2004

Bluetooth Enabled Ad-hoc Networks: Performance Evaluation of a Self-healing Scatternet Formation Protocol

Rade Trimceski

Michigan Technological University

Bluetooth Enabled Ad-hoc Networks: Performance Evaluation of a Self-healing Scatternet Formation Protocol

by Rade Trimceski

A THESIS submitted in partial fulfillment of the requirements for the degree of:
MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

MICHIGAN TECHNOLOGICAL UNIVERSITY
2004

Attribution-ShareAlike 1.0

License

THE WORK (AS DEFINED BELOW) IS PROVIDED UNDER THE TERMS OF THIS CREATIVE COMMONS PUBLIC LICENSE ("CCPL" OR "LICENSE"). THE WORK IS PROTECTED BY COPYRIGHT AND/OR OTHER APPLICABLE LAW. ANY USE OF THE WORK OTHER THAN AS AUTHORIZED UNDER THIS LICENSE IS PROHIBITED.

BY EXERCISING ANY RIGHTS TO THE WORK PROVIDED HERE, YOU ACCEPT AND AGREE TO BE BOUND BY THE TERMS OF THIS LICENSE. THE LICENSOR GRANTS YOU THE RIGHTS CONTAINED HERE IN CONSIDERATION OF YOUR ACCEPTANCE OF SUCH TERMS AND CONDITIONS.

1. Definitions

1. "Collective Work" means a work, such as a periodical issue, anthology or encyclopedia, in which the Work in its entirety in unmodified form, along with a number of other contributions, constituting separate and independent works in themselves, are assembled into a collective whole. A work that constitutes a Collective Work will not be considered a Derivative Work (as defined below) for the purposes of this License.
2. "Derivative Work" means a work based upon the Work or upon the Work and other pre-existing works, such as a translation, musical arrangement, dramatization, fictionalization, motion picture version, sound recording, art reproduction, abridgment, condensation, or any other form in which the Work may be recast, transformed, or adapted, except that a work that constitutes a Collective Work will not be considered a Derivative Work for the purpose of this License.
3. "Licensor" means the individual or entity that offers the Work under the terms of this License.
4. "Original Author" means the individual or entity who created the Work.
5. "Work" means the copyrightable work of authorship offered under the terms of this License.
6. "You" means an individual or entity exercising rights under this License who has not previously violated the terms of this License with respect to the Work, or who has received express permission from the Licensor to exercise rights under this License despite a previous violation.

2. Fair Use Rights. Nothing in this license is intended to reduce, limit, or restrict any rights arising from fair use, first sale or other limitations on the exclusive rights of the copyright owner under copyright law or other applicable laws.

3. License Grant. Subject to the terms and conditions of this License, Licensor hereby grants You a worldwide, royalty-free, non-exclusive, perpetual (for the duration of the applicable copyright) license to exercise the rights in the Work as stated below:

1. to reproduce the Work, to incorporate the Work into one or more Collective Works, and to reproduce the Work as incorporated in the Collective Works;
2. to create and reproduce Derivative Works;
3. to distribute copies or phonorecords of, display publicly, perform publicly, and perform publicly by means of a digital audio transmission the Work including as incorporated in Collective Works;
4. to distribute copies or phonorecords of, display publicly, perform publicly, and perform publicly by means of a digital audio transmission Derivative Works;

The above rights may be exercised in all media and formats whether now known or hereafter devised. The above rights include the right to make such modifications as are technically necessary to exercise the rights in other media and formats. All rights not expressly granted by Licensor are hereby reserved.

4. Restrictions. The license granted in Section 3 above is expressly made subject to and limited by the following restrictions:

1. You may distribute, publicly display, publicly perform, or publicly digitally perform the Work only under the terms of this License, and You must include a copy of, or the Uniform Resource Identifier for, this License with every copy or phonorecord of the Work You distribute, publicly display, publicly perform, or

publicly digitally perform. You may not offer or impose any terms on the Work that alter or restrict the terms of this License or the recipients' exercise of the rights granted hereunder. You may not sublicense the Work. You must keep intact all notices that refer to this License and to the disclaimer of warranties. You may not distribute, publicly display, publicly perform, or publicly digitally perform the Work with any technological measures that control access or use of the Work in a manner inconsistent with the terms of this License Agreement. The above applies to the Work as incorporated in a Collective Work, but this does not require the Collective Work apart from the Work itself to be made subject to the terms of this License. If You create a Collective Work, upon notice from any Licensor You must, to the extent practicable, remove from the Collective Work any reference to such Licensor or the Original Author, as requested. If You create a Derivative Work, upon notice from any Licensor You must, to the extent practicable, remove from the Derivative Work any reference to such Licensor or the Original Author, as requested.

2. You may distribute, publicly display, publicly perform, or publicly digitally perform a Derivative Work only under the terms of this License, and You must include a copy of, or the Uniform Resource Identifier for, this License with every copy or phonorecord of each Derivative Work You distribute, publicly display, publicly perform, or publicly digitally perform. You may not offer or impose any terms on the Derivative Works that alter or restrict the terms of this License or the recipients' exercise of the rights granted hereunder, and You must keep intact all notices that refer to this License and to the disclaimer of warranties. You may not distribute, publicly display, publicly perform, or publicly digitally perform the Derivative Work with any technological measures that control access or use of the Work in a manner inconsistent with the terms of this License Agreement. The above applies to the Derivative Work as incorporated in a Collective Work, but this does not require the Collective Work apart from the Derivative Work itself to be made subject to the terms of this License.

3. If you distribute, publicly display, publicly perform, or publicly digitally perform the Work or any Derivative Works or Collective Works, You must keep intact all copyright notices for the Work and give the Original Author credit reasonable to the medium or means You are utilizing by conveying the name (or pseudonym if applicable) of the Original Author if supplied; the title of the Work if supplied; in the case of a Derivative Work, a credit identifying the use of the Work in the Derivative Work (e.g., "French translation of the Work by Original Author," or "Screenplay based on original Work by Original Author"). Such credit may be implemented in any reasonable manner; provided, however, that in the case of a Derivative Work or Collective Work, at a minimum such credit will appear where any other comparable authorship credit appears and in a manner at least as prominent as such other comparable authorship credit.

5. Representations, Warranties and Disclaimer

1. By offering the Work for public release under this License, Licensor represents and warrants that, to the best of Licensor's knowledge after reasonable inquiry:

    1. Licensor has secured all rights in the Work necessary to grant the license rights hereunder and to permit the lawful exercise of the rights granted hereunder without You having any obligation to pay any royalties, compulsory license fees, residuals or any other payments;

    2. The Work does not infringe the copyright, trademark, publicity rights, common law rights or any other right of any third party or constitute defamation, invasion of privacy or other tortious injury to any third party.

2. EXCEPT AS EXPRESSLY STATED IN THIS LICENSE OR OTHERWISE AGREED IN WRITING OR REQUIRED BY APPLICABLE LAW, THE WORK IS LICENSED ON AN "AS IS" BASIS, WITHOUT WARRANTIES OF ANY KIND, EITHER EXPRESS OR IMPLIED INCLUDING, WITHOUT LIMITATION, ANY WARRANTIES REGARDING THE CONTENTS OR ACCURACY OF THE WORK.

6. Limitation on Liability. EXCEPT TO THE EXTENT REQUIRED BY APPLICABLE LAW, AND EXCEPT FOR DAMAGES ARISING FROM LIABILITY TO A THIRD PARTY RESULTING FROM BREACH OF THE WARRANTIES IN SECTION 5, IN NO EVENT WILL LICENSOR BE LIABLE TO YOU ON ANY LEGAL THEORY FOR ANY SPECIAL, INCIDENTAL, CONSEQUENTIAL, PUNITIVE OR EXEMPLARY DAMAGES ARISING OUT OF THIS LICENSE OR THE USE OF THE WORK, EVEN IF LICENSOR HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

7. Termination

1. This License and the rights granted hereunder will terminate automatically upon any breach by You of the terms of this License. Individuals or entities who have received Derivative Works or Collective Works

from You under this License, however, will not have their licenses terminated provided such individuals or entities remain in full compliance with those licenses. Sections 1, 2, 5, 6, 7, and 8 will survive any termination of this License.

2. Subject to the above terms and conditions, the license granted here is perpetual (for the duration of the applicable copyright in the Work). Notwithstanding the above, Licensor reserves the right to release the Work under different license terms or to stop distributing the Work at any time; provided, however that any such election will not serve to withdraw this License (or any other license that has been, or is required to be, granted under the terms of this License), and this License will continue in full force and effect unless terminated as stated above.

8. Miscellaneous

1. Each time You distribute or publicly digitally perform the Work or a Collective Work, the Licensor offers to the recipient a license to the Work on the same terms and conditions as the license granted to You under this License.
2. Each time You distribute or publicly digitally perform a Derivative Work, Licensor offers to the recipient a license to the original Work on the same terms and conditions as the license granted to You under this License.
3. If any provision of this License is invalid or unenforceable under applicable law, it shall not affect the validity or enforceability of the remainder of the terms of this License, and without further action by the parties to this agreement, such provision shall be reformed to the minimum extent necessary to make such provision valid and enforceable.
4. No term or provision of this License shall be deemed waived and no breach consented to unless such waiver or consent shall be in writing and signed by the party to be charged with such waiver or consent.
5. This License constitutes the entire agreement between the parties with respect to the Work licensed here. There are no understandings, agreements or representations with respect to the Work not specified here. Licensor shall not be bound by any additional provisions that may appear in any communication from You. This License may not be modified without the mutual written agreement of the Licensor and You.

Creative Commons is not a party to this License, and makes no warranty whatsoever in connection with the Work. Creative Commons will not be liable to You or any party on any legal theory for any damages whatsoever, including without limitation any general, special, incidental or consequential damages arising in connection to this license. Notwithstanding the foregoing two (2) sentences, if Creative Commons has expressly identified itself as the Licensor hereunder, it shall have all rights and obligations of Licensor.

Except for the limited purpose of indicating to the public that the Work is licensed under the CCPL, neither party will use the trademark "Creative Commons" or any related trademark or logo of Creative Commons without the prior written consent of Creative Commons. Any permitted use will be in compliance with Creative Commons' then-current trademark usage guidelines, as may be published on its website or otherwise made available upon request from time to time.

Creative Commons may be contacted at http://creativecommons.org/.

This thesis, "Bluetooth Enabled Ad-hoc Networks: Performance evaluation of a Self-healing Scatternet Formation Protocol," is hereby approved in partial fulfillment of the requirements for the Degree of MASTER OF SCIENCE IN ELECTRICAL ENGINEERING.

DEPARTMENT:

Electrical and Computer Engineering

Signatures:

Thesis Advisor _____

Dr. Brian T. Davis

Department Chair _____

Dr. Timothy J. Schulz

Date _____

Abstract

Bluetooth wireless technology is a robust short-range communications system designed for low power (10 meter range) and low cost. It operates in the 2.4 GHz Industrial Scientific Medical (ISM) band and it employs two techniques for minimizing interference: a frequency hopping scheme which nominally splits the 2.400 - 2.485 GHz band in 79 frequency channels and a time division duplex (TDD) scheme which is used to switch to a new frequency channel on 625 μs boundaries. During normal operation a Bluetooth device will be active on a different frequency channel every 625 μs, thus minimizing the chances of continuous interference impacting the performance of the system.

The smallest unit of a Bluetooth network is called a piconet, and can have a maximum of eight nodes. Bluetooth devices must assume one of two roles within a piconet, master or slave, where the master governs quality of service and the frequency hopping schedule within the piconet and the slave follows the master's schedule. A piconet must have a single master and up to 7 active slaves. By allowing devices to have roles in multiple piconets through time multiplexing, i.e. slave/slave or master/slave, the Bluetooth technology allows for interconnecting multiple piconets into larger networks called scatternets.

The Bluetooth technology is explored in the context of enabling ad-hoc networks. The Bluetooth specification provides flexibility in the scatternet formation protocol, outlining only the mechanisms necessary for future protocol implementations. A new protocol for scatternet formation and maintenance - mscat - is presented and its performance is evaluated using a Bluetooth simulator. The free variables manipulated in this study include device activity and the probabilities of devices performing discovery procedures. The relationship between the role a device has in the scatternet and it's probability of performing discovery was examined and related to the scatternet topology formed. The results show that mscat creates dense network topologies for networks of 30, 50 and 70 nodes. The mscat protocol results in approximately a 33% increase in slaves/piconet and a reduction of approximately 12.5% of average roles/node. For 50 node scenarios the set of parameters which creates the best determined outcome is unconnected node inquiry probability (UP) = 10%, master node inquiry probability (MP) = 80% and slave inquiry probability (SP) = 40%. The mscat protocol extends the Bluetooth specification for formation and maintenance of scatternets in an ad-hoc network.

Table of Content

List of Figures

List of Tables

Chapter 1
Introduction

As wireless networks are gaining a stronghold in the communications market there is a new classification of wireless networks being investigated called ad-hoc networks. The Webster definition for the latin word *adhoc* reads “for the particular end or case at hand without consideration of wider application” which describes the foundational idea of an ad-hoc network. The network is created at a particular instance in time t = x, and does not exist at any point in time t, for t < x. The National Institute of Standards and Technology (NIST) defines a wireless ad-hoc network as a collection of autonomous nodes that communicate with each other by forming a multi-hop radio network and maintaining connectivity in a decentralized manner. Since the nodes communicate over wireless links, they have to contend with the effects of radio communication, such as noise, fading, and interference. In addition, the links typically have less bandwidth than in a wired network. Each node in a wireless ad hoc network functions as both a host and a router, and the control of the network is distributed among the nodes. The network topology is in general dynamic, because the connectivity among the nodes may vary with time due to node departures, new node arrivals, and the possibility of having mobile nodes [Nist04].

The main reason for investigating ad-hoc networks is the fact that they require no networking infrastructure, i.e. routers, access points etc., thus the financial and technical overhead for creating such a network is minimal or non existent. Ad-hoc networks are particularly interesting for scenarios where one does not want or cannot deploy and manage an infrastructure, in example:

- spontaneous meetings (at business/school campuses, airports)
- mobile sensor networks

- In-car convenience network
- Highway information networks
- battle field information networks

There are multiple specifications being used for digital wireless communications today. Two widely discussed in the context of mobile device networking are the IEEE 802.11 standard (WiFi) and Bluetooth also known as IEEE 802.15. These protocols are not in direct competition, in fact some modern laptops include devices supporting both technologies. The manufacturers of 802.11 chipsets predict a rise in 802.11 users from 16 million in 2001 to 60 million in 2006 in part because 802.11 is targeted at enabling wireless local area networking (WLAN) [Meritt02]. Bluetooth requires lower power consumption than 802.11 but it provides limited communication range of 10 meters. It is targeted at Personal Area Networks (PANs) which operate in the design space of interconnecting personal mobile devices. Even while targeted at different applications, they have a significant overlap. The potential exists for Bluetooth to see the same growth patterns 802.11 is currently exhibiting for a different segment of the wireless technology market, i.e. printers, wired appliances, and low-cost, low-power devices. If this prediction holds true the future of ad-hoc networks that use Bluetooth as underlying technology is bright.

Formation of an ad-hoc network poses great engineering challenges regardless of the underlying technology. Furthermore, the study of ad-hoc network performance differs from the analysis of classical/wireless network topologies in the sense that there are many additional parameters that govern the performance of ad-hoc networks. The most important distinction is that in an ad-hoc network the formation of the network itself is one of the critical optimization parameters. This is not an issue in wired or more long-term/ stable wireless networks. Along the same line of reasoning, the network's self-healing capability is an important parameter that requires individual attention due to the dynamic nature of particular applications of ad-hoc networks which explicitly require this capability.

The desirable performance characteristics of an ad-hoc network are somewhat dependent on the application, thus a mobile sensor ad-hoc network might strive to

integrate a new node as fast as possible while an ad-hoc network created at a meeting might strive for the maximum bandwidth utilization. Authentication, formation, maintenance and routing characteristics of an ad-hoc network can be changed to suit the needs of each of these applications.

The challenges in the field of ad-hoc networking can be categorized as: forming and maintaining the network, routing through the network, and ensuring quality of service on the network. The work presented in this document deals with the challenge of forming and maintaining the Bluetooth enabled ad-hoc network. In the rest of the document the words scatternet and network are used interchangeably.

Chapter 2
The Bluetooth Technology

Bluetooth wireless technology is a short-range communications system with the following features: robustness, low power, and low cost. Many features of the core specification are optional, allowing product differentiation. The Bluetooth core system consists of an RF transceiver, baseband, and protocol stack as shown in Figure 2.1. The system offers services that enable the connection of devices and the exchange of a variety of classes of data between these devices [BTSpec03]. This chapter will introduce the key

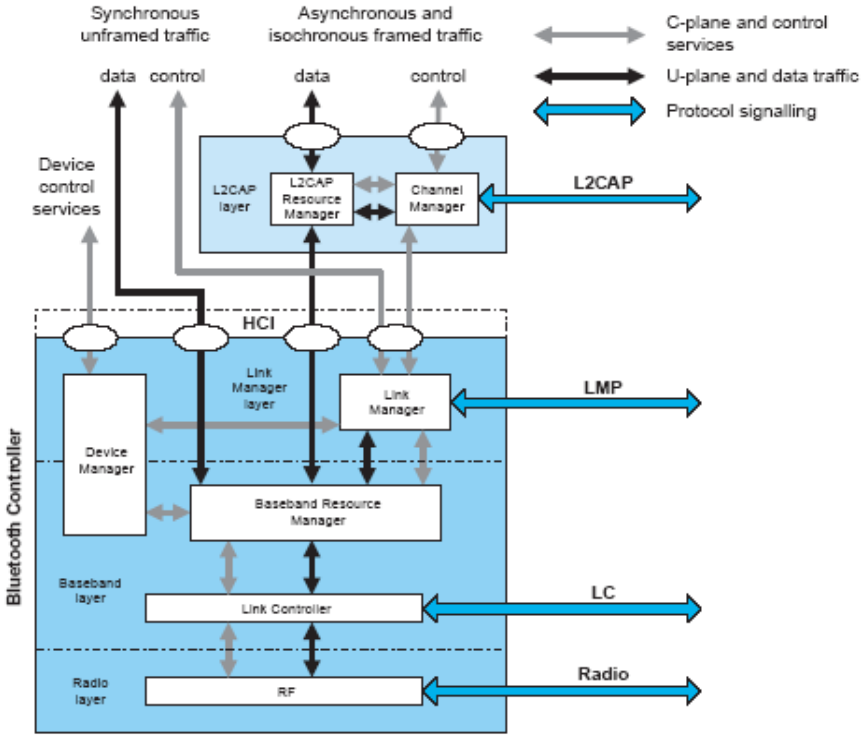


Figure 2.1: Overview of the Bluetooth protocol [BTSpec03]
The Bluetooth controller is usually implemented as a separate piece of hardware from the host computer.

components of each of the four layers depicted in Figure 2.1 and explain their role in the overall operation of the system.

The Bluetooth RF (physical layer) contains the radio-frequency (RF) hardware which does the actual radio transmissions of the data packets. The RF operation uses a shaped, binary FM modulation to minimize transceiver complexity. The symbol rate is 1 Megasymbol per second (Ms/s) supporting the bit rate of 1 Megabit per second (Mb/s) [BTSpec03]. In order to minimize the interference in the 2.4 GHz range the Bluetooth system employs two techniques, a frequency hopping scheme which nominally splits the 2.400 - 2.485 GHz band in 79 frequency channels, see Section 2.2 — Frequency Hopping Kernel, and a time division duplex (TDD) scheme which is used to switch to a new frequency channel every 625 μs or a multiple of 625 μs. During normal operation a Bluetooth device will transmit or listen for a radio transmission every 625 μs on a different frequency channel thus minimizing the chances of continuous interference impacting the performance of the system.

The baseband layer in Figure 2.1 contains the link controller and the baseband resource manager. The detailed operation of the baseband resource manager is not fundamental for the work presented here so there is little discussion on it. For further information the reader is referred to the “Baseband Specification” section in [BTSpec03].

The link controller governs how a piconet (see Section 2.1 for the definition of a piconet) is established and how devices can be added to and released from the piconet, the smallest network of devices. Several states of operation of the devices are defined to support these functions. In addition, the link controller defines states that enable the operation of several piconets with one or more common members, i.e. the scatternet [BTSpec03].

The link manager, which is part of the link manager layer, is the hardware device which provides the mechanism for interpreting the messages passed via the Link Manager Protocol or LMP. The Link Manager Protocol (LMP) is used to control and negotiate all aspects of the operation of the Bluetooth connection between two devices. This includes

the set-up and control of logical transports and logical links, and for control of physical links. The Link Manager Protocol is used to communicate control packets between the Link Managers (LM) on the two devices which are connected to each other. All LMP messages apply solely to the physical link and logical transports between the sending and receiving devices.

Figure 2.1 shows the Host Controller Interface (HCI) above the link manager layer. The dotted line encapsulating the HCI indicates that the layers located above the HCI, i.e. the L2CAP layer, are implemented on the host computer, most likely in software.

The packets exchanged between two Bluetooth devices can be broadly categorized as packets sent over an ACL transport link or over a SCO/eSCO transport link. The asynchronous connection-oriented (ACL) transport link is used to carry:

- control signalling
- best effort asynchronous user data

The ACL logical transport uses a simple 1-bit ARQN/SEQN scheme to provide simple channel reliability [BTSpec03].

The synchronous connection-oriented (SCO) transport channel is used for transmitting packets on point-to-point SCO link unique to the two devices doing the communication. The SCO logical transport, which carries the SCO packets, can be considered to be a circuit-switched connection between the two devices. These links can carry 64 Kb/s of information and typically this information is an encoded voice stream. Three different SCO configurations exist, offering a balance between robustness, delay and bandwidth consumption [BTSpec03]. The work presented in this document considers only ACL communication thus the SCO packets and the SCO logical transport layers that enable the voice streams are not discussed in great detail.

2.1 Bluetooth Piconet

The Bluetooth technology employs the piconet as a means of inter-device communication. The piconet, being defined as two or more Bluetooth units sharing the

same radio channel [BTSpec03], is limited in the number of devices that can actively participate in communication. In a piconet the master can have up to 7 active slaves and up to 256 parked slaves. An active slave is a device within a piconet that has one ACL logical transport to the piconet master, known as the default ACL. The default ACL is created between the master and the slave when a device joins a piconet (connects to the basic piconet physical channel). This default ACL is assigned a logical transport address (LT_ADDR) by the piconet master. This LT_ADDR is also used to identify the active physical link when required or as a piconet active member identifier, effectively for the same purpose [BTSpec03].

A Bluetooth device has two defining characteristics which are crucial to piconet creation and data transfer. These are the *unique* 48-bit Bluetooth address which is assigned by the manufacturer and the free running 28-bit clock of the device whose least significant bit (LSB) ticks in units of 312.5 μs, i.e. half a time slot, giving a clock rate of 3.2 kHz [BTSpec03].

In a piconet, one device (the master) defines the frequency hopping scheme which it derives from its Bluetooth address and its clock. The master also determines the quality of service given to its slaves - the other devices in the piconet which follow their master's frequency and time hopping schedule. When the piconet is established every actively participating device in the piconet, i.e. all the slaves, know the exact value of their master's clock and Bluetooth address. The Bluetooth manual calls for the master to transmit on odd clock cycles, while a slave transmits on the even clock cycle. A slave device is allowed to transmit in the even cycle if and only if it received a special, POLL, packet in the preceding (odd) clock cycle.

Figure 2.2 shows several piconet topologies. One master may be actively

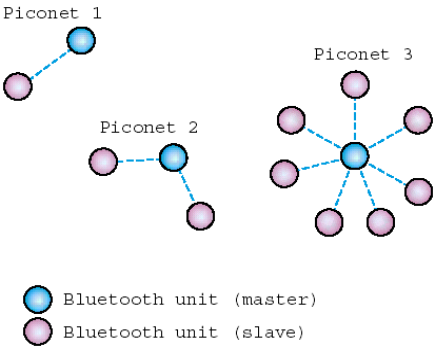


Figure 2.2: Various piconets

connected to at most seven slaves. Each slave in turn can be shared among multiple masters and master/slave roles can occur where one device is the master of a number of slaves and also operates as a slave of another master. Not only is the piconet limited in the number of participating devices, but is further constrained by each devices communication range on the physical layer of 10 meters. The master and slave roles that devices assume are assigned during the connection phase and the host application is unaware of its Bluetooth role in the network.

2.2 Frequency Hopping Kernel

The Bluetooth system operates in the 2.4 GHz ISM band, specifically in the 2400 - 2483.5 MHz frequency range. The core specification defines 79 radio frequency (RF) channels as: f=2402+k MHz, k=0,...,78. This implies that there is a 2 MHz lower guard band and 3.5 MHz upper guard band. Devices hop across the 79 channels and stay on one channel for 625 μs or one clock tick during which they either transmit or receive data.

The pseudo-random RF channel hopping sequence together with the packet (slot) timing and an access code is what defines a physical channel. The Bluetooth specification defines four physical channels:

- basic piconet physical channel - used during normal piconet operation;
- adapted piconet physical channel - alternative for normal piconet operation;
- page scan physical channel - used during the page and page scan sub-states;

- inquiry scan physical channel - used during the inquiry and inquiry scan sub-states;

The hopping sequence for each physical channel is determined by a portion of the Bluetooth device address and the selected hopping sequence. The phase in the hopping sequence is determined by the Bluetooth clock. All physical channels are subdivided into time slots whose length is different depending on the physical channel. Within the physical channel, each reception or transmission event is associated with a time slot or time slots. For each reception or transmission event an RF channel is selected by the hop selection kernel [BTSpec03]. The minimum hop rate is 1600 hops/s in the CONNECTION state and the maximum is 3200 hops/s in the INQUIRY and PAGE sub-states, see Figure 2.4 for overview of the different states. During the INQUIRY and PAGE sub-states the device transmits two packets in a single 625 μs slot in an effort to find a device with which it is attempting to establish a connection.

Devices in a piconet use either the basic piconet physical channel or the adapted piconet physical channel. In both cases a specific frequency hopping pattern is used, which is algorithmically determined by certain fields in the Bluetooth address and clock of the master. The basic hopping pattern is a pseudo-random ordering of the 79 frequencies in the ISM band. The hopping pattern may be adapted to exclude a portion of the frequencies that are used by interfering devices [BTSpec03]. The adaptive hopping technique improves Bluetooth co-existence with static (non-hopping) ISM systems when these are co-located and was introduced in the Bluetooth specification version 1.2. An example of a static system operating in the ISM band is a 802.11b/g wireless network.

2.3 Packet Types - Description

As it was discussed in Chapter 2 the Bluetooth technology supports two basic types of transport channels between devices which enable both data and voice communication. The ACL transport channel is primarily used for exchanging data and control packets while the eSCO/SCO transport channel is used for reliable point-to-point voice packets. This section discusses the different types of packets that are used to

communicate over ACL and eSCO/SCO links with emphasis on packets used over ACL links since this study omits the discussion of eSCO/SCO links.

