel, half-duplex transceiver model. A multiple channel transceiver would offer the possibility of advanced Medium Access control.

6.3.2 Link Layer

The Link Layer could be modified to provide a measure of local congestion by monitoring the carrier detect of the Physical Layer. This load measurement could then be used to control the relay cost of the node: as the airwaves become congested, the node would advertise a higher relay cost and as a result, future messages would tend to “flow around” the congested area.

It would be worth testing a “passive acknowledgment” system. Once the Link Layer relays a message to its neighbor, if the Link Layer “hears” that the message has been broadcast by its neighbor, then it knows that the message was received correctly. If it doesn’t hear the relayed message, then the Link Layer can retransmit the message without involving error recovery from higher networking levels. This assumes reciprocity of transmitter/receiver systems.

6.3.3 Medium Access Layer

The “sluffing” mechanism described in Section 3.6, “Medium Access Layer,” on page 34 is effective at eliminating long latencies, but it’s at the expense of some network reliability: some packets that would otherwise have reached their destination intact are lost. There may be a way to dynamically detect when it’s appropriate to sluff backlogged messages in order to achieve the dual goals of low latency and high reliability.

6.3.4 Network Layer

Currently, all routing choices are made in terms of *routing cost*, which is primarily determined by the hop count. A more complete system would incorporate a *confidence* factor: a new, unexplored route would advertise a low confidence, a stable route that has been in use for a long time would advertise a correspondingly high confidence. This approach could increase throughput and reduce latencies by homing in on a few select nodes that are known to be able to relay the message. Over time, stationary nodes will form routes with a higher confidence level than those formed by mobile nodes.

Hyphos, as defined, assumes a completely flat addressing scheme. A system with multiple transmitter channels could take on some of the characteristics of a cellular system with the goal of establishing a hierarchical addressing scheme. Alternatively, a Hyphos network could organize itself into “zones” using the reaction-diffusion techniques described in [Abel95].

The natural branching nature of communications in a mesh of wireless nodes offers great promise for multicast architectures: each Hyphos node can determine if it’s on a path between a multicast server and one or more multicast clients. By dynamically controlling the size of its data caches, it can reduce the amount of “upstream” traffic required for error recovery.

6.3.5 Transport Layer

The model for simulating traffic at the Transport Layer is simplistic. The results of the simulator could be trusted better if a more sophisticated algorithm for generating network traffic patterns were employed.

It would be worthwhile implementing TCP, UDP, or some form of “IP-Lite” [Poor96] protocol atop the transport layer.

6.4 Acknowledgments

One year ago I was a rank neophyte in the field of networking and communications. Any progress I’ve made in the field must be attributed to a host of people I’ve met along the way; I’ll do my best to thank a few of them here.

First, I’d like to thank Mike Hawley, not just for his role as advisor, but also for being a good mentor and friend. When I first arrived at MIT, I gave Mike the challenge to get me to “think big” and to work outside the doctrine of pragmatism I’d learned through years of work in the commercial world. Mike rose to the challenge and has pushed me, gently and with great humor, to stretch my imagination well beyond its previous boundaries.

I’ve had the great fortune to meet and work with Dave Morgan. Dave is the official liaison for the Motorola Fellows Program, a program that has provided me with very tangible and much appreciated support. But Dave’s support has extended much further than his official duties. Early on in my work, Dave gave me a jump start in the field by introducing me to people, research papers, and industry standards. While I was still learning the fundamentals, Dave’s ongoing encouragement gave me the confidence to press on. Subsequently, he has become an effective exponent of my work both within and without Motorola and has opened many doors.

My readers, Andy Lippman and Steve McGeady, both took substantial time from their schedules to review and discuss my work as it evolved. They challenged my assumptions, forced me to defend my claims, argued alternate views, and helped me to differentiate the substantial and the unimportant aspects of my work. Their detailed comments were invaluable in shaping the written thesis.

David Tennenhouse graciously agreed to be an “associate reader”. As an expert in networking, David helped me get oriented quickly in the early phases of my work, and pointed me towards numerous experts in the field. I fired off many e-mail

messages starting with “David Tennenhouse gave me your name...” and every one elicited a response. David also impressed upon me the fact that reducing theory to practice by building hardware would differentiate this work from any purely theoretical research. The circuit design is now underway.

Tim Sheppard, a recent MIT EECS graduate, dedicated several hours of telephone and e-mail time to tutor me about low-level RF communication theory. Tim’s own thesis and subsequent communications convinced me to re-examine the model I was using in my simulation; Chapter 5 of this thesis, “A revised transmitter/receiver model” is a direct consequence of Tim’s influence.

I was pleasantly surprised at the number of networking gurus from outside of MIT who freely shared their comments, offered advice, sent pointers to references and mailed copies of research papers. Scott Corson (University of Maryland), Rohit Dube (University of Maryland), Randy Katz (U. C. Berkeley), Barry Leiner (MCC), William Kaiser (UCLA), Radia Perlman (Novel), Keith Scott (UCLA), Mark Weiser (Xerox) all offered valuable help.

My colleagues and cohorts within MIT made the day-to-day research a pleasure. Ongoing discussions with Manish Tuteja and Ben Denckla helped shape key ideas in this work. Roger Kermode and Henry Holtzman graciously answered all of my questions about networking standards and practice. Members of Neil Geshenfeld’s group, notably Joe Paradiso, Rehmi Post, and Matt Reynolds tutored me on low-level electronics and RF issues. Olin Shivers gave me a kick in the psychic pants when I needed one. Mike Goertz, newly arrived at MIT, laid the foundations for the first hardware prototypes.

