Adaptive bandwidth allocation for bridge downlink operation

A thesis presented
by

Muhi Ahmed Ibne Khair

to
The Department of Computer Science
in partial fulfillment of the requirements
for the degree of
Master of Science
in the subject of

Computer Science

The University of Manitoba
Winnipeg, Manitoba
February 2008

Thesis advisor Author

Jelena Mišić **Muhi Ahmed Ibne Khair**

Adaptive bandwidth allocation for bridge downlink operation

Abstract

The Wireless Mesh Network (WMN) is an emerging technology that has many potential applications with a huge consumer demand. There is a big demand to develop or choose among existing ones, suitable Medium Access Control protocol which matches communication needs of WMN. The IEEE 802.15.3 MAC for Wireless Personal Area Network (WPAN) is a potential candidate for mesh networking over small distances. One approach to forming the WMN using IEEE 802.15.3 technology is to interconnect two or more piconets using brigde devices. Piconet inter-connection through bridging techniques is an interesting problem, since it enables coverage of larger areas and accomodates larger number of devices. Bridge design in IEEE 802.15.3 involves concepts of parent and child piconets which basically partition the bandwidth between parent and child piconets on TDMA basis. Bridge device has to be a member in parent piconet and coordinator in child piconet. Due to the resemblance to the similar approach in Bluetooth technology we will refer to this approach as to Master -Slave bridge. The scheduling of channel time to the bridge device is a challenging task. We investigate the performance of master-slave bridge (MS-bridge) that interconnects two IEEE 802.15.3 piconets in a parent-child manner. We have designed an adaptive bandwidth allocation algorithm for bridge bandwidth allocation, and examined the impact of the value of the smoothing constant and threshold hysteresis on the throughput, blocking probability, bridging delay, and average queue size for the downlink queue at the bridge device.

Contents

List of Figures

List of Tables

Acknowledgments

I convey my profound gratitude to ALLAH, the most merciful and gracious, who has created me and made me complete this work successfully.

I would like to thank my advisor Dr. Jelena Mišić who provided me with the opportunity and encouraged me to explore a new research area. Dr. Mišić enthusiasticly guided me with her extensive research experience throughout my thesis work. She taught me how to follow the innovative paths of research and always showed me the right directions. My special thanks goes to Dr. Vojislav Mišić for his continuous support for Artifex and valuable suggestions during our research meetings. Dr. Vojislav Mišić was always there for me whenever I faced any problem regarding my research work and advised me as required.

My heartful thanks goes to my parents whose endless support, encouragement, and love made me finish my thesis. Thanks to my sister and brother-in-law. I also remember my little nephew who always keeps me cheerful.

I would like to thank Faculty of Graduate Studies for International Graduate Student Entrence Scholarship, Faculty of Science for Science Student Scholarship, and Department of Computer Science for Graduate Fellowship. I would also link to thank my supervisor Dr. Jelena Mišić for supporting me with Research Assistantship.

This thesis is dedicated to my parents.

Chapter 1

Introduction

The rapid growth of wireless communication has created a strong demand for mesh networking in wireless applications. Wireless mesh networks (WMNs) are dynamically self-organized and self-configured, with the nodes belonging to different small area networks automatically establishing an ad hoc network and maintaining the mesh connectivity [2, 1]. Mesh networking is considered to be the key technology for the next generation of wireless networking such as broadband home networking, building automation, vehicle automation, and enterprise networking to support many potential applications. The applications of Wireless Personal Area Networks (WPANs) include, but are not limited to data transfer among various devices e.g. camcorder to PC, home gateway to portable devices, security camera to central server, digital images from a robot to a PC (disaster recovery), etc. These applications have requirements of high data rate, low cost, and quality of service provisions such as bridging delay and throughput with few transmission errors. The IEEE 802.15.3 standard [8] for WPANs is designed to fulfill the above requirements over small distances and the standard can be employed for mesh networking. Consumer demand for mesh networks opens a wide research area in both academia and industry for enhancement and expansion of WPANs. Proper utilization of channel time has an impact on network performance, which is a prime concern for the researchers. Research has been done to evaluate the performance of Medium Access Control (MAC) protocol for a single piconet, but the performance of 802.15.3 based mesh networks is a new area of networking research. The performance of a mesh network significantly depends on the interconnection between piconets. The quality of the interconnection in turn depends on how the bridge device handles the traffic that passes through the bridge device. Operations of the bridge devices have impact on overall network (mesh performance) performance and channel allocation schemes for bridges need to be explored.

The IEEE 802.15.3 standard defines the physical layer and MAC protocol for WPANs. It is intended for small area networks such as home networks and has a higher data rate than other WPANs and Wireless Local Area Networks (WLANs). We can expand the coverage of a home area network or a small area network by interconnecting two or more piconets [1]. The IEEE 802.15.3 standard is also envisioned as the enabling technology for IPTV in home-networking. The 802.15.3 MAC protocol is also a potential candidate for another emerging field in wireless networking, called Ultra Wide Band (UWB) technology. WPANs can achieve very high data rate with the help of UWB technology [5]. UWB application demands and current activities of IEEE 802.15.3 standard are described on Wimedia Alliance website [1].

We can use 802.15.3 MAC as the building block of 802.15.3 based Wireless Mesh

[1]A piconet is a small network of two or more different types of mobile devices that can communicate with each other

Network (WMN) or 802.15.3 scatternet[2] formation. Mesh network and scatternet are two general terms that can be used interchangeably. We need to inter-connect the piconets to form a scatternet. We can inter-connect two piconets through bridging [3] at the MAC protocol layer. For successful communication between two piconets, the devices in both network and especially the bridge device need guaranteed channel time. Therefore, we need a scheduling policy to ensure the proper allocation of bandwidth to the bridge device as well as to the devices of different piconets in a WMN. This is a complex task when we use a single channel for all the piconets that form the mesh network.

Bandwidth allocation to the bridge device is a common problem in scatternet formation in all networks such as bluetooth, mobile ad hoc network (MANET), and IEEE 802.15.3 network. Data exchange between two piconets is not possible without inter-connection, which requires a bridge device. The bridge device needs proper bandwidth allocation for data exchange and for its own operation. We investigated the scatternet formation methods of bluetooth and MANET to find the possibilities of employing them for 802.15.3 piconet inter-connection. These methods are not compatible for 802.15.3 WPAN as the bridging mechanisms are based on MAC layer characteristics that are unique for each type of network and incompatible for each other. The recent works of IEEE 802.15.3 scatternet formation deals with optimization of coverage areas, connection data rate optimization over distance [19] and number of piconet optimization [17]. Effective piconet inter-connection in terms of bandwidth allocation for bridge is still an open issue that need to be studied.

[2]A scatternet is formed with two or more piconets

[3]Bridging is the mechanism of connecting two or more piconets through a device

The main focus of this research is to design an adaptive bandwidth allocation algorithm for bridge operation and evaluate the performance of the designed algorithm. First we performed experiments to observe the performance of non adaptive bandwith allocation to bridge devices. Further, we designed and described a bandwidth allocation algorithm that can be applied for bridge up link and down link operation. The analysis of the results of non adaptive bandwidth allocation experiments helped us to determine the parameter values, such as initial amount and maximum amount of bandwidth allocation for our adaptive bandwidth allocation experiments. Bandwidth allocation is mainly based on time division multiple access (TDMA) [12] and follows the 802.15.3 MAC superframe (duration of channel time for a single cycle) structure. We have designed our algorithm to adaptively allocate bandwidth for bridge down link operation and also introduced hysteresis threshold to prevent rapid change in bandwidth allocation when queue size has large and quick variations. We performed separate experiments for adaptive bandwidth allocation and adaptive bandwidth allocation with hysteresis system. Introduction of hysteresis system shows performance improvement over adaptive bandwidth allocation algorithm.

We organised the rest of the thesis as follows. In Chapter 2 we describe the basic properties of IEEE 802.15.3 standard and discuss the concepts of child piconet and neighbour piconet provided in the standard. In Chapter 3 we talk about the piconet interconnection strategies in different networks of 802.15 standards family with related works. Then in Chapter 4 we describe the network model and superframe structure for our experiments. In this chapter we also lay out the design of piconet interconnection through MS-bridge with respect to our experiments. Further in chapter 5 we give detail description of our simulation model and provide the class diagrams of the model. In Chapter 6 , 7 and 8 we describe our bandwidth allocation algorithms, experimental design, and analysis of simulation results. Finally we summarize our results and conclude in Chapter 9.

Chapter 2

Basic properties of IEEE 802.15.3 standard

The IEEE 802.15.3 standard [8] for high data rate Wireless Personal Area Networks (HR-WPANs) is designed to fulfill the requirements of high data rate suitable for multimedia applications whilst ensuring low end-to-end delay. It is also designed to provide easy reconfigurability and high resilience to interference, since it uses the unlicensed Industrial, Scientific, and Medical (ISM) band at 2.4GHz which is shared with a number of other communication technologies such as WLAN (802.11b/g) and Bluetooth (802.15.1), among others.

Devices in 802.15.3 networks are organized in small networks called piconets, each of which is formed, controlled, and maintained by a single dedicated device referred to as the piconet coordinator (PNC). The network is formed in an ad hoc fashion: upon discovering a free channel, the PNC capable device starts the piconet by simply transmitting period beacon frames; other devices that detect those frames then

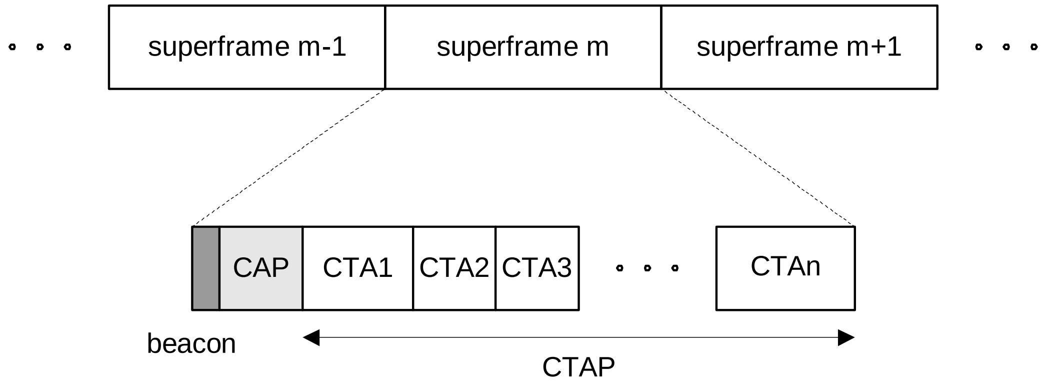


Figure 2.1: IEEE 802.15.3 Superframe format

request admission, or association (as it is referred to in the 802.15.3 standard). The coordinator duties include transmission of periodic beacon frames for synchronization, admission of new devices to the piconet, as well as allocation of dedicated time periods to allow unhindered packet transmission by the requesting device. When a device needs to send data, it sends a request to the PNC. If resources are available, the PNC allocates required channel time allocations (CTAs) in the next superframe for that device and informs it in a subsequent beacon. The device then sends data to the destination device in the allocated CTAs on a peer-to-peer basis.

Time in an 802.15.3 piconet is structured in superframes delimited by successive beacon frame transmissions from the piconet coordinator. The structure of the superframe is shown in Figure 2.1. Each superframe contains three distinct parts: the beacon frame, the contention access period (CAP), and the channel time allocation period (CTAP). During the Contention Access Period, devices compete with each other for access; a form of CSMA-CA algorithm is used. This period is used to send requests for CTAs (defined below) and other administrative information, but also for smaller amounts of asynchronous data.

Channel Time Allocation Period contains a number of individual sub-periods (referred to as Channel Time Allocation, or CTA) which are allocated by the piconet coordinator upon explicit requests by the devices that have data to transmit. Requests for CTAs are sent during the Contention Access Period; as such, they are subject to collision with similar requests from other devices. The decision to grant the allocation request or not rests exclusively with the piconet coordinator, which must take into account the amount of resources available – most often, the traffic parameters of other devices in the network and the available time in the superframe. If a device is allocated a CTA, other devices may not use it, and contention-free access is guaranteed. CTA allocation is announced in the next beacon frame; it may be temporary or may last until explicit deallocation by the piconet coordinator. Typically, CTAs are used to send commands and larger quantities of isochronous and asynchronous data.

Special CTAs known as Management Channel Time Allocation (MCTA) are used for communication and dissemination of administrative information from the piconet coordinator to the devices, and vice versa. There are three types of MCTAs defined in the standard - Association, Open, and Regular MCTA. Association and Open MCTAs use the Slotted Aloha [12] medium access technique, while Regular MCTAs use the TDMA mechanism.

Reliable data transfer in 802.15.3 networks is achieved by utilizing acknowledgements and retransmission of non-acknowledged packets. The standard defines three acknowledgment modes:

- no acknowledgement (No-ACK) is typically used for delay sensitive but loss tolerant traffic such as multimedia (typically transferred through UDP or some similar protocol);
- immediate acknowledgement (Imm-ACK) means that each packet is immediately acknowledged with a small packet sent back to the sender of the original packet; and
- delayed acknowledgement (Dly-ACK), where an acknowledgment packet is sent after successfully receiving a batch of successive data packets; obviously, this allows for higher throughput due to reduced acknowledgment overhead – but the application requirements must tolerate the delay incurred in this case, and some means of selective retransmission must be employed to maintain efficiency.