Each Bluetooth packet consists of an access code, a header, and a payload as shown in Figure 2.3. Every packet starts with an access code. If a packet header follows,

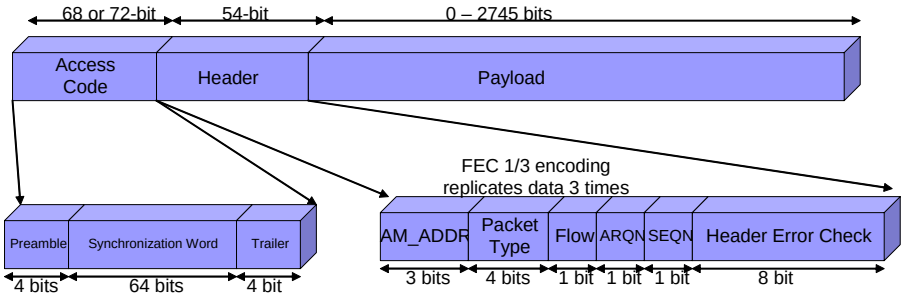


Figure 2.3: Generic Bluetooth packet structure

The access code is 72 bits long only if it is followed by a packet header otherwise it is 68 bits long. The packet header consists of 18 bits encoded with 1/3 Forward Error Correction code resulting in a

the access code is 72 bits long, otherwise the access code is 68 bits long and is known as a shortened access code. The shortened access code does not contain a trailer. This access code is used for synchronization, DC offset compensation and identification of a device. The access code identifies all packets exchanged on a physical channel: all packets sent in the same physical channel are preceded by the same access code. In the receiver of the device, a sliding correlator correlates against the access code and triggers when a threshold is exceeded. This trigger signal is used to determine the receive timing. The shortened access code is used in paging, inquiry, and park. In this case, the access code itself is used as a signalling message and neither a header nor a payload is present [BTSpec03].

The header contains all the control information associated with the packet and the link, such as the address of the slave for which the packet is intended. The 4 bit packet type filed distinguishes between sixteen different types of packets. The interpretation of the type code depends on the logical transport address (AM_ADDR) in the packet. Based on the AM_ADDR it will be determined whether the packet is sent on an SCO logical transport, an eSCO logical transport, or an ACL logical transport. Then it can be

determined which type of SCO packet, eSCO packet, or ACL packet has been received. The type code also determines how many slots the current packet will occupy as described in Table 2.1. This allows the non-addressed receivers to refrain from listening to the

Segment	Type code $b_3b_2b_1b_0$	Slot occupancy	SCO logical transport	eSCO logical transport	ACL logical transport
1	0000	1	NULL	NULL	NULL
	0001	1	POLL	POLL	POLL
	0010	1	FHS	reserved	FHS
	0011	1	DM1	reserved	DM1
2	0100	1	undefined	undefined	DH1
	0101	1	HV1	undefined	MSCAT
	0110	1	HV2	undefined	undefined
	0111	1	HV3	EV3	undefined
	1000	1	DV	undefined	undefined
	1001	1	undefined	undefined	AUX1
3	1010	3	undefined	undefined	DM3
	1011	3	undefined	undefined	DH3
	1100	3	undefined	EV4	undefined
	1101	3	undefined	EV5	undefined
4	1110	5	undefined	undefined	DM5
	1111	5	undefined	undefined	DH5

Table 2.1: Packet types for ACL and eSCO/SCO logical transport links.

This table describes the packets which can be sent on different transport links. The packet type with code 0b0101 belonging to segment 2 was left undefined for the ACL logical transport. For the purpose of implementing the MSCAT protocol this packet type was reserved for MSCAT specific communications.

channel for the duration of the remaining slots. The payload contains the actual message information if this packet is a higher layer protocol message, i.e. an L2CAP, LM protocol message or even user data being passed down the Bluetooth stack [Bray02].

For the implementation of the mscat protocol the type code 0x5 was defined for the ACL logical transport to contain packets relating to the operation of the protocol. The Bluetooth specification left this type code undefined for an ACL logical transport so there are no conflicts in utilizing this code for our specific purpose.

2.3.1 Common Packets - Structure and Purpose

There are five common kinds of packets. In addition to the types listed in segment 1 of Table 2.1, the ID packet is also a common packet type but is not listed in segment 1 because it does not have a packet header.

- The identity or ID packet consists of the device access code (DAC) or inquiry access code (IAC) when the device sending the ID packet is in the inquiry state. It has a fixed length of 68 bits. It is a very robust packet since the receiver uses a bit correlator to match the received packet to the known bit sequence of the ID packet [BTSpec03].
- The NULL packet has no payload and consists of the channel access code and packet header only. Its total (fixed) length is 126 bits. The NULL packet may be used to return link information to the source regarding the success of the previous transmission (ARQN), or the status of the RX buffer (FLOW). The NULL packet does not have to be acknowledged by the receiving device [BTSpec03].
- The POLL packet does not have a payload. Upon reception of a POLL packet the slave shall respond with a packet even when the slave does not have any information to send unless the slave has scatternet commitments in that timeslot. This return packet is an implicit acknowledgement of the POLL packet. This packet can be used by the master in a piconet to poll the slaves. Slaves do not transmit the POLL packet. In contrast to the NULL packet, it requires a confirmation from the recipient [BTSpec03].
- The FHS packet is a special control packet containing, among other things, the Bluetooth device address and the clock of the sender. The payload contains 144 information bits plus a 16-bit CRC code. The payload is coded with a rate 2/3 FEC with a gross payload length of 240 bits [BTSpec03].

The packets listed above are used for connection establishment, device discovery and inter-piconet communication as described in Section 2.4 — Connection Establishment and Section 2.1 — Bluetooth Piconet.

2.3.2 Mscat Packets

For the purpose of implementing the mscat protocol the following packet types with header type code 0x5 were added: *CONNFAIL*, *CONNSUCCESS*, *CONNTERM*, *IDENTREQ*, *IDENT* and *GOTOPAGE*. The *CONNFAIL*, *CONNSUCCESS*, and *CONNTERM* are packets that contain the Bluetooth address of a newly discovered device in the payload. The *IDENTREQ*, *IDENT* and *GOTOPAGE* packets are used in stage 2 of the mscat protocol or the reorganization of an existing scatternet. Section 4.2 — Mscat - Implementation contains detailed information about the packet types listed here and how they are used in the state machine of the mscat protocol.

2.4 Connection Establishment

Before two devices can exchange data in their piconet they must go through the connection procedures as outlined in the Bluetooth specification [BTSpec03]. Figure 2.4.A shows the state diagram of the link controller and Figure 2.4.B shows the packets exchanged between the devices during the connection establishment states.

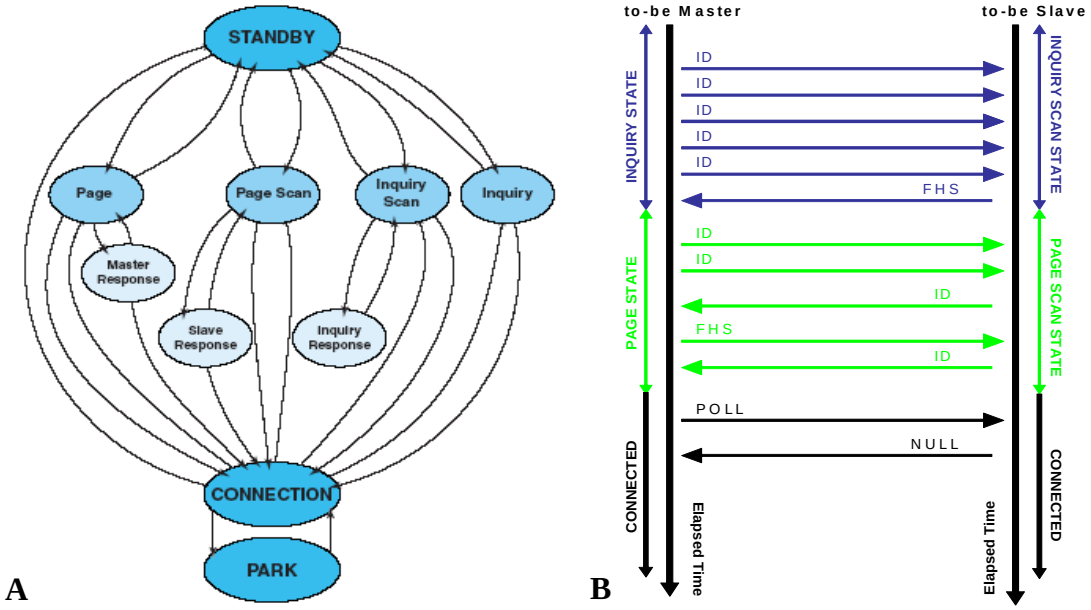


Figure 2.4: Link controller states and connection establishment procedure

A: The device that starts the connection establishment procedure by entering the Inquiry state will be the master of the connection where as the device entering the Inquiry Scan state will be the slave device. The host computer is unaware of the master/slave role as it is pertinent only to the link controller and link manager, see Figure 2.1.

B: The Inquiring device transmits ID packets until it hears a responding FHS packet from the Inquiry Scanning device. Then the devices move into Page/Page Scan respectively and they exchange another set of ID packets. The to-be-master device then send an FHS packet which the Page Scanning device acknowledges with an ID packet. After this the master sends the first POLL packet which the slave acknowledges with a NULL packet. At this point the devices are connected in a piconet.

When a device is initially turned on it is in the standby state. When a device has established a connection with at least one other device it is in the connection state. This section will explain how devices move from the standby state to the connection state, effectively establishing a new connection with another device.

In order to initiate connection establishment, two devices must be in the appropriate inquiring states, i.e. one device in the inquiry state while the other device is in

the inquiry scan state. The inquiring device will become the master of the piconet and therefore from now on it will be referred to as the *to-be-master.* The inquiry scanning device will become the slave of the piconet thus from now on it will be referred to as the *to-be-slave*. The *to-be-master* transmits specially constructed packets called ID packets which contain a General Inquiry Access Code (GIAC) indicating that this device is in inquiry and is looking for devices with which to establish a connection. The *to-be-slave* listens for ID packets and once it receives one it moves to the inquiry response sub-state and responds to the ID packet with a special Frequency Hopping Sequence (FHS) packet. The FHS packet contains the 48-bit Bluetooth address and the clock of the *to-be-slave* device. The *to-be-slave* may choose to randomly back off before sending its FHS packet thus minimizing the chances of interfering with another inquiry scanning device that heard the same ID packet from the *to-be-master.* In either case, once the FHS packet is sent, the *to-be-slave* moves to the page scan state. Upon the reception of the FHS packet, the *to-be-master* moves to the page state. Because the two devices in inquiry and inquiry scan do not know each others clocks or Bluetooth addresses the probability of both devices being on the same frequency channel at the same time is very small. The Bluetooth specification attempts to alleviate this problem by making the hopping sequence of the Inquiring device twice as fast, but even with this provision the sequence of inquiry and FHS packet exchange can last up to 10.24 seconds.

The *to-be-master* (now in the page state) transmits ID packets on the frequency channel that it derives from the *to-be-slave*'s Bluetooth address and clock[1]. Once the *to-be-slave* (now in the page scan state) receives the ID packet it moves to the Slave Response sub-state and acknowledges the ID packet of the *to-be-slave* with its own ID packet that it sends to the *to-be-master.* After this packet exchange there is coarse synchronization between the two devices. Upon the reception of the *to-be-slave*'s ID packet the *to-be-master* moves to the Master Response sub-state and transmits its own FHS packet. Once the *to-be-slave* receives the FHS packet it acknowledges with an ID packet and switches to using the *to-be-master*'s address and clock for frequency hopping. The *to-be-master* switches to its own address and clock for frequency hopping when it

1.Note that the FHS packet sent by the *to-be-slave* in inquiry response contained it's address and clock.

receives the acknowledgement ID packet from the *to-be-slave*. The first packet it sends on the new frequency channel is the POLL packet which the *to-be-slave* acknowledges with a NULL packet at which point the connection is established. The time spent in page / page scan by the devices is on the order of 10 milliseconds, which is orders of magnitude less than the time spent in the inquiry / inquiry scan states. During this process the master assigns a unique, to the piconet, 3-bit address to the slave which it will use to address it in the connected state. The Bluetooth specification requires that a device periodically go into the inquiry or page states, even when they are connected.

2.5 Bluetooth Scatternet

As mentioned in Section 2.1 — Bluetooth Piconet, a slave device can be shared among multiple masters and master/slave roles can occur where one device is the master of a number of slaves and also operates as a slave of another master in a different piconet. This is the mechanism by which the Bluetooth specification allows for the joining of two piconets to form a scatternet, thus increasing both the effective range of the network and the number of active devices participating in the network. A scatternet is defined as two or more piconets co-located in the same area with or without inter-piconet communication [BTSpec03]. Figure 2.5 below, shows a scatternet built of several piconets, where the

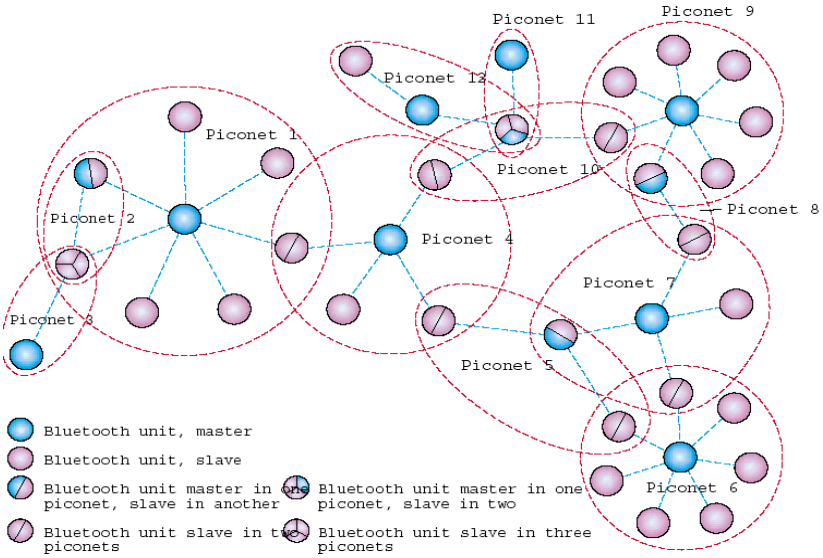


Figure 2.5: Bluetooth scatternet consisting of 12 piconets

joining nodes for the inter-piconet communications have master/slave and slave/slave roles.

Slave/slave and master/slave devices or nodes are sometimes referred to as *gateway* nodes in the literature because they bridge two piconets. The gateway nodes participate in multiple piconet communication based on time multiplexing. For example, in Figure 2.5 the gateway node that bridges piconets 1 and 4 will spend some time frequency hopping, transmitting and receiving in piconet 1 (based on the frequency hopping schedule derived from the master of piconet 1), then it will leave piconet 1 and start frequency hopping, transmitting and receiving in piconet 4 (based on the frequency hopping schedule derived from the master of piconet 4). The Bluetooth specification allows for several alternatives to be used for piconet switching. The aforementioned mechanism of the gateway node leaving one piconet and joining the other without negotiating or informing either master is the simplest one to implement. The gateway node can negotiate with its active master and ask to be put in the *park* state. While in *park* the device will not be polled by the master so it can effectively leave the piconet and join the other piconet where it will initiate a un-park and proceed with regular piconet operation. This method of switching between piconets is more time consuming and energy inefficient.

Figure 2.6 depicts the scenario where a master/slave node switches between the two piconets that it bridges. Recalling that a master is the device governing the transmit and receive schedules as well as the frequency hopping in the piconet, when it leaves the piconet where it acts as a master the slaves of that piconet cannot perform any transmissions effectively paralyzing the entire piconet for the time that their master is “gone”.

The Bluetooth specification leaves a significant amount of flexibility in the specifics of how Bluetooth scatternets are formed and managed. The devices that are turned on and are in the standby state need to make a decision weather they are going to enter the inquiry or inquiry scan state. Furthermore the devices that are already connected can perform node discovery operations by either entering the inquiry or inquiry scan state.

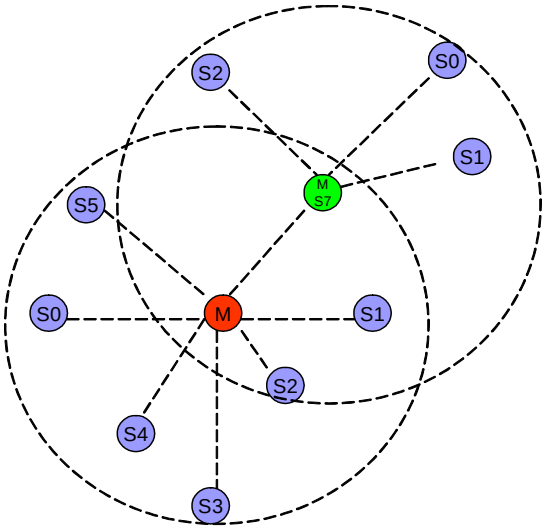


Figure 2.6: Two piconets connected by a master/slave gateway node

How often and in which state devices go are questions which are left open by the Bluetooth specification. The formation of different network topologies also depends on what types of nodes go into inquiry or inquiry scan and how often they do so. In example, should master nodes perform inquiry or inquiry scan more often or should slave nodes perform inquiry or inquiry scan more often? These questions which undoubtedly have impact on the formed network topologies are left open in the Bluetooth specification. It is the goal of the work presented in this document to extend the specification by providing a protocol for scatternet formation and maintenance which will strive to form acceptable network topologies in acceptable time.

Chapter 3
Problem Definition

Scatternet formation protocol in the context of this document refers to both the *initial formation* of the scatternet as well as the *scatternet maintenance*. The elements of the protocol that dominate activity when all nodes are turned on at the same time is referred to as the *initial formation* part, whereas the part that gets executed periodically at each node in order to discover newly introduced nodes in the network is referred to as the *scatternet maintenance* part. Some researchers make the distinction between the *initial formation* and *scatternet maintenance* in order to explore the one without considering the other. In this research the scatternet formation protocol considers both the *initial formation* and *scatternet maintenance* because in a dynamic environment, such as mobile sensor network, the two cannot be distinguished from each other.

The primary focus of the work presented in this document is scatternet formation. In Chapter 2 - Section 2.1 the concept of a Bluetooth piconet was introduced and in Section 2.5 the concept of a Bluetooth scatternet was introduced. The first broader classification of the solutions for scatternet formation protocols proposed so far in the literature distinguishes between protocols that require radio vicinity of all nodes (single hop topologies) and protocols that work in the more general multi-hop scenario. All of the solutions are localized, in the sense that the protocols are executed at each node with the sole knowledge of its immediate neighbors (nodes in its transmission range) [Basagni04a].

For the purpose of this work a scatternet formation protocol has the following desirable properties as outlined in [Basagni04a]:

1. The produced scatternets should be connected.
2. Resilience to disconnections in the network. The protocol should be able to operate in the connected components of the network.

3. Piconet size limited to eight nodes, to avoid the overhead associated with parking and unparking slaves.
4. Distributed and localized operations. A protocol should be executed at each node with information available at the node itself locally (e.g., knowledge of one and two-hop neighbors).
5. Self-healingness. A protocol should react to changes in the network topology to maintain a scatternet that retains all the properties of the scatternet initially formed.

The problem of designing a scatternet formation protocol that works for the multi-hop scenario so far has not addressed the self-healing property which is listed above. It is one of the main goals of the work presented in this document to address this issue and propose a possible solution. The self-healing property is of extreme importance in specific applications of ad-hoc networks, i.e. mobile sensor networks and battlefield data networks. It is with these applications in mind that the design space of self-healing scatternet formation protocols is examined.

3.1 Preliminary Research

The idea of using Bluetooth as the underlaying technology for creating an ad-hoc network was born in the fall of 2002 as a result of an IEEE Computer Society sponsored competition which required participants to use the Bluetooth technology. For further detail about the initial efforts and ideas the reader is referred to [CSIDC02].

An excellent overview of the current literature on the problem of scatternet formation for both single-hop and multi-hop scenarios is presented in [Basagni03] and [Basagni04a]. A single-hop scenario is defined as a scenario where each node in the network is within communications range of every other node. The multi-hop scenario is less restraining and is defined as a scenario where each node is placed within communication range of at least one other node in the simulation space.

Solutions for the single-hop scatternet formation problem are presented in [Salo01], [Law03], and [Tan02a]. The solution proposed in [Salo01] is based on a leader election process to collect topology information at the leader. Then, a centralized algorithm is run at the leader to assign the roles to the network nodes. In order to achieve

desirable scatternet properties, the centralized scheme executed by the leader requires that the number of network nodes is 36 [Basagni03]. The authors of [Salo01] impose the constraint that nodes cannot leave the scatternet while the scatternet formation protocol is being executed. The protocols presented in [Law03] and [Tan02a] run over single-hop topologies with no limitations on the number of nodes. However, the resulting scatternet is a tree which limits efficiency and robustness as pointed out in [Basagni04a]. Law et. al.'s proposed scatternet formation algorithm achieves O(log n) time complexity and O(n) message complexity but the authors fail to address the issue of a master's bandwidth utilization [Law03]. Even though the authors of [Law03] constrain their analysis to single-hop networks, and Basagni et. al in [Basagni03] classify this algorithm as single-hop it can be deduced from [Law03], but not proven, that their algorithm will work for the more general case of multi-hop topologies. It is believed that the authors of [Law03] constrained their work to single-hop topologies in order to mathematically prove that their algorithm works.

Among the solutions that apply to the more general case of multi-hop topologies, the scatternet formation protocol described in [Zaruba01] requires that the protocol is initiated by a designated node (the blueroot) and generates a tree-like scatternet. The blueroot starts the formation procedure by acquiring as slaves its one hop neighbors. These, in turn, start contacting their own neighbors (those nodes that are two hops from the root) trying to acquire them as their slaves and so on, in a wave expansion fashion, until the whole tree is constructed. The obtained scatternet has piconets with an unbounded number of slaves. A procedure based on geometric properties of networks of devices scattered in the plane is described to re-configure the tree so that each master has no more than seven slaves. This allows the master to avoid the time and bandwidth consuming operation of parking and unparking slaves [Basagni04a].

[Li02a], [Wang02], [Petrioli02], and [Petrioli03] are solutions to the scatternet formation problem, for multi-hop topologies, that do not organize the network in a tree-like structure. All of the protocols presented in these publications explore the design space of the *initial formation* of the scatternet without considering the *scatternet maintenance*.

This implies that a new node which comes in radio range of the already-formed scatternet will not be merged in the network. For applications such as mobile sensor networks this is unacceptable.

The work presented in this document is similar to the work presented in [Chiasserini03]. Since the work presented in [Chiasserini03] does not include numerical results it is difficult to compare the two. Furthermore, the work presented here explores what parameters impact the speed with which a new node is merged in the scatternet.

3.2 Goals and Assumptions

This document presents a scatternet formation and maintenance protocol, see Chapter 4, and attempts to answer the following questions in the context of scatternet formation protocols for multi-hop scenarios:

1. What is the average time needed for a connected scatternet to be formed?
2. What is the average time needed for a new node to be merged in a scatternet and how can this time be minimized?
3. What is the number of piconets that form the scatternet when related to the total number of nodes in the scatternet?
4. What is the average number of slaves per piconet?
5. What is the average number of roles per node in the scatternet?
6. What is the number of master/slave nodes in comparison to the total number of nodes in the scatternet?
7. What is the mean shortest path between any two nodes in the connected scatternet?

To answer the first question this study makes the assumption that nodes in the simulation are turned on in an interval of 100 ms. The reason for this assumption is two fold. If we are to attempt to measure the average time required to form a connected scatternet but nodes are allowed to enter and leave the simulation space it is impossible to pick a single point in time which will be considered the starting point because all points in time are equally viable choices. On the other hand, making all nodes turn on at the same point in time would be physically impossible thus we have settled for the aforementioned assumption. The node placement in the simulated space is such that each node is within communications range of at least one more node. This assumption greatly simplifies the programs required to perform the graph analysis and statistics reporting. Another

assumption made pertinent to answering this question is that nodes are not mobile, i.e. once placed in the simulation space they are not allowed to move.

To answer the second question the assumption is made that a connected scatternet is formed prior to the new node joining the simulation. This is done for two reasons: if we want to measure the time required to merge a node in the network a network must exist prior to the node's arrival and in order to measure time duration an initial starting point in time is required.

No assumptions are made in order to answer questions three, four, five and six. For question number seven, what is the mean shortest path between any two nodes in the connected scatternet it is assumed that a connected scatternet containing all the nodes is present. Furthermore, it is assumed that all of the nodes in the simulation space can form a connected scatternet, i.e. there are no set of nodes that are out of communications range of the other set of nodes.

Questions three, four, five and six are direct indicators of how good the network topology is. For a fixed set of nodes in a simulation it is desirable for the network topology to contain minimal number of piconets. Lesser the number of piconets that form the scatternet denser the network topology will be. In ad-hoc networks dense network topologies are desirable because data packets would have to be routed through less hops on average. The average number of slaves per piconet is an extension of the aforementioned measurement of how good a given network topology is as is the average number of roles per node in the scatternet. The number of shared master/slave nodes when compared to the total number of nodes in the simulation is a good indicator of the efficiency of the network topology. As discussed in Section 2.5 — Bluetooth Scatternet when a master/slave node acts as a slave the piconet in which it was acting as a master is essentially suspended since the master which is coordinating all the communication is absent. Note that even slave-to-slave communication is impossible when the master is absent because in a piconet all data packets are routed through the master.

The list of questions in this section is a guideline that will be used in measuring how good a particular network topology is. The average time to form a network and merge a node in a network are not direct indicators of how good a network topology is but nonetheless they are important indicators of the overall network formation and maintenance protocol design. For mobile ad-hoc sensor networks the formation time of the network and the ability to quickly merge a new node are just as important factors as is the formed network topology.

This work concerns itself with the scatternet formation and maintenance problem. Ideally a researcher exploring this problem would have to consider routing and quality of service (QoS) as an integral part of the network topology performance but due to time and resource constraints this study makes the following assumptions about routing and QoS. The nodes in the simulation have perfect routing information which is updated instantaneously. This implies that when a node discovers a new node it can instantly and accurately determine if the discovered node already belongs to the connected scatternet to which the discoverer node belongs. This assumption about ideal routing allows us to examine the scatternet formation and maintenance design space while acknowledging that for a more through study a routing protocol must be implemented. The quality of service is another important issue that needs to be considered together with the scatternet formation and maintenance problem. However QoS is outside the scope of this study and only the default round-robin QoS scheme for inter-piconet communication is implemented. Shared slaves and master/slave nodes switch piconets based on a timer.

Chapter 4

MSCAT - Self-Healing Scatternet Formation and Maintenance Protocol

The design of mscat was motivated by several observations made in a preliminary study [CSIDC02]. It was observed that all inter- and intra-piconet data traffic must go through at least one master node and all inter-piconet traffic must go through at least one bridge node. Furthermore, when a master node is performing tasks such as inquiry or page there can be no data transfer between any two nodes in the piconet. From the aspect of time required to merge a new node in the network it was observed that it is beneficial to dynamically adjust the probabilities of devices going into inquiry or inquiry scan depending on their role in the network. From these observations it can be concluded that master nodes together with bridge nodes have a great potential to be bandwidth bottlenecks in the scatternet. Yet the master nodes in certain scenarios are crucial players in the process of merging a new node in the network.

4.1 Mscat - Protocol Description

The design objectives for mscat are that it should strive to maximize the bandwidth utilization, minimize the delay incurred by packets traversing the network, allow for a new nodes to join the network rapidly, maintain a connected scatternet while nodes leave the network, and be completely distributed executing tasks that depend on the node’s role in the network. The scatternet formation protocol must be complied with by each Bluetooth node in order to maintain up to date scatternet topology. Mscat allows for unconnected nodes, slaves, masters, and shared master/slave acting as masters to execute inquiry or inquiry scan. Which operation, i.e. inquiry or inquiry scan, is executed will depend on the role that the node has in the network and the status/number of the one hop neighbors.