Finally, expert editing advice and wordsmithing were provided by John Underkoffler and Stacy Koprince. Any remaining typographical errors and split infinitives are mine.

7.0 References

- [Abel95] Hal Abelson, Tom Knight, Gerald Sussman. "Amorphous Computing." MIT LCS internal White Paper. Draft of October 14, 1995.
- [Bert92] Dimitri Bertsekas, Robert Gallager. "Data Networks" Second Edition. *Prentice Hall* Englewood Cliffs, New Jersey, 1992.
- [Bult96] K. Bult, A. Burstein, D. Chang, M. Dong, M. Dielling, E. Kruglick, J. Ho, F. Lin, T. H. Lin, W. J. Kaiser, R. Mukai, P. Nelson, F. L. Newburg, K. S. J. Pister, G. Pottie, H. Sanchez, O. M. Stafsuff, K. B. Tan, C. M. Ward, G. Yng, S. Xue, H. Marcy, J. Yao. "Wireless Integrated Microsensors." *Proceedings of the 1996 Hilton Head Transducers Conference*, June 1996.
- [Carl86] A. Bruce Carlson. "Communication Systems," Third Edition. McGraw-Hill Book Company, 1986.
- [Carv96] Phillip Carvey. "BodyLAN." *IEEE Circuits and Devices*, V4 No 12 July 1996
- [Cors95] M. S. Corson and A. Ephremides, "A Distributed Routing Algorithm for Mobile Wireless Networks," *ACM/Baltzer Journal on Wireless Networks*, Vol. 1, No. 1, pp. 61-82, February 1995.
- [Deme94] Alan Demers, Scott Elrod, Christopher Kantarkiev, Edward Richley. "A Nano-Cellular Local Area Network Using Near-Field RF Coupling." *Internal publication CSL-94-8* Xerox Corporation, Palo Alto Research Center. October 1994.
- [Dube96] Rohit Dube, Cynthia D. Rais, Kuang-Yeh Wang, Satish K. Tripathi. "Signal Stability based Adaptive Routing (SSA) for Ad-Hoc Mobile Networks." *IEEE Personal Communications* pp. 36-45. February 1997.
- [Garc95] J. J. Garcia-Lune-Aceves, Jochens Behrens, "Distributed, Scalable Routing Based on Vectors of Link States," *IEEE Journal on Selected Areas in Communications*, Vol. 13, No. 8, pp 1383-1395, October, 1995.
- [John94] David B. Johnson. "Routing in Ad Hoc Networks of Mobile Hosts," *IEEE Workshop on Mobile Computing Systems and Applications*, December 1994
- [John96] David B. Johnson, David A. Maltz. "Dynamic Source Routing in Ad Hoc Wireless Networks." In *Mobile Computing*, edited by Tomasz Imielinski and Hank Korth, Kluwer Academic Publishers, 1996.
- [Klei87] Leonard Kleinrock and John Silvester. "Spatial reuse in multihop packet radio networks." *Proceedings of the IEEE*, 75(1):156-167, January 1987.

- [Laue95] Gregory S. Lauer. "Packet-radio Routing." In *Routing in Communications Networks*, edited by Martha SE. Steenstrup, chapter 11, pages 55-76. Prentice-Hall, Englewood Cliffs, New Jersey, 1995.
- [Metc76] Robert M. Metcalfe and David R. Boggs. "Ethernet: Distributed packet switching for local computer networks." *Communications of the ACM* 19,7 July 1976, pp 395-404.
- [Park97] V. Park, M.S. Corson. "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks," *Proceedings IEEE INFOCOM*, April 1997.
- [Perl92] Radia Perlman. "Interconnections: Bridges and Routers." Addison-Wesley Publishing Company 1992.
- [Poor96] Robert Poor, Rehmi Post, et al. "The Filament Chip: a lightweight connection to networks." MIT Media Lab, Things That Think Consortium. Internal publication, October 1996, available on-line as <http://ttd.media.mit.edu/pia/Research/Filament/index.html>
- [Rapp96] Theodore S. Rappaport. "Wireless Communications, Principals and Practice." Prentice-Hall Inc., Upper Saddle River, New Jersey, 1995.
- [Schr91] Michael D. Schroder, Andrew D. Birrell, Michael Burrows, Hal Murray, Roger M. Needham, Thomas L. Rodeheffer, Edwin H. Satterthwaite, Charles P. Thacker. "Autonet: A High-Speed, Self-Configuring Local Area Network Using Point-to-Point Links." *IEEE Journal on Selected Areas in Communications* Vol. 9, No. 8, pp. 1318-1335, October 1991
- [Scot95] Keith Scott and Nicholas Bambos. "The Self-Organizing Wireless Adaptive Network (SWAN) Protocol for Communication Among Mobile Users." *Globe-com '95*, pp. 355-359, 1995
- [Shep95] Timothy Jason Shepard. Decentralized Channel Management in Scalable Multihop Spread-Spectrum Packet Radio Networks. MIT, EECS Thesis, 1995
- [Tane96] Andrew S. Tanenbaum. "Computer Networks." Third Edition. Prentice Hall PTR, Upper Saddle River, New Jersey, 1996.
- [Tute97] Manish Tuteja. "AnchoredDisplays: The Web on Walls," *Proceedings of Conference on Human Factors in Computing Systems (CHI '97)*, ACM, Atlanta, March 1997.
- [Wolf88] Stephen Wolfram. "Mathematica™: A System for Doing Mathematics by Computer." Addison-Wesley Publishing Company. 1988