The 802.15.3 standard contains provisions for the coexistence of multiple piconets in the same (or partially overlapping) physical space. Since the data rate is high, up to 55 Mbps, the channel width is large and there are, in fact, only five channels available in the ISM band for use of 802.15.3 networks. If 802.11-compatible WLAN (or, perhaps, several of them) is/are present in the vicinity, the number of available channels is reduced to only three in order to prevent excessive interference between the networks adhering to two standards. As a result, the formation of multiple piconets must utilize time division multiplexing, rather than the frequency division one, as is the case with Bluetooth. Namely, a piconet can allocate a special CTA during which another piconet can operate. There are two types of such piconets: a child piconet and a neighbour piconet.

A child piconet is the one in which the piconet coordinator is a member of the parent piconet. It is formed when a PNC-capable device which is a member of the parent piconet sends a request to the parent piconet coordinator, asking for a special CTA known as a private CTA. Regular CTA requests include the device addresses of both the sender and the receiver; a request for a private CTA is distinguished by virtue of containing the same device address as both the sending and the receiving node. When the parent piconet coordinator allocates the required CTA, the child piconet coordinator may begin sending beacon frames of its own within that CTA, and thus may form another piconet which operates on the same channel as the parent piconet, but is independent from it. The private CTA is, effectively, the active portion of the superframe of the child piconet. The child superframe consists, then, of this private CTA which can be used for communication between child piconet coordinator (PNC) and its devices (DEVs); the remainder of the parent superframe is reserved time – it can't be used for communication in the child piconet.

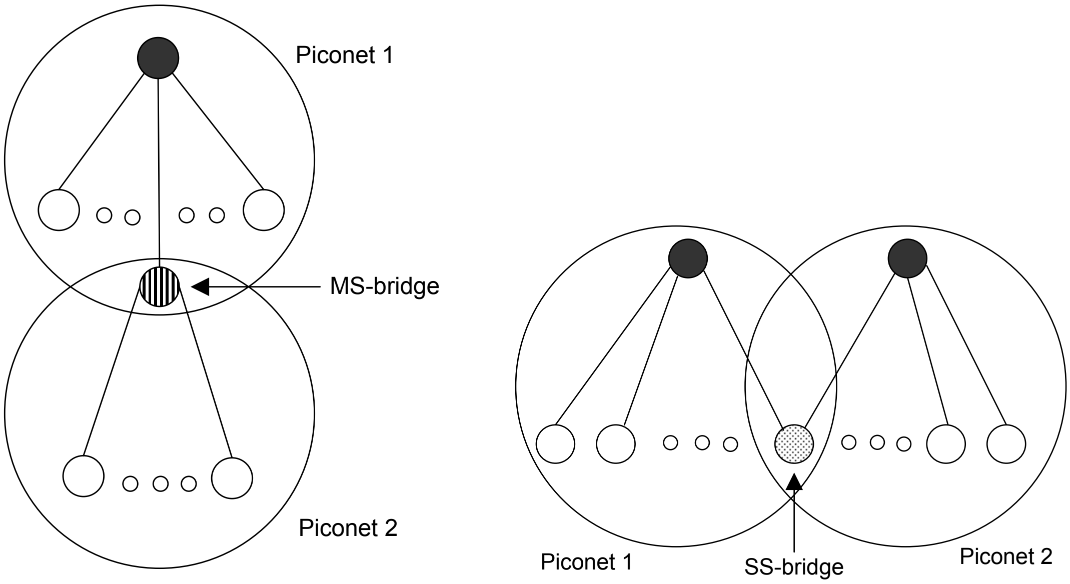
The 802.15.3 standard also provides the concept of the neighbour piconet, which is intended to enable an 802.15.3 piconet to coexist with another network that may or may not use the 802.15.3 communication protocols; for example, an 802.11 WLAN in which one of the devices is 802.15.3-capable. A PNC-capable device that wants to form a neighbour piconet will first associate with the parent piconet, but not as an ordinary piconet member; the parent piconet coordinator may reject the association request if it does not support neighbour piconets. If the request is granted, the device then requests a private CTA from the coordinator of the parent piconet. once a private CTA is allocated, the neighbour piconet can begin to operate. The neighbour piconet coordinator may exchange commands with the parent piconet coordinator, but no data exchange is allowed. In other words, the neighbour piconet is simply a means to share the channel time between the two networks. Since, unlike the child piconet, data communications between the two piconets are not possible, this mechanism is unsuitable for the creation of multi-piconet networks, and, consequently, for mesh networking.

Chapter 3

Piconet interconnection strategies

Two common approaches, namely MS-bridge and SS-bridge are used for piconet inter-connection in different networks regardless of their technology. The choice of the connection depends on the location of the bridge device within a scatternet. The inter-connection will be established by an MS-bridge if a PNC capable device is present between two piconets. On the other hand an SS-bridge can be used if a device resides within the coverage area of two piconets. In an MS-bridge, Figure 3.1(a), the bridge device is the PNC for Piconet 1 and a normal member of Piconet 2. In an SS-bridge, Figure 3.1(b), the bridge device is a normal member in both the piconets. The inter-connection techniques of these two approaches are as follows:

Master-Slave bridge: From the standpoint of piconet interconnection, the child piconet mechanism allows for simple implementation of the Master-Slave interconnection topology, since the two piconets are linked through the child piconet coordinator which partakes in both of them, and thus can act as the bridge. Figure 3.1(a) schematically shows such topology in which Piconet 2 is the child of Piconet 1; the



(a) Master-Slave (MS-bridge) (b) Slave-Slave (SS-bridge)

Figure 3.1: Inter-connection of two piconets through MS and SS bridge

child piconet PNC is also acting as a Master-Slave bridge that links the piconets. The timing relationship of superframes in parent and child piconets is shown in Figure 3.1(a), where the top part corresponds to the parent piconet and the bottom part to the child piconet. Note that the distinction is logical rather than physical, since the piconets share the same RF channel.

A given piconet can have multiple child piconets, and a child piconet may have another child of its own. Obviously, the available channel time is shared between those piconets, at the expense of decreased throughput and increased delay; but the effective transmission range may be increased.

Slave-Slave bridge: This interconnection topology may also be implemented, but in a slightly more complex manner. Namely, direct communication between the members of different piconets is not possible; the only shared device is the PNC of the child piconet. If an ordinary device wants to act as a bridge, it must explicitly associate with both parent and child piconets, and obtain a distinct device address in each of them. In this manner, multiple bridges may exist between the two piconets. The topology of two piconets interconnected through a Slave-Slave bridge is shown in Figure 3.1(b). Note that, in this case, the piconet may be linked through a parent-child relationship; but they could also use different RF channels, with a certain penalty because of the need for the bridge to synchronize with two independently running superframe structures. However, SS-bridge approach is out of the scope of our experiment.

3.1 Related work

The IEEE 802.15 series of protocols defines MAC protocols for different WPANs. These WPANs vary with different technologies and data rates, such as Zigbee for low rate, Bluetooth for medium rate, and Wimedia for high rate, but they all are designed to cover small area and use single and/or limited channels.

Bluetooth is a medium data rate network and uses polling [14] to manage the medium access of the nodes. There are three bridging approaches in bluetooth, namely rendezvous, adaptive rendezvous and walk-in scheduling that can be used to form a scatternet. Rendezvous is the point when data between two piconets are exchanged through the bridge device and there will be problem if the two piconets fail to synchronize at this point according to the scheduled time. Moreover, rendezvous based scheduling does not scale up well [14]. In the walk-in approach the bridge switches between the piconets without any prior scheduling information and does not have any rendezvous point. Its disadvantage is synchronization delay though its scalability is better than the rendezvous approach. Also this approach can degrade the performance of real-time applications since the bridge does not visit the piconets in any ordered fashion. These scheduling policies can use either MS-bridge (Figure 3.1(a)) or SS-bridge (Figure 3.1(b)). Johansson et el. [11] designed a distributed scheduling algorithm to inter-connect bluetooth piconets employing MS-bridge and SS-bridge. The MS-bridge takes into account the neighbours' scheduling information to schedule channel time to its piconet. We can centrally allocate channel time with the help of neighbour piconet's scheduling information. This algorithm can also adapt to any topological change locally without affecting the whole network. Therefore, we have chosen MS-bridge to inter connect our piconets. MS-bridge is a simple approach as it operates directly under the control of parent PNC. Other way we can say that for MS-bridges, the connection is already there and does not require additional channel time to establish a connection. However, non-local traffic (packets coming from one piconet to another) will experience higher bridging delay if channel time is not allocated properly to the bridge device.

The IEEE 802.15.4 standard is provided for low rate WPANs which have different application demands. The bridging of 802.15.4 is different from bluetooth and 802.15.3. The idea is to send data to the sink node from different source nodes. During data exchange the bridge periodically visits the sink piconet to deliver the data collected from the source piconet. There is an inactive period in the superframe of 802.15.4 MAC [9]. To reduce data loss probability due to the inactive period, the nodes collect redundant data. In 802.15.3 networks, redundant data collection is not desirable, for example in real-time multimedia applications duplicate packets containing the same data is not acceptable. Misic [13] et al. have studied the performance of 802.15.4 beacon enabled MS-bridges by queuing analytical approach with satisfactory results for the CSMA/CA MAC. Piconets in 802.15.3 and 802.15.4 are beacon enabled and both have time periods allocated by the PNC. The 802.15.4 standard uses guaranteed time allocation (GTA) period for data exchange and the 802.15.3 standard uses CTA period. I have studied the 802.15.4 bridging problem and found GTA is being used for the bridge data exchange. I used CTA period for bridge operation in my experiments, where a long CTA time were allocated by the parent PNC similar to 802.15.4 bridging.

Xiao [18] has performed a detailed performance evaluation of the IEEE 802.15.3 [8] and IEEE 802.15.3a [6] standards through simulation and mathematical analysis. He has also done a throughput analysis of the 802.11 [7] protocol, which uses backoff with counter freezing during inactive portions of the superframe. The freezing and backoff techniques are essentially the same in the 802.11 and 802.15.3 MACs, except that different ways of calculating the backoff time are utilized. The backoff and freezing have an impact on the performance of the network; especially the backoff has a direct impact on the delay parameter. Large backoff windows can result in longer delays. On the other hand, small backoff windows may increase the probability of collisions. Xiao used the backoff procedures defined in the 802.11 and 802.15.3 MAC specifications; this work gives us performance of the protocol in terms of throughput over various payload sizes. From the performance experiments of single piconet we realized that bandwidth allocation is an important factor in network performance. We also realized that bandwidth allocation to bridge device for piconet interconnection is an interesting area of research.

Yin and Leung [19] have optimized the connection data rate of a scatternet. The connection data rate is defined as the data rate of a connection from one device to a randomly chosen device from the scatternet. The random device can be within the same piconet or from another piconet of the scatternet. They have defined the problem as finding an optimized coverage area for the piconet to achieve expected scatternet connection data rate (ESCR). Their results suggest that the piconet radius of 10 m has a good ESCR and the standard defines the piconet size as 10 m. Therefore, in my experiments I will assume the piconet coverage area is 10 m.

Tan and Guttag [16] designed an appointment based scheduling algorithm based on incoming and outgoing traffic patterns to allocate channel time to the devices of a piconet. Tan and Guttag's work inspired us to design an adaptive CTA allocation algorithm that allocates CTA based on queue status.

Chapter 4

Interconnecting IEEE 802.15.3 piconets through bridge device

Bridging of two IEEE 802.15.3 based networks involves complex issues such as inter-connection of piconets, synchronization of parent-child piconets, and scheduling of channel time. The inter-connection between two or more piconets needs a bridge device and allocation of channel time for that bridge device. In this chapter first we discuss the superframe structure of our experiments. Then we describe IEEE 802.15.3 based network model where an MS-bridge establishes an interconnection between two piconets followed by channel time allocation techniques for bridge operation.

4.1 Superframe structure

We have designed the superframe for our experiments based on the structure showed in Figure 4.1. In Figure 4.2 we draw the structure of the superframe that we

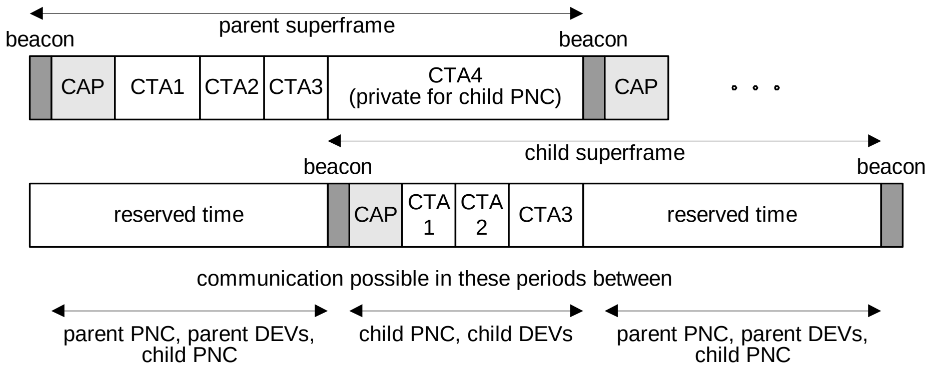


Figure 4.1: Communication between parent and child piconet within a superframe (Taken from the IEEE 802.15.3 Std.)

used in our experiments. The whole superframe which is basically under the control of parent PNC is divided into two portions, one for parent piconet and other for child piconet. Each portion has the same structure with beacon period, CAP and CTA period with reserved CTAs for bridge operations. In Figure 4.2 bridge-UL indicates time for uplink operation and bridge-DL indicates time for downlink operation. We assigned x number of CTAs for uplink traffic and y number of CTAs for downlink traffic. x and y may have constant values or we can change their values based on traffic condition. In our experiments we have varied the value of y to analyse bridge downlink performance. Bridge uplink performance can also be analysed by varying the x value. Since the downlink traffic volume is high, we performed our experiments to analyse the performance of downlink operation. During downlink, bridge device forwards packets to the devices of child piconet. During uplink operation the bridge device delivers the packets received from the devices of child piconet to their intended destination devices in the parent piconet.