The mscat protocol has two main stages, a node discovery stage (stage 1) and piconet restructuring stage (stage 2). The node discovery stage of mscat is executed by a node which is performing inquiry and its role in the network is a slave. In the inquiry procedure the slave is looking for new devices to join the network and when a new node is discovered, by receiving an FHS packet as outlined in Section 2.4 — Connection Establishment, the slave proceeds to forward the FHS packet to its master who will attempt to merge the new node in the scatternet. The outcome of this stage can be that the master successfully merges the newly discovered node, or that the master fails to merge the new node. This stage of mscat attempts to address the problem of piconet density in the scatternet by trying to make all newly discovered nodes attempt to join an already existing piconet rather than form a new one. Maintaining high piconet density is important because the average shortest path between any two nodes in the scatternet is greatly affected by it. To extend this argument, having short routing paths between any two nodes in the scatternet is dependent on the average shortest path and having short routing paths is extremely desirable property for any kind of network and especially for an ad-hoc network.

In the case when the master fails to merge the new node, the slave who originally discovered the node will initiate stage 2, the piconet restructuring stage, to attempt to merge the new node in the scatternet. In the process of doing so the slave will attempt to restructure the piconet(s) involved in the simplest and fastest way possible. When the slave is finished executing stage 2 of the mscat protocol the newly discovered node will be merged in the scatternet in some fashion.

Figure 4.1 shows the pseudo-code describing the operation of the mscat protocol. The function `MSCAT(node u)`is executed at each node when the node needs to perform the inquiry or inquiry scan procedure. The probability **`p(u)`** of the device going into inquiry can be different depending on the current role the device has in the network.

Figure 4.2.A-B depicts a scenario where an unconnected node N1 is discovered by slave S1. S1 is executing inquiry and N1 inquiry scan. As outlined in Section 2.4 — Connection Establishment, S1 is flooding the channel with ID packets in an effort to find a

```
MSCAT(node u){
   x <- rand[0,1];
   if(x < p(u))
      u performs INQ
      if(new node v found)
         CONNECT(u, v)
      else
         return
   else
      u performs INQ SCAN
}
```

```
CONNECT(node u, node v){
   if(u is a master)
      u pages v
      if successful
         u connects to v
         return
      elese
         return
   else if(u is a slave)
      mu <- u's master
      mu pages v
      if successful
         mu connects to v
         return
   u pages v
   if unsuccessful
      return;
   u connects to v
   if(v was a slave)
      u assumes master/slave role
      v assumes shared slave role
      return
   else if(v was a master)
      u initiates role-reversal
      u assumes slave/slave role
      v assumes master role
      return
   else if(v was unconnected node)
      u initiates role-reversal
      u assumes slave/slave role
      v assumes master role
      return
```

Figure 4.1: Pseudo code - mscat protocol

p(u) is the probability of the node going into inquiry. This probability can be different for an unshared slave **p(us)**, shared slave **p(uss)**, unconnected node **p(un),** and master node **p(ms)**.

new node. When N1 receives the ID packet it doesn't know anything about the sender of the packet since the ID packet doesn't contain any node specific information. N1 then responds with an FHS packet and proceeds to move into page scan where it will wait to be paged. S1 in the mean time forwards N1's FHS packet to its master M. Upon reception of this packet M moves into page and pages N1 which is still executing the page scan procedure. When M establishes a connection with N1 it informs its slave S1 that the new

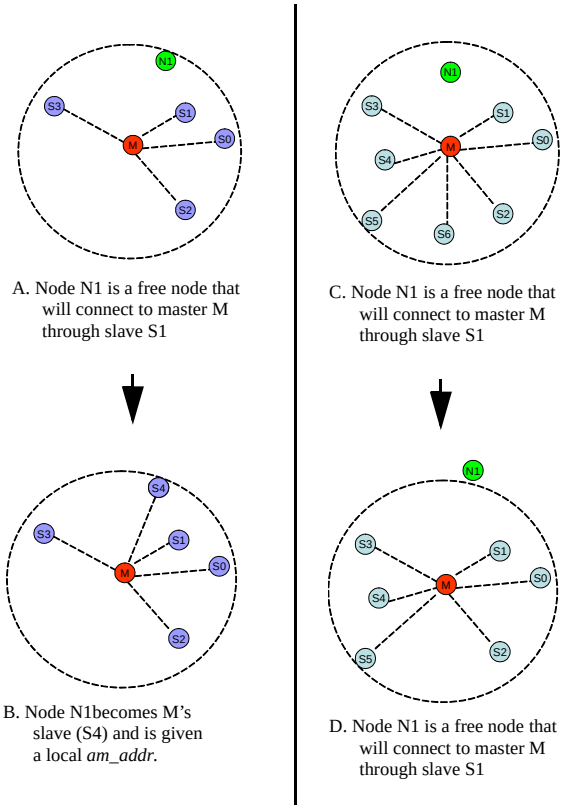


Figure 4.2: Mscat stage 1 invocation scenarios

A-B:In this figure an unconnected device N1 was discovered by slave S1. S1 forwarded the N1's FHS packet to its master M who successful merges N1 in its piconet. Mscat stage 1 is successful.

C:N1 cannot be merged by M because M already has a fully populated piconet with 7 slaves. Mscat stage 1 is unsuccessful.

D: N1 is out of M's communications range so they cannot connect.Mscat stage 1 is unsuccessful.

node is successfully merged in the piconet. Figure 4.2.C-D show the scenarios in which the master M cannot merge N1 in its piconet. In the case of Figure 4.2.C the master M has a full piconet (7 active slaves) and cannot merge a new node. In the case of Figure 4.2.D the master M is out of N1's communication range thus they cannot establish a connection. In both cases the master M will inform the slave which initiated mscat's stage 1, in this example S1, that the newly discovered node cannot be merged in the scatternet. Upon receiving information that N1 cannot be merged by M, S1 will start executing stage 2 of the mscat protocol.

In general stage 2 of mscat can be initiated by two events. When the slave that is executing mscat receives a *CONN_FAIL* packet from its master it proceeds to execute

stage 2 of mscat. The second event that can trigger a slave to start executing stage 2 of mscat is a time-out event for the execution of mscat's stage 1 protocol. Regardless of how stage 2 is initiated the goal is for the slave node to merge the newly discovered node in the scatternet.

Figure 4.3.A-F depicts the scenarios where stage 2 of the mscat protocol will be

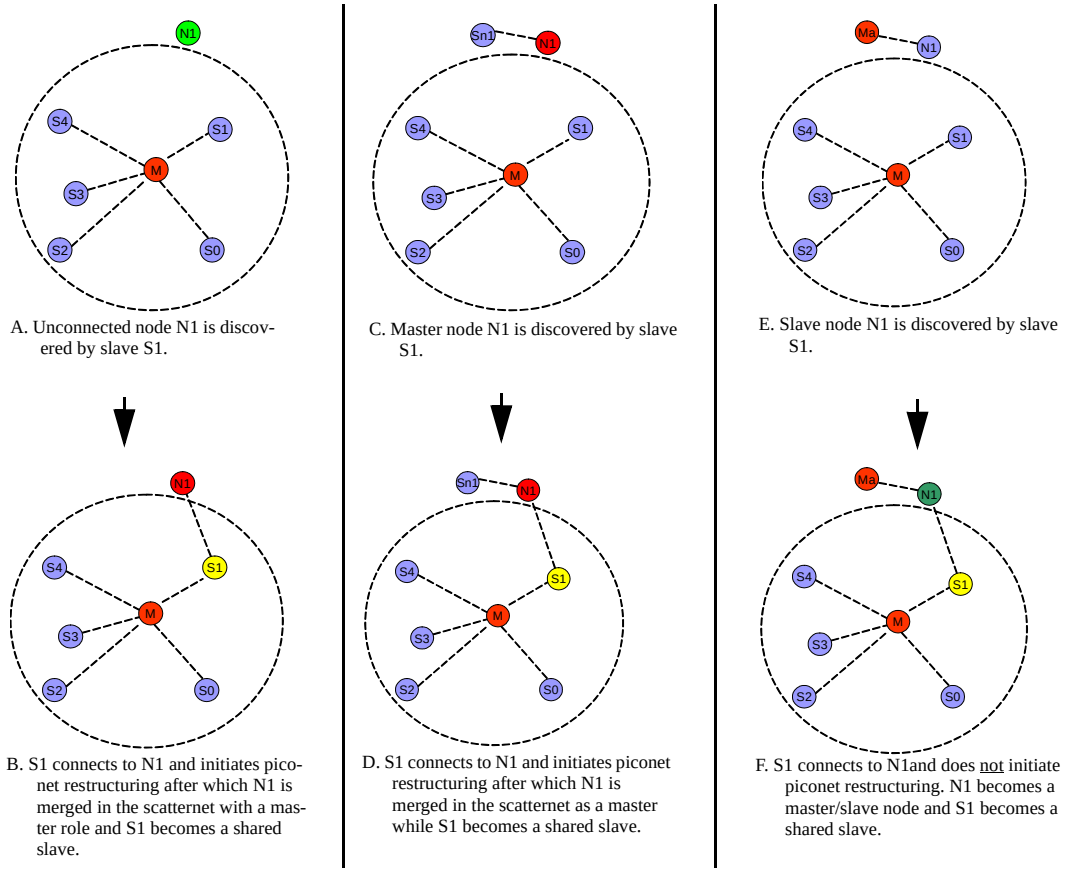


Figure 4.3: Mscat stage 2 invocation scenarios

A-B: Nodes N1 and S1 establish a connection after which they exchange information about their roles. S1 initiates a piconet restructuring and N1 becomes a master with S1 becoming a shared slave.
C-D: Nodes N1 and S1 establish connection after which they exchange information about their role. S1 initiates a piconet restructuring and N1 retains its master role while S1 becomes a shared slave.
E-F: Nodes N1 and S1 establish a connection after which they exchange information about their roles. S1 does not initiate piconet restructuring thus N1 remains a master/slave node and S1 remains shared slave.

executed. In Figure 4.3.A N1 is a unconnected node discovered by slave S1. S1 has no information about N1's role in the network until they establish a connection at which point S1 sends to N1 a *IDENTREQ* packet, requesting N1 to identify its role in the network. N1 responds with a *IDENT* packet which contains the information about the current role of

the node. Upon reception of the *IDENT* packet S1 can infer the role of N1 prior to them establishing a connection. S1 deduces that N1 was an unconnected node and initiates a piconet restructuring by sending a *GOTOPAGE* packet. This will put N1 in the page state and S1 in the page scan state thus positioning N1 to become the new master and S1 to become a shared slave. It is important to note that in stage 2 of mscat there is a temporary piconet which is formed between the discovered node and the discovering node, in this case N1 and S1. Figure 4.3 does not show this temporary piconet but it does show the final piconet structure after stage 2 of mscat is done executing (Figure 4.3.B). In this example (Figure 4.3.A) the piconet restructuring is desirable because a good network topology will be preserved if N1 becomes a master node and S1 becomes a shared slave node.

If N1 is already a connected node, the new connection will either merge two scatternets or create a redundant connection within one scatternet. Detecting a redundant connection will require information about the larger scatternet, which will require the use of a routing table. Since redundant connections can reduce network diameter it may be desirable to keep such a connection, but the rules governing this are outside the scope of this study. For the purpose of evaluating the mscat algorithm through simulation perfect routing information knowledge will be assumed for each node in the connected scatternet.

In Figure 4.3.C-D the newly discovered node N1 is a master so the execution of stage 2 of mscat will merge two scatternets. S1 will start executing stage 2 of mscat by attempting to page N1. When successful they form a temporary connection (piconet) in which S1 is a master and N1 is a slave. S1 sends an *IDENTREQ* packet to N1 but in this case the *IDENT* packet sent by N1 to S1 will identify the prior role of N1 as a master so S1 will initiate a piconet restructuring by sending a *GOTOPAGE* packet to N1. The final network topology is depicted in Figure 4.3.D and it will contain S1 now acting as a shared slave and N1 will retain the role it had prior to the merging, a master.

The last possibility is shown in Figure 4.3.E where the discovered node N1 is acting as a slave and the connection of N1 and S1 will merge two scatternets. When S1 pages N1 and they connect S1 becomes a master of the connection with N1. This makes

S1 a master/slave node, acting as a master to N1 and slave to M. On the other hand N1 becomes a shared slave between Ma and S1. This is the least desirable scenario for mscat because S1 will not initiate a piconet restructuring this leaving it as a master/slave. The decision not to have S1 attempt network restructuring is because it would be too time consuming and some instances it might dissolve an already existing piconet. This second point comes to light when one considers what would happen if S1 initiated a piconet restructuring with N1. N1 would have to become a master to S1 thus becoming a master/slave node and S1 will have to become a shared slave between N1 and M. This creates a chain of two bridge nodes which is worse than having a S1 become a master/slave and N1 a shared slave which would take even more time.

Having a master/slave node in the network may reduce bandwidth utilization and greatly impact the performance of the network. As previously explained in “Bluetooth Scatternet” on page 15, when the master/slave node leaves the piconet in which it functions as master, the rest of the nodes must wait for the master to return. Simulations are planned which will examine under which conditions this becomes a significant problem.

4.2 Mscat - Implementation

The implementation of mscat in the simulator used for this study required introducing six new packets with type code 0x5 or 0b0101 in the header. The payloads of these packets differ and will be explained in this section.

The mscat protocol is initiated when a slave that is performing the inquiry procedure discovers a new node which is not currently present in the scatternet. The slave performs a breadth first search (BFS) on the connected topology to which it belongs looking for the newly discovered node. This BFS search is the implementation of the assumption that each node has perfect routing information, for more detail see Section 3.2 — Goals and Assumptions. If the newly discovered node is not in the connected scatternet the slave creates a *DATA* packet to its master with the payload of the packet being the FHS packet that it received from the discovered node. At this point the node sets a boolean flag

`executingMscat_stage1` to true which indicates that his node is somewhere in the mscat stage 1 state machine. Each tick a timer called `T_mscat_stage1TOCounter` is decremented. If the slave doesn't hear back from its master about the status of the node it discovered it proceeds to stage 2 of mscat. The master which receives the *DATA* packet initiates stage 1 of mscat and attempts to connect to the node it received information for from its slave. There are 3 possible outcomes for the master:

- it successfully connects to the node and sends *CONNSUCCESS* packet to its slave;
- it fails to connect to the node and sends *CONNFAIL* packet to its slave;
- it find the node in the connected scatternet and sends *CONNTERM* packet to its slave;

The *CONNSUCCESS* packet contains the Bluetooth address of the discovered and merged node in the payload. Upon reception of this packet the slave terminates the execution of stage 1 of mscat and proceeds with normal operations. The *CONNFAIL* packet contains the Bluetooth address of the discovered node and when the slave receives this packet it proceeds to execute stage 2 of mscat. The *CONNTERM* packet also contains the Bluetooth address of the discovered node in the payload section of the packet and it indicates that the discovered node is now present in the connected scatternet so the slave stops executing stage 1 of mscat and proceeds with normal operation.

In stage 2 of the mscat protocol the slave that discovered the new node will attempt to connect to the discovered node. To indicate that device is executing stage 2 of mscat it sets a boolean flag `executingMscat_stage2` to true and starts decrementing a counter `T_mscat_stage2TOCounter` every tick. If the devices connect the initiator of the connection sends an *IDENTREQ* packet which doesn't contain any payload but rather just asks the recipient of the packet to identify it's role in the network. The recipient of the *IDENTREQ* packet responds with *IDENT* packet which in the payload contains the role of the device in the network (a 2 bit value). If the devices do not connect before `T_mscat_stage2TOCounter` expires the slave will reset its boolean flag and timer and it will move to regular operation in the scatternet. For description of what happens as a consequence of these packet exchanges see Section 4.1 — Mscat - Protocol Description.

Chapter 5
Simulation Environment

For the purpose of testing the performance of the algorithm presented in Chapter 4 two simulators were developed. A Java based MTUScat simulator was developed to test the feasibility of the ideas presented in this document and Mscat is being developed as a complete test bed for detailed testing and analysis of the proposed protocol. MTUScat was used to generate the preliminary results which were presented in [CSIDC02].

5.1 Preliminary simulator - MTUScat

MTUScat was developed in Java as an initial simulator to test the scatternet formation algorithm. It's only intent was to yield a starting point for the development of the scatternet formation algorithm and give a basic idea of the performance of the algorithm. The code was created solely for the simulation of the scatternet formation algorithm and it is limited in that respect.

Several other limitations exist in MTUScat, the largest being the lack of a frequency hopping engine. Nodes in this simulator have no concept of frequency hopping, and thus do not implement the frequency hopping scheme specified in [BTSpec03]. Instead, the simulator operates on an average time to perform the inquiry and paging procedures. Another limitation of the simulator is the time scale that the simulator runs at. Currently, one clock tick of the simulator is approximately equal to 500 Bluetooth time cycles, or approximately 0.3 seconds real time. Also, no routing information is maintained by the nodes, and no packets are routed across the scatternet. The nodes are aware of the piconets surrounding them and use information on nodes within a 3 piconet radius when making connection decisions.

The simulator is highly configurable and allows for sophisticated node behavior. It allows nodes to power on and off, move with various speeds, and pick destinations either at random or with a Gaussian distribution around a given location. Nodes can be introduced at different times during the simulation and have the capability to be removed from the simulation. MTUScat has a graphical user interface which allows for the visualization of nodes, their movements, connections and role in the scatternet. The simulator has the ability to either read parameters from the command line or from a file specified at run time. Figure 5.1is an example of a typical run of MTUScat. In this figure,

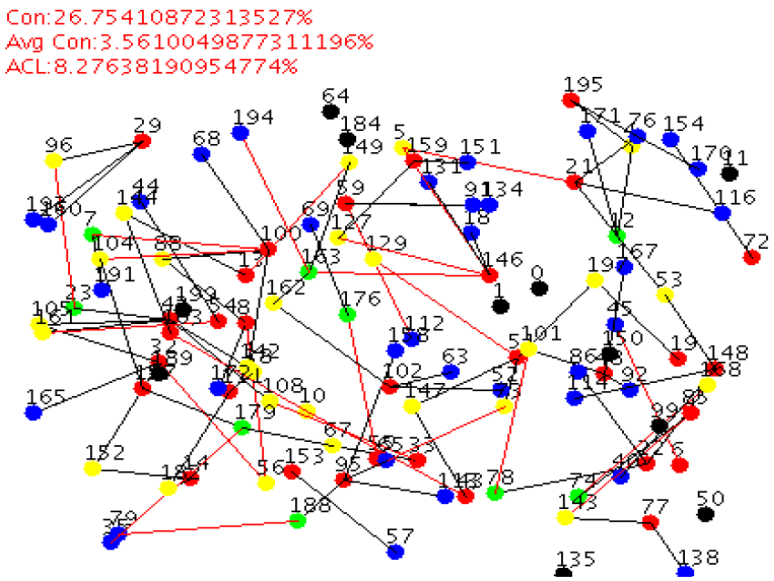


Figure 5.1: Screenshot of the MTUScat simulator

the master nodes are red the unshared slaves are blue, the shared slaves appear yellow, the master/slaves are green and the unconnected nodes are black. Also the links between the nodes are shown in black, links that are about to exceed the 10 meter transmission radius are shown in red.

The General Operations Director (GOD) has the ability to give accurate statistics with regards to connectivity and node statistics. The GOD of MTUScat allows for several parameters to be collected at any instant in time. These parameters include the percentages of nodes that are currently operating in the five roles of the scatternet, the average number

of nodes that are connected at any given instant and connectivity history of the nodes in the simulation.

MTUScat offers the option of running simulations on several different topologies with different movement patterns. Nodes are allowed to move within a rectangular two dimensional plane. The distribution of nodes is random, and two major categories of node placement is available. A hot spot configuration is available that allows the user to specify the number and locations for large node groupings to occur. These areas will have a high node concentration and nodes will frequently move form one area of high concentration to another. These areas exhibit a gaussian distribution around the center point, specified by the user and nodes moving from one spot of high concentration to another will do so in a linear fashion.

The other topology provided by MTUScat is a random distribution. With this topology, it is possible to set speeds, waits and turn on times to emulate a variety of situations. Using this, conference room scenarios with little node movement can be simulated. Also, it is possible to make the nodes highly dynamic where the time that is spent at each destination is very low.

5.2 SimScat

The Java based SimScat simulator implements all the necessary parts of the Bluetooth stack shown in Figure 2.1. Specifically the simulator implements the radio layer, the link controller, the baseband resource manager, parts of the link manager as well as parts of the device manager. The L2CAP layer is not simulated but the notion of connection source and sync are used for the purpose of simulating traffic which will be sent through the network. The simulator in its current stage assumes perfect routing information is available to all of the nodes in the network thus it provides each node with up to date information about topology changes. This can be easily changes at a later date as the code is object oriented and modular. The assumption that there is perfect routing makes evaluating mscat easier and faster. It is acknowledged that for a more detailed study

one has to consider the routing in the scatternet as an integral part of the overall design but this is outside the scope of the current and proposed work.

SimScat is cycle accurate thus one tick in the simulation corresponds to 625 μs. It implements the frequency hopping engine as outlined in [BTSpec03]. The simulator accurately implements the inquiry/inquiry scan/page/page scan states and their sub-states and it utilizes all the necessary timers which are defined in [BTSpec03].

In terms of generating statistics from the simulations SimScat generates a trace file in which each event that takes place in the simulation is reported. Helper programs written in perl and C++ are then used on the trace file to extract statistics about the simulation. This was done because some problems as parsing text lend themselves to using perl while other problems like graph analysis are better candidates for C++. SimScat does have some statistics generating capability like master bandwidth utilization, fully connected scatternet formation time reporting and number of connected scatternets in the simulation space in case a fully connected scatternet was not formed.

The simulator does not simulate SCO links and higher layers of the Bluetooth stack. In the simulation nodes within a piconet can exchange data packets but data exchanges through the scatternet are not possible because no routing protocols are implemented. Another constraint of the simulator is the granularity of the simulation time which is 625 μs. This becomes an issue in the inquiry and page states where the frequency hopping kernel selects a new transmit/receive frequency every 312.5 μs. The simulator ignores the packet sent on a non-aligned edge in these two states thus degrading the maximum possible speed at which new devices discover each other in the inquiry/inquiry scan states. Another limitation of the simulator is the radio propagation model. Currently the simulator delivers all packets to a receiver which is within 10 meters of the transmitter. The packet is not delivered only when there is another transmission which the receiver picks up on the same frequency channel.

SimScat does not allow for 3 way shared devices, so the possible roles for a node in the simulation are: unconnected, master, slave, shared master/slave, and shared slave/

slave. Shared slaves do not perform device discovery operations because the simulator limits them to 2-way shared slaves. Shared master/slave nodes when acting as a master can enter inquiry provided that they don't have 7 slaves in their piconet.

Figure 5.2 shows the graphical front-end for the simulator which was used for

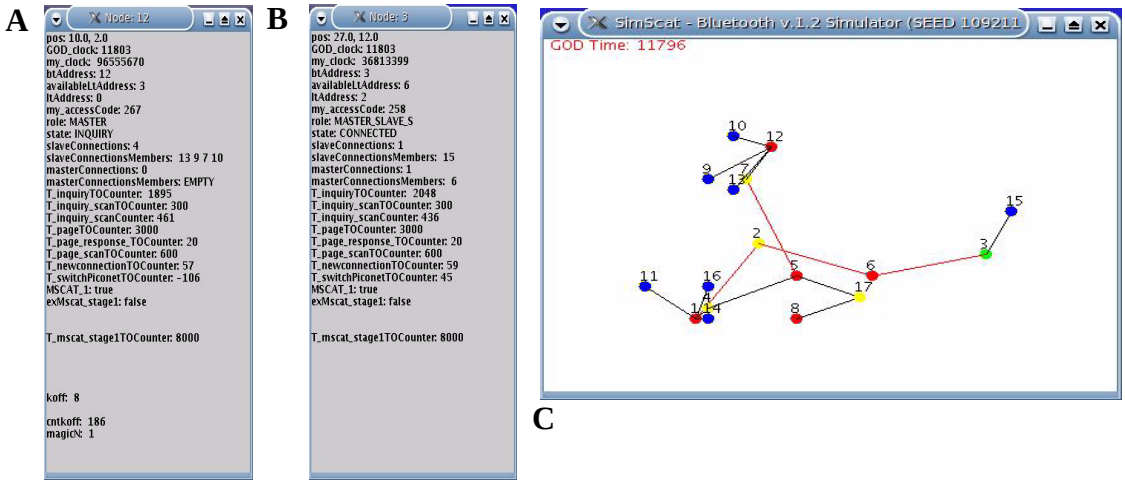


Figure 5.2: Screenshot of SimScat with two open node windows

In SimScat a red node indicates a master, blue nodes are slaves, yellow nodes are shared slaves, green nodes are shared master/slaves and black nodes are unconnected nodes. The node window is updated in real time and provides

visualizing network topologies and debugging the simulator. When the graphical front-end is enabled (through a configuration file) the user can click on any of the nodes displayed in the simulation window and a node window will be opened. As can be seen in Figure 5.2.A and Figure 5.2.B the node window displays real-time information about the node's local clock, simulation clock, timers, current role and state, master and slave connection lists, frequency hopping kernel pertinent information and mscat pertinent information when the node is executing one of the two mscat stages.

The correct operation of the simulator was ensured by using several debugging methods. The simulator code contained `assert()`functions which reported when the simulator had entered code that was logically should have never been executed. There are different levels of severity of `assert()`functions but all of them report an assert event in the simulation trace file while some of them terminate the simulation after reporting the cause of the assert in the trace file. Several perl scripts were written to process the trace file as the simulation was running and the output from these programs combined with the

information contained in the node windows and the graphical front-end proved essential to verifying the operation of the simulator. The use of the Java programming language and the Java API allowed for many of the “null pointer”, “no such element” etc. exceptions to be caught and fixed. Most of the bugs in the early stage of the development were caught thanks to the exception handling of the built in data structures that were used in the simulator, i.e. LinkedList’s, ArrayLists, Arrays and more complex objects.

Chapter 6
Results

The data presented in this section was obtained by running simulations with 155 unique input configurations. Each configuration file passed to the simulator was executed 5 times with different random number generator seed. The simulation space was a 2-D plane with consistent size for all simulations, x = 30 m and y = 30 m. All timer values and a sample configuration file used in the simulations are listed in Appendix A. The placement of nodes in the simulation plane was done with the constraint that each newly placed node must be in communications range of at least one already existing node. This constraint allows for much simpler implementation of the programs required for graph analysis for the simple reason that each simulation scenario is guaranteed to have a visibility graph which contains all the nodes placed in the simulation. This existence of a visibility graph that encompasses all nodes in the simulation space implies that in almost all cases a scatternet can be formed that will connect all the nodes.

As outlined in Section 3.2 — Goals and Assumptions this study attempts to answer two main questions, how does the mscat protocol perform and what values for the unconnected probability, master probability and slave probability give optimal performance of the mscat protocol. Section 6.1 presents data that compares the network topologies formed with and without mscat while Section 6.2 presents data that compares the impact of varying parameters on the mscat protocol's performance.

6.1 Mscat - Performance Analysis

This section explores the effect of varying master, slave, and unconnected node probability of going into inquiry on the resulting network topology that is formed with and

without the mscat protocol. The simulation results presented in this section use the same random seed for simulations with mscat turned on and off. This provides for accurate comparison of the data presented in Figure 6.1 and Figure 6.2.

It is important to note that each figure on the y-axis shows a measure of how good a network topology is but this can be misleading if each figure is interpreted independently of the others. For example, if for a given inquiry probability we see that the average slaves/piconet is at its maximum this does not necessarily mean that the network topology is optimal because we have not considered the other measures of how good a network topology is, like mean shortest path, average roles per node, number of piconets or number of master/slave nodes. The network formation time is another measure that might be of interest to be examined even though it does not directly affect the formed network topology.

Figure 6.1, “Average slaves/piconet vs. inq probability” contains a set of three charts showing the average slaves/piconet as a function of different inquiry probabilities. As discussed in Section 3.2 — Goals and Assumptions, average slaves/piconet is an important measure of a good network topology. More slaves per piconet indicates denser piconets which is desirable because for a network with fixed number of nodes higher slaves per piconet translates into less piconets and smaller network diameter. Each point on the charts is an average of five simulation runs with different random seeds and the y-error bars show +/- one standard deviation from the average for each data point.

Figure 6.1.A shows the impact of varying the probability of master nodes going into inquiry on the average slaves/piconet. This chart has six lines, three for networks created without mscat and three for networks formed with mscat. Each pair of same color lines shows average slaves/piconet for fixed number of nodes with mscat on and off. It can be concluded that for 30, 50 and 70 nodes the networks formed with the mscat algorithm have greater number of average slaves per piconet. This chart indicates that a good choice for the master inquiry probability for network formation with mscat turned on is 80%. The simulations used to produce this chart used constant slave inquiry probability of 30% and unconnected inquiry probability of 10%.

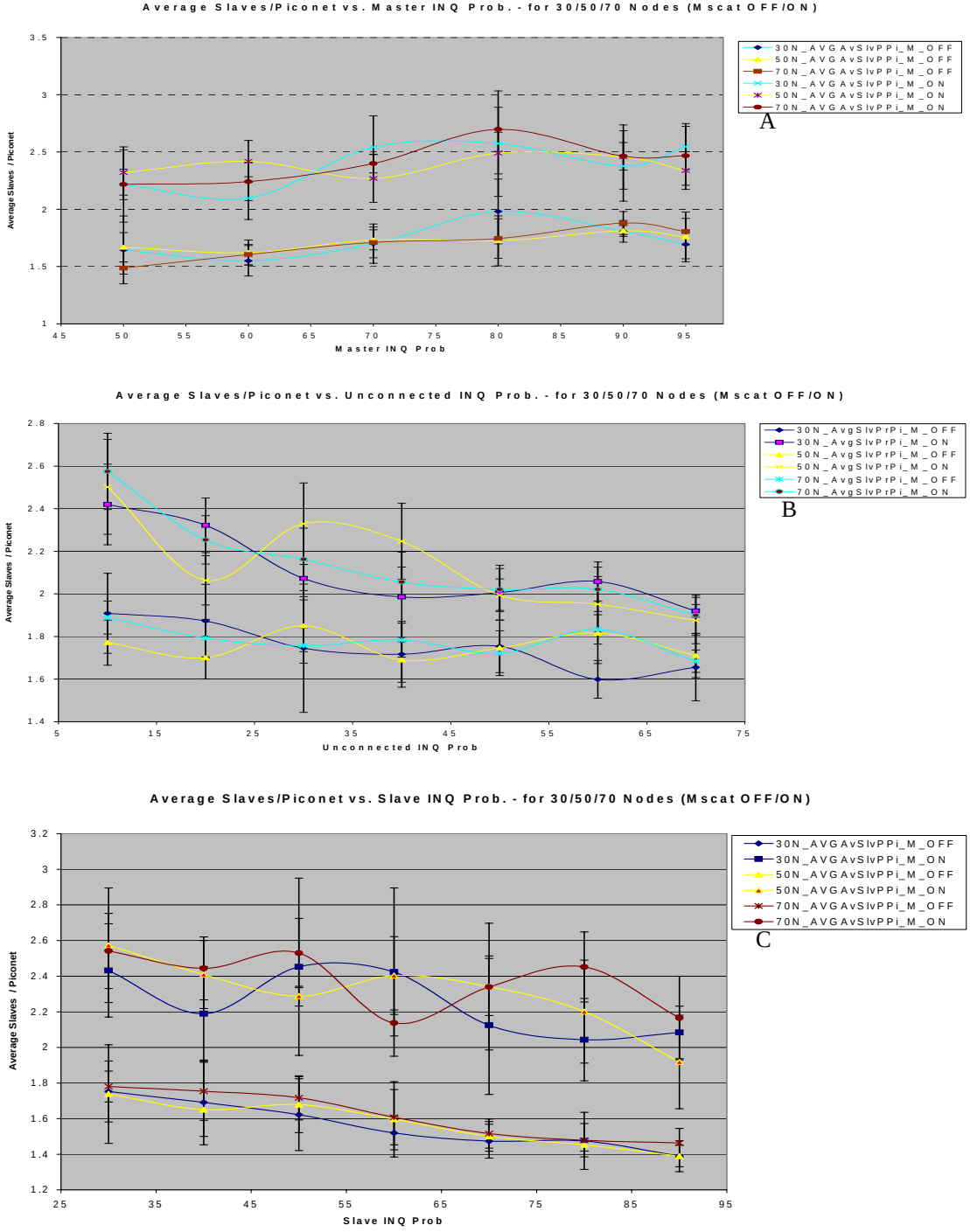


Figure 6.1: Average slaves/piconet vs. inq probability

A. Average slaves/piconet as a function of master inq probability
(slave inq probability = 30%; unconnected inq probability = 10%)

B. Average slaves/piconet as a function of unconnected inq probability
(master inq probability = 90%; slave inq probability = 30%)

C. Average slave/piconet as a function of slave inq probability
(unconnected inq probability = 10%; master inq probability = 90%)

Figure 6.1.B shows the impact of varying the probability of unconnected nodes going into inquiry on the average slaves/piconet. This chart has three pairs of same color lines which show the average slaves/piconet for fixed number of nodes, 30, 50 and 70 with and without the mscat protocol. This chart as the previous one shows that network topologies created with mscat have higher number of slaves/piconet. Furthermore, the chart shows that for an unconnected inquiry probability of 10% we get highest number of average slaves/piconet when using mscat. For this chart the master inquiry probability was held constant at 90% and the slave inquiry probability was held constant at 30%.

Figure 6.1.C shows the impact of varying the probability of slave nodes going into inquiry on the average slaves/piconet. Same as the previous two charts in Figure 6.1this one has three pairs of same color lines which show the average slaves/piconet for fixed number of nodes 30, 50, and 70 with and without the mscat protocol. The general trend for network topologies formed without mscat is that as the probability of a slave node going into inquiry increases the number of average slaves/piconet decrease. This trend is also observable for topologies formed with the mscat protocol except that the data is more oscillatory and with a higher standard deviation. Nonetheless if we observe the points for all M_ON lines at slave inquiry probability of 30% and 90% we can conclude that we obtain denser networks when using slave inquiry probability of 30%.

Figure 6.2, “Average roles/node vs. inq probability” contains three charts that showing the effect of varying the inquiry probabilities on the average roles/node. The average roles/node is a good indicator of how good the formed network topology is because a high number of roles per node translates into reduced throughput performance due o the overhead (an average of 2 slots) associated with piconet switching. Master-slave roles are inefficient since all the communications in the piconet of the master that switches to be slave have to suspended until the node returns to acting as a master [Basagni04a].

Figure 6.2.A show a chart with three pairs of same color lines for each of the 30, 50 and 70 node simulation runs comparing the network formed with and without the mscat protocol. From the chart it can be inferred that network topologies formed with the mscat protocol have lower average roles/node. The general trend in this chart is that as the

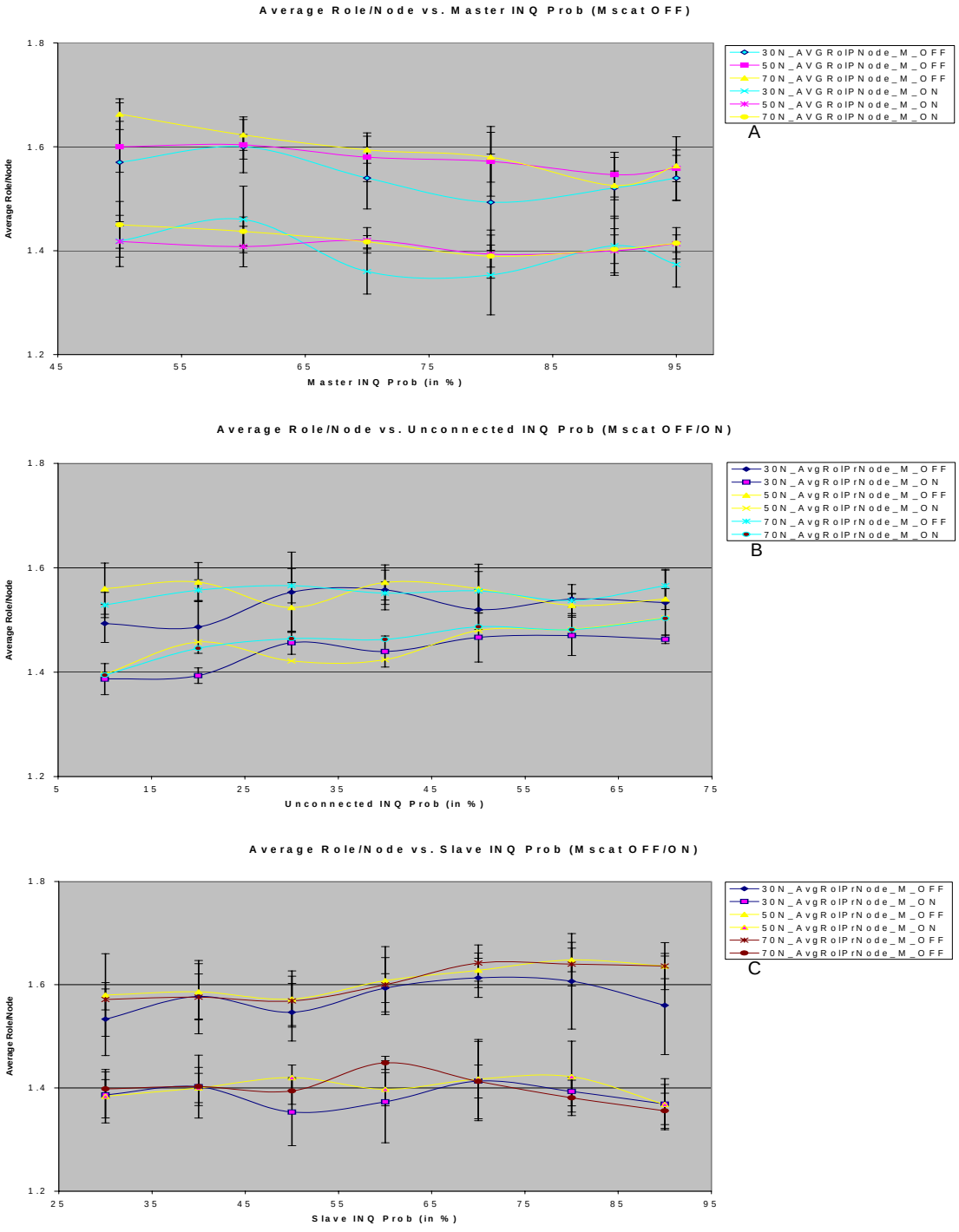


Figure 6.2: Average roles/node vs. inq probability

A. Average roles/node as a function of master inquiry probability
(slave inquiry probability = 30%, unconnected inquiry probability = 10%)
B. Average roles/node as a function of unconnected inquiry probability
(master inquiry probability = 90%, slave inquiry probability = 30%)
C. Average roles/node as a function of slave inquiry probability
(master inquiry probability = 90%, unconnected inquiry probability 10%)

master inquiry probability is increased from 50% to 95% we obtain network topologies with lower average roles/node with 90% giving optimal average roles/node for 50 and 70 nodes and 95% for 30 node networks with mscat turned on.

Figure 6.2.B shows the impact that varying the unconnected inquiry probability has on the number of average roles/node. Again there are three sets of same color lines for each of the 30, 50 and 70 nodes with and without mscat and it is distinguishable that for each simulation scenario the network topologies formed with mscat yield lower average roles/node. For this particular measure of good network topology the optimal value for unconnected inquiry probability is 10%.

Figure 6.2.C shows the impact that varying the slave inquiry probability has on the number of average roles/node. There are three pairs of same color lines for each of the 30, 50 and 70 nodes with mscat turned on and off. For networks formed with the mscat protocol slave inquiry probability of 90% yields optimal number of average roles/node.

Figure 6.3 shows another measure of good network topology, the mean shortest path (msp) between any two nodes in the scatternet. As briefly discussed in Section 3.2 it is desirable for the network topology to have smaller mean shortest path between any two nodes in the scatternet because smaller msp translates into smaller average packet delivery time. In Figure 6.3 the group of lines towards the bottom of the graph are the msp for the visibility graph of the simulated scenarios. The graph shows that for scatternet topologies formed with the mscat protocol have lower mean shortest path in comparison to scatternet topologies formed without the mscat protocol.

6.2 Mscat - Impact of Parameters on Performance

Section 6.1showed that network topologies formed with the mscat protocol have better properties than network topologies formed without it. This section attempts to answer the question of what are the parameters for the mscat protocol that provide an *optimal* network topology. Note that the word *optimal* in the context of this research

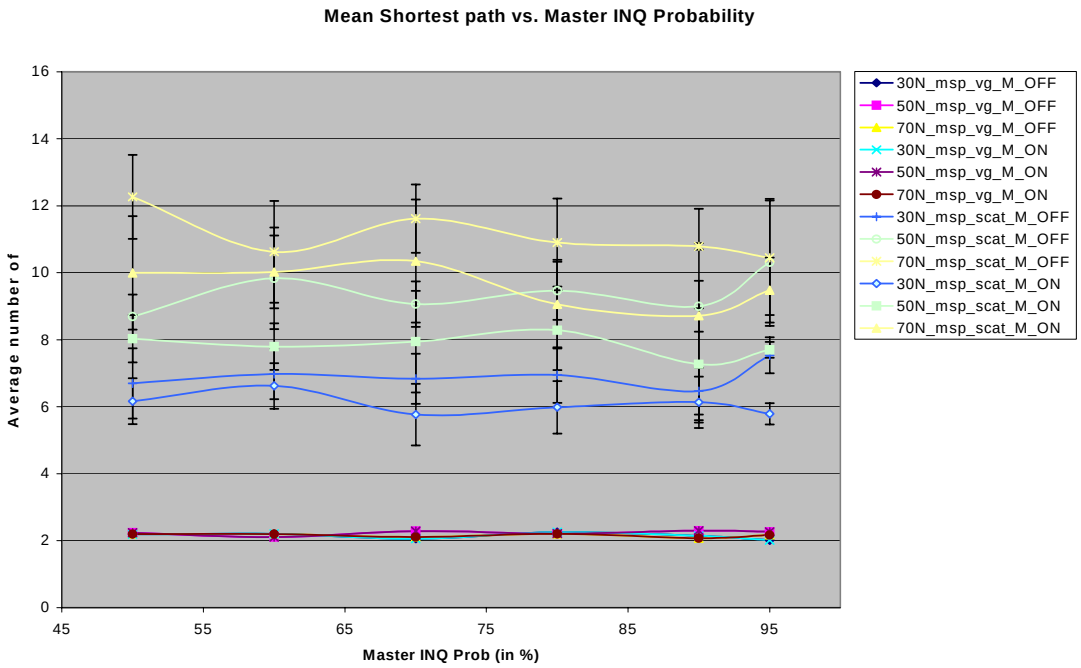


Figure 6.3: Mean shortest path vs. master inq probability

This figure shows the mean shortest path (msp) between any two nodes in the scatternet as a function of master inq. probability. There are three sets of two same color lines which compare the msp for scatternets formed with and without mscat. The collection of lines toward the bottom of the figure is the msp for the visibility graph of the nodes.

means a network topology that has properties which were outlined in Section 3.2 — Goals and Assumptions.

For the results presented in this section 35 configuration files were used as simulation input. For each configuration file the simulator ran 5 simulations with different random number generator seeds yielding a total of 175 simulations. All 35 configurations contained 50 randomly placed nodes and used the mscat protocol. The simulation space was a 2-D plane, 30 x 30 meters large.

Table 6.1 shows a summary of the varied parameters for the 35 configuration files and the seeds used. As the table shows there were 25 unique seeds used for each set of simulations that varied the slave inquiry probability (SP). This was done in order to ensure consistency of node placement in the simulation space. A side effect of this effort to maintain consistency among all configurations is that the same seed used for random node

Constants		Variables		Number of simulation repetitions for each data point	Random number generator seeds used
UP Inq Prob (%)	Number of nodes	SP Inq Prob (%)	MP Inq Prob (%)		
10	50	30	{50, 60, 70, 80, 90}	5	25 unique seeds
10	50	40	{50, 60, 70, 80, 90}	5	same 25 seeds used
10	50	50	{50, 60, 70, 80, 90}	5	same 25 seeds used
10	50	60	{50, 60, 70, 80, 90}	5	same 25 seeds used
10	50	70	{50, 60, 70, 80, 90}	5	same 25 seeds used
10	50	80	{50, 60, 70, 80, 90}	5	same 25 seeds used
10	50	90	{50, 60, 70, 80, 90}	5	same 25 seeds used

Table 6.1: Summary of simulation configurations

This table shows a summary of the free variables, variables held constant, number of simulation repetitions per configuration file and simulation seeds used to produce the data presented in this section.

placement was used for connection establishment thus yielding pathologically similar network formation patterns. This problem can be alleviated by using two random number generators with two unique seeds. One seed can be used exclusively for node placement while the other for inquiry/inquiry scan and page/page scan related operations.

Figure 6.4 shows the percent of different roles of nodes present in a network of 50 nodes. We are particularly interested in the percent of nodes with a master/slave role even though the figure shows all the possible roles nodes assume in the network. As discussed in Section 3.2 — Goals and Assumptions, the number of shared master/slave nodes when compared to the total number of nodes in the simulation is a good indicator of the efficiency of the network topology with smaller number of them being desirable.

From Figure 6.4 we can see that the number of master/slave nodes is very low for the following configurations: SP[1] = 30%, MP[2] = 90%; SP = 40%, MP = 80% and SP = 50%, MP = 80%. The configuration SP = 40%, MP = 80% has lower number of master nodes with comparison to the configuration SP = 30%, MP = 90% leading to the conclusion that the SP = 40%, MP = 80% configuration has smaller number of piconets thus yielding a denser network topology. Figure 6.5 shows the number of piconets as a function of master and slave inquiry probability. By examining Figure 6.5 we can see that

1. SP - Slave inquiry probability
2. MP - Master inquiry probability

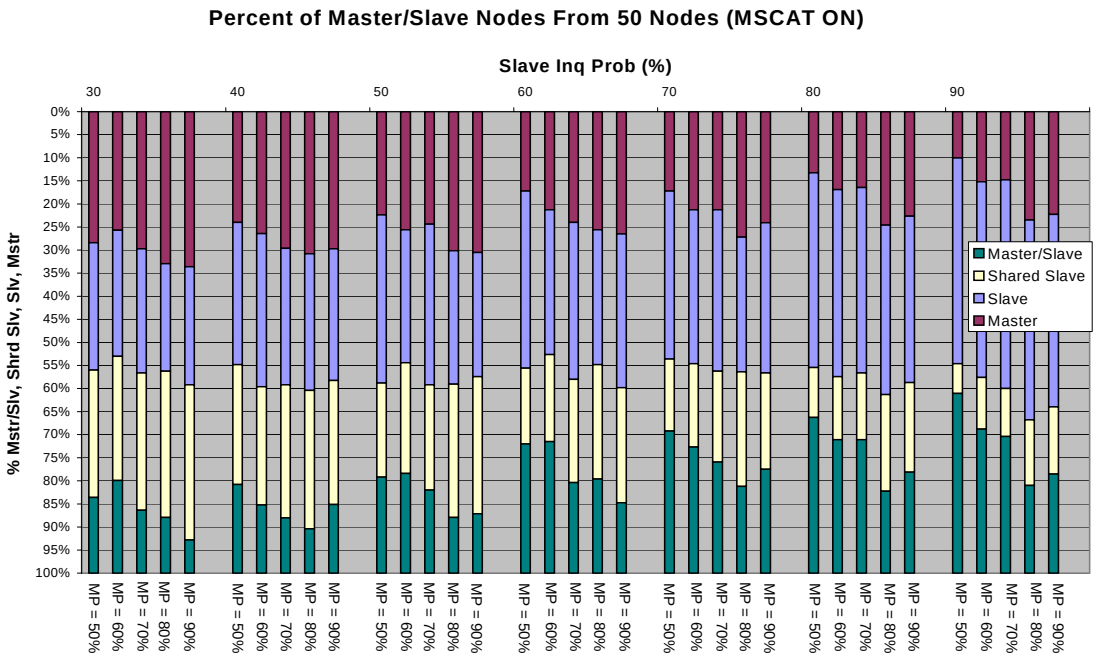


Figure 6.4: Breakdown of percentage of roles nodes assume for 50 nodes

This figure shows the percentage of roles that nodes have for each configuration scenario. Green bars show percent of master/slave nodes, white bars show percent of shared slave nodes, blue bars show percent of slave nodes and red bars show percent of master nodes. All configurations had 50 nodes.

a SP = 40% yields the smallest number of piconets for MP = {50%, 60%, 70%, 80%} for 50 node scenarios.

Figure 6.6 shows the average slaves/piconet as a function of both master and slave inquiry probabilities. For this figure the unconnected node inquiry probability (UP) was set to 10% and the number of nodes was fixed to 50. For master inquiry probabilities (MP) of 50, 60, 70 and 80 percent it can be seen from the figure that a slave probability of 40% yields the highest number of average slaves/piconet. For an MP of 90% the highest number of average slaves/piconet is reached with SP of 30%. The SP = 40%, MP = 80% configuration yields the highest number of average slaves/piconet, 2.471 slaves/piconet.

Figure 6.4, Figure 6.5 and Figure 6.6 show three measures of how good a network topology is. If we compare the results of these three figures we can conclude that the configuration UP = 10%, SP = 40%, MP = 80% yields a network topology that has optimal number of slaves/piconet, master/slave nodes and number of piconets. A network

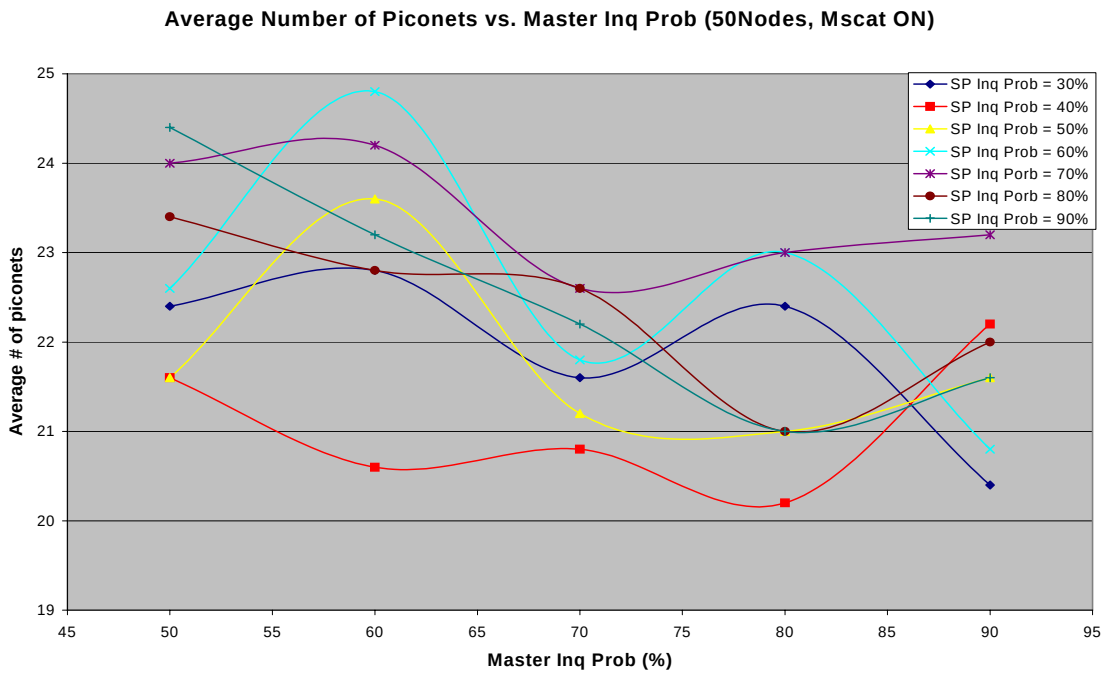


Figure 6.5: Average number of piconets vs. master and slave inq probability

This figure shows the average number of piconets as a function of master and slave inquiry probability. Smaller number of piconets for a fixed number of nodes indicates denser network topology. For this graph the UP was held constant at 10%.

topology formed with these parameters will have optimal density and minimal number of average hops between any two nodes in the network.

6.3 Mscat - Self-healing Analysis

As outlined in Section 4.1 the mscat protocol is initiated by a node performing inquiry. Since nodes in the network periodically go into inquiry, as they are required by the Bluetooth specification, the mscat protocol is capable of merging a new node in an already existing scatternet. In order to answer the question of how long does it take to merge a new node in an already existing scatternet twenty one configurations were run. The slave inquiry probability was varied from 30% to 90% in 10% increments while the master inquiry probability was held constant at 80% and the unconnected node inquiry probability was held at 10%. Scenarios containing 30, 50 and 70 nodes were examined. For each configuration file the simulator ran 5 simulations with unique random number generator seeds.

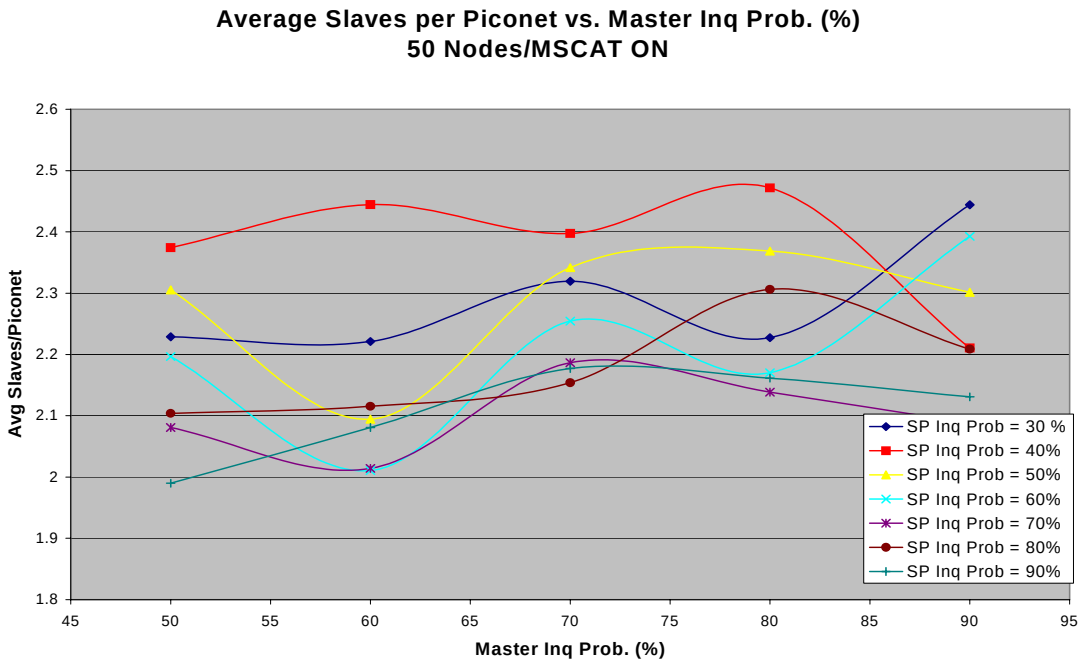


Figure 6.6: Average slaves/piconet vs. master inq prob and slave inq prob

This figure shows the average slaves/piconet as a function of the master and slave inquiry probability. 2.471 is the maximum number of slaves/piconet, obtained with the configuration SP = 40%, MP = 80%. Unconnected node (UP) inquiry probability was set to 10%.