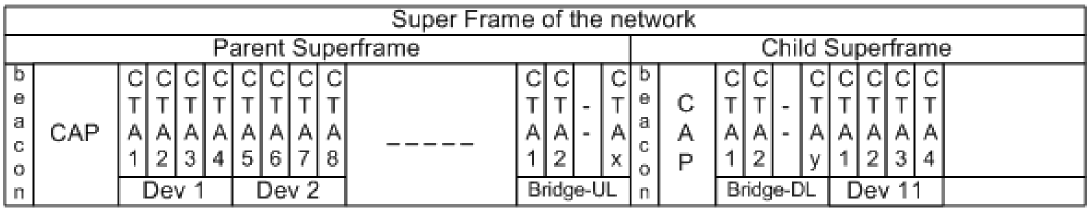


Figure 4.2: Super frame structure. x=number of CTAs for up load and y=number of CTAs for down load

The superframe size of our experiments is as follows:

$$SF_L = B_p + CAP_p + CTAP_p + B_c + CAP_c + CTAP_c \tag{4.1}$$

In equation 4.1 SF_L stands for superframe length, p stands for parent piconet and c stands for child piconet. B_p and B_c represents the duration of beacon period for parent piconet and child piconet respectively. The beacon period for both parent piconet and child piconet is 100 μs. Parent piconet CAP time is 3600 μs and child piconet CAP is 2000 μs. CTAP is the time duration of all the CTAs that are assigned to all the devices of a piconet. $CTAP_p$ and $CTAP_c$ include x and y number of CTAs respectively for bridge operations. The size of each CTA is 894 μs, which is determined based on the packet size so that each CTA can be used to send a single packet. We have selected above values based on the specification given in IEEE 802.15.3 standard [8].

4.2 Network model

We show the network model of our experiment in Figure 4.2 where an MS-bridge establishes connection between a parent piconet and a child piconet. The parent pi-

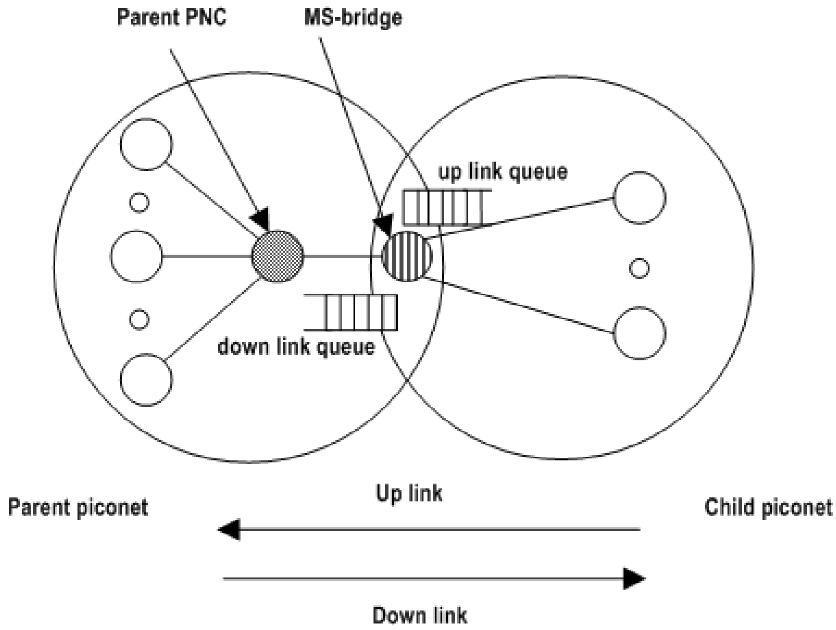


Figure 4.3: Piconet inter-connection through MS-bridge

conet consists of nine member devices and the child piconet consists of five member devices excluding the bridge device. Here bridge device acts as the PNC of the child piconet and a member of the parent piconet. We can see from Figure 4.2 that the bridge device falls under the coverage area of parent piconet and it has its own coverage area. Each device in the network generates local traffic with locality probability P_L and non-local traffic with $1 - P_L$ probability. The basic MS- bridging operation requires that the bridge device maintains two queues for the inter-connection between two piconets, one queue for uplink traffic and other for downlink traffic. The traffic flow from parent piconet to child piconet is defined as down link and child piconet to parent piconet as up link. The downlink experiences more traffic flow than the uplink because there are more devices in the parent piconet than the child piconet. Therefore, there will be more non-local traffic coming from the devices of parent piconet to the devices of child piconet. The bridge device gets CTA to exchange packet in each superframe for both uplink and downlink traffic. The uplink data transfer happens during parent piconet operation and bridge device forwards packets from child piconet to the devices of parent piconet. The downlink operation happens during the child superframe portion. From Figure 4.1 we can see that a private CTA from the parent superframe is assigned for child piconet operation and it is structured as the superframe of child piconet. The child PNC has beacon and CAP period same as the parent PNC and also assigns CTAs to its devices same as the parent PNC. Now, for bridge uplink operation parent piconet can reserve CTAs specifically for bridge operation from parent superframe portion other than the one indicated as private in Figure 4.1 for child PNC. For downlink operation same way the child PNC will reserve CTAs from child superframe portion for bridge operation. In next section we describe in detail about the superframe structure of our experiments.

4.3 CTA allocation for bridge operation

The IEEE 802.15.3 standard defines the basic bandwidth unit as CTA in the superframe. In our experiments we quantify the amount of bandwidth in terms of CTAs. We will use these terms bandwidth and CTAs interchangeably to imply amount of bandwidth allocation to piconet devices. Allocation of CTA to bridge device has direct impact on the performance of bridge operation. Bandwidth allocation to piconet devices and especially bridge has to match traffic arrival pattern. Traffic passing through the bridge is influenced by applications running on devices and placement of devices across the interonnected piconets. Usual practice in modeling traffic arrival pattern to a single device is to use some random process e.g. Poisson process [10]. If we use constant number of CTAs for different amount of traffic, we can not ensure proper utilization of channel time. Through adaptive CTA allocation we can maximize the channel time utilization and minimize the allocation of unused channel time. Therefore, it is important to take into account the traffic load in CTA allocation. Traffic intensity is indicated by length of packet queues. In this thesis we have first observed deficiencies of constant bandwidth allocation for bridge device. Based on those observations we have developed adaptive algorithm for bridge bandwidth allocation based on historical data about queue lengths. Further, we extended adaptive CTA allocation algorithm by introducing hysteresis threshold to improve the performance of bridge operation. Therefore, our research had three phases. First we evaluated the bridge performance with a simple approach to study its performance limits. Then we designed an adaptive CTA allocation algorithm and analysed its performance. Finally we extended our algorithm to enhance the bridge performance. In chapter 6, 7, and 8 we describe these three approaches with simulation results and analysis.

Chapter 5

Simulation design and parameters

We used discrete event simulation using Petri-net based simulator, Artifex [15], [3] to build our network model and evaluate the performance of our algorithms. Artifex provides graphical interface to build simulation model where we can design our network functionalities an in object oriented fashion. An object can be defined using graphical elements, among which transitions, places, and links are the most basic ones. In Figure 5.3(a) we can see transitions, places, and links in the shape of rectangles, circles, and arrow lines respectively. Transitions are the elements where we implement the actions or logic of an event by coding. Places work as storage that can hold data units in the form of tokens. Tokens are structured data that we can define in Artifex. Transitions and places are connected by links. A transition triggers when it fetches token from the connecting input places.

Objects send and receive data between each other through interfaces. We can define a set of input and output places under an interface. The input place of an object is connect to the output place of another object. Tokens are passed to and from objects carrying simulation information through the interfaces. Artifex also allows us to split the functionalities of an object into different pages which are basically connected with each other. There are also initial and final action sections in Artifex. In initial action we can initialize the variables, which needs initial values. In final action section we can take measurement of the parameters. In our model we have three objects: PNC(piconet coordinator class), Medium (wireless medium class), and Device (normal member device class). In the following sections we describe the objects of our model and their functionalities.

5.1 PNC

In Figure 5.1 we illustrate the functionalities of PNC class. The PNC class has one input place and two output places. GETTING_DATA:PACKET is an input place and SENDING_ACK:PACKET and PNC_OUT:BCON are output places. PACKET and BCON are two different token types. In our simulator we represent different types of packets such as data packet, acknowledgement packet, beacon packet etc. by tokens. Name of token types with the place names indicates what type of token an input or output place can send or receive. Each input place in one object is connected with an output place in another object and vice versa. The connected input and output place pair must be defined with same type of token. The PNC generates beacon at the beginning of each superframe and sends them to all devices in the network through medium class. Beacons are sent by PNC_OUT output place. PNC receives data packet or command packets by GETTING_DATA input place. After getting a data packet, it sends acknowledgement packet through SENDING_ACK output place.

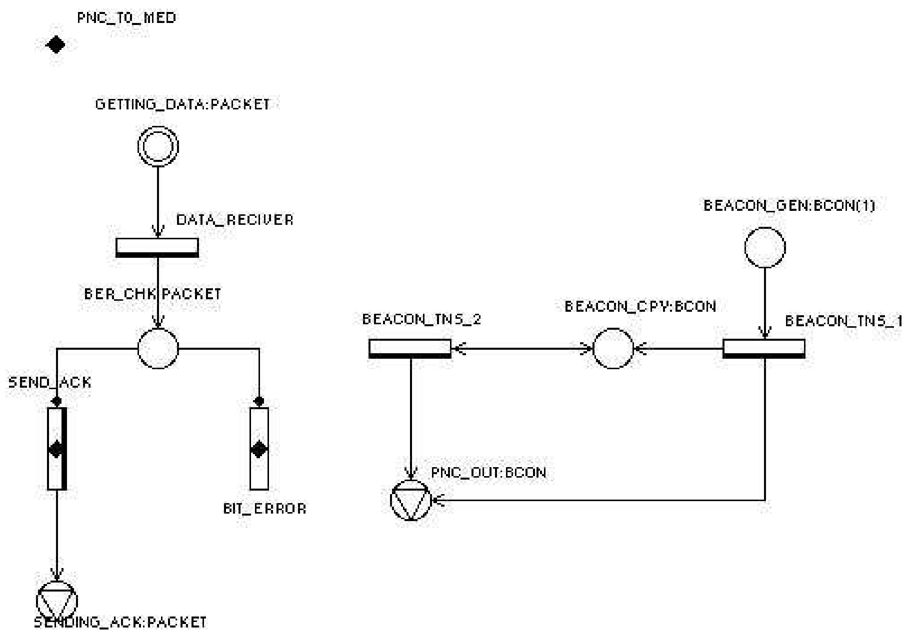


Figure 5.1: Object diagram of piconet co-ordinator

Our adaptive bandwidth allocation algorithm and adaptive bandwidth with hysteresis threshold algorithm runs in PNC class.

5.2 Medium

In Figure 5.2 we can see the Medium class has two input places and six output places. The input places are RECEIVER:PACKET and RCV_FRM_PNC:BCON. The RECEIVER place can receive PACKET type tokens, which may be data, command or acknowledgement packet from either PNC class or device class. The RCV_FRM_PNC place receives BCON type token, which is beacon signal that comes from the PNC class. The output places are OUTPUT:BKOFF, BEACON_TO_DEVICE:BCON,

DATA_TO_DEV:PACKET, DATA_TO_PNC:PACKET, ACK_TO_DEVICE:PACKET, and CALCULATE:PER_EVAL. We have two new token types here: BKOFF and PER_EVAL. BKOFF token type is nothing but clock pulse of our virtual clock. Each pulse is considered to be one μ second. All the events in our simulation happen based on this clock time. Through OUTPUT place the clock sends timing information to all the devices in the network. BEACON_TO_DEVICE place passes the beacon signal from the PNC class to all the devices. DATA_TO_DEV and DATA_TO_PNC places are connected to input places in DEVICE class and PNC class respectively and passes data or command packets to the intended destination. ACK_TO_DEVICE place sends the acknowledgement packets coming from one device to another. CALCULATE place is used to collect the measurement information of different parameters and uses token type PER_EVAL.

The transmission delay of packets is implemented in the Medium class at TRANSMISSION_DELAY transition. Transmission delay is calculated based on packet size and data rate. When Medium class receives a packet it becomes busy and we indicate that with a variable. Medium class sends the status of medium by BKOFF token with every clock pulse to the devices. If a packet arrives while medium status is busy, then collision happens and we check that through CHK_MED place. After a collision both packets are discarded.