Figure 6.7 shows the merging time as a function of the slave inquiry probability. From the figure it is evident that network topologies of 30, 50 and 70 nodes created without the mscat protocol provide for a shorter merging time of a new node. This is expected because in a network created with the mscat protocol upon a slave discovering a new node it attempts to have it's master connect to it first, time consuming activity which is not avoidable.

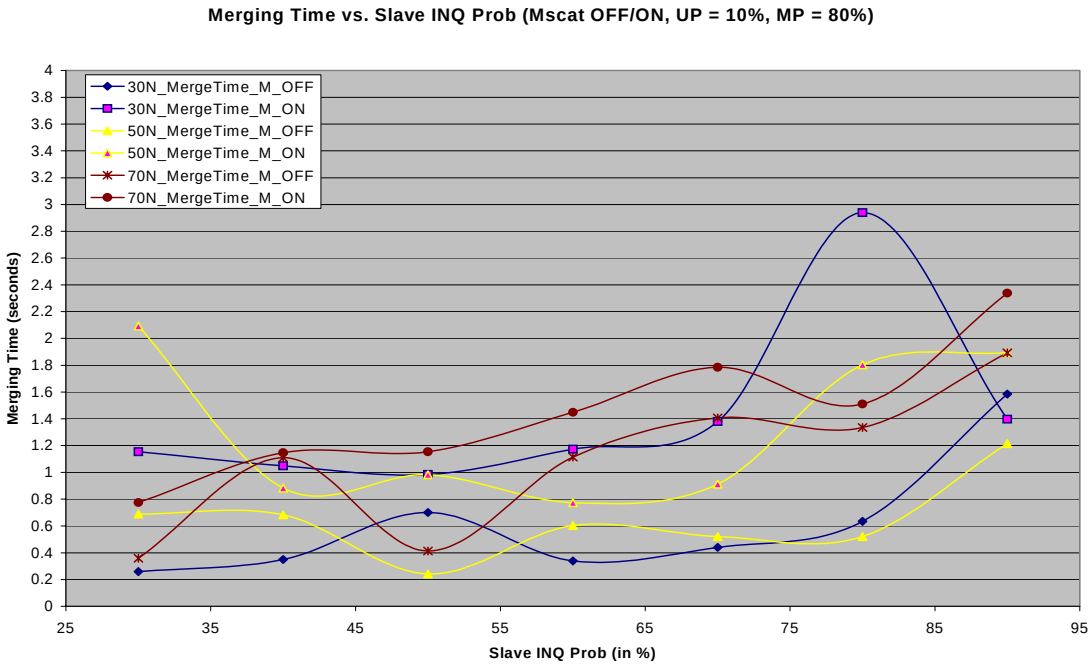


Figure 6.7: Merging time as a function of slave inquiry probability

This figure shows the merging time as a function of the slave inquiry probability. The unconnected probability is held constant at 10% and the master inquiry probability is held constant at 80%.The merging times for networks with mscat are higher than the merging times for networks formed without mscat.

Chapter 7
Conclusion

The work presented in this masters of science thesis document includes two major contributions to the field of Bluetooth enabled ad-hoc networks. The first one is the new scatternet formation and maintenance protocol and the second one is the examination of the self-healing property of the protocol. Since there has been only one publication that explores the self-healing property of a scatternet formation protocol this work is unique and different in the sense that it is a first attempt at gathering numerical results that will quantify the properties associated with self-healing. Furthermore this work shows that for specific scenarios a scatternet formation protocol can and should be designed to also act as a scatternet maintenance protocol.

The results from Chapter 6 show that mscat creates dense network topologies for networks of 30, 50 and 70 nodes. Networks using the mscat protocol have approximately 33% increase in slaves/piconet and a reduction of approximately 12.5% of average roles/ node. The mean shortest path between any two nodes in a scatternet was approximately 14.7% lower for scatternets formed with the mscat protocol. For 50 node scenarios the set of parameters which creates the best determined outcome is unconnected node inquiry probability (UP) = 10%, master node inquiry probability (MP) = 80% and slave inquiry probability (SP) = 40%.

The merging time of a new node was found to be higher for scatternets using the mscat protocol. This indicates that the mscat protocol creates denser networks at the expense of time required to merge new nodes.

7.1 Future Work

The work presented in this document makes certain assumptions which can be removed or relaxed in order to get a better picture about the scatternet topologies formed with the proposed protocol. The assumption that nodes in the simulation can have perfect and instantaneous routing information is not realistic. Implementing a real routing protocol and examining how it affects the performance of the scatternet formation and maintenance protocol can be the next step in this study.

The way shared slaves and master/slave nodes switch piconets in the simulator is rudimentary. Currently the decision to switch piconets is based on a simple timer `T_switchPiconetTO` which is set at the beginning of the simulation. A more intelligent way of deciding when a shared slave or master/slave node switches piconets can be implemented for a more accurate measure of the performance of the scatternet topology. This is a quality of service issue which was not the main focus of this study but nonetheless it should be investigated and included in a more through study.

The simulator can also be extended to implement a more accurate free space radio propagation model. As discussed in Section 5.2, the simulator currently implements very rudimentary interference model, thus a receiver will receive a packet only if the sender is within 10 meter radius and there are no other transmissions in the receiver's radio range on the same frequency.

7.2 Acknowledgments

The author extends his gratitude to the department of Electrical and Computer Engineering for supporting the research by a fellowship and a teaching assistant position. Thank you to Dr. Brian T. Davis for his advice and guidance in the past 3 years. The author would also like to thank the following colleagues who where involved in the efforts to complete the work presented in this document (in no particular order):

- Jonathan Kaus (undergraduate CpE)- whose exceptional programming skills and sheer intelligence are invaluable to the success of this project.

- Matthew Merry (graduate student U. of Michigan) - who worked on this project more than 2 years ago and who's ideas influenced this research.
- Joseph Nievelt (undergraduate CS) - whose exceptional programming skills and sheer intelligence are invaluable to the success of this project.

A special thank you is extended to all the open source developers who created and perfected the tools used in programming and running the simulator on the GNU/Linux operating system.

Appendix A

General Parameters For Simulations

A.1 Timer Values

Below is an excerpt from a configuration file that shows the timer values used for all of the simulations which generated the data presented in this document.

```
##############################
# Timer values in simulation
##############################
# Commented values are in form (low-default-high), indicating the range
# that the value can be in to be specification compliant.  The defaults are very
# long range, in that over a very long time, a lot of devices will get into the
# network.  But to obtain high speed connections, different values need to be
# used.
# Note: We don't have a Page Scan Interval Timeout (length of time between
# consecutive page scans) and PAGE_SCAN_TO should be set to less than that value.
# We go into Page_scan every time an inquiry_scan finds somebody, so at maximum,
# we will go into page_scan every (T_INQUIRY_SCAN + T_INQUIRY_SCAN_TO) ticks.
# Thus, use that value as the upper bound of T_PAGE_SCAN_TO.  Maximally it should
# be 4096.

T_PAGE_TO 3000                  # (1-8192-65440)
T_PAGE_SCAN_TO 600              # (17-18-?-4096) right now, less than 2348
T_PAGE_RESPONSE_TO 20           # (?-8-?)
T_INQUIRY_TO 2048               # (2048-?-98308)
T_INQUIRY_SCAN_TO 300           # (17-18-4096)
# this timer indicates how often a node moves from CONNECTED
# to either INQUIRY or INQUIRY SCAN states
T_INQUIRY_SCAN 500             # (18-4096-4096)  +/- 10% of value chosen randomly (so max setting is
3723)

# For the supervision timeouts, they shoudl be at least:
# T_PAGE_RESPONSE_TO + T_NEW_CONNECTION_TO + T_SWITCH_PICONET_TO + 2*T_INQUIRY_SCAN +
#    max((T_INQUIRY_TO + T_PAGE_TO),(T_INQUIRY_SCAN_TO + T_PAGE_SCAN_TO))
# with values currently set, this is about > 6500...
T_SUPERVISION_TO 7000          # (1-32000-65440)
T_SLAVE_SUPERVISION_TO 7000    # (1-32000-65440)
T_NEW_CONNECTION_TO 60          # (?-32-?)

T_INTERLACED_PAGE_SCAN_TO 2048   # Replacement PAGE_SCAN_TO when unconnected.  still needs to obey
T_PAGE_SCAN_TO ranges
T_SWITCH_PICONET_TO 50          # shared devices want to switch their current piconet after this amount
of time

# MSCAT related timer(s)
T_MSCAT_TO_STAGE1 8000          # good value should be around 3 * T_PAGE_TO, maybe :)
T_MSCAT_TO_STAGE2 80            # for stage 2 of mscat algorithm
```

A.2 Sample Config File

Below is a sample configuration file used for a simulation.

```
# Author: Rade Trimceski
# This is an Mscat.java input configuration file
# Any line in this file that starts with a # is a comment and
# is ignored by the program parser.
#
#

# GUI parameter, FALSE - GUI turned off; TRUE - GUI turned on
GUI TRUE

# Random seed to be used, comment out if you want to use the system
# clock to generate the random seed. If you know the seed used for
# a particular simulation and want to replicate it, use the same seed
#SEED 1081193916071

# Debugging parameters, TRUE - DEBUG ON; FALSE - DEBUG OFF
DEBUG_BTNode TRUE
DEBUG_Mscat TRUE

# Use stage 1 of the mscat protocol
MSCAT_1 TRUE
# Use stage 2 of the mscat protocol
MSCAT_2 TRUE

# Time (in Bluetooth ticks, i.e. 625 us) at which the sumulation terminates
ENDTIME 400000

SIMPLE_ADR TRUE

# Number of nodes in the simulation
SIMPLE_NODES 50

# How big is the simulation space in X,Y coordinates (in meters)
MAX_X 30
MAX_Y 30

# Scale and Edge are parameters for the GUI, if no GUI is used you can omit them
# Bigger SCALE number means bigger GUI window.
# EDGE is the boundary space around where the nodes go, so that there can't be a node all the way at the
edge of the
# window.  Its value is in pixels.
SCALE 10
EDGE 50

# Probabilites for each node in the simulation, depending on its role
# in the simulation.
# The numbers you can input are double and should be between 0 and 1
# Play nice, I don't do any error checking
UNCONNECTED_INQ_PROB 0.1
MASTER_INQ_PROB 0.9
SLAVE_INQ_PROB 0.3
SHARED_S_S_INQ_PROB 0.004


################################
# Timer values in simulation
################################
# Commented values are in form (low-default-high), indicating the range
# that the value can be in to be specification compliant.  The defaults are very
```

```
# long range, in that over a very long time, a lot of devices will get into the
# network.  But to obtain high speed connections, different values need to be
# used.
# Note: We don't have a Page Scan Interval Timeout (length of time between
# consecutive page scans) and PAGE_SCAN_TO should be set to less than that value.
# We go into Page_scan every time an inquiry_scan finds somebody, so at maximum,
# we will go into page_scan every (T_INQUIRY_SCAN + T_INQUIRY_SCAN_TO) ticks.
# Thus, use that value as the upper bound of T_PAGE_SCAN_TO.  Maximally it should
# be 4096.

T_PAGE_TO 3000                  # (1-8192-65440)
T_PAGE_SCAN_TO 600              # (17-18-?-4096) right now, less than 2348
T_PAGE_RESPONSE_TO 20           # (?-8-?)
T_INQUIRY_TO 2048               # (2048-?-98308)
T_INQUIRY_SCAN_TO 300           # (17-18-4096)
# this timer indicates how often a node moves from CONNECTED
# to either INQUIRY or INQUIRY SCAN states
T_INQUIRY_SCAN 500             # (18-4096-4096)  +/- 10% of value chosen randomly (so max setting is
3723)

# For the supervision timeouts, they shoudl be at least:
# T_PAGE_RESPONSE_TO + T_NEW_CONNECTION_TO + T_SWITCH_PICONET_TO + 2*T_INQUIRY_SCAN +
#    max((T_INQUIRY_TO + T_PAGE_TO),(T_INQUIRY_SCAN_TO + T_PAGE_SCAN_TO))
# with values currently set, this is about > 6500...
T_SUPERVISION_TO 7000          # (1-32000-65440)
T_SLAVE_SUPERVISION_TO 7000    # (1-32000-65440)
T_NEW_CONNECTION_TO 60          # (?-32-?)

T_INTERLACED_PAGE_SCAN_TO 2048   # Replacement PAGE_SCAN_TO when unconnected.  still needs to obey
T_PAGE_SCAN_TO ranges
T_SWITCH_PICONET_TO 50          # shared devices want to switch their current piconet after this amount
of time

# MSCAT related timer(s)
T_MSCAT_TO_STAGE1 8000          # good value should be around 3 * T_PAGE_TO, maybe :)
T_MSCAT_TO_STAGE2 80            # for stage 2 of mscat algorithm
```

Appendix B

Tables of Data

B.1 Table For Section 6.1

The following tables contains the data presented in Section 6.1 — Mscat - Performance Analysis.

MP_PROB_M_OFF	RUN	NODES_MOFF	CONN_M_OFF	UCON_M_OFF	avg_slv_pi_M_OFF	mstrs_M_OFF	slvs_M_OFF	ss_M_OFF	ms_M_OFF	avg_role_M_OFF	scat_tim_M_OFF	bw_util_M_OFF	msp_visb_M_OFF	msp_scat_M_OFF
50	1	30	30	0	1.933333333	7	9	6	8	1.466666667	10.128125	37.54963317	2.17011	5.51264
	2	30	30	0	1.260869565	6	2	5	17	1.733333333	14.585	42.3180888	2.27586	7.42069
	3	30	30	0	1.933333333	8	8	7	7	1.466666667	12.525	38.04461597	2.4092	6.12874
	4	30	30	0	1.45	8	3	7	12	1.633333333	17.21375	38.78246574	2.03218	7.72414
	5	30	29	1	1.647058824	6	7	5	11	1.551724138		42.07498054	2.05665	
60	1	30	30	0	1.4	10	2	8	10	1.6		37.30200102	2.02069	
	2	30	30	0	1.705882353	6	7	6	11	1.566666667	9.285625	41.36504934	2.28736	6.36552
	3	30	30	0	1.45	5	6	4	15	1.633333333	16.355625	41.2613228	2.29885	6.46667
	4	30	30	0	1.666666667	8	4	8	10	1.6	8.024375	38.24830822	2.25747	7.06207
	5	30	30	0	1.526315789	9	3	8	10	1.6	10.258125	38.42730252	2.15632	8.0092
70	1	30	30	0	1.705882353	6	8	5	11	1.533333333	11.236875	41.4963464	1.84138	7.33333
	2	30	30	0	1.933333333	9	7	8	6	1.466666667	9.926875	32.35457028	1.98161	6.03678
	3	30	30	0	1.705882353	6	8	5	11	1.533333333	16.78125	39.88195941	2.09885	6.69425
	4	30	30	0	1.705882353	8	6	7	9	1.533333333	8.49375	37.02848118	2.07126	6.26667
	5	30	30	0	1.45	8	3	7	12	1.633333333	15.899375	36.97474072	2.29885	7.83908
80	1	30	30	0	2.416666667	9	10	8	3	1.366666667	11.279375	30.74355161	2.26207	6.51034
	2	30	30	0	2.071428571	9	8	8	5	1.433333333	28.75625	29.85913962	2.2046	7.28966
	3	30	30	0	1.875	10	4	10	6	1.533333333	20.814375	32.23741141	2.22989	5.85747
	4	30	30	0	1.875	10	4	10	6	1.533333333	16.34625	33.3749008	2.57011	7.0046
	5	30	30	0	1.666666667	9	3	9	9	1.6	9.51625	33.23176453	1.9931	8.07126
90	1	30	29	1	1.8125	9	5	8	7	1.517241379	27.544375	33.39485217	2.12874	6.50246
	2	30	30	0	1.8125	10	5	9	6	1.5		31.48419649	2.17011	
	3	30	29	1	1.666666667	10	1	10	8	1.620689655	18.126875	30.55677883	1.84598	6.26847
	4	30	30	0	1.8125	8	7	7	8	1.5	28.77125	34.51464592	2.36782	7.37471
	5	30	30	0	1.933333333	9	7	8	6	1.466666667	9.249375	31.07931576	2.21609	5.70575
95	1	30	30	0	1.526315789	11	1	10	8	1.6	6.303125	29.99628205	2.05977	7.74023
	2	30	30	0	1.705882353	10	4	9	7	1.533333333	40.52625	31.07471529	1.84828	7.1931
	3	30	30	0	1.611111111	11	2	10	7	1.566666667	12.55875	28.36548592	2.06897	8.38621
	4	30	30	0	1.8125	10	5	9	6	1.5	13.99875	31.20467512	2.04138	7.28736
	5	30	30	0	1.8125	11	4	10	5	1.5	11.52	28.17958378	2.04828	7.06207
50	1	50	50	0	1.515151515	11	6	11	22	1.66	11.378125	41.72888675	2.31673	8.57306
	2	50	50	0	1.814814815	10	13	10	17	1.54	9.415	40.42434692	2.23184	8.1551
	3	50	50	0	1.612903226	10	9	10	21	1.62	6.91125	41.83007102	2.02286	9.83755
	4	50	50	0	1.785714286	11	11	11	17	1.56	7.2825	40.97229382	2.29633	8.41388
	5	50	50	0	1.612903226	12	7	12	19	1.62	11.284375	40.87213235	2.32163	8.47347
60	1	50	50	0	1.75	13	9	13	15	1.56	14.035625	37.62453977	1.96	11.3543
	2	50	50	0	1.633333333	8	13	7	22	1.58	17.018125	42.16688183	2.29714	10.7388
	3	50	50	0	1.5625	10	8	10	22	1.64	18.01125	40.90991451	2.01878	8.69796
	4	50	50	0	1.470588235	12	4	12	22	1.68	14.11375	41.46569183	2.17796	10.5967
	5	50	50	0	1.689655172	9	13	8	20	1.56	10.189375	40.18263902	2.07755	7.78776
70	1	50	50	0	1.666666667	12	8	12	18	1.6	17.606875	39.19768906	2.13878	9.50531
	2	50	50	0	1.689655172	14	7	14	15	1.58	15.85125	37.16434717	2.38694	9.63673
	3	50	50	0	1.724137931	11	9	12	18	1.6	12.044375	39.26627999	2.31755	9.34531
	4	50	50	0	1.884615385	15	10	14	11	1.5	11.80625	35.45130611	2.2702	8.84653
	5	50	50	0	1.7	10	9	11	20	1.62	9.5225	39.77744101	2.32	7.98204
80	1	50	50	0	1.515151515	10	7	10	23	1.66	18.49375	40.15466423	2.49469	9.69796
	2	50	50	0	1.851851852	14	9	14	13	1.54	18.488125	35.95370967	2.07837	9.03918
	3	50	50	0	1.580645161	11	9	10	20	1.6	14.285625	37.14901299	2.26367	10.2163
	4	50	50	0	2.041666667	13	13	13	11	1.48	8.4275	33.98988365	2.2351	10.1984
	5	50	50	0	1.633333333	11	10	10	19	1.58	17.6275	37.31001986	1.96408	8.17714
90	1	50	50	0	1.785714286	15	7	15	13	1.56	11.5725	32.72864726	2.24653	8.67184
	2	50	50	0	1.814814815	15	8	15	12	1.54	9.1975	33.34806841	2.23429	9.36245
	3	50	50	0	1.814814815	15	9	14	12	1.52	14.531875	32.58433854	2.45143	9.46204
	4	50	49	1	1.75862069	13	6	14	16	1.612244898	19.6225	35.87231952	2.38857	7.82143
	5	50	50	0	1.884615385	13	12	12	13	1.5	14.80875	35.07922869	2.1902	9.67918
95	1	50	49	1	1.689655172	15	6	14	14	1.571428571	14.9975	33.74342219	2.44	12.4286
	2	50	50	0	1.484848485	16	2	15	17	1.64	13.071875	33.40120559	2.22041	10.0637
	3	50	50	0	2.083333333	15	11	15	9	1.48	17.348125	31.25571749	2.32245	7.56327
	4	50	50	0	1.814814815	17	7	16	10	1.52	17.99625	30.45186728	2.13143	11.7184
	5	50	50	0	1.724137931	12	9	12	17	1.58	10.23625	37.20334872	2.21878	9.76082
50	1	70	70	0	1.533333333	12	13	12	33	1.642857143	10.216875	42.88047423	2.22899	12.648
	2	70	70	0	1.489361702	15	6	17	32	1.7	9.336875	41.69214607	2.08406	10.419
	3	70	70	0	1.533333333	14	12	13	31	1.628571429		42.43034571	2.3412	
	4	70	70	0	1.408163265	14	8	13	35	1.685714286	12.523125	42.10549018	2.12671	12.7478
	5	70	70	0	1.468085106	13	11	12	34	1.657142857	17.87125	42.37685149	2.20952	13.2393
60	1	70	70	0	1.533333333	13	13	12	32	1.628571429	14.621875	41.29745439	2.18675	11.0033
	2	70	70	0	1.674418605	14	11	16	29	1.642857143	12.691875	39.73918232	2.17184	8.52133
	3	70	70	0	1.604651163	17	11	16	26	1.6	15.49	39.80859054	2.10228	12.1176
	4	70	70	0	1.5	15	9	15	31	1.657142857	11.583125	40.11500216	2.37226	11.8344
	5	70	70	0	1.707317073	11	18	11	30	1.585714286	9.925	42.11136876	2.15859	9.65135
70	1	70	70	0	1.731707317	17	10	19	24	1.614285714	11.080625	37.38291651	2.15942	12.328
	2	70	70	0	1.75	20	10	20	20	1.571428571	15.445	35.59442869	2.01201	12.699
	3	70	70	0	1.533333333	17	9	16	28	1.628571429	27.33875	39.26491998	2.07288	11.9702
	4	70	70	0	1.894736842	23	7	25	15	1.571428571	14.01625	33.05718993	2.22319	10.5607
	5	70	70	0	1.642857143	20	9	19	22	1.585714286	13.74875	36.83610771	2.07039	10.511
80	1	70	70	0	1.613636364	13	12	14	31	1.642857143	7.615625	40.9959469	2.21656	9.52091
	2	70	70	0	2	18	14	20	18	1.542857143	15.99375	35.55512416	2.18634	11.2923
	3	70	70	0	1.725	19	11	19	21	1.571428571	22.598125	35.62584606	2.15776	9.89814
	4	70	70	0	1.815789474	18	15	17	20	1.528571429	13.429375	35.44886605	2.24928	10.9238
	5	70	70	0	1.568181818	17	10	16	27	1.614285714	11.981875	37.70391839	2.21408	12.8534
90	1	70	70	0	2.028571429	21	13	22	14	1.514285714	14.19125	32.77789595	2.05424	10.752
	2	70	70	0	1.769230769	22	11	20	17	1.528571429	22.481875	32.75362476	2.18054	10.6633
	3	70	70	0	1.820512821	19	11	20	20	1.571428571	12.42375	34.42828581	2.06087	10.7656
	4	70	70	0	1.864864865	20	14	19	17	1.514285714	7.7275	34.7962157	1.95155	10.9284
	5	70	70	0	1.916666667	20	15	19	16	1.5	11.224375	34.16914335	2.12133	10.8124
95	1	70	70	0	1.75	21	9	21	19	1.571428571	8.92875	33.207908	2.12257	11.3404
	2	70	70	0	1.868421053	22	9	23	16	1.557142857	28.168125	31.05351821	2.26667	7.93582
	3	70	69	1	1.642857143	21	7	20	21	1.594202899	13.73625	33.37656677	2.19586	9.98465
	4	70	69	1	1.820512821	24	5	25	15	1.579710145	23.27625	30.8913394	2.12961	10.4365
	5	70	70	0	1.944444444	23	11	23	13	1.514285714	20.26625	30.41520837	2.15238	12.5317