5.3 Device

The Device class is the biggest class of our model with complex and different functionalities. We reduced the complexity of this class by distributing the function-

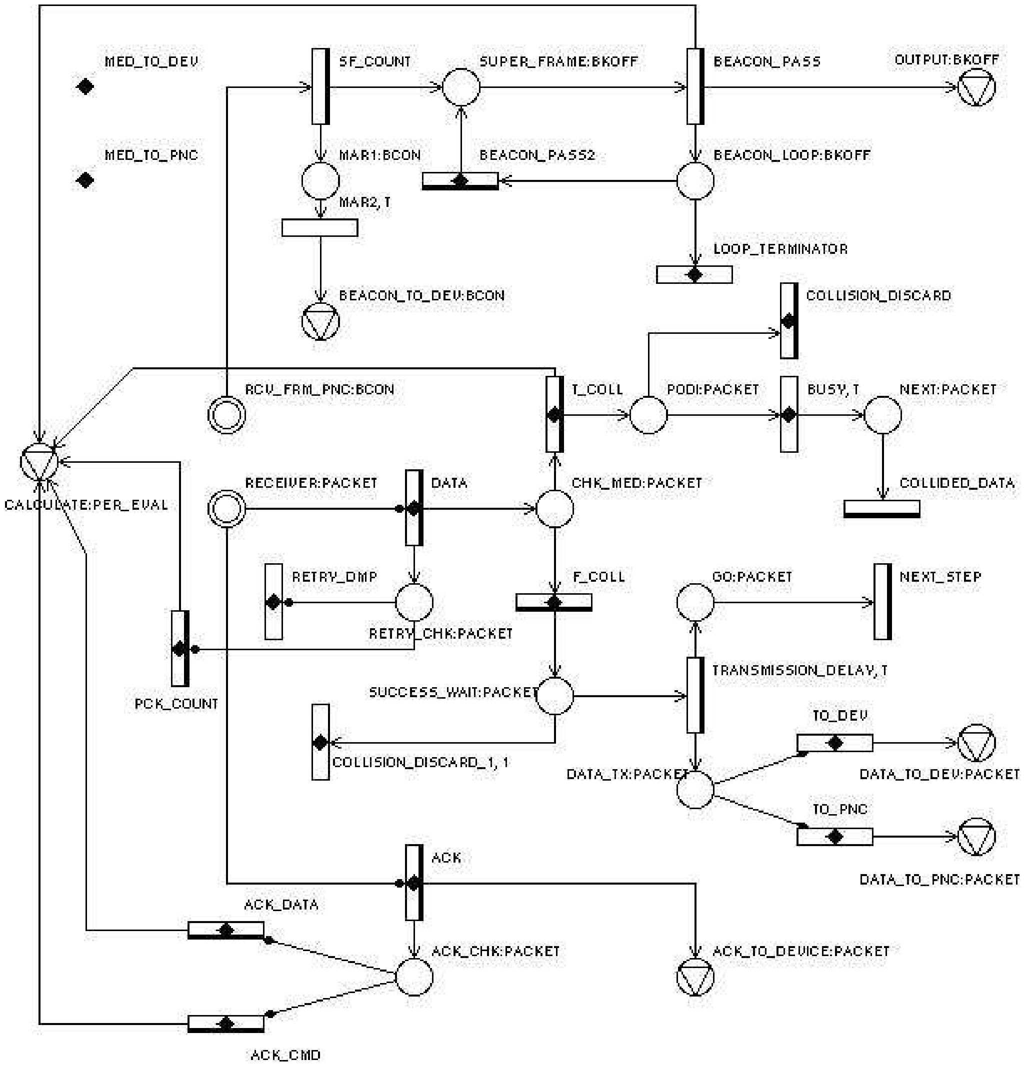
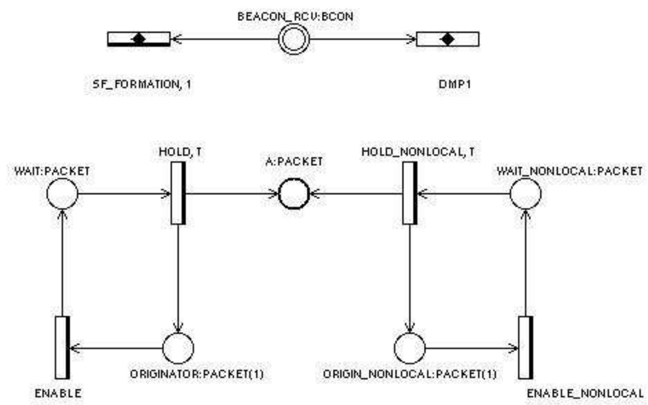


Figure 5.2: Object diagram of medium

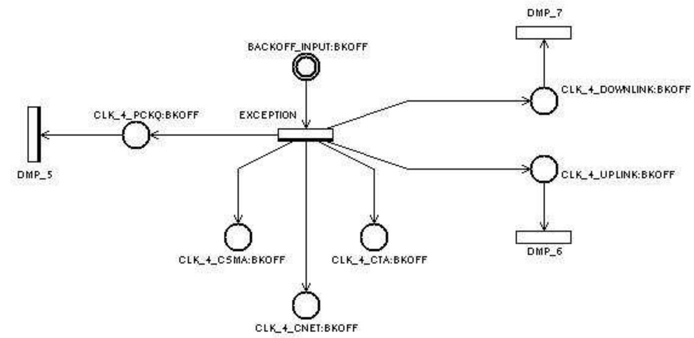
alities in different pages. The functionalities of Device class are: packet generation, distributing clock time, formation of superframe, csma-ca protocol, channel access during CTAP, data transmission and data receiving. We show the diagrams of above operations in Figures 5.3, 5.4, 5.5, 5.6 and 5.7 and describe them in the following paragraphs.

Figure 5.3(a) shows the main page of our Device class. We named it main page because the operation of this class starts by generating packets. There is only one input place, which is BEACON_RCV:BCON as we can see from Figure 5.3(a). Beacon packets come from the PNC class through Medium class with the superframe and channel time information for the devices. After receiving beacon the devices update their necessary variables according to the provided information in the beacon. Packets are generated by ORIGINATOR:PACKET and ORIGIN_NONLOCAL:PACKET places for local and non-local traffic respectively. The codes for random packet generation are written at ENABLE and ENABLE_NONLOCAL transitions. Figure 5.3(b) shows the diagram for sending timing information to different parts of the Device class for respective events. The BACKOFF_INPUT:BKOFF input place receives the clock time and medium status information by BKOFF tokens. Same BKOFF token is copied by EXCEPTION transition into different places that are connected to different pages of the Device class. Thus same clock time reaches all around the device class simultaneously preventing clock drifting. The overdrawn places in Figure 5.3(b) indicates that this place connects two pages. Figure 5.3(c) shows the diagram for child superframe formation.

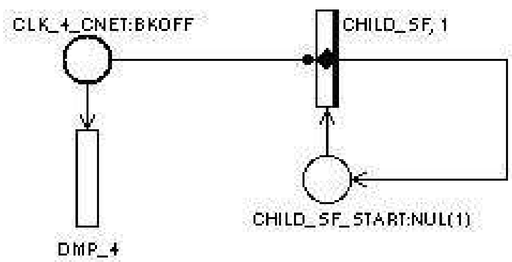
The operation of csma-ca protocol during CAP period is shown in Figure 5.4(a) which runs in CSMA page. ACC_1:PACKET is the entry point for packets that arrives during CAP period. We drew the block diagram of csma-ca algorithm from the description of IEEE 802.15.3 standard in Figure 5.6. The artifex diagram in Figure 5.4(a) is the implementation of block diagram in Figure 5.6. The transition BKWINDOWCALCULATE calculates the time for backoff before channel access by



(a) Packet generation



(b) Clock sending time for different events



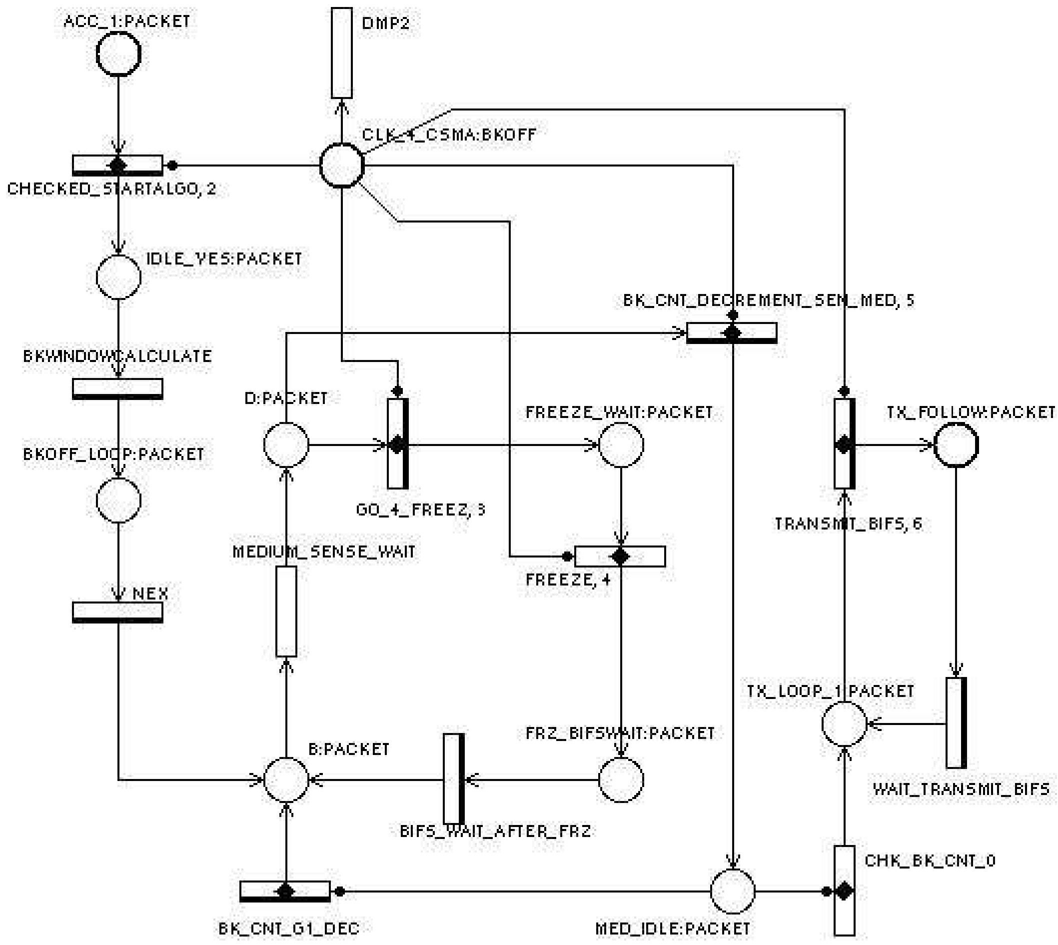
(c) Child superframe formation

Figure 5.3: Object diagram of devices: packet generation, sending clock time and child superframe formation

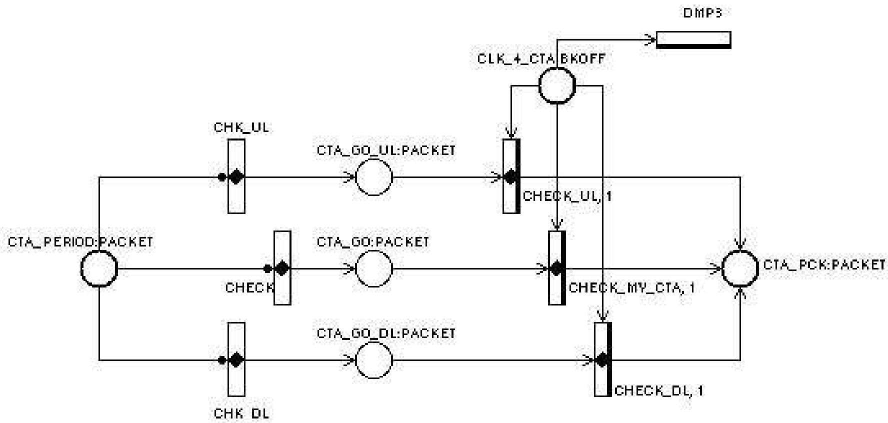
a packet. We implemented the backoff process in the form of looping as we can see in Figure 5.4(a) that transitions BK_CNT_G1_DEC, MEDIUM_SENSE_WAIT, and BK_CNT_DECREAMENT_SEN_MED forms a loop and there is another loop inside it for freezing operation that freezes backoff counter if the medium gets busy during backoff process. Once the backoff counter reaches 0, the packet gets channel access and starts transmission. Finally, a packet successfully goes through csma-ca process after it exits through TX_FOLLOW place which is connected to RxTx page.

In Figure 5.4(b) we show the operation of CTA period. Here we drew both normal device CTA operation and bridge device CTA operation. A packet has three ways to go from the place CTA_PERIOD. The CHECK transition triggers if the packet is a local traffic, otherwise CHECK_UL or CHECK_DL triggers based on the packet's destination whether it is uplink or downlink traffic.

In Figure 5.5 we show the operation of transmission page RxTx. In this page we implemented the following activities: packet transmission, retry process, and acknowledgement sending and receiving. Here we have three input places and two output places. The input places are BACKOFF_INPUT:BKOFF, ACK_INPUT:PACKET and DATA_FROM_MED:PACKET. The BACKOFF_INPUT place provides the timing information. The ACK_INPUT place receives the acknowledgement packets and DATA_FROM_MED place receives data packets. The place DATA_FROM_MED is also connected with dataReceive page and we will discuss about data receiving operation in the next paragraph. The output places are TRANSMIT_DATA:PACKET and CALCULATE:PER_EVAL. All packets whether coming during CAP or CTA period are transmitted through TRANSMIT_DATA output place. Whenever a token



(a) Operation of csma-ca protocol during CAP



(b) Operation of CTA period

Figure 5.4: Object diagram of devices: activities during csma and CTA period

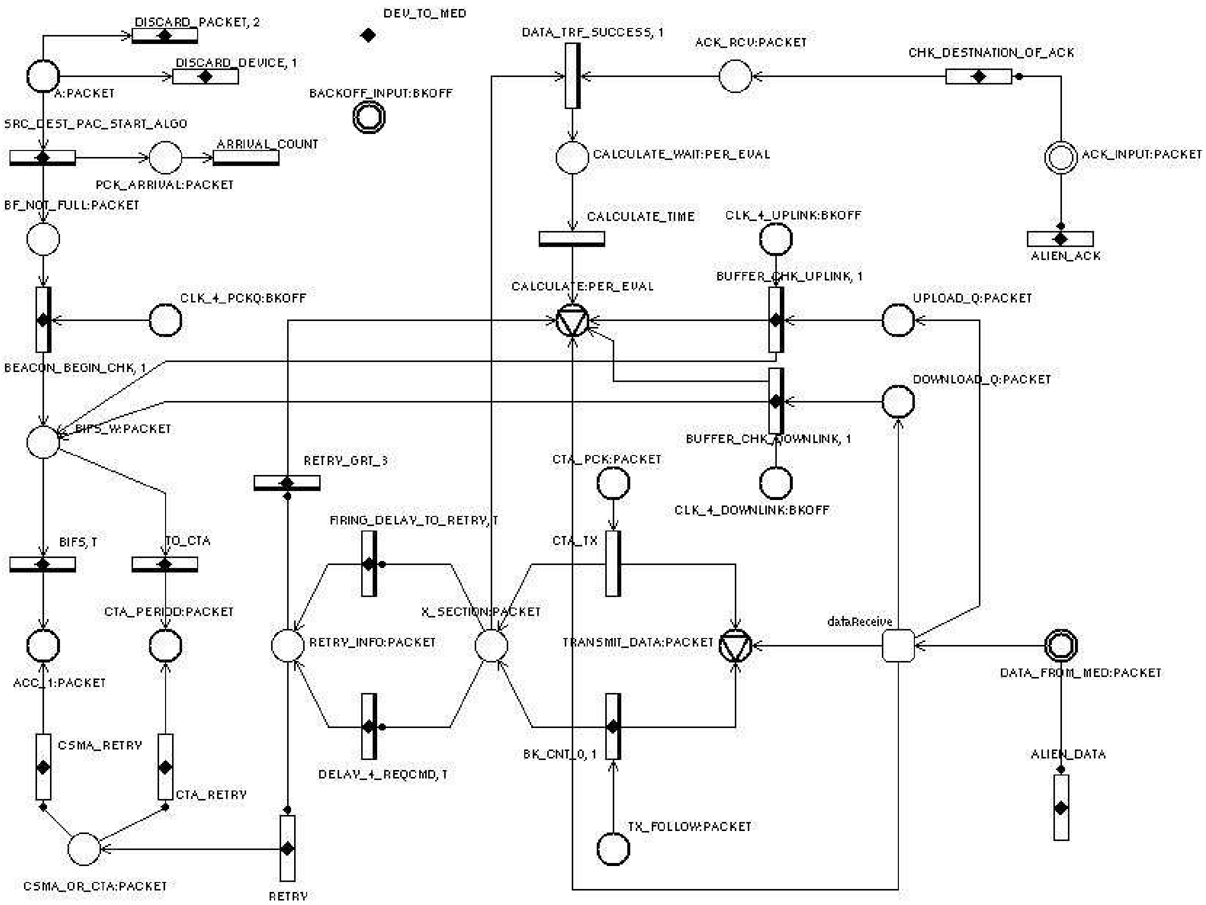


Figure 5.5: Object diagram of devices: data transmission

passes through TRANSMIT_DATA place, the same token is copied in X_SECTION place. The purpose of copying the token is retransmission in case of failure to receive acknowledgement within retransmission timeout period. After receiving an acknowledgement packet the data token is fetched from X_SECTION by triggering the DATA_TRF_SUCCESS transition.

The page dataReceive has one input place and two output places. All three places are connected with RxTx page. The input place DATA_FROM_MED receives packets

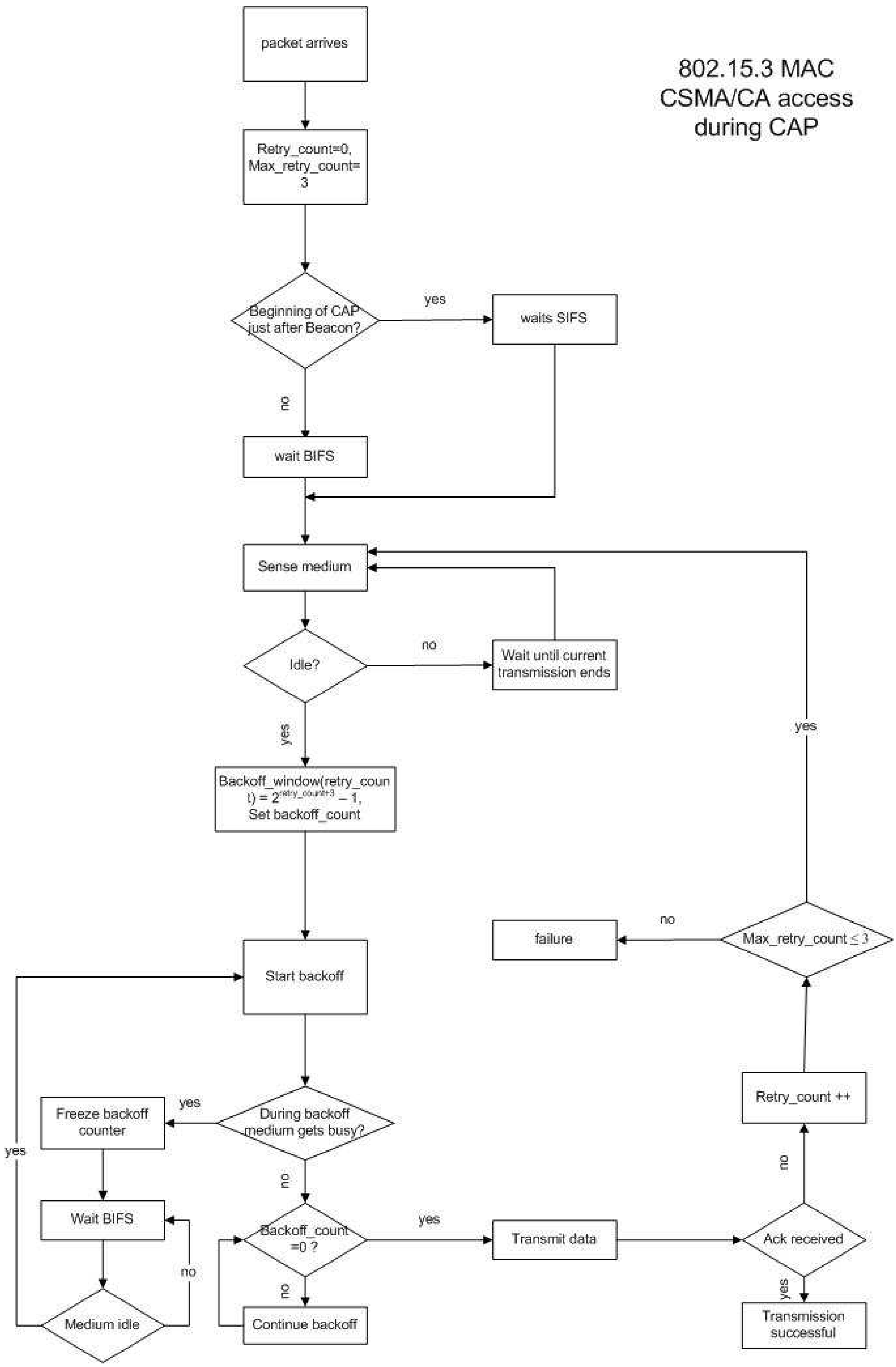


Figure 5.6: Block diagram of csma-ca algorithm

from medium. Here packets are checked for correct destination by CHK_DST transition. After that packets are checked again for bit errors. We randomly generate bit error and discard packets accordingly. We used a uniform random number generator for producing errors within a data packet. After successfully receiving a packet an acknowledgement packet is sent through TRANSMIT_DATA place by NO_FWD transition. In this page we also implement the packet forwarding by bridge device. We can see in Figure 5.7 that from place SELECT_DIRECTION a packet can go to either parent piconet or child piconet. The places UPLOAD_Q and DOWNLOAD_Q represents the queues for uplink and downlink traffic respectively.

5.4 Model Validation

We determined the transient period by moving average method taking measurement of throughput. We run 5 replications each with length of 20 seconds and 4000 observations. The gap between each observation is 5000 μ seconds. The observation numbers in X axis of Figure 5.8 represents subsequent 50th observation value starting from the first one. In each replication we changed the seed for random number generator for packet arrival. We measured the throughput of downlink traffic at each observation point. Then we applied moving average method [10] on the collected data with different window (W) parameters and found our transient period in between observation number 1100 and 1200. Therefore, we choose our transient period till the 1200 observation point, which is 6 seconds. In Figure 5.8 we can see the chart of moving averages with different window sizes.

To determine the run length of our simulation we performed another experiment

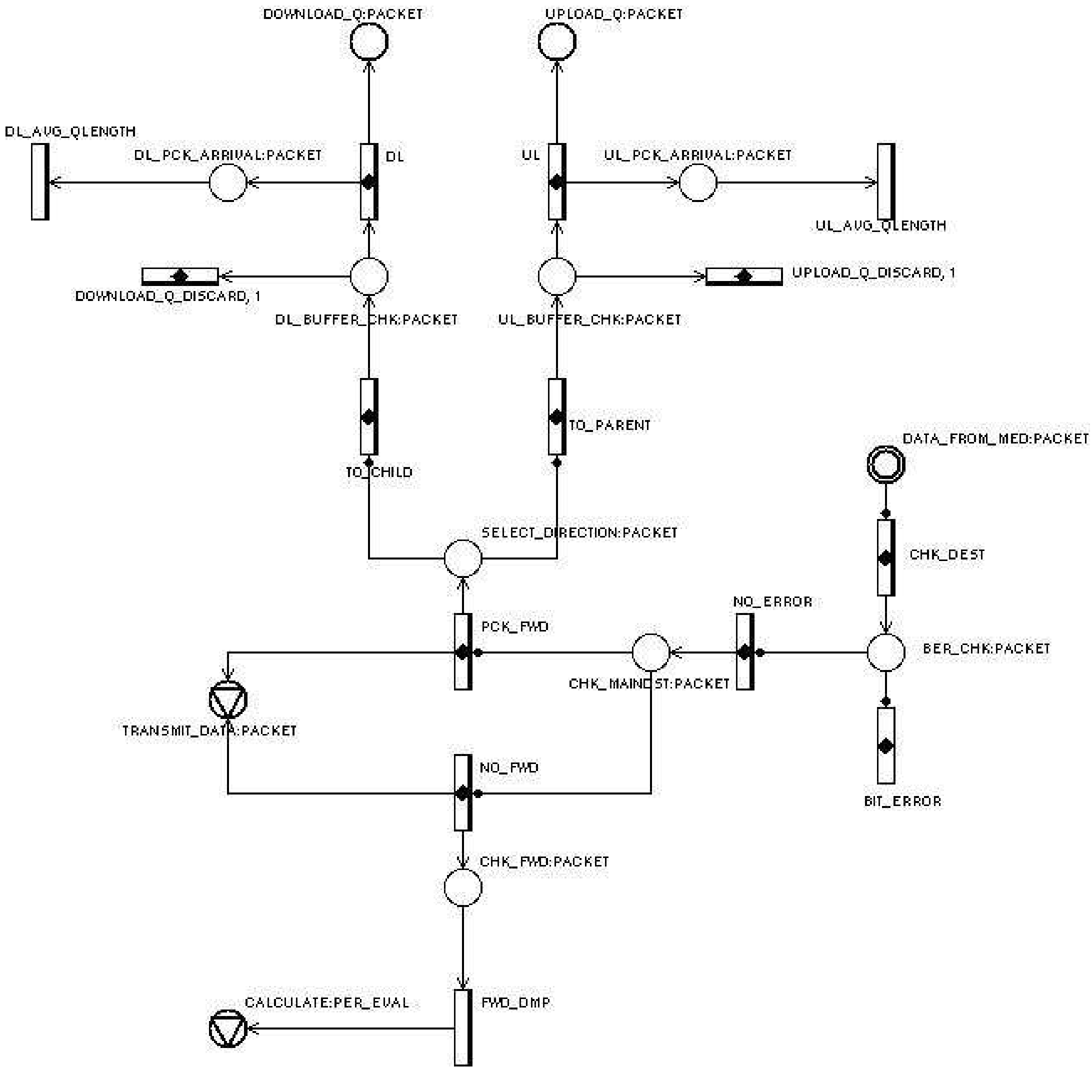


Figure 5.7: Object diagram of devices: packet receiving and bridge operation

Measurement	**Throughput**	**Blocking probability**	**Average queue size**	**Bridging delay**
Mean	0.044	0.061	42.8	765953
90%CI	0.040, 0.049	0.016, 0.107	26.3, 59.3	471156, 1060749

Table 5.1: Sample mean and confidence interval of measurement parameters.