UP_PROB_M_OFF	RUN	NODES_MOFF	CONN_M_OFF	UCON_M_OFF	avg_slv_pi_M_OFF	mstrs_M_OFF	slvs_M_OFF	ss_M_OFF	ms_M_OFF	avg_role_M_OFF	scat_tim_M_OFF	bw_util_M_OFF	msp_visb_M_OFF	msp_scat_M_OFF
10	1	30	30	0	1.8125	9	6	8	7	1.5	16.14375	32.76014503	2.15172	6.2
	2	30	30	0	2.142857143	9	6	10	5	1.5	8.38875	30.37071347	1.88046	5.42529
	3	30	30	0	1.705882353	8	6	7	9	1.533333333	16.7825	34.87796971	2.53563	7.03448
	4	30	30	0	1.8125	10	5	9	6	1.5	9.10875	31.18268864	2.3931	7.10115
	5	30	30	0	2.071428571	8	9	7	6	1.433333333	7.846875	30.51691288	2.10805	6.89429
20	1	30	30	0	1.933333333	10	6	9	5	1.466666667	16.405	29.26836701	2.08966	7.72644
	2	30	30	0	1.933333333	10	6	9	5	1.466666667	7.918125	29.21146738	1.91954	8.37241
	3	30	30	0	2.071428571	12	5	11	2	1.433333333	20.025625	23.80260019	2.2069	6.43908
	4	30	30	0	1.8125	10	5	9	6	1.5	11.334375	29.61085894	2.10805	7.17471
	5	30	30	0	1.611111111	9	4	8	9	1.566666667	13.485625	34.22185979	2.5908	8.41379
30	1	30	30	0	1.611111111	11	2	10	7	1.566666667	49.123125	31.66847834	2.58161	9.71034
	2	30	30	0	1.526315789	10	2	9	9	1.6	15.623125	31.95608326	2.18621	8.46207
	3	30	30	0	1.45	10	1	9	10	1.633333333	44.051875	33.00903018	2.11724	7.62299
	4	30	30	0	2.071428571	12	5	11	2	1.433333333	15.121875	24.46347892	2.38851	7.09425
	5	30	30	0	2.066666667	10	4	11	5	1.533333333	29.4825	27.8656134	2.04828	6.95862
40	1	30	29	1	1.473684211	11	0	10	8	1.620689655		31.40508796	2.22906	
	2	30	30	0	1.705882353	12	2	11	5	1.533333333	22.10375	28.24683266	2.16552	7.21839
	3	30	30	0	1.705882353	11	3	10	6	1.533333333	26.735	28.15695879	2.4	8.4046
	4	30	30	0	1.823529412	12	1	12	5	1.566666667	25.42125	26.46925667	2	6.05287
	5	30	30	0	1.875	9	5	9	7	1.533333333	27.10875	31.87583244	2.09195	6.62759
50	1	30	30	0	1.611111111	13	0	12	5	1.566666667	20.70875	25.59713596	2.12874	6.15862
	2	30	30	0	1.8125	13	2	12	3	1.5	19.860625	26.07679228	2.1931	7.15172
	3	30	30	0	1.705882353	11	3	10	6	1.533333333	10.936875	28.84162019	1.90805	7.63218
	4	30	30	0	1.705882353	12	2	11	5	1.533333333	28.595	26.75241947	2.4	8.43218
	5	30	30	0	1.933333333	10	6	9	5	1.466666667	23.650625	28.15800468	2.18851	8.02069
60	1	30	30	0	1.473684211	13	0	11	6	1.566666667		28.51262723	2.16552	
	2	30	30	0	1.555555556	10	4	8	8	1.533333333		31.13398393	2.13103	
	3	30	30	0	1.647058824	13	2	11	4	1.5		26.60046916	2.31494	
	4	30	30	0	1.611111111	11	2	10	7	1.566666667	13.875	29.87891864	2.03218	7.42069
	5	30	30	0	1.705882353	10	4	9	7	1.533333333	41.1125	29.70548446	2.13103	7.6023
70	1	30	30	0	1.473684211	11	2	9	8	1.566666667		30.10806328	2.58391	
	2	30	30	0	1.705882353	14	0	13	3	1.533333333	19.18125	25.14273989	2.4046	6.86897
	3	30	30	0	1.526315789	11	1	10	8	1.6	44.95875	28.35660042	2.2	8.07126
	4	30	30	0	1.705882353	12	2	11	5	1.533333333	23.014375	26.37437033	2.50345	7.87356
	5	30	30	0	1.866666667	14	3	12	1	1.433333333		22.67975135	2.25747	
10	1	50	50	0	1.612903226	14	5	14	17	1.62	16.94625	35.86472169	2.33143	11.7412
	2	50	50	0	1.75862069	14	6	15	15	1.6	9.66375	35.09472863	2.32	8.2702
	3	50	50	0	1.884615385	16	9	15	10	1.5	19.411875	31.76489168	2.04816	7.87265
	4	50	50	0	1.851851852	14	9	14	13	1.54	16.554375	33.62976745	2.21061	9.02286
	5	50	50	0	1.75	13	10	12	15	1.54	7.423125	35.93401387	2.09878	9.43102
20	1	50	50	0	1.666666667	17	3	17	13	1.6	19.67	32.26303381	2.34857	9.72898
	2	50	49	1	1.548387097	17	2	16	14	1.612244898		33.32358827	2.34776	
	3	50	49	1	1.75	19	3	18	9	1.551020408	15.93	28.83744399	2.46286	7.94218
	4	50	50	0	1.814814815	19	5	18	8	1.52	26.36875	28.6747867	2.15347	11.6727
	5	50	50	0	1.724137931	18	3	18	11	1.58	17.480625	30.90545482	2.06612	8.80327
30	1	50	50	0	1.884615385	17	8	16	9	1.5	23.32375	30.23926737	2.43184	8.3951
	2	50	50	0	1.724137931	14	7	14	15	1.58	38.06625	33.96755955	2.50367	7.63347
	3	50	50	0	2.041666667	19	8	18	5	1.46	23.223125	26.65255401	2.16	9.89143
	4	50	50	0	1.814814815	19	5	18	8	1.52	18.34	28.47883936	2.03265	9.18612
	5	50	50	0	1.785714286	18	4	18	10	1.56	20.578125	30.57176737	2.19918	10.1731
40	1	50	50	0	1.580645161	17	3	16	14	1.6	26.345	31.02556483	2.25959	13.8955
	2	50	50	0	1.814814815	18	6	17	9	1.52	33.255	29.56463245	2.37061	11.5224
	3	50	50	0	1.580645161	19	1	18	12	1.6	45.84	30.00808796	2.06612	10.6139
	4	50	50	0	1.75	17	5	17	11	1.56		31.27428087	2.10204	
	5	50	50	0	1.724137931	18	3	18	11	1.58	17.626875	31.19992047	2.17061	10.111
50	1	50	50	0	1.75	23	1	21	5	1.52	29.01125	25.04447106	2.03837	9.86565
	2	50	50	0	1.75	20	2	20	8	1.56	19.390625	26.80276916	2.01224	9.25306
	3	50	50	0	1.75	21	2	20	7	1.54	41.07	26.8097518	2.29796	13.5363
	4	50	50	0	1.75	20	3	19	8	1.54	30.969375	29.16086524	2.20735	8.54939
	5	50	50	0	1.733333333	17	1	19	13	1.64	47.59125	29.62576266	2.12408	7.67755
60	1	50	50	0	1.851851852	18	5	18	9	1.54	29.440625	29.49008075	2.05224	7.60571
	2	50	50	0	1.6	21	1	19	9	1.56		28.4169551	2.07918	
	3	50	50	0	1.814814815	21	3	20	6	1.52		25.94579582	2.15102	
	4	50	50	0	1.814814815	19	5	18	8	1.52	34.671875	26.60998792	2.28571	10.3159
	5	50	50	0	2	18	7	18	7	1.5	26.1075	28.62527982	2.14939	8.73306
70	1	50	50	0	1.714285714	19	5	17	9	1.52		28.33586921	2.26612	
	2	50	50	0	1.814814815	22	2	21	5	1.52	31.395625	25.30038338	2.03673	11.1918
	3	50	50	0	1.548387097	20	2	17	11	1.56		28.28502587	2.23102	
	4	50	50	0	1.689655172	21	2	19	8	1.54	28.554375	26.35374799	2.29959	9.14041
	5	50	50	0	1.785714286	18	4	18	10	1.56	26.47375	29.06907419	2.1249	9.44245
10	1	70	70	0	1.794871795	23	8	23	16	1.557142857	24.1725	31.21532373	2.23313	10.8725
	2	70	70	0	1.916666667	23	12	22	13	1.5	10.08	31.60006056	2.20828	9.72091
	3	70	70	0	1.842105263	22	10	22	16	1.542857143	13.184375	32.89471385	2.14824	9.65424
	4	70	69	1	1.891891892	20	12	20	17	1.536231884	12.30875	33.68159706	2.24886	10.7298
	5	70	69	1	2	22	12	22	13	1.507246377	13.353125	30.88948324	2.15114	12.896
20	1	70	70	0	1.815789474	26	7	25	12	1.528571429	14.9	28.04875611	2.03934	13.947
	2	70	70	0	1.794871795	23	8	23	16	1.557142857	20.2325	30.69101702	2.0559	9.3706
	3	70	70	0	1.642857143	26	3	25	16	1.585714286	21.980625	30.22035881	2.14824	12.6923
	4	70	70	0	1.868421053	24	7	25	14	1.557142857	23.739375	30.92889891	2.17805	9.47412
	5	70	70	0	1.842105263	23	8	24	15	1.557142857	18.7	30.65805517	2.31553	10.8919
30	1	70	70	0	1.75	29	1	29	11	1.571428571	18.335625	24.94898938	2.13954	10.9901
	2	70	70	0	1.864864865	27	7	26	10	1.514285714	19.313125	26.76306725	2.05549	13.7184
	3	70	70	0	1.731707317	25	3	26	16	1.6	23.911875	30.37143158	2.15362	10.9081
	4	70	70	0	1.642857143	25	4	24	17	1.585714286	20.72875	29.37571826	2.13085	16.2257
	5	70	70	0	1.794871795	27	4	27	12	1.557142857	17.723125	28.00990839	2.21615	10.8464
40	1	70	70	0	1.725	26	5	25	14	1.557142857	24.14875	29.41281315	2.50807	13.4766
	2	70	70	0	1.891891892	25	8	25	12	1.528571429	15.31125	28.71481947	2.18965	9.3205
	3	70	70	0	1.707317073	25	5	24	16	1.571428571	24.501875	29.34590126	2.2294	5.21739
	4	70	70	0	1.75	25	5	25	15	1.571428571	13.94125	29.29392966	2.30849	12.0468
	5	70	70	0	1.842105263	29	4	28	9	1.528571429	26.27	26.33568889	2.06832	11.0878
50	1	70	69	1	1.769230769	25	7	23	14	1.536231884	32.7775	29.35424897	2.18758	7.03666
	2	70	70	0	1.658536585	27	4	25	14	1.557142857		29.89608951	2.32464	
	3	70	70	0	1.815789474	32	2	30	6	1.514285714	23.470625	24.47455624	2.2323	11.7494
	4	70	70	0	1.568181818	24	3	23	20	1.614285714	42.601875	31.55867878	2.06335	12.077
	5	70	70	0	1.794871795	28	3	28	11	1.557142857	32.335	27.8305267	2.19337	11.3665
60	1	70	70	0	1.815789474	27	6	26	11	1.528571429	37.085	27.94644844	2.10352	12.7284
	2	70	70	0	1.842105263	29	3	29	9	1.542857143	31.82125	26.20799285	2.18261	11.0489
	3	70	70	0	1.725	26	5	25	14	1.557142857	22.469375	28.97426143	2.18841	11.8029
	4	70	70	0	1.891891892	27	6	27	10	1.528571429	20.76875	26.54947528	2.10104	11.0248
	5	70	70	0	1.891891892	28	5	28	9	1.528571429	41.015625	26.70383415	2.01946	10.846
70	1	70	70	0	1.682926829	28	3	26	13	1.557142857	46.524375	25.77082915	2.06501	10.8195
	2	70	70	0	1.658536585	31	0	29	10	1.557142857		26.93178433	2.10807	
	3	70	70	0	1.682926829	26	4	25	15	1.571428571		30.47990559	2.12961	
	4	70	70	0	1.627906977	25	2	25	18	1.614285714	31.2025	30.39678002	2.39089	13.1648
	5	70	70	0	1.769230769	30	3	28	9	1.528571429	46.433125	25.79371374	2.05135	9.54161

SP_PROB_M_OFF	RUN	NODES_MOFF	CONN_M_OFF	UCON_M_OFF	avg_slv_pi_M_OFF	mstrs_M_OFF	slvs_M_OFF	ss_M_OFF	ms_M_OFF	avg_role_M_OFF	scat_tim_M_OFF	bw_util_M_OFF	msp_visb_M_OFF	msp_scat_M_OFF	
30	1	30	30	0	1.578947368	10	1	10	9	1.633333333	23.776875	32.74654752	2.03908	6.77011	
	2	30	30	0	1.611111111	7	6	6	11	1.566666667	15.1875	34.30866096	2.14483	6.8069	
	3	30	30	0	1.933333333	8	8	7	7	1.466666667	8.398125	34.33721706	2.44368	6.87816	
	4	30	30	0	1.705882353	7	7	6	10	1.533333333		37.31048299	2.98851		
	5	30	30	0	1.933333333	10	6	9	5	1.466666667		30.39496636	2.12414		
40	1	30	30	0	1.705882353	8	6	7	9	1.533333333	10.01375	35.48584052	2.12874	8.23908	
	2	30	30	0	1.526315789	7	4	7	12	1.633333333	12.228125	38.28428074	1.92874	7.53103	
	3	30	30	0	2.066666667	9	5	10	6	1.533333333	14.016875	30.59090981	2.14483	5.75862	
	4	30	30	0	1.45	6	6	4	14	1.6	15.6425	40.17230394	2.03908	1.8023	
	5	30	29	1	1.705882353	9	3	9	8	1.586206897		32.16855349	2.11576		
50	1	30	30	0	1.526315789	7	5	6	12	1.6	14.34625	36.93098385	2.29195	7.82529	
	2	30	30	0	1.526315789	8	4	7	11	1.6	14.361875	36.81864705	2.81839	7.70575	
	3	30	30	0	1.705882353	7	7	6	10	1.533333333	12.16375	37.04717682	2.35632	6.84368	
	4	30	30	0	1.933333333	8	8	7	7	1.466666667	23.2575	34.38808956	1.98621	7.47126	
	5	30	30	0	1.421052632	7	7	4	12	1.533333333		38.35641766	2.09655		
60	1	30	30	0	1.45	6	5	5	14	1.633333333	31.23	38.60929661	2.4023	7.35862	
	2	30	30	0	1.611111111	8	5	7	10	1.566666667	25.024375	34.6540757	2.33793	7.89885	
	3	30	30	0	1.611111111	7	6	6	11	1.566666667	45.510625	34.85974013	2.42989	7.03448	
	4	30	30	0	1.526315789	8	4	7	11	1.6	19.969375	34.56656554	2.03908	6.46667	
	5	30	30	0	1.4	8	4	6	12	1.6		35.73713979	2.47816		
70	1	30	30	0	1.526315789	8	4	7	11	1.6	28.1625	34.00012582	1.8092	7.61149	
	2	30	30	0	1.45	4	7	3	16	1.633333333	12.830625	43.05925057	2.41839	7.9954	
	3	30	30	0	1.380952381	6	4	5	15	1.666666667	38.515	38.64666291	2.18851	8.31034	
	4	30	30	0	1.4	7	5	5	13	1.6	24.74375	38.95117876	2.70575	4.27816	
	5	30	30	0	1.611111111	8	5	7	10	1.566666667	42.22625	34.19040123	2.22529	6.2092	
80	1	30	30	0	1.611111111	6	7	5	12	1.566666667	24.343125	38.85444216	1.62069	5.64828	
	2	30	30	0	1.380952381	5	5	4	16	1.666666667		39.82868419	2.48276		
	3	30	30	0	1.647058824	7	8	5	10	1.5		36.2175436	2.0023		
	4	30	30	0	1.473684211	8	5	6	11	1.566666667	33.333125	36.52679628	2.6	5.90805	
	5	30	30	0	1.260869565	7	1	6	16	1.733333333		39.91035149	2.12644		
90	1	30	30	0	1.388888889	9	8	4	9	1.433333333		33.77853391	2.13333		
	2	30	30	0	1.35	7	6	4	13	1.566666667	28.090625	36.67323215	2.02989	1.12414	
	3	30	30	0	1.318181818	5	4	4	17	1.7		41.86981981	2.12414		
	4	30	30	0	1.421052632	6	8	3	13	1.533333333		37.10869668	2.21609		
	5	30	30	0	1.473684211	5	8	3	14	1.566666667	10.6775	39.03670157	2.29885	3.14943	
30	1	50	50	0	1.689655172	10	12	9	19	1.56	12.3025	38.52894819	2.25714	9.08653	
	2	50	50	0	1.484848485	14	4	13	19	1.64	42.058125	36.15909592	2.34286	10.222	
	3	50	50	0	1.961538462	16	7	17	10	1.54	9.0875	31.27619046	2.36816	6.35184	
	4	50	50	0	2.083333333	11	15	11	13	1.48	21.085625	35.82228765	2.2098	8.42776	
	5	50	50	0	1.470588235	14	2	14	20	1.68	13.861875	36.02926384	2.2702	9.72408	
40	1	50	50	0	1.580645161	14	6	13	17	1.6	10.49625	36.90131024	2.32082	9.12	
	2	50	50	0	1.612903226	15	4	15	16	1.62	12.4275	34.66087417	2.14531	9.05633	
	3	50	50	0	1.884615385	14	11	13	12	1.5	22.46	33.54931732	2.28408	10.458	
	4	50	50	0	1.484848485	13	5	12	20	1.64	23.44125	37.63817663	2.09224	10.711	
	5	50	49	1	1.689655172	14	7	13	15	1.571428571	16.6775	35.27724403	1.97469	9.87075	
50	1	50	49	1	1.689655172	11	10	10	18	1.571428571	17.185625	36.16292722	2.30612	9.26531	
	2	50	50	0	1.689655172	13	9	12	16	1.56	32.05875	34.10082248	2.20327	8.17469	
	3	50	50	0	1.441176471	13	4	12	21	1.66	45.88375	37.6289467	2.17388	10.7959	
	4	50	50	0	1.689655172	13	9	12	16	1.56	19.331875	35.0957169	2.14939	9.20327	
	5	50	49	1	1.884615385	15	9	14	11	1.510204082	24.614375	33.42638354	2.12898	8.34524	
60	1	50	50	0	1.484848485	10	8	9	23	1.64	30.345	39.96133599	1.93878	12.1722	
	2	50	50	0	1.689655172	16	6	15	13	1.56	36.534375	32.87095568	2.23347	9.76082	
	3	50	50	0	1.923076923	13	11	13	13	1.52	19.453125	33.53754532	2.16245	9.47265	
	4	50	50	0	1.4	13	3	12	22	1.68	27.40625	35.81459244	2.17714	9.90041	
	5	50	50	0	1.484848485	13	5	12	20	1.64	36.8925	35.64224059	2.13143	9.04163	
70	1	50	50	0	1.53125	14	5	13	18	1.62	26.970625	35.79392272	2.21224	10.3176	
	2	50	50	0	1.441176471	10	7	9	24	1.66		38.89839537	2.57551		
	3	50	50	0	1.454545455	12	7	10	21	1.62	28.59875	37.67072413	2.25469	7.33143	
	4	50	50	0	1.633333333	12	9	11	18	1.58	35.6	36.28036917	2.45061	9.13633	
	5	50	50	0	1.441176471	13	4	12	21	1.66	35.9375	36.36833653	2.2	11.9698	
80	1	50	50	0	1.484848485	10	8	9	23	1.64	40.675625	39.05841769	2.20898	9.60408	
	2	50	50	0	1.4	7	9	6	28	1.68	39.6225	43.05684713	2.17306	12.68	
	3	50	50	0	1.441176471	8	9	7	26	1.66		41.21554639	2.16735		
	4	50	50	0	1.454545455	12	7	10	21	1.62	34.46125	36.4265966	2.2449	4.46857	
	5	50	50	0	1.484848485	10	8	9	23	1.64	45.096875	37.97379714	2.10612	12.031	
90	1	50	50	0	1.4	7	8	7	28	1.7		41.44004578	2.04		
	2	50	50	0	1.454545455	9	10	7	24	1.62		40.50196492	2.33469		
	3	50	50	0	1.454545455	8	11	6	25	1.62		41.16706219	2.17061		
	4	50	50	0	1.393939394	10	11	6	23	1.58		36.91164632	2.19184		
	5	50	50	0	1.243243243	13	4	9	24	1.66	13.84375	37.90041755	2.04816	3.64571	
30	1	70	70	0	1.794871795	21	10	21	18	1.557142857	9.941875	34.27391343	2.28282	10.7226	
	2	70	70	0	1.918918919	18	14	19	19	1.542857143	24.915625	34.76239655	2.10849	10.5752	
	3	70	70	0	1.707317073	19	10	19	22	1.585714286	14.91375	35.58234782	2.12671	9.9118	
	4	70	70	0	1.775	21	8	22	19	1.585714286	25.893125	34.01997584	2.15569	10.4381	
	5	70	70	0	1.707317073	23	6	23	18	1.585714286	21.465625	32.44646302	2.25797	12.0472	
40	1	70	69	1	1.75	16	13	16	24	1.579710145	17.4075	37.16993683	2.15901	11.1675	
	2	70	70	0	1.6	17	6	19	28	1.671428571	23.91	36.29715569	2.22277	9.45466	
	3	70	70	0	1.682926829	21	9	20	20	1.571428571	14.3075	34.01607237	2.23892	12.2145	
	4	70	70	0	2.029411765	19	18	18	15	1.471428571	18.860625	34.01918405	1.96149	10.7772	
	5	70	70	0	1.707317073	21	8	21	20	1.585714286	23.48	34.80956021	2.0646	11.9176	
50	1	70	70	0	1.666666667	22	6	22	20	1.6	19.69875	33.62563721	2.06294	13.8576	
	2	70	70	0	1.864864865	17	17	16	20	1.514285714	45.45375	36.51588631	2.21366	9.12754	
	3	70	70	0	1.555555556	19	7	18	26	1.628571429	23.214375	36.48974659	2.19048	15.2816	
	4	70	70	0	1.682926829	18	12	17	23	1.571428571	18.378125	35.87657853	2.24928	9.83271	
	5	70	70	0	1.815789474	17	16	16	21	1.528571429	36.026875	34.89003488	2.24431	12.1839	
60	1	70	70	0	1.5	14	12	12	32	1.628571429	35.376875	38.65774509	2.06501	8.54534	
	2	70	70	0	1.5	16	9	15	30	1.642857143	28.80375	38.59152592	2.02153	12.6182	
	3	70	70	0	1.642857143	15	14	14	27	1.585714286	24.465625	38.23546223	2.12919	10.7482	
	4	70	70	0	1.533333333	20	6	19	25	1.628571429	16.190625	36.20358349	2.21284	9.82236	
	5	70	70	0	1.864864865	15	19	14	22	1.514285714	44.33625	37.4109654	2.17723	8.08903	
70	1	70	70	0	1.5	15	10	14	31	1.642857143	27.004375	39.42762056	2.25259	14.636	
	2	70	70	0	1.642857143	19	10	18	23	1.585714286		35.03738913	2.07164		
	3	70	69	1	1.416666667	15	7	14	33	1.68115942	33.4125	38.45783928	2.2617	11.0443	
	4	70	70	0	1.52173913	12	12	12	34	1.657142857	26.89875	41.49604822	2.02816	9.09358	
	5	70	70	0	1.5	18	7	17	28	1.642857143		35.96201591	1.95321		
80	1	70	70	0	1.416666667	13	11	11	35	1.657142857	40.335625	41.21886435	2.13375	13.1408	
	2	70	70	0	1.38	15	6	14	35	1.7	39.103125	40.08895771	2.13209	17.1776	
	3	70	70	0	1.604651163	15	13	14	28	1.6		37.82014016	2.28323		
	4	70	70	0	1.446808511	16	9	14	31	1.642857143		38.83669833	2.17308		
	5	70	70	0	1.545454545	15	13	13	29	1.6		39.29631448	2.17888		
90	1	70	69	1	1.446808511	11	12	10	36	1.666666667		38.99967436	2.15445		
	2	70	70	0	1.446808511	12	13	10	35	1.642857143		39.67628595	2.1441		
	3	70	70	0	1.395833333	16	9	13	32	1.642857143	41.37375	38.07643983	2.2824	7.04928	
	4	70	70	0	1.604651163	12	16	11	31	1.6		39.46153962	2.09275		
	5	70	70	0	1.425531915	13	13	10	34	1.628571429		38.19048317	2.32133		

MP_PROB_M_ON	RUN	NODES_MON	CONN_M_ON	UCON_M_ON	avg_slv_pi_M_ON	mstrs_M_ON	slvs_M_ON	ss_M_ON	ms_M_ON	avg_role_M_ON	scat_tim_M_ON	bw_util_M_ON	msp_visb_M_ON	msp_scat_M_ON
50	1	30	29	1	2.153846154	7	10	6	6	1.413793103		36.71249059	2.17011	
	2	30	30	0	1.8125	9	6	8	7	1.5	19.96875	34.67351401	2.27586	6.49425
	3	30	30	0	2.230769231	8	10	7	5	1.4	12.965	32.57942215	2.4092	6.61839
	4	30	30	0	2.727272727	10	9	10	1	1.366666667	13.906875	28.65788709	2.03218	5.37701
	5	30	29	1	2.153846154	5	12	4	8	1.413793103		39.53935507	2.05665	
60	1	30	30	0	1.933333333	9	7	8	6	1.466666667	6.60375	31.32501179	2.02069	6.90345
	2	30	30	0	2.071428571	7	10	6	7	1.433333333	12.154375	31.7793904	2.28736	7.3977
	3	30	30	0	2.416666667	9	10	8	3	1.366666667	16.04125	26.54055686	2.29885	6.54713
	4	30	30	0	2	9	6	9	6	1.5	17.2425	32.21520789	2.25747	6.70115
	5	30	30	0	2.066666667	10	4	11	5	1.533333333	17.110625	30.11247444	2.15632	5.54943
70	1	30	30	0	2.636363636	9	10	9	2	1.366666667	15.5475	27.845788	1.84138	5.34943
	2	30	30	0	2.230769231	9	9	8	4	1.4	7.4675	27.26455545	1.98161	6.77241
	3	30	30	0	2.9	8	13	7	2	1.3	9.895	26.07795216	2.09885	5.38851
	4	30	30	0	2.636363636	11	9	10	0	1.333333333	21.21125	23.03716547	2.07126	6.67356
	5	30	30	0	2.307692308	9	9	8	4	1.4	27.7975	28.92457899	2.29885	4.63908
80	1	30	30	0	2.9	8	13	7	2	1.3	10.649375	27.14977083	2.26207	5.11494
	2	30	30	0	2.9	9	12	8	1	1.3	9.73875	24.14462756	2.2046	5.85057
	3	30	30	0	2.9	9	12	8	1	1.3	15.828125	24.13709269	2.22989	5.4069
	4	30	30	0	2.230769231	11	7	10	2	1.4	32.965	26.36716605	2.57011	6.53333
	5	30	30	0	1.933333333	12	4	11	3	1.466666667	17.598125	25.77871257	1.9931	7.01609
90	1	30	30	0	2.727272727	9	10	9	2	1.366666667	10.025625	24.20671592	2.12874	6.0023
	2	30	30	0	2.071428571	10	7	9	4	1.433333333	46.61625	26.17584366	2.17011	6.1908
	3	30	30	0	2.384615385	11	5	12	2	1.466666667	12.614375	22.49594304	1.84598	5.09425
	4	30	30	0	2.636363636	9	11	8	2	1.333333333	13.4075	23.34931194	2.36782	6.12184
	5	30	29	1	2.071428571	11	5	10	3	1.448275862	17.385	24.43134009	2.21609	7.25123
95	1	30	30	0	2.5	12	6	12	0	1.4	8.951875	19.73713043	2.05977	5.77931
	2	30	30	0	2.9	9	12	8	1	1.3	10.615	22.23638101	1.84828	6.18161
	3	30	30	0	2.416666667	11	7	11	1	1.4	8.43375	21.89752367	2.06897	5.66897
	4	30	30	0	2.416666667	10	9	9	2	1.366666667	45.19625	22.59616329	2.04138	5.34023
	5	30	30	0	2.5	12	6	12	0	1.4	8.305	19.78646675	2.04828	5.97241
50	1	50	50	0	2.227272727	15	14	14	7	1.42	15.676875	31.36650886	2.31673	8.16816
	2	50	49	1	2.55	12	16	13	8	1.428571429	19.84	32.72186635	2.23184	7.19898
	3	50	50	0	2.45	14	17	13	6	1.38	10.931875	30.4793338	2.02286	8.90286
	4	50	50	0	2.333333333	12	18	11	9	1.4	23.886875	36.12528958	2.29633	7.84327
	5	50	50	0	2.041666667	13	14	12	11	1.46		36.95321304	2.32163	
60	1	50	50	0	2.380952381	13	16	13	8	1.42	14.095625	34.04150368	1.96	6.88163
	2	50	50	0	2.380952381	13	16	13	8	1.42	26.2525	33.51272173	2.29714	7.40571
	3	50	50	0	2.380952381	15	14	15	6	1.42	16.16875	29.76666182	2.01878	7.75265
	4	50	50	0	2.722222222	14	19	13	4	1.34	9.42125	30.46383899	2.17796	8.3151
	5	50	50	0	2.227272727	11	17	11	11	1.44	9.26	33.41224792	2.07755	8.6049
70	1	50	50	0	2.227272727	16	12	16	6	1.44	13.686875	26.5975792	2.13878	9.86286
	2	50	50	0	2.130434783	16	12	15	7	1.44	12.11	30.05029897	2.38694	7.69143
	3	50	50	0	2.227272727	14	15	13	8	1.42	23.713125	31.77737861	2.31755	7.83592
	4	50	50	0	2.130434783	15	14	13	8	1.42	15.381875	27.50554571	2.2702	5.70694
	5	50	50	0	2.631578947	17	14	17	2	1.38	8.755	25.04166784	2.32	8.61796
80	1	50	50	0	2.578947368	14	18	13	5	1.36	12.069375	28.01001043	2.49469	7.78857
	2	50	49	1	2.722222222	10	22	9	8	1.346938776	12.934375	31.40500527	2.07837	7.40816
	3	50	50	0	2.227272727	18	11	17	4	1.42	13.365625	24.30010898	2.26367	8.81061
	4	50	50	0	2.45	18	13	17	2	1.38	11.33125	25.78855592	2.2351	10.1314
	5	50	50	0	2.476190476	12	15	14	9	1.46	16.070625	30.41066611	1.96408	7.28571
90	1	50	50	0	2.333333333	15	15	14	6	1.4	22.824375	26.06035725	2.24653	7.75184
	2	50	50	0	2.777777778	11	21	11	7	1.36	13.151875	34.33352073	2.23429	6.14857
	3	50	50	0	2.318181818	18	9	19	4	1.46	10.736875	24.09254097	2.45143	9.13224
	4	50	50	0	2.130434783	17	12	15	6	1.42	19.30125	26.16265419	2.38857	5.01143
	5	50	50	0	2.722222222	13	19	13	5	1.36	13.40375	29.14536562	2.1902	8.28653
95	1	50	50	0	2.333333333	16	14	15	5	1.4	17.51125	24.52936318	2.44	7.45959
	2	50	50	0	2.130434783	17	11	16	6	1.44	19.471875	25.09897159	2.22041	8.09959
	3	50	50	0	2.227272727	17	12	16	5	1.42	17.908125	23.85223308	2.32245	7.62367
	4	50	49	1	2.5	17	12	17	3	1.408163265	16.36125	26.83387979	2.13143	7.63946
	5	50	50	0	2.5	17	13	17	3	1.4	9.11875	23.14924041	2.21878	7.66531
50	1	70	70	0	2.1875	17	21	17	15	1.457142857	11.7175	34.30916663	2.22899	9.00994
	2	70	70	0	2.029411765	17	17	19	17	1.514285714	10.91625	32.96835372	2.08406	12.5089
	3	70	70	0	2.3	21	20	20	9	1.414285714	21.579375	32.39398174	2.3412	10.906
	4	70	69	1	2.1875	19	18	19	13	1.463768116	15.705625	34.31369482	2.12671	9.20929
	5	70	70	0	2.379310345	17	25	16	12	1.4	12.96125	35.30187659	2.20952	8.34907
60	1	70	70	0	2.225806452	20	20	19	11	1.428571429	22.016875	33.03997885	2.18675	9.16108
	2	70	70	0	2.464285714	19	23	19	9	1.4	10.738125	28.34622396	2.17184	8.97764
	3	70	70	0	2.029411765	20	17	19	14	1.471428571	10.5775	31.50042977	2.10228	11.523
	4	70	70	0	2.333333333	20	20	20	10	1.428571429	16.02	31.1975058	2.37226	9.69234
	5	70	70	0	2.15625	19	19	19	13	1.457142857	8.335	31.8937328	2.15859	10.7524
70	1	70	70	0	2.448275862	23	17	24	6	1.428571429	13.046875	27.91348169	2.15942	10.6708
	2	70	70	0	2.413793103	21	20	21	8	1.414285714	15.583125	29.16312934	2.01201	8.81449
	3	70	70	0	2.333333333	20	20	20	10	1.428571429	13.899375	29.33880883	2.07288	10.2224
	4	70	70	0	2.3	20	21	19	10	1.414285714	9.62875	28.19371754	2.22319	13.265
	5	70	70	0	2.5	23	19	23	5	1.4	21.63125	27.65837145	2.07039	8.76025
80	1	70	69	1	2.692307692	20	22	21	6	1.391304348	9.58	28.01076979	2.21656	9.78602
	2	70	70	0	2.958333333	17	28	18	7	1.357142857	12.45	27.4345922	2.18634	8.08571
	3	70	70	0	2.413793103	23	18	23	6	1.414285714	15.894375	25.92877034	2.15776	11.0298
	4	70	70	0	2.730769231	22	20	24	4	1.4	20.213125	25.11238742	2.24928	7.89938
	5	70	70	0	2.692307692	20	23	21	6	1.385714286	11.2975	28.23445803	2.21408	8.50269
90	1	70	70	0	2.258064516	24	15	24	7	1.442857143	23.139375	25.44877265	2.05424	12.4836
	2	70	70	0	2.464285714	21	22	20	7	1.385714286	11.829375	23.88195991	2.18054	3.76812
	3	70	70	0	2.5	20	22	20	8	1.4	11.2525	24.89413412	2.06087	8.53375
	4	70	70	0	2.555555556	21	23	20	6	1.371428571	27.0975	26.26339999	1.95155	10.1217
	5	70	70	0	2.535714286	23	18	24	5	1.414285714	26.104375	27.36623638	2.12133	8.68489
95	1	70	70	0	2.84	23	20	25	2	1.385714286	11.64875	21.53518231	2.12257	8.41573
	2	70	70	0	2.1875	25	13	25	7	1.457142857	12.86625	25.54289563	2.26667	9.55114
	3	70	70	0	2.592592593	22	21	22	5	1.385714286	15.014375	23.85583333	2.19586	8.7913
	4	70	70	0	2.413793103	22	18	23	7	1.428571429	12.238125	24.34719756	2.12961	10.9184
	5	70	70	0	2.3	24	17	23	6	1.414285714	24.129375	24.93746023	2.15238	9.73043