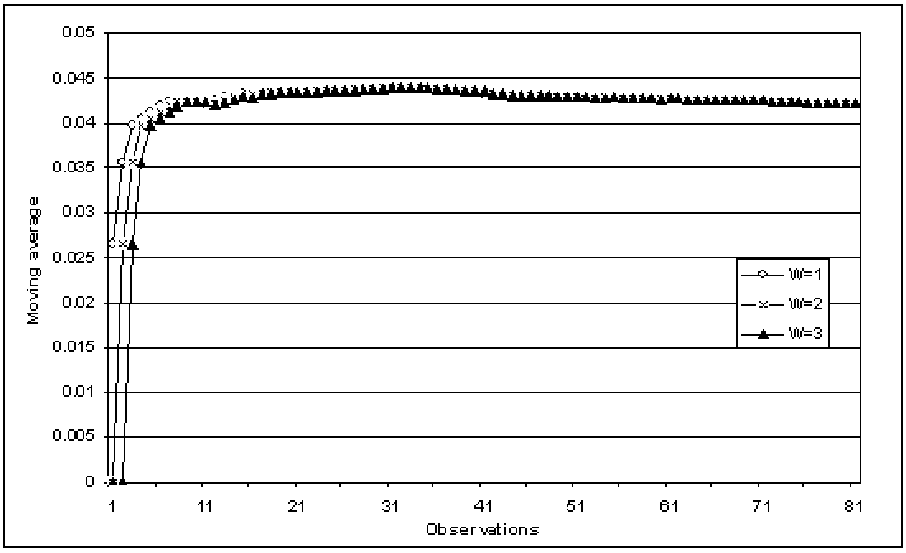


Figure 5.8: Transient interval determination by moving average method

of 10 replications each with a different seed value. We performed this experiment by fixing all the parameters and took the measurement after the transient period. We used probability of traffic locality 0.75 and arrival rate of 50 packets/second. We assigned 5 CTAs for downlink traffic, 3 CTAs for uplink traffic and 4 CTAs for each device. We checked the model with the above parameter values and varying the seed values to validate simulation results. Since our basic model is same, we do not have to check every configuration with different parameter values. We took measurement of throughput, blocking probability of packets, average queue size, and bridging delay of downlink queue. We run the simulation with different run lengths and found out the length where the means of our measured parameters fall within 90% confidence interval. If sample mean is x for m observations and s is the sample standard deviation, then the 90% confidence interval for the population mean is given by $(x - z_{1-\alpha/2} s/\sqrt{n}, x + z_{1-\alpha/2} s/\sqrt{n})$ [10] where $z_{1-\alpha/2}$ is the $(1-\alpha/2)$-quantile of a unit normal variate. Applying the above formula on our simulation outputs we obtained our results with a run length of 10 seconds where the width of 90% confidence interval for sample means are within 5% of population mean. In Table 5.1 we show the results of our experiments.

Chapter 6

Non-adaptive bandwidth allocation

Non-adaptive bandwidth allocation is simple approach and does not take into account traffic load. In this static approach the PNC allocates constant amount of bandwidth to the bridge device in each superframe. The amount of bandwidth is determined by number of CTAs. The purpose of non-adaptive CTA allocation is to measure bridge capability in terms of throughput and blocking probability of packets. We observe how much delay packets experience with fixed number of CTA allocation in terms of bridging delay. The hypothesis of our experiments is that bridge device performance improves with the increase of bandwidth allocation. The results of our experiments prove that our hypothesis is true. In the following sections first we will describe the simulation environment, and then we will show simulation results and discuss the results.

6.1 Experimental environment

We performed three experiments each with different amount of allocated bandwidth (CTAs) for bridge down link operation. We measured throughput, packet blocking probability, average queue size, and bridge delay of packets of bridge down link queue varying packet arrival rate and traffic locality probability. We calculated throughput as the ratio of successfully transmitted data to total offered load over the simulation period. The bridging delay of packets includes packet resident time within the bridge device plus the time it takes to reach its destination. We used traffic locality probability to generate local traffic that does not go through bridge device, within parent piconet or child piconet.

The variable parameters of our experiments are packet arrival rate and traffic locality probability. We have chosen to vary these two parameters because the bridge performance depends on traffic load. Packet arrival rate and traffic locality probability determines the traffic load. We designed our experiments as such where we varied the arrival rate from 10 to 50 packets/s with increasing steps of 10 and traffic locality probability from 0.65 to 0.9 increasing in steps of 0.05. Number of allocated CTAs for each device is four. For bridge operation we have allocated three CTAs for uplink. Since we are interested in bridge downlink performance, we performed three experiments each with different amount of allocated bandwidth for downlink operation. We increased bandwidth allocation for each subsequent experiment and assigned bandwidth in the amount of 3, 5 and 7 CTAs respectively.

We performed experiments to observe the effect of small amount of bandwidth allocation (e.g. 1 and 2 CTAs) by varying number of CTAs and packet arrival rate. We varied the number of CTA allocation from 1 CTA to 5 CTAs. The results showed that small amount of bandwidth (e.g. 1 and 2 CTAs) allocation causes low performance of bridge operation. The performance of bridge device increases with the increase of CTA allocation. Therefore, first we have performed non adaptive bandwidth allocation experiment with 3 CTAs. Then we did our experiments with 5 and 7 CTAs. We decided to limit the number of allocated CTAs to 7 because the maximum limit of superframe length defined in IEEE 802.15.3 standard exceeds if we assign more than 7 CTAs.

The total run time of our simulation was 16 seconds including the warm up period of 6 seconds and a 10 seconds measurement period. We have used packet size of 1200 bytes and data rate of 11 Mbps. We also used bit error rate (BER) of 0.0001 to simulate wireless transmission error. We list all parameters of our experiments in Table 6.1.

6.2 Results and analysis

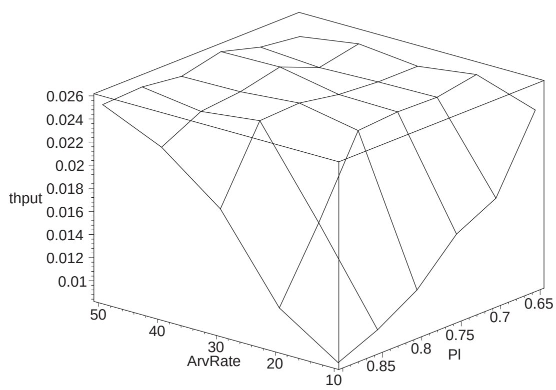
Figure 6.1 shows the simulation results with the number of allocated CTAs 3. We see in Figure 6.1(a) the throughput increases with the increase of packet arrival rate and decrease of locality probability (P_L), but it starts to flatten from the point of about arrival rate 30 packets/s and locality probability 0.8. We expect increase in throughput with the increase of traffic volume. From Figure 6.1(a) we realize that due to insufficient channel time the throughput graph does not show a smooth increase from low traffic load to high traffic load. High blocking probability and larger average queue size explains the reason of low throughput. High blocking probability indicates

Name	**Type**	**Value**	**Increasing Step**
Packet arrival rate	Variable	10 to 50 packets/s	10
Traffic locality probability	Variable	0.65 to 0.9	0.05
Bandwidth for member devices	Fixed	4 CTAs	-
Bandwidth for bridge uplink	Fixed	3 CTAs	-
Bandwidth for bridge downlink	Fixed	3/5/7 CTAs	-
Packet size	Fixed	1200 bytes	-
Data rate	Fixed	11 Mbps	-
BER	Fixed	0.0001	-

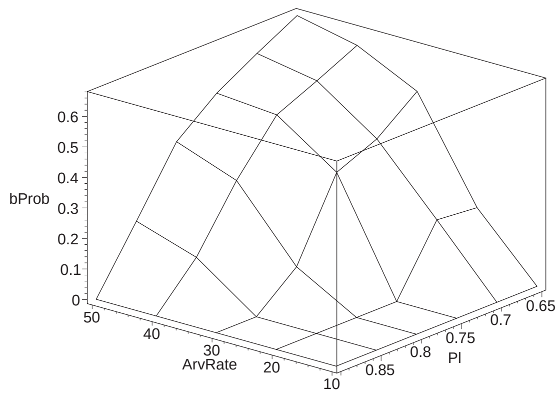
Table 6.1: List of parameters for the experimetns of non adaptive bandwidth allocation.

that packet loss is high at the bridge device. Packet loss happens due to insufficient amount of channel time (CTAs) for bridge operation. In Figure 6.1(d) we see longer bridge delay due to the same reason. We also observe an increasing jump in the delay graph in Figure 6.1(d) in spite of low arrival rate and high locality probability. High locality probability means bridge device will experience less traffic. We observe from all the graphs in Figure 6.1 that low arrival rate and small nonlocal traffic locality probability results in low throughput, blocking probability, average queue size and bridging delay. The low values of these measurement parameters are usual at this point. The performance of bridge degrades with the increase of arrival rate and traffic locality probability and this allows us to think of further improvement of performance.

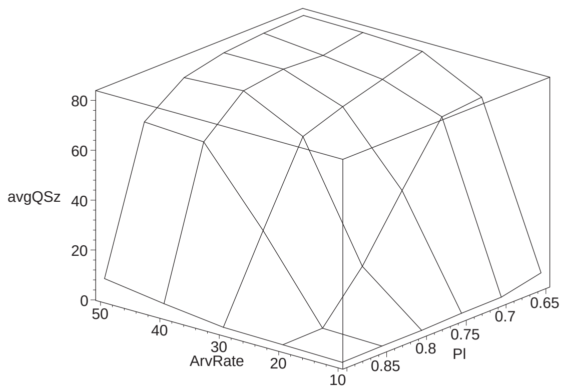
In Figure 6.2 we see throughput, packet blocking probability, average queue size,



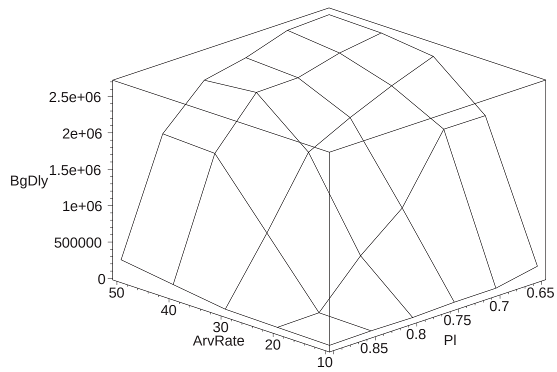
(a) Throughput



(b) Blocking probability



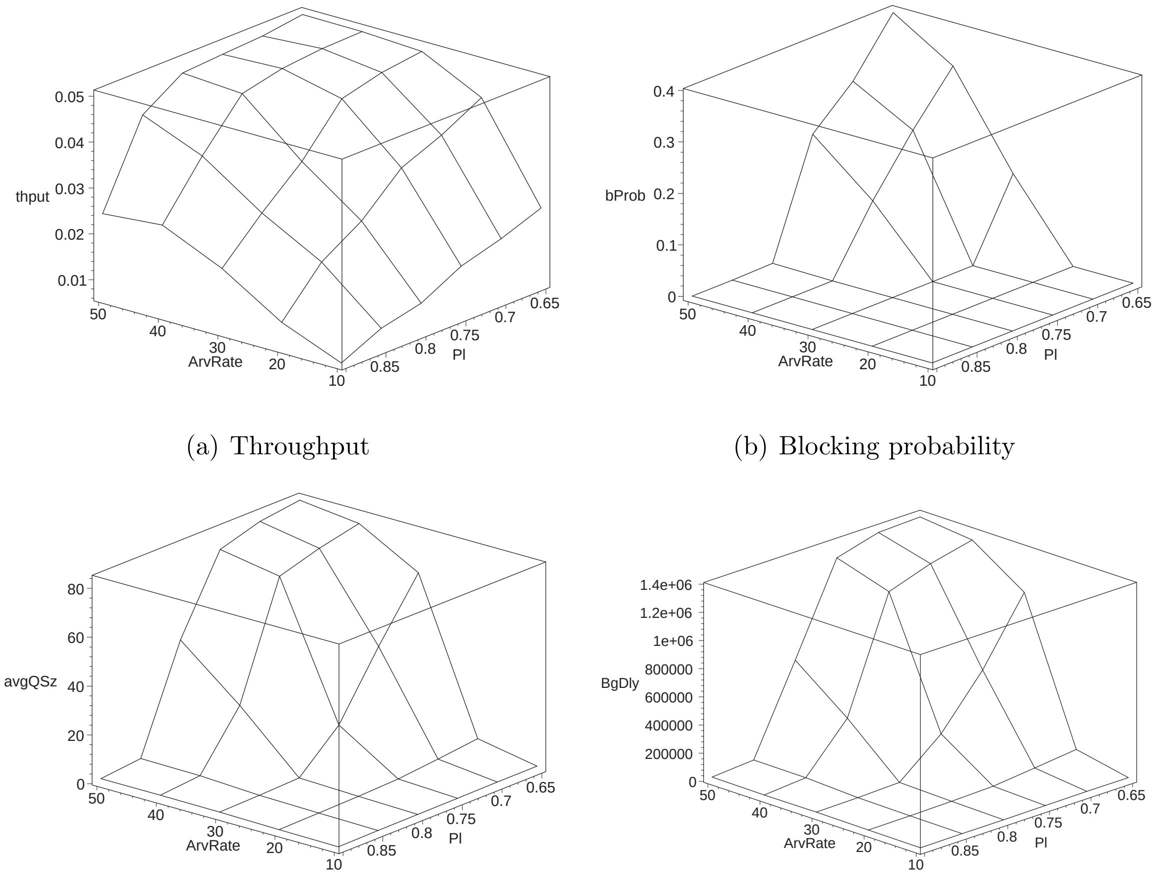
(c) Average queue size



(d) bridge delay of packets

Figure 6.1: Throughput, blocking probability, average queue size and bridging delay when locality probability and packet arrival rate are varied. Allocated CTAs for downlink operation are 3.

and bridge delay of packets with allocation of 5 CTAs for bridge operation. Here we see our expected performance improvement. Still throughput flattens with high volume of traffic. The blocking probability of packets is also high and it has an increasing pattern. The reason of increasing pattern of the blocking probability is the increase of packet arrival rate. At some point of time there might not be enough traffic compared to allocated CTAs and the channel time may remain unused. Therefore,



(a) Throughput

(b) Blocking probability

(c) Average queue size

(d) bridge delay of packets

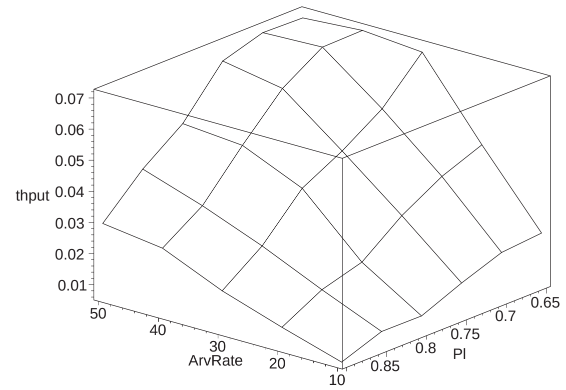
Figure 6.2: Throughput, blocking probability, average queue size and bridging delay when locality probability and packet arrival rate are varied. Allocated CTAs for downlink operation are 5.

fixed CTA allocation is not a good choice as we want maximum utilization of our channel time. High packet blocking probability indicates we have high packet loss rate. The bridge delay is less than the previous experiment with 3 CTAs. If we can increase CTA allocation according to traffic demand, we will achieve better results than fixed or random CTA allocation.