UP_PROB_M_ON	RUN	NODES_M_ON	CONN_M_ON	UCON_M_ON	avg_slv_pi_M_ON	mstrs_M_ON	slvs_M_ON	ss_M_ON	ms_M_ON	avg_role_M_ON	scat_tim_M_ON	bw_util_M_ON	msp_visb_M_ON	msp_scat_M_ON
10	1	30	30	0	2.416666667	8	11	7	4	1.366666667	24.428125	26.17514502	2.15172	6.18851
	2	30	30	0	2.307692308	11	6	11	2	1.433333333	10.714375	22.92411889	1.88046	6.53563
	3	30	30	0	2.230769231	12	6	11	1	1.4	10.946875	20.73342429	2.53563	6.63218
	4	30	30	0	2.416666667	11	8	10	1	1.366666667		23.82332129	2.3931	
	5	30	30	0	2.727272727	9	10	9	2	1.366666667	15.263125	31.88854276	2.10805	5.70575
20	1	30	30	0	2.416666667	11	8	10	1	1.366666667	6.85125	22.65619392	2.08966	6.92874
	2	30	30	0	2.230769231	10	8	9	3	1.4	15.96625	25.39110042	1.91954	7.26897
	3	30	30	0	2.230769231	12	6	11	1	1.4	12.411875	21.82479423	2.2069	7.91034
	4	30	30	0	2.5	10	8	10	2	1.4	12.914375	23.83197088	2.10805	5.73103
	5	30	30	0	2.230769231	12	6	11	1	1.4	8.995	23.10188781	2.5908	6.3931
30	1	30	29	1	2.142857143	11	4	11	3	1.482758621	24.0225	23.66002849	2.58161	6.57143
	2	30	30	0	1.933333333	13	3	12	2	1.466666667	8.7375	23.71074618	2.18621	7.52414
	3	30	30	0	2.071428571	14	3	13	0	1.433333333	12.478125	20.11526353	2.11724	6.97011
	4	30	30	0	2.071428571	12	5	11	2	1.433333333	12.460625	24.14780216	2.38851	6.35862
	5	30	30	0	2.142857143	12	4	12	2	1.466666667	23.13875	23.22237475	2.04828	7.4069
40	1	30	29	1	1.6875	14	1	12	2	1.482758621		23.86616026	2.22906	
	2	30	30	0	2.071428571	13	4	12	1	1.433333333	13.744375	22.29887876	2.16552	6.11724
	3	30	30	0	2.071428571	11	6	10	3	1.433333333	27.97125	24.82343924	2.4	6.71954
	4	30	29	1	1.866666667	14	2	12	1	1.448275862		22.38073847	2	
	5	30	30	0	2.230769231	13	5	12	0	1.4	29.74	21.31251381	2.09195	6.55862
50	1	30	30	0	1.933333333	13	3	12	2	1.466666667	14.54375	22.76787881	2.12874	6.65747
	2	30	30	0	1.933333333	13	3	12	2	1.466666667	21.07875	21.78038837	2.1931	7.22989
	3	30	30	0	2	12	2	13	3	1.533333333	12.213125	23.9010789	1.90805	6.06667
	4	30	30	0	2.230769231	12	6	11	1	1.4	20.09125	21.83612426	2.4	7.56092
	5	30	30	0	1.933333333	13	3	12	2	1.466666667	12.005	24.0681192	2.18851	8.1954
60	1	30	30	0	2.142857143	12	5	11	2	1.433333333	31.104375	23.89354513	2.16552	4.1977
	2	30	30	0	2.071428571	13	4	12	1	1.433333333		22.73537855	2.13103	
	3	30	29	1	1.933333333	13	1	13	2	1.517241379		23.43961894	2.31494	
	4	30	30	0	2	11	4	11	4	1.5	13.4	25.90399543	2.03218	7.05977
	5	30	30	0	2.142857143	11	5	11	3	1.466666667	22.666875	24.01296016	2.13103	6.03218
70	1	30	29	1	1.866666667	12	4	10	3	1.448275862		25.13685813	2.58391	
	2	30	30	0	1.933333333	14	2	13	1	1.466666667	11.934375	23.23169279	2.4046	7.2092
	3	30	30	0	1.933333333	14	2	13	1	1.466666667	24.225625	21.95600551	2.2	9.45057
	4	30	30	0	1.933333333	13	3	12	2	1.466666667	24.980625	24.53350152	2.50345	6.6092
	5	30	30	0	1.933333333	12	4	11	3	1.466666667	20.42375	23.00937909	2.25747	7.77241
10	1	50	50	0	2.333333333	15	15	14	6	1.4	9.044375	26.75422308	2.33143	9.70612
	2	50	50	0	2.333333333	17	13	16	4	1.4	18.239375	25.47140819	2.32	9.24163
	3	50	50	0	2.833333333	16	15	17	2	1.38	20.88	23.17024564	2.04816	6.80082
	4	50	50	0	2.380952381	17	12	17	4	1.42	14.854375	24.46520912	2.21061	7.61551
	5	50	50	0	2.631578947	15	16	15	4	1.38	13.856875	25.4706725	2.09878	8.05714
20	1	50	50	0	2.041666667	18	9	17	6	1.46		27.00027504	2.34857	
	2	50	50	0	1.96	17	9	16	8	1.48	34.540625	28.36035152	2.34776	11.8098
	3	50	50	0	1.96	19	7	18	6	1.48	43.675	25.9408352	2.46286	9.80327
	4	50	49	1	2.227272727	16	12	15	6	1.428571429	14.91375	25.13802121	2.15347	10.6616
	5	50	50	0	2.130434783	20	8	19	3	1.44	17.09375	23.34094031	2.06612	9.34286
30	1	50	50	0	2.428571429	18	10	19	3	1.44	20.635625	23.89197424	2.43184	8.81796
	2	50	50	0	2.227272727	19	10	18	3	1.42	22.42	24.74516317	2.50367	9.81061
	3	50	50	0	2.041666667	22	5	21	2	1.46	22.570625	22.36977568	2.16	10.2784
	4	50	50	0	2.5	19	11	19	1	1.4	25.969375	21.39177363	2.03265	8.43673
	5	50	49	1	2.45	18	12	17	2	1.387755102	15.325625	22.47683305	2.19918	9.04592
40	1	50	50	0	2.333333333	20	10	19	1	1.4	16.51375	20.97820256	2.25959	9.60408
	2	50	50	0	2.227272727	18	11	17	4	1.42	18.05625	24.64574192	2.37061	8.8898
	3	50	50	0	2.130434783	19	9	18	4	1.44	21.47125	23.96360214	2.06612	10.1967
	4	50	50	0	2.041666667	19	8	18	5	1.46	17.351875	23.2410323	2.10204	8.59592
	5	50	50	0	2.5	17	13	17	3	1.4	17.90375	23.10852448	2.17061	9.93306
50	1	50	50	0	1.884615385	22	3	21	4	1.5	24.45875	23.88552192	2.03837	9.37306
	2	50	50	0	2	22	4	21	3	1.48	27.515625	22.51908866	2.01224	7.60327
	3	50	50	0	2.041666667	20	7	19	4	1.46	21.79	24.00369845	2.29796	10.36
	4	50	50	0	2.083333333	22	4	22	2	1.48	26.87375	22.32938621	2.20735	8.02286
	5	50	50	0	1.96	23	3	22	2	1.48	24.2375	22.01982279	2.12408	9.26204
60	1	50	50	0	2.130434783	20	8	19	3	1.44	18.424375	22.41490267	2.05224	8.35265
	2	50	50	0	1.884615385	21	5	19	5	1.48	24.4325	23.84543637	2.07918	4.87755
	3	50	50	0	1.814814815	20	4	19	7	1.52	49.591875	25.76224017	2.15102	9.91837
	4	50	50	0	2.041666667	21	6	20	3	1.46	14.0325	23.9586893	2.28571	8.89469
	5	50	49	1	1.884615385	21	3	20	5	1.510204082	19.884375	25.15901018	2.14939	9.27211
70	1	50	50	0	2.041666667	22	5	21	2	1.46	21.750625	22.22229695	2.26612	7.72735
	2	50	50	0	1.884615385	22	3	21	4	1.5	19.57125	23.51115729	2.03673	9.13143
	3	50	50	0	1.75	20	3	19	8	1.54	37.3275	27.69245927	2.23102	9.40653
	4	50	50	0	1.884615385	20	5	19	6	1.5	29.18625	26.12260556	2.29959	10.0735
	5	50	50	0	1.814814815	21	3	20	6	1.52	46.166875	24.57706622	2.1249	8.96653
10	1	70	70	0	2.5	23	19	23	5	1.4	15.785	24.10098487	2.23313	8.68199
	2	70	70	0	2.76	24	22	23	1	1.342857143	11.556875	20.47169309	2.20828	10.9172
	3	70	70	0	2.76	21	25	20	4	1.342857143	11.489375	24.57703781	2.14824	10.8277
	4	70	70	0	2.35483871	21	15	24	10	1.485714286	14.218125	25.8186001	2.24886	10.1843
	5	70	70	0	2.5	24	18	24	4	1.4	28.7075	24.04602033	2.15114	9.65921
20	1	70	70	0	2.290322581	27	12	27	4	1.442857143	12.818125	23.22026035	2.03934	10.4697
	2	70	70	0	2.4	25	13	27	5	1.457142857	35.12875	24.20482033	2.0559	10.2799
	3	70	70	0	2.3	25	16	24	5	1.414285714	16.184375	22.10275867	2.14824	11.5843
	4	70	70	0	2.121212121	25	12	25	8	1.471428571	17.900625	25.27976179	2.17805	10.3731
	5	70	70	0	2.15625	26	13	25	6	1.442857143	27.15375	25.22560559	2.31553	11.9992
30	1	70	70	0	2.121212121	28	9	28	5	1.471428571	19.005	23.53466517	2.13954	9.54824
	2	70	69	1	1.971428571	31	4	30	4	1.492753623	36.703125	23.20371644	2.05549	10.7084
	3	70	70	0	2.379310345	28	14	27	1	1.4	18.543125	20.47934317	2.15362	10.7946
	4	70	70	0	2.151515152	30	5	32	3	1.5	15.108125	22.35999041	2.13085	12.583
	5	70	70	0	2.1875	29	9	29	3	1.457142857	18.523125	23.02733452	2.21615	11.3723
40	1	70	70	0	2.029411765	28	9	27	6	1.471428571	17.995625	23.42589398	2.50807	10.5967
	2	70	70	0	2.029411765	29	8	28	5	1.471428571	16.52	22.58503563	2.18965	11.9151
	3	70	70	0	2.090909091	28	11	26	5	1.442857143	28.69625	23.81096682	2.2294	8.95362
	4	70	70	0	2.15625	27	12	26	5	1.442857143	18.693125	23.69377864	2.30849	10.6534
	5	70	70	0	1.971428571	30	6	29	5	1.485714286	26.1525	22.85505767	2.06832	13.1139
50	1	70	70	0	1.944444444	29	5	29	7	1.514285714	45.8	24.81423072	2.18758	10.9325
	2	70	70	0	2.029411765	28	8	28	6	1.485714286	19.215	22.52651943	2.32464	13.5851
	3	70	70	0	1.971428571	29	7	28	6	1.485714286	26.096875	22.84657327	2.2323	12.2029
	4	70	69	1	2.1875	30	8	29	2	1.449275362	41.4575	21.5766379	2.06335	10.15
	5	70	70	0	1.971428571	30	5	30	5	1.5	26.4675	22.7220275	2.19337	13.6174
60	1	70	70	0	1.864864865	31	4	29	6	1.5	26.779375	23.50900551	2.10352	10.6994
	2	70	70	0	2.121212121	26	11	26	7	1.471428571	37.595	23.78168043	2.18261	10.1992
	3	70	70	0	1.971428571	27	9	26	8	1.485714286	41.135625	25.47072336	2.18841	11.9462
	4	70	69	1	2.058823529	28	7	28	6	1.492753623	36.01125	23.98803769	2.10104	11.8065
	5	70	70	0	2.090909091	29	9	28	4	1.457142857	20.935	22.67837171	2.01946	12.1404
70	1	70	69	1	1.916666667	30	4	29	6	1.507246377	36.52	22.35446874	2.06501	9.59761
	2	70	70	0	1.864864865	30	5	28	7	1.5	42.751875	23.97148565	2.10807	3.75776
	3	70	70	0	1.769230769	26	7	24	13	1.528571429	27.99	26.85404432	2.12961	9.35031
	4	70	69	1	2.029411765	30	6	29	4	1.47826087	33.140625	22.54109208	2.39089	14.2967
	5	70	70	0	1.916666667	29	6	28	7	1.5	29.564375	23.658148	2.05135	12.0654

SP_PROB_M_ON	RUN	NODES_M_ON	CONN_M_ON	UCON_M_ON	avg_slv_pl_M_ON	mstrs_M_ON	slvs_M_ON	ss_M_ON	ms_M_ON	avg_role_M_ON	scat_tim_M_ON	bw_util_M_ON	msp_visb_M_ON	msp_scat_M_ON	
30	1	30	30	0	2.416666667	10	9	9	2	1.366666667	15.206875	23.97588642	2.03908		6.92184
	2	30	30	0	2.071428571	10	7	9	4	1.433333333	18.54125	25.36373356	2.14483		6.09885
	3	30	30	0	2.307692308	9	8	9	4	1.433333333	21.056875	27.55906733	2.44368		6.22759
	4	30	30	0	2.727272727	10	9	10	1	1.366666667	31.261875	24.16528491	2.98851		5.09885
	5	30	30	0	2.636363636	10	10	9	1	1.333333333	22.90375	21.15673198	2.12414		6.14943
40	1	30	30	0	2.142857143	7	9	7	7	1.466666667	15.533125	32.85631937	2.12874		7.24828
	2	30	30	0	2.153846154	10	9	8	3	1.366666667		30.81260361	1.92874		
	3	30	30	0	2.636363636	7	13	6	4	1.333333333	20.07875	35.51733312	2.14483		6.24368
	4	30	30	0	1.933333333	7	9	6	8	1.466666667	13.283125	30.9311022	2.03908		7.2023
	5	30	29	1	2.076923077	12	6	10	1	1.379310345		22.76991061	2.11576		
50	1	30	30	0	2.636363636	8	12	7	3	1.333333333	24.26375	25.69206915	2.29195		6.36322
	2	30	30	0	2.071428571	9	8	8	5	1.433333333	28.609375	26.71804291	2.81839		6.08966
	3	30	30	0	3.222222222	9	13	8	0	1.266666667	41.971875	20.07250557	2.35632		5.66667
	4	30	30	0	2	10	8	8	4	1.4		27.31138352	1.98621		
	5	30	30	0	2.333333333	10	10	8	2	1.333333333		25.20062408	2.09655		
60	1	30	30	0	2.071428571	8	8	8	6	1.466666667		31.21566141	2.4023		
	2	30	30	0	2.071428571	9	8	8	5	1.433333333	18.63125	26.64563536	2.33793		5.90115
	3	30	30	0	2.416666667	11	8	10	1	1.366666667	35.553125	23.34375582	2.42989		7.47356
	4	30	30	0	3.222222222	9	13	8	0	1.266666667	28.005625	20.55614328	2.03908		6.0069
	5	30	30	0	2.333333333	8	12	6	4	1.333333333		28.39155961	2.47816		
70	1	30	30	0	1.705882353	9	5	8	8	1.533333333	32.95625	27.75205761	1.8092		5.7977
	2	30	30	0	2	9	9	7	5	1.4		27.40879749	2.41839		
	3	30	30	0	2.636363636	9	11	8	2	1.333333333	35.129375	24.92214804	2.18851		6.62069
	4	30	30	0	2.416666667	8	11	7	4	1.366666667	30.09375	27.14145788	2.70575		6.78621
	5	30	30	0	1.866666667	10	7	8	5	1.433333333		27.92003053	2.22529		
80	1	30	30	0	2.416666667	6	13	5	6	1.366666667	18.663125	32.45965994	1.62069		6.58391
	2	30	30	0	2	8	10	6	6	1.4		30.17881321	2.48276		
	3	30	30	0	2.071428571	7	10	6	7	1.433333333	44.9525	31.35696519	2.0023		5.88966
	4	30	30	0	1.928571429	6	13	3	8	1.366666667		32.10318243	2.6		
	5	30	30	0	1.8	7	11	4	8	1.4		33.12240934	2.12644		
90	1	30	30	0	1.866666667	7	10	5	8	1.433333333		32.90532841	2.13333		
	2	30	30	0	2.153846154	6	13	4	7	1.366666667		31.82798063	2.02989		
	3	30	29	1	2.166666667	7	13	4	5	1.310344828		30.24066819	2.12414		
	4	30	30	0	2.230769231	7	11	6	6	1.4	42.804375	27.7368894	2.21609		5.3954
	5	30	30	0	2	8	12	5	5	1.333333333		24.42146622	2.29885		
30	1	50	50	0	3.0625	15	20	14	1	1.3	14.668125	25.09266447	2.25714		9.42857
	2	50	50	0	2.173913043	16	12	15	7	1.44	15.19125	24.31878591	2.34286		8.39429
	3	50	50	0	2.5	15	15	15	5	1.4	8.9725	25.46035799	2.36816		7.18041
	4	50	50	0	2.5	17	13	17	3	1.4	18.963125	31.29583032	2.2098		7.51837
	5	50	50	0	2.631578947	17	14	17	2	1.38	20.5975	22.27063426	2.2702		7.62041
40	1	50	50	0	2.227272727	16	13	15	6	1.42	16.95375	28.0955062	2.32082		8.65388
	2	50	50	0	2.631578947	15	16	15	4	1.38	16.07625	24.03656231	2.14531		6.82694
	3	50	50	0	2.227272727	15	14	14	7	1.42	26.066875	26.95826968	2.28408		7.91755
	4	50	50	0	2.380952381	16	13	16	5	1.42	15.014375	25.20709555	2.09224		10.0996
	5	50	50	0	2.578947368	14	18	13	5	1.36	11.67	25.16706	1.97469		8.09633
50	1	50	50	0	2.333333333	14	16	13	7	1.4	21.575	27.25551064	2.30612		9.41224
	2	50	50	0	2.333333333	15	15	14	6	1.4	23.94875	24.29146713	2.20327		7.77796
	3	50	50	0	2.318181818	15	12	16	7	1.46	23.050625	27.40306884	2.17388		8.08082
	4	50	50	0	2.227272727	16	13	15	6	1.42	33.274375	26.22742573	2.14939		9.77061
	5	50	50	0	2.227272727	17	12	16	5	1.42	36.613125	28.90864983	2.12898		8.51184
60	1	50	50	0	2.227272727	12	17	11	10	1.42	31.380625	31.19781752	1.93878		10.2049
	2	50	50	0	2.578947368	15	17	14	4	1.36	36.1	25.03116933	2.23347		7.2751
	3	50	50	0	2.631578947	14	17	14	5	1.38	43.168125	26.23284492	2.16245		6.20816
	4	50	50	0	2.130434783	13	15	12	10	1.44	34.809375	29.1851362	2.17714		8.80163
	5	50	49	1	2.45	15	15	14	5	1.387755102	21.30875	25.77065094	2.13143		8.77041
70	1	50	49	1	2.9375	14	21	12	2	1.285714286		22.84577691	2.21224		
	2	50	50	0	2.041666667	13	14	12	11	1.46	36.38125	29.50997952	2.57551		8.76163
	3	50	50	0	2.380952381	13	16	13	8	1.42	45.275	27.67435345	2.25469		7.01306
	4	50	50	0	2.130434783	14	14	13	9	1.44	39.744375	28.8440564	2.45061		8.26449
	5	50	50	0	2.217391304	14	12	15	9	1.48	37.723125	25.96559519	2.2		6.58449
80	1	50	50	0	2.272727273	15	13	15	7	1.44	49.611875	27.86617254	2.20898		8.82857
	2	50	50	0	2.4	10	22	8	10	1.36		33.24962532	2.17306		
	3	50	50	0	2.526315789	10	23	8	9	1.34		33.84709899	2.16735		
	4	50	49	1	1.846153846	14	11	12	12	1.489795918		29.40676843	2.2449		
	5	50	50	0	1.96	11	15	10	14	1.48	38.779375	32.14846013	2.10612		9.48082
90	1	50	50	0	1.62962963	8	21	2	19	1.42		36.2274682	2.04		
	2	50	50	0	2.043478261	11	19	8	12	1.4		30.78489162	2.33469		
	3	50	50	0	1.869565217	12	21	6	11	1.34		30.26718171	2.17061		
	4	50	50	0	2.3	13	20	10	7	1.34		29.73989754	2.19184		
	5	50	50	0	1.75	11	22	4	13	1.34		32.38085378	2.04816		
30	1	70	70	0	2.5	25	17	25	3	1.4	13.54375	22.01013763	2.28282		9.78054
	2	70	70	0	2.84	20	24	21	5	1.371428571	15.508125	25.38627747	2.10849		8.64431
	3	70	70	0	2.413793103	25	16	25	4	1.414285714	18.96625	23.90093355	2.12671		10.1561
	4	70	70	0	2.3	24	17	23	6	1.414285714	11.2075	25.79913206	2.15569		11.4199
	5	70	69	1	2.653846154	23	19	24	3	1.391304348	10.3775	22.34926915	2.25797		10.5269
40	1	70	70	0	2.692307692	21	24	20	5	1.357142857	19.549375	24.21877276	2.15901		8.71014
	2	70	70	0	2.258064516	22	17	22	9	1.442857143	16.8625	25.90389979	2.22277		10.7164
	3	70	70	0	2.333333333	21	19	21	9	1.428571429	46.82375	27.24812831	2.23892		9.3793
	4	70	70	0	2.555555556	19	25	18	8	1.371428571	20.456875	29.00841506	1.96149		8.74824
	5	70	70	0	2.379310345	19	22	19	10	1.414285714	14.101875	27.5635553	2.0646		11.6269
50	1	70	70	0	2.482758621	22	19	22	7	1.414285714	24.880625	26.553865	2.06294		8.69896
	2	70	70	0	2.730769231	20	23	21	6	1.385714286	21.348125	23.77187796	2.21366		9.34617
	3	70	70	0	2.225806452	21	19	20	10	1.428571429	19.61	27.37271733	2.19048		10.5296
	4	70	70	0	2.555555556	18	26	17	9	1.371428571	21.526875	30.02645007	2.24928		9.74037
	5	70	70	0	2.653846154	21	23	21	5	1.371428571	14.410625	23.1736833	2.24431		9.62774
60	1	70	70	0	2.090909091	19	19	18	14	1.457142857	23.11	32.80748815	2.06501		10.0559
	2	70	70	0	2.258064516	22	18	21	9	1.428571429	24.94125	24.80675942	2.02153		11.67
	3	70	70	0	2.090909091	19	19	18	14	1.457142857	32.82625	31.86872322	2.12919		9.88696
	4	70	70	0	2.090909091	20	18	19	13	1.457142857	40.359375	29.52225984	2.21284		10.8356
	5	70	70	0	2.15625	21	18	20	11	1.442857143	22.541875	28.23515451	2.17723		8.82195
70	1	70	70	0	2.464285714	17	26	16	11	1.385714286	31.006875	28.78074742	2.25259		11.87
	2	70	70	0	2.555555556	20	24	19	7	1.371428571		26.11056695	2.07164		
	3	70	70	0	2.225806452	19	21	18	12	1.428571429	41.5025	29.394456	2.2617		11.0335
	4	70	70	0	2.258064516	16	23	16	15	1.442857143	48.95625	30.74524003	2.02816		10.1077
	5	70	69	1	2.193548387	21	18	20	10	1.434782609		27.830692	1.95321		
80	1	70	70	0	2.3	18	23	17	12	1.414285714	37.249375	28.60478863	2.13375		9.57847
	2	70	69	1	2.791666667	16	30	15	8	1.333333333		31.30840946	2.13209		
	3	70	70	0	2.444444444	16	29	14	11	1.357142857		33.03612343	2.28323		
	4	70	70	0	2.379310345	18	24	17	11	1.4	41.471875	31.48257906	2.17308		11.4489
	5	70	70	0	2.344827586	16	26	15	13	1.4		29.97323131	2.17888		
90	1	70	70	0	2.064516129	12	33	6	19	1.357142857		32.38231098	2.15445		
	2	70	67	3	2.066666667	14	28	9	16	1.373134328		34.2577555	2.1441		
	3	70	69	1	1.878787879	16	26	10	17	1.391304348		30.43618905	2.2824		
	4	70	70	0	2.423076923	14	35	9	12	1.3		28.70370214	2.09275		
	5	70	70	0	2.392857143	15	30	12	13	1.357142857		28.57619472	2.32133		

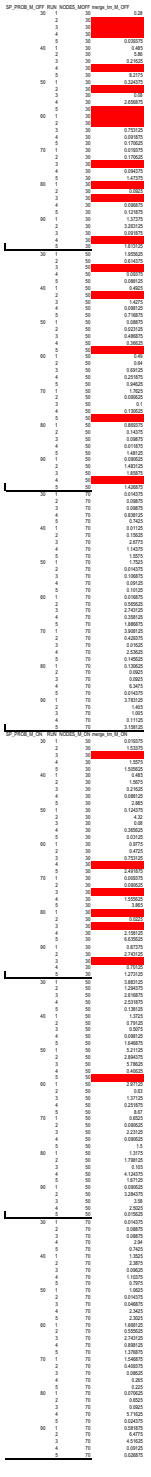
B.2 Table for Section 6.2

The following tables contains the data presented in Section 6.2 — Mscat - Impact of Parameters on Performance.