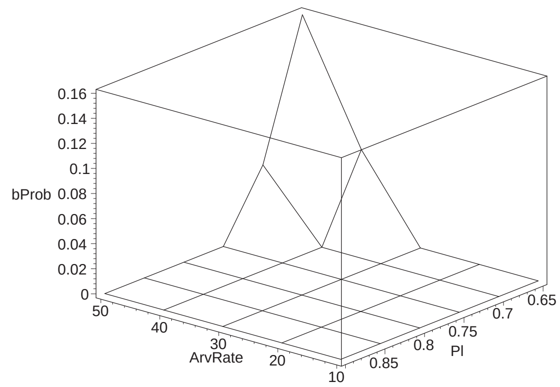
In Figure 6.3 we show the simulation results with 7 CTAs for bridge operation.

All four graphs for throughput, packet blocking probability, average queue size, and bridge delay show significant improvement over previous simulation results. This means increasing the number of CTA allocation definitely improves bridge performance. We observe very low blocking probability of packets and bridge delay, but increasing the number of CTA allocation results in longer superframe. Due to long superframe the local packets experience longer waiting time to get access to channel time. Long superframe also violates the superframe length specification of IEEE 802.15.3 standard. Moreover, the improvement in average queue size is not significant compared to throughput, bridge delay and blocking probability. From all the graphs in Figure 6.1, Figure 6.2 and Figure 6.3 we learn that we can improve bridge performance by increasing CTAs and should pay carefull attention in the design of CTA allocation algorithm.

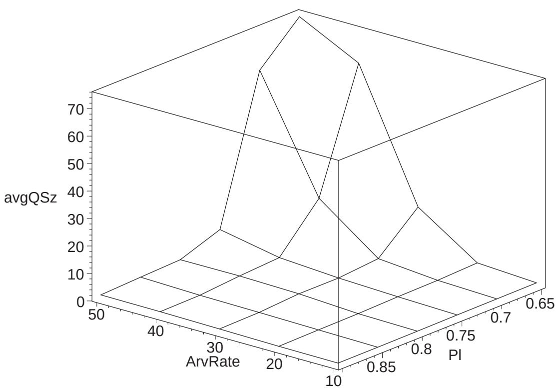
From Figure 6.2 and Figure 6.3 we can see all the graphs except the throughput graph are flat at the bottom and gradually the graphs goes up. This flat areas in the graphs means that there are unused bandwidth when we assign fixed 5 or 7 CTAs for low packet arrival rate and small amount of traffic. The results of non-adaptive bandwidth allocation experiments inspire us for the design of an adaptive bandwidth allocation algorithm, which we describe in the next chapter with experiment results and analysis.



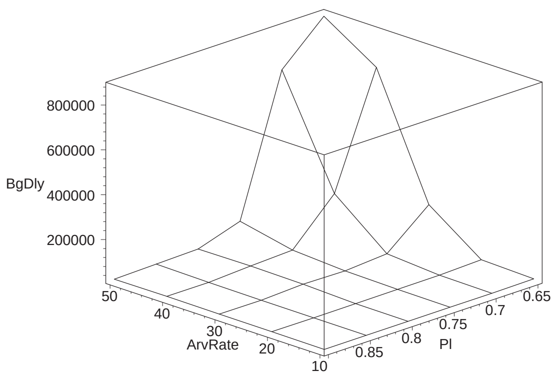
(a) Throughput



(b) Blocking probability



(c) Average queue size



(d) bridge delay of packets

Figure 6.3: Throughput, blocking probability, average queue size and bridging delay when locality probability and packet arrival rate are varied. Allocated CTAs for downlink operation are 7.

Chapter 7

Adaptive bandwidth allocation to bridge device

Bandwidth allocation for bridge operation should account the traffic condition at the bridge device. Performance analysis of non-adaptive bandwidth allocation directs us to further research and explore the option of dynamic bandwidth allocation for bridge device operations. We designed the adaptive bandwidth allocation algorithm as such where the bandwidth is allocated in terms of CTAs based on previous history of traffic condition. The analysis of experiment results from Chapter 6 gives us expected result for three CTAs for small amount of traffic. Therefore, we initially assigned three CTAs to the bridge device in our adaptive bandwidth allocation algorithm and gradually increased bandwidth allocation based on traffic status. In the following section first we describe our adaptive bandwidth allocation algorithm.Then we show the results of our experiments and analyse them.

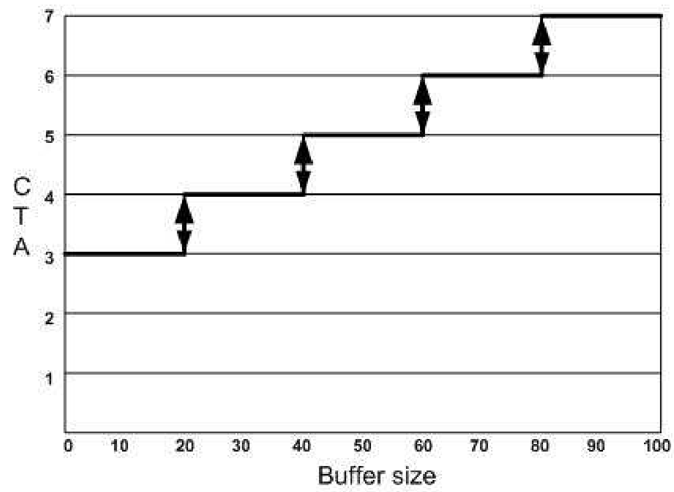


Figure 7.1: Queue size threshold increase-decrease diagram

7.1 Algorithm

The adaptive bandwidth allocation algorithm (Algorithm 1) runs at the PNC of parent piconet to adaptively allocate CTAs for downlink bridge operation. The bridge device sends average queue size to PNC during CAP. This average queue size is calculated over the simulation period in each superframe. Initially we assign three CTAs to bridge. Then based on previous queue sizes of bridge device we calculate an estimated queue size using the exponential weighted moving average (EWMA) method [12] for prediction of number of CTAs for bridge device. Eventually we allocate bandwidth to bridge device based on the estimated queue size. The EWMA method gives us exponential smoothing on queue size estimation using the following formula

$Q_e[n+1] = \alpha * Q_a[n] + (1-\alpha) * Q_e[n]$,

where Q_e is estimated queue size, Q_a is average queue size and α is called the smoothing constant or EWMA constant of EWMA method. The estimation of EWMA method is based on the past values of our considered parameter (in our case average queue size). We can control the effect of past values on our estimation by EWMA constant. The value of α is $0 < \alpha < 1$. Higher α value means our estimation will be influenced more by the past values.

The buffer size for down link queue is 100 and we varied number of CTAs from 3 to 7. The variation of CTA allocation has five steps from 3 to 7. Based on buffer size of 100 and 5 steps of CTA allocation we determined the threshold points on buffer size to increase or decrease CTA allocation. In Figure 7.1 we can see the threshold points at which up arrow and down arrow respectively indicate increase and decrease of CTA allocation. The vertical axis of the diagram in Figure 7.1 represents number of CTAs and the horizontal axis represents the buffer size and also shows the five ranges. Dividing the buffer size with number of steps of CTA change, we determine the length of ranges within which the estimated queue size falls. We calculate the ranges in our algorithm using the following formula: $(i-1)*\beta+1, i*\beta$, where the value of i goes from 1 to 5 and β is the length of ranges with a value of 20. For example if estimated value of queue size falls within the range of 1 and 20 (inclusive), we assign 3 CTAs. For each subsequent range we assign 4, 5, 6, and 7 CTAs respectively.

γ is the adjustment variable in Algorithm 1. It does not have any impact on the logical part of our algorithm. We add the γ value with an index variable to calculate the proper amount of CTA for bridge device. We use $qSizeIndex$ to determine the upper bound of a threshold range within which estimated queue size falls. In line 5 of Algorithm 1 we see how $qSizeIndex$ is used to find out the threshold value. The value of *qSizeIndex* starts from 1 and subsequently increases by one until it reaches the desired threshold value. Based on this *qSizeIndex* we determine the CTA allocation. Since we start CTA allocation from 3 and the index value starts from 1, we adjust the difference between these two values by γ. We obtained $\gamma = 2$ from this difference.

Algorithm 1 Adaptive CTA allocation for down link

CTAforDLqueue=y
averageDLqSize = sent at previous superframe from bridge device
estimatedQsize=(1-EWMAconstant)*estimatedQsize+EWMAconstant *averageDLqSize)
qSizeIndex=1
while estimatedQsize> β * qSizeIndex **do**
    qSizeIndex++;
end while
CTAforDLqueue=qSizeIndex+ γ

7.2 Experimental environment

In our experiments we fixed the number of CTAs for up link to three and adaptively determined the number of CTAs to be allocated to the bridge device for down link operation. We decided to fix the up-link CTAs because there is less traffic, which comes from child piconet destined for parent piconet than the opposite direction. On the other hand more traffic comes from parent piconet destined for child piconet due to higher number of devices in the parent piconet. We have measured throughput, packet blocking probability, and average queue size and bridge delay of packets of bridge

Name	**Type**	**Value**	**Increasing Step**
Packet arrival rate	Variable	10 to 50 packets/s	10
Traffic locality probability	Variable	0.65 to 0.9	0.05
Bandwidth for member devices	Fixed	4 CTAs	-
Bandwidth for bridge uplink	Fixed	3 CTAs	-
Bandwidth for bridge downlink	dynamic	3/4/5/6/7 CTAs	1
Packet size	Fixed	1200 bytes	-
Data rate	Fixed	11 Mbps	-
BER	Fixed	0.0001	-
EWMA constant	Fixed	0.3/0.5/0.7	-

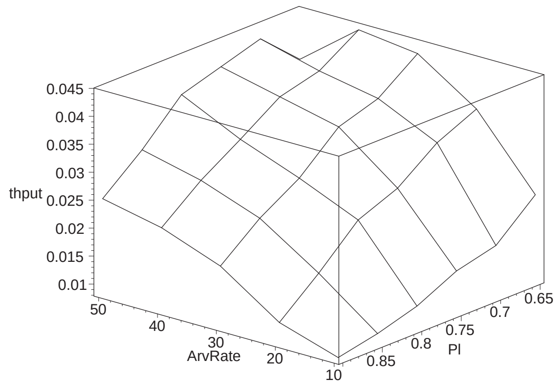
Table 7.1: List of parameters for the experiments of adaptive bandwidth allocation.

down link queue as function of packet arrival rate and traffic locality probability. We varied packet arrival rate from 10 to 50 packets/s and traffic locality probability from 0.65 to 0.9 with an increasing step of 0.05. We performed three experiments with the same configuration mentioned above for three different EWMA constant values. We ran our simulations for 6 seconds as warm up period to reach the steady state and took the measurement over simulation period of 10 seconds. We list the parameter names and values in Table 7.1.

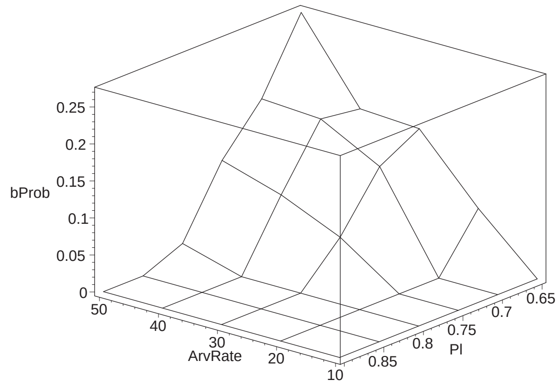
7.3 Results and analysis

In Figure 7.2 we show the results of our experiments for adaptive CTA allocation to bridge device with EWMA constant 0.3. Throughput of down link queue increases with the increase of packet arrival rate and decrease of traffic locality probability. We observe throughput tends to drop at highest point. This indicates that the system has reached saturation point with adaptive CTA allocation. Now, if we look at packet blocking probability graph in Figure 7.2(b), we observe an increasing pattern with the increase of down link traffic volume at bridge device. Gradually the average queue size flattens. Increasing packet blocking probability and flatten average queue size explain us the drop of throughput for high traffic load. Throughput reaches its maximum limit because of buffer overflow, not because of limitation of system capacity. Higher queue size creates longer delay and we see that in Figure 7.2(d) from bridge delay graph. If we compare the results of non-adaptive CTA allocation with five fixed CTAs in Figure 6.2(d) and adaptive CTA allocation in Figure 7.2(d), we find the difference between delay values are very high. Therefore, we have to control adaptive CTA allocation to decrease delay.

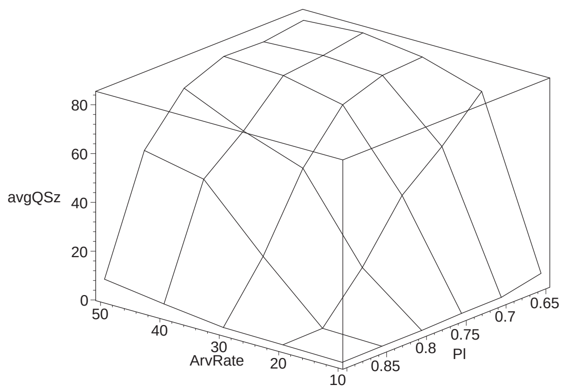
In Figure 7.3 we show the results of our experiments with EWMA constant value 0.5. We see all the graphs have same pattern as the graphs with EWMA constant value 0.3 in Figure 7.2. Although packet blocking probability decreases slightly, average queue size and bridge delay do not have any significant change in their values. Average queue size and bridge delay both flattens after a sudden straight increase. This sudden increase happens due to the increasing factor, which we use to determine the number of CTA allocation. We increase or decrease number of CTA allocation with the change of β (threshold increasing factor). In some cases we have drastic drop in CTA allocation with very small decrease in queue size. For example if the estimated queue size decreases to 59 from 61, number of CTA allocation drops from 6 to 5.



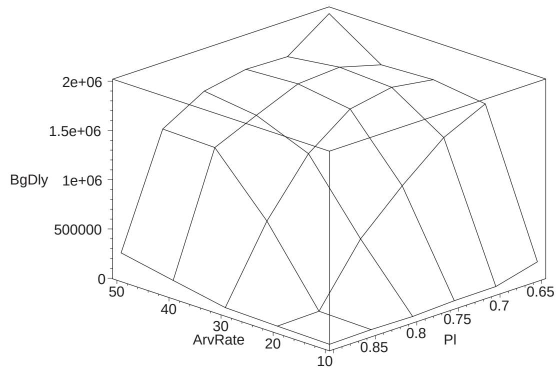
(a) Throughput



(b) Blocking probability



(c) Average queue size

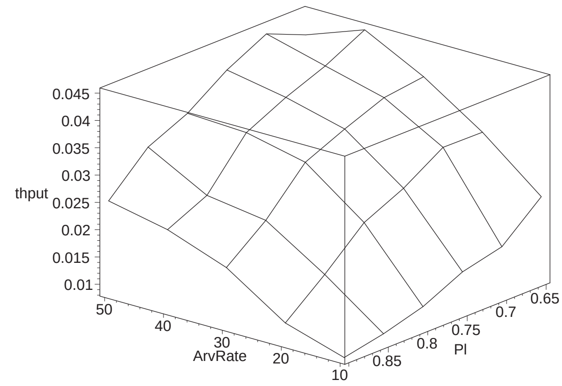


(d) bridge delay of packets

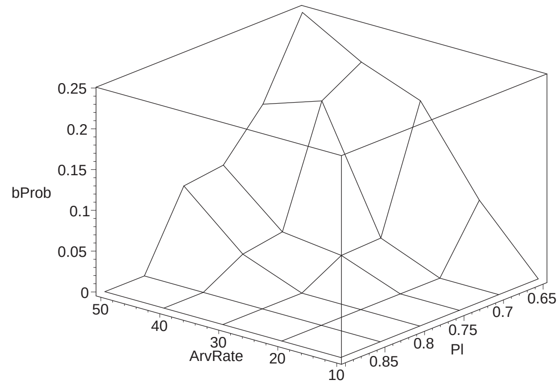
Figure 7.2: Adaptive bandwidth allocation: throughput, blocking probability, average queue size and bridging delay when locality probability and packet arrival rate are varied. EWMA constant is set to 0.3.

This drop in CTA allocation is a big change compare to the queue size decrease from 61 to 59. We need to prevent this sudden drop in CTA allocation to achieve better network performance. Findings of this analysis dictate us for further improvement of our adaptive CTA allocation algorithm (Algorithm 1).

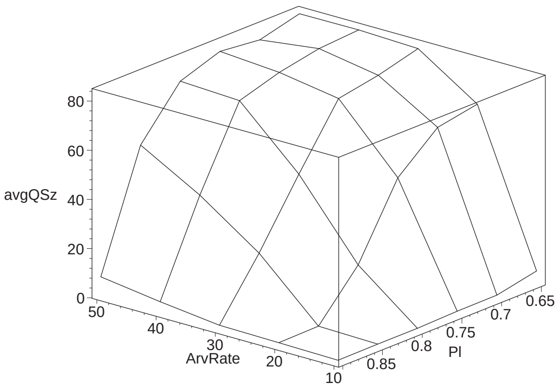
In Figure 7.4 we show throughput, packet blocking probability, average queue size, and bridge delay of packets with EWMA value 0.7. We observe same patterns in all



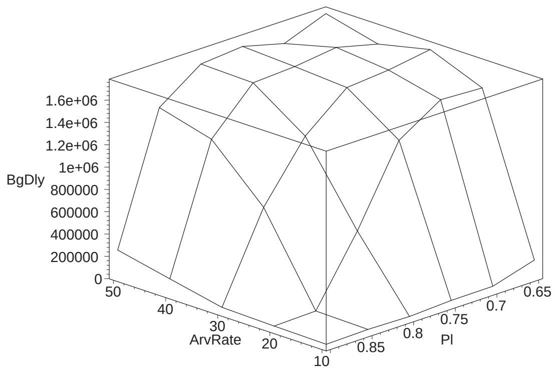
(a) Throughput



(b) Blocking probability



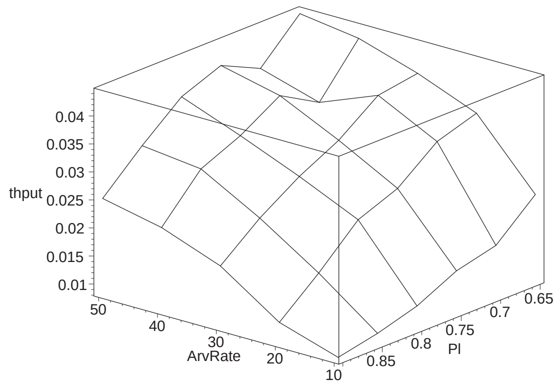
(c) Average queue size



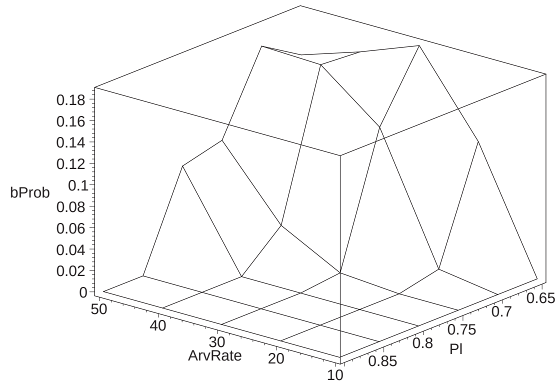
(d) bridge delay of packets

Figure 7.3: Adaptive bandwidth allocation: throughput, blocking probability, average queue size and bridging delay when locality probability and packet arrival rate are varied. EWMA constant is set to 0.5.

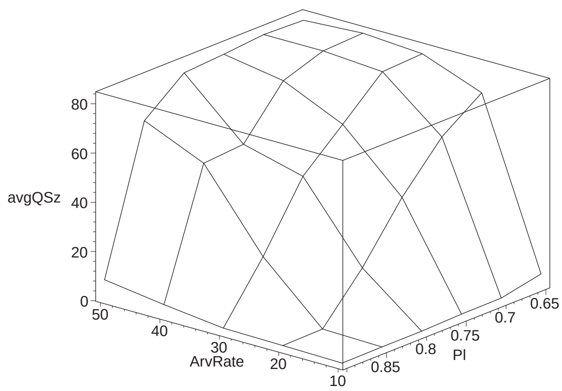
four graphs as before. We have some improvements in packet blocking probability, but the bridge delay is still high. In all three experiments we observe that the average queue size and bridge delay graphs have same pattern as we expect. Analysing the results of all three experiments we can say that the EWMA constant does not have significant impact on our algorithm. However, a medium value of EWMA constant (in our case 0.5) shows better performance.



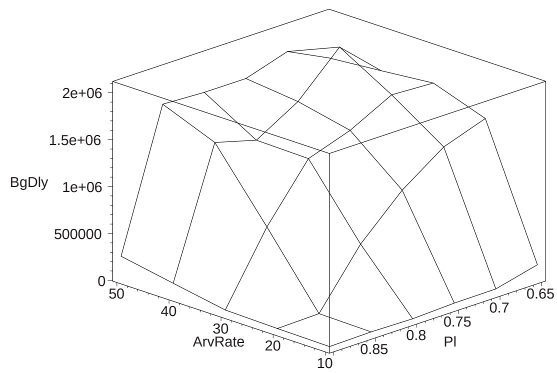
(a) Throughput



(b) Blocking probability



(c) Average queue size



(d) bridge delay of packets

Figure 7.4: Adaptive bandwidth allocation: throughput, blocking probability, average queue size and bridging delay when locality probability and packet arrival rate are varied. EWMA constant is set to 0.7.

Chapter 8

Adaptive bandwidth allocation with hysteresis

The simulation results of adaptive bandwidth allocation algorithm in section 7.3 indicates that there is scope for performance improvement; especially the packets have long bridging delay. The average queue size and blocking probability are also high. From the analysis of the results in Chapter 7 we found that slight change (increase or decrease) of estimated queue size arround the threshold value affects the bandwidth allocation. The minimum change in bandwidth allocation i.e. one CTA increase or decrease has significant impact on the bridge performance. Therefore, we extended our adaptive bandwidth allocation algorithm by introducing hysteresis control over threshold of estimated queue size. Basically we try to keep the CTA allocation unchanged once it goes up by redefining the threshold value. In other words, we want to prevent change of bandwidth allocation due to small variation of traffic.

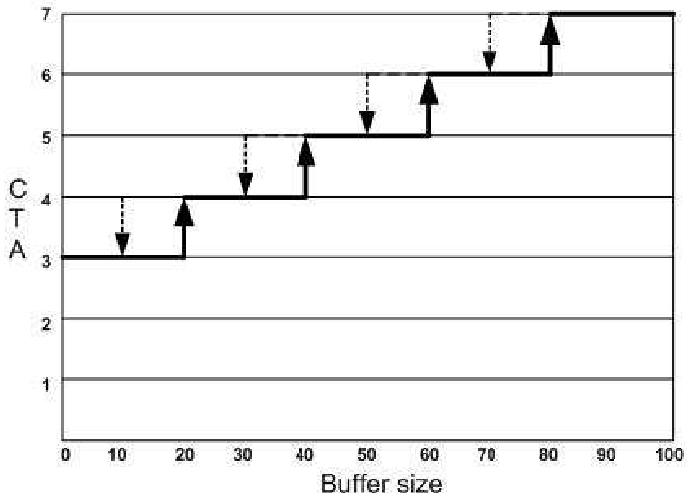


Figure 8.1: Queue size threshold increase-decrease diagram

Hysteresis is a nonlinear phenomenon, which establishes relationship between 'variable input' and 'variable output' of a system [4]. In Figure 8.1 we draw the hysteresis diagram of our extended bandwidth allocation algorithm. We observe cyclic increase and decrease in queue size threshold which in turn causes cyclic increase or decrease in CTA allocation. In our hysteresis system estimated queue size is control parameter and number of CTA allocation is output value which we want to control. In the following sections we first discuss our extended algorithm and then show our simulation results with analysis.

8.1 Algorithm

Algorithm 2 shows our extended version of adaptive CTA allocation algorithm with hysteresis threshold control. First we determine the number of CTAs to be

Algorithm 2 Adaptive CTA allocation with hysteresis

CTAforDLqueue=3

averageDLqSize = sent at previousSuperFrame from bridge device

estimatedQsize = (1 - EWMAconstant) $*$ estimatedQsize + EWMAconstant $*$ averageDLqSize

qSizeIndex=1

while estimatedQsize> $\beta *$qSizeIndex **do**

  qSizeIndex++;

end while

CTAforDLqueue=qSizeIndex+ γ

if estimatedQsize<averageDLqSize **then**

  threshold=$(\beta * queueSizeIndex) - \delta$

  if estimatedQsize>threshold **then**

    CTAforDLqueue+1

  end if

end if

allocated using Algorithm 1. After that we check whether our estimated queue size is smaller than the previous value of average queue size. If the estimated queue size is smaller than average queue size then we determine a new threshold value. The new threshold value is the middle point of the range where estimated queue size falls. For example for the range of 1 to 20 the mid point is 10. In Algorithm 2 δ has a value of 10. We find the upper value of our threshold range and then reduce it by δ to obtain the mid point (new threshold value) as follows: $i * \beta - \delta$. Further, we compare our estimated queue size again with this new threshold value. If the estimated queue size is larger than the threshold value then we increase CTA allocation by one, otherwise keep number of CTA allocation unchanged. In Figure 8.1 we show the hysteresis behaviour of our algorithm as increase or decrease of CTA allocation with the change of threshold value. The diagram shows us the number of CTA allocation on the basis of queue status and the increase of CTA allocation with the increase of queue size. We can also see when the queue size decreases, we do not change the CTA allocation until the queue size comes down to the middle point of a range.

8.2 Results and analysis

The simulation environment for adaptive bandwidth allocation with hysteresis is same as adaptive