SP	MP	RUNS	NODES_MON	CONN_M_ON	UCON_M_ON	avg_slv_pi_M_ON	mstrs_M_ON	slvs_M_ON	ss_M_ON	ms_M_ON	avg_role_M_ON	scat_tim_M_ON	bw_util_M_ON	msp_visb_M_ON	msp_scat_M_ON
30	50	1	50	50	0	2.041666667	15	12	14	9	1.46	10.799375	31.76785189	2.09878	9.54367
		2	50	50	0	2.45	13	18	12	7	1.38	13.593125	31.27190596	2.41061	8.02612
		3	50	50	0	2.318181818	15	11	17	7	1.48	17.12625	32.22332693	2.03918	7.42857
		4	50	50	0	2.45	14	17	13	6	1.38	12.42375	30.08370491	2.14367	6.81551
		5	50	50	0	1.884615385	14	11	13	12	1.5	17.2075	34.67711912	2.22367	8.14776
	60	1	50	50	0	2.173913043	13	13	14	10	1.48	13.40375	35.49295983	2.06367	8.95918
		2	50	50	0	2.578947368	15	17	14	4	1.36		33.01939406	2.52653	
		3	50	50	0	2.260869565	11	14	13	12	1.5	12.87375	34.56980308	2.26367	6.91837
		4	50	50	0	2.130434783	13	15	12	10	1.44	16.423125	33.67490928	2.1902	10.2294
		5	50	49	1	1.961538462	12	9	14	14	1.571428571	17.848125	33.37409957	2.33796	7.5034
	70	1	50	50	0	2.318181818	14	13	15	8	1.46	13.39375	27.72840574	2.26694	8.40653
		2	50	50	0	2.428571429	14	14	15	7	1.44	12.754375	29.95484043	2.33469	7.69143
		3	50	50	0	2.173913043	18	10	17	5	1.44	14.496875	27.97542863	2.17143	6.98857
		4	50	49	1	2.227272727	13	15	12	9	1.428571429	16.2375	31.10527455	2.09551	8.33163
		5	50	50	0	2.45	15	15	15	5	1.4	15.829375	27.13069081	1.88	7.60816
	80	1	50	49	1	2.227272727	17	11	16	5	1.428571429	32.47125	25.17287105	2.34204	8.93197
		2	50	50	0	2.272727273	15	13	15	7	1.44	17.816875	26.70472307	2.03837	7.39837
		3	50	50	0	2.380952381	16	14	15	5	1.4	20.0225	26.18840301	2.13224	6.29959
		4	50	50	0	2.333333333	17	13	16	4	1.4	21.9125	23.7541349	2.4351	9.20653
		5	50	50	0	1.923076923	17	7	17	9	1.52	11.901875	26.62682533	2.09633	9.09796
	90	1	50	50	0	2.45	16	12	18	4	1.44	9.914375	23.24418608	2.24816	8.40327
		2	50	50	0	2.882352941	17	17	16	0	1.32	13.02875	20.610388	2.02776	8.33061
		3	50	50	0	2.333333333	18	12	17	3	1.4	13.86625	23.32316453	2.0098	8.55102
		4	50	50	0	2.380952381	18	11	18	3	1.42	22.330625	23.49418115	2.0302	8.53878
		5	50	50	0	2.173913043	15	12	15	8	1.46	24.454375	27.18243793	2.2351	8.59265
40	50	1	50	50	0	2	14	11	14	11	1.5	13.009375	32.6095108	2.09878	7.87102
		2	50	50	0	2.736842105	12	17	14	7	1.42	16.95625	32.65118726	2.41061	8.31755
		3	50	50	0	2.45	11	20	10	9	1.38	10.33125	34.9688687	2.03918	6.90531
		4	50	50	0	2.380952381	11	18	11	10	1.42	14.445	36.37878342	2.14367	6.31265
		5	50	50	0	2.304347826	12	11	16	11	1.54	11.985	33.41381109	2.22367	7.17796
	60	1	50	50	0	1.96	15	12	13	10	1.46	28.915	33.52032839	2.06367	4.65551
		2	50	50	0	2.736842105	15	14	17	4	1.42	21.13875	28.90081413	2.52653	8.56735
		3	50	50	0	2.526315789	11	22	9	8	1.34		34.48738559	2.26367	
		4	50	50	0	2.5	12	18	12	8	1.4	13.768125	29.92794642	2.1902	7.77551
		5	50	50	0	2.5	13	17	13	7	1.4	17.883125	32.82228757	2.33796	8.47184
	70	1	50	50	0	2.5	14	16	14	6	1.4	13.18	26.86951086	2.26694	8.34857
		2	50	50	0	2.45	14	17	13	6	1.38	16.845625	32.72483176	2.33469	9.47755
		3	50	50	0	2.578947368	15	17	14	4	1.36	14.04125	26.21846753	2.17143	8.86939
		4	50	50	0	2.125	16	9	17	8	1.5	16.735	27.22591415	2.09551	7.55265
		5	50	50	0	2.333333333	15	15	14	6	1.4	18.41375	27.73655905	1.88	9.2098
	80	1	50	50	0	2.130434783	17	11	16	6	1.44	13.291875	26.40921975	2.34204	9.07837
		2	50	50	0	2.684210526	16	14	17	3	1.4	11.22875	26.66195502	2.03837	7.1551
		3	50	50	0	2.631578947	14	17	14	5	1.38	13.75625	24.46720719	2.13224	7.77143
		4	50	50	0	2.333333333	16	14	15	5	1.4	25.09625	25.28239701	2.4351	8.91673
		5	50	50	0	2.578947368	14	18	13	5	1.36	17.4875	26.32422525	2.09633	8.78531
	90	1	50	50	0	2.45	13	18	12	7	1.38	14.244375	27.68275577	2.24816	9.02612
		2	50	50	0	2.333333333	15	15	14	6	1.4	14.70375	25.75492306	2.02776	8.57388
		3	50	49	1	2	16	12	13	8	1.428571429		29.08423694	2.0098	
		4	50	50	0	2.041666667	16	11	15	8	1.46	27.65	28.36188577	2.0302	7.38939
		5	50	50	0	2.227272727	14	15	13	8	1.42	22.123125	28.42779176	2.2351	9.7551
50	50	1	50	50	0	2.041666667	11	16	10	13	1.46	19.955625	36.49647574	2.09878	8.31347
		2	50	50	0	2.333333333	9	21	8	12	1.4	28.75	38.48256597	2.41061	8.57959
		3	50	50	0	2.882352941	12	22	11	5	1.32	22.6325	30.57149573	2.03918	9.03184
		4	50	50	0	2.227272727	11	18	10	11	1.42	12.195	38.20202105	2.14367	8.38449
		5	50	50	0	2.041666667	13	14	12	11	1.46	14.901875	31.17869228	2.22367	10.3012
	60	1	50	50	0	2.380952381	13	16	13	8	1.42	19.891875	30.85359628	2.06367	8.67347
		2	50	50	0	2	14	11	14	11	1.5		35.96267001	2.52653	
		3	50	50	0	2.130434783	10	18	9	13	1.44	29.3025	35.28450011	2.26367	7.33306
		4	50	50	0	1.96	11	15	10	14	1.48	15.40875	35.236126	2.1902	11.4506
		5	50	50	0	2	16	12	14	8	1.44		32.83368582	2.33796	
	70	1	50	50	0	2.631578947	14	17	14	5	1.38	10.628125	28.1691935	2.26694	7.74204
		2	50	50	0	1.96	17	9	16	8	1.48	29.151875	30.85758576	2.33469	7.78776
		3	50	50	0	2.45	11	20	10	9	1.38	16.019375	29.5478849	2.17143	9.22041
		4	50	50	0	2.333333333	8	22	7	13	1.4	13.4725	33.55439553	2.09551	7.40653
		5	50	50	0	2.333333333	11	19	10	10	1.4	16.094375	30.70961579	1.88	8.59102
	80	1	50	49	1	2.227272727	15	13	14	7	1.428571429	17.57625	26.26382696	2.34204	8.13776
		2	50	50	0	2.227272727	15	14	14	7	1.42	40.25625	28.80301104	2.03837	8.27673
		3	50	50	0	2.428571429	14	15	14	7	1.42	20.87125	28.24985537	2.13224	8.79265
		4	50	50	0	2.578947368	16	16	15	3	1.36	18.424375	24.21006676	2.4351	8.58204
		5	50	50	0	2.380952381	15	14	15	6	1.42	11.1575	24.73105727	2.09633	9.70122
	90	1	50	50	0	2.272727273	14	14	14	8	1.44	12.6925	25.923948	2.24816	10.7894
		2	50	49	1	2.041666667	14	11	14	10	1.489795918		30.93181245	2.02776	
		3	50	50	0	2.333333333	14	16	13	7	1.4	28.381875	28.28138796	2.0098	9.55592
		4	50	50	0	2.227272727	19	10	18	3	1.42	18.0725	23.50537445	2.0302	11.2212
		5	50	50	0	2.631578947	15	16	15	4	1.38	24.975625	25.38588363	2.2351	6.76245
60	50	1	50	50	0	2.333333333	9	21	8	12	1.4	21.144375	39.95563877	2.09878	7.97388
		2	50	50	0	2.227272727	9	20	8	13	1.42	19.60875	43.68932557	2.41061	7.63102
		3	50	50	0	2.227272727	9	20	8	13	1.42	20.10125	36.04896002	2.03918	7.84082
		4	50	50	0	1.923076923	9	15	9	17	1.52	22.035	38.28444497	2.14367	9.06204
		5	50	50	0	2.272727273	7	20	8	15	1.46	13.008125	36.63976294	2.22367	6.85796
	60	1	50	50	0	2.333333333	10	20	9	11	1.4	24.748125	34.13783281	2.06367	9.22204
		2	50	50	0	1.655172414	9	14	7	20	1.54		39.29567004	2.52653	
		3	50	50	0	2.333333333	11	19	10	10	1.4	25.82125	34.09479777	2.26367	7.79755
		4	50	49	1	1.689655172	10	11	9	19	1.571428571	17.621875	35.75654101	2.1902	7.67517
		5	50	50	0	2.041666667	13	14	12	11	1.46	21.726875	31.10828207	2.33796	7.72735
	70	1	50	50	0	2.45	13	17	13	7	1.4	23.540625	27.96782646	2.26694	7.49469
		2	50	50	0	2.227272727	12	17	11	10	1.42	24.754375	31.49731153	2.33469	9.46449
		3	50	50	0	2.130434783	12	16	11	11	1.44	17.67125	29.33171791	2.17143	7.79755
		4	50	50	0	2.130434783	11	17	10	12	1.44	24.971875	33.43246335	2.09551	7.97714
		5	50	50	0	2.333333333	12	18	11	9	1.4	33.985	32.2672095	1.88	8.98776
	80	1	50	50	0	2.083333333	11	15	11	13	1.48	37.916875	31.98128313	2.34204	7.18449
		2	50	50	0	2.173913043	13	13	14	10	1.48	27.319375	31.43941967	2.03837	7.15429
		3	50	50	0	2.130434783	13	15	12	10	1.44	16.561875	29.13384872	2.13224	7.94612
		4	50	50	0	2.578947368	13	19	12	6	1.36	20.098125	27.36054115	2.4351	8.04
		5	50	50	0	1.884615385	14	11	13	12	1.5	28.6775	31.90563023	2.09633	10.4033
	90	1	50	50	0	2.272727273	15	13	15	7	1.44	49.109375	26.31860426	2.24816	7.99837
		2	50	50	0	2.130434783	12	16	11	11	1.44	35.60125	31.69224965	2.02776	7.95265
		3	50	50	0	2.227272727	14	15	13	8	1.42	34.126875	30.89181207	2.0098	7.30204
		4	50	49	1	2.882352941	11	22	10	6	1.326530612	26.609375	26.94383817	2.0302	8.62415
		5	50	50	0	2.45	14	17	13	6	1.38	29.024375	26.85834545	2.2351	8.37959
70	50	1	50	50	0	1.92	10	17	8	15	1.46		37.68449414	2.09878	
		2	50	50	0	2.130434783	6	22	5	17	1.44	23.801875	41.68823708	2.41061	8.27265
		3	50	50	0	2.578947368	8	24	7	11	1.36	34.86875	35.90567607	2.03918	7.13714
		4	50	50	0	1.923076923	12	12	12	14	1.52	30.50875	34.02960303	2.14367	8.91265
		5	50	50	0	1.851851852	7	16	7	20	1.54	39.133125	40.73572723	2.22367	10.3935
	60	1	50	50	0	2.227272727	10	19	9	12	1.42	24.353125	32.71255688	2.06367	6.7902
		2	50	50	0	2	12	16	10	12	1.44		34.72078675	2.52653	
		3	50	50	0	1.846153846	10	16	8	16	1.48		40.17205533	2.26367	
		4	50	49	1	1.814814815	10	13	9	17	1.530612245	28.51625	37.10243674	2.1902	7.84354
		5	50	50	0	2.181818182	11	19	9	11	1.4		35.28877415	2.33796	
	70	1	50	50	0	2.227272727	12	17	11	10	1.42	21.994375	29.85047506	2.26694	6.61469
		2	50	49	1	2.041666667	8	18	7	16	1.469387755	23.10625	37.20062908	2.33469	9.20918
		3	50	50	0	2.130434783	12	16	11	11	1.44	26.48125	32.81429388	2.17143	7.97714
		4	50	50	0	2.45	11	20	10	9	1.38	34.978125	30.75176727	2.09551	8.25061
		5	50	50	0	2.083333333	10	16	10	14	1.48	30.020625	34.3381599	1.88	10.6506
	80	1	50	50	0	1.884615385	16	9	15	10	1.5	48.025625	28.83480909	2.34204	8.4849
		2	50	50	0	2.333333333	15	15	14	6	1.4	28.0425	25.7962558	2.03837	8.40327
		3	50	50	0	2.473684211	10	23	8	9	1.34		30.07753993	2.13224	
		4	50	50	0	1.96	14	12	13	11	1.48	39.180625	30.571067	2.4351	8.14286
		5	50	50	0	2.041666667	13	14	12	11	1.46	36.1975	29.44457121	2.09633	9.28653
	90	1	50	50	0	2.333333333	11	19	10	10	1.4	38.1675	31.78443215	2.24816	7.85714
		2	50	50	0	2.041666667	14	14	12	10	1.44	33.073125	30.74001527	2.02776	8.22776
		3	50	50	0	1.92	11	16	9	14	1.46		33.17668357	2.0098	
		4	50	49	1	1.916666667	13	14	11	11	1.448979592		32.12841108	2.0302	
		5	50	50	0	2.227272727	11	18	10	11	1.42	35.11375	31.24072318	2.2351	8.70449
80	50	1	50	50	0	1.884615385	6	19	5	20	1.5	27.276875	37.50022985	2.09878	7.19592
		2	50	50	0	2.333333333	7	23	6	14	1.4	40.440625	38.997509	2.41061	8.44408
		3	50	50	0	2.130434783	6	22	5	17	1.44	24.918125	35.80342505	2.03918	7.11102
		4	50	49	1	2.285714286	7	23	5	14	1.387755102		38.44970439	2.14367	
		5	50	50	0	1.884615385	7	18	6	19	1.5	28.536875	39.43743693	2.22367	9.88327
	60	1	50	50	0	2.130434783	8	20	7	15	1.44	34.83375	36.64105879	2.06367	6.85633
		2	50	50	0	1.92	10	17	8	15	1.46		36.46368462	2.52653	
		3	50	49	1	2.3	7	24	5	13	1.367346939		33.99727188	2.26367	
		4	50	50	0	2	6	22	4	18	1.44		39.37918009	2.1902	
		5	50	50	0	2.227272727	11	18	10	11	1.42	20.185	31.74074662	2.33796	8.38939
	70	1	50	50	0	2	8	20	6	16	1.44		37.48609695	2.26694	
		2	50	50	0	2.227272727	7	22	6	15	1.42	26.333125	35.64468001	2.33469	7.36653
		3	50	50	0	2.173913043	11	16	11	12	1.46	29.75	30.08134452	2.17143	7.59837
		4	50	50	0	2.130434783	5	23	4	18	1.44	33.643125	37.56256026	2.09551	8.64653
		5	50	49	1	2.238095238	10	19	9	11	1.408163265		35.62876141	1.88	
	80	1	50	50	0	2.19047619	13	20	9	8	1.34		29.39519857	2.34204	
		2	50	50	0	2.526315789	13	20	11	6	1.34		26.52052661	2.03837	
		3	50	49	1	2.473684211	9	23	7	10	1.346938776		31.06366284	2.13224	
		4	50	50	0	2.380952381	12	17	12	9	1.42	49.213125	28.94832474	2.4351	9.74286
		5	50	49	1	1.96	14	11	13	11	1.489795918		30.64644155	2.09633	
	90	1	50	49	1	2.4	11	19	10	9	1.387755102		32.90262575	2.24816	
		2	50	50	0	1.846153846	14	12	12	12	1.48		31.07012205	2.02776	
		3	50	50	0	2.526315789	9	24	7	10	1.34		32.56611041	2.0098	
		4	50	49	1	2.043478261	10	18	8	13	1.428571429		33.11447236	2.0302	
		5	50	49	1	2.227272727	12	16	11	10	1.428571429	46.76375	30.06894358	2.2351	8.22449
90	50	1	50	49	1	1.846153846	6	18	5	20	1.510204082		43.63516914	2.09878	
		2	50	50	0	2.45	5	26	4	15	1.38	32.420625	40.15809995	2.41061	6.96816
		3	50	50	0	1.807692308	5	22	2	21	1.46		42.15234182	2.03918	
		4	50	50	0	1.846153846	4	22	2	22	1.48		41.34501383	2.14367	
		5	50	50	0	2	5	23	3	19	1.44		40.82638225	2.22367	
	60	1	50	50	0	2.285714286	8	23	6	13	1.38		36.327016	2.06367	
		2	50	50	0	1.807692308	7	20	4	19	1.46		39.87104112	2.52653	
		3	50	50	0	2.086956522	8	21	6	15	1.42		37.40867726	2.26367	
		4	50	50	0	2.041666667	7	20	6	17	1.46	28.621875	37.28852171	2.1902	8.15592
		5	50	50	0	2.181818182	8	22	6	14	1.4		37.26728715	2.33796	
	70	1	50	50	0	2.045454545	9	24	4	13	1.34		33.36406106	2.26694	
		2	50	50	0	2.041666667	5	22	4	19	1.46	47.536875	37.9773264	2.33469	7.30694
		3	50	50	0	1.96	7	19	6	18	1.48	41.673125	38.03348979	2.17143	7.52571
		4	50	50	0	2.611111111	8	27	5	10	1.3		31.55678352	2.09551	
		5	50	50	0	2.227272727	8	21	7	14	1.42	43.596875	35.53690021	1.88	8.89469
	80	1	50	50	0	1.913043478	12	22	5	11	1.32		29.6952058	2.34204	
		2	50	50	0	2	12	21	7	10	1.34		29.50469993	2.03837	
		3	50	50	0	2.263157895	12	25	6	7	1.26		30.53189791	2.13224	
		4	50	47	3	2.5	11	21	8	7	1.319148936		26.01033486	2.4351	
		5	50	50	0	2.130434783	11	18	9	12	1.42		31.50069715	2.09633	
	90	1	50	49	1	2.090909091	9	21	6	13	1.387755102		32.22026269	2.24816	
		2	50	50	0	2.043478261	11	19	8	12	1.4		29.30043624	2.02776	
		3	50	50	0	2.368421053	11	24	7	8	1.3		30.06972275	2.0098	
		4	50	48	2	1.913043478	13	18	7	10	1.354166667		30.75080961	2.0302	
		5	50	50	0	2.238095238	11	21	8	10	1.36		30.4384103	2.2351	

B.3 Table for Section 6.3

The following tables contains the data presented in Section 6.3 — Mscat - Self-healing Analysis.



Bibliography

[Agga00] A. Aggrawal, M. Kapoor, L. Ramachandran, A. Sarkar, “Clustering Algorithms for wireless ad-hoc networks,” Proc. 4th Int’l Workshop Discrete Algorithms and Methods for Mobile Computing and Communications, (DIALM for Mobility 2000), ACM Press, 2000, pp. 54 - 63.

[Basagni03] Stefano Basagni, Raffaele Bruno, Chira Petrioli, “A Performance Comparison of Scatternet Formation Protocols for Networks of Bluetooth Devices,” Proc. of the first IEEE Intl. Conf. on Pervasive Computing and Communications (PerCom’03), IEEE 2003

[Basagni04a] Stefano Basagni, Raffaele Bruno, Gabriele Mambrini, Chira Petrioli, “Comparative Performance Evaluation of Scatternet Formation Protocols for Networks of Bluetooth Devices,” to be published in Wireless Networks 10, pp.197-213, Kluwer Academic Publishers 2004

[BTSpec03] *Bluetooth Core specification version 1.2*, Bluetooth SIG, November 03 https://www.bluetooth.org/

[Bray02] Jennifer Bray and Charles F. Sturman, *Bluetooth 1.1: Connect Without Cables*, Prentice Hall, New Jersey, 2002, pp.49.

[Chiasserini03] C.F. Chiasserini, M. Ajmone Marsan, E. Baralis and P. Garza, “Towards feasible topology formation algorithms for Bluetooth-based WPANs,” Proc. of IEEE HICSS-36, Jan. 2003, Big Island, Hawaii.

[CSIDC02] M. Merry, J. Nievelt, R. Trimceski, B. Davis, *Examination of Bluetooth Scatternet Formation for an Instant Messenger Application*, tech. report, IEEE Computer Society International Design Competition, 2002.

[Law03] C. Law, A. Mehta, K. Siu, “A New Bluetooth Scatternet Formation Protocol,” *Mobile Networks and Applications*, vol 8, issue 5, October 2003,pp. 485 - 498.

[Meritt02] Rick Merritt. “Chip makers question need to redraw 802.11 map.” Electrical Engineering Times, May 02, 2002. http:// www.siliconstrategies.com/story/OEG20020502S0062.

[Nist04] Advanced Network Technologies Division - National Institute of Standards and Technology, http://w3.antd.nist.gov/wahn_home.shtml (August 26,2004)

[Petrioli02] C. Petrioli and S. Basagni, “Degree-constrained multihop scatternet formation for Bluetooth networks,” Proc. of the IEEE Globecom 2002, pp. 222 - 226, Taipei, Taiwan, ROC, Vol. 1, November 2002.

[Petrioli03] C. Petrioli, S. Basagni and I. Chlamtac, “Configuring BlueStars: Multihop scatternet formation for Bluetooth networks,” IEEE Transactions on Computers 52(6), pp. 779-790, Special Issue on Wireless Internet (2003).

[Salo01] T. Salonidis, P. Bhagwat, L. Tassiulas, R. LaMaire, “Distributed topology construction of Bluetooth personal area networks,” Proc. 20th Ann. Joint Conf. IEEE CS and Communication Soc., (INFOCOM 2001), April 2001.

[Li02a] X. Li and I. Stojmenovic, “Partial Delaunay triangulation and degreelimited localized Bluetooth scatternet formation,” Proc. of AD-HOC NetwOrks and Wireless (ADHOC-NOW), September 2002, Fields Institute, Toronto, Canada.

[Tan02a] G. Tan, A. Miu, J. Guttag and H. Balakrishnan, “An efficient scatternet formation algorithm for dynamic environment”, Proceedings of the IASTED Communications and Computer Networks (CCN), Cambridge, MA (4ñ6 November 2002).

[Tan02b] G. Tan, *Self-organizing Bluetooth scatternets*, master’s thesis, Department Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, January 2002.

[Wang02] Z.Wang, R.J. Thomas and Z. Haas, “BlueNet - A new scatternet formation scheme,” Proc. of the 35th Hawaii Intl. Conf. on System Science (HICSS-35), January 2002, Big Island, Hawaii.

[Zaruba01] G. Zaruba, S. Basagni, I. Chlamtac, “BlueTrees - scatternet formation to enable Bluetooth-based personal area networks,”, Proc. of the IEEE Intl. Conf. on Communications, ICC 2001, Helsinki, Finland, Vol. 1 (11-14 June) pp. 273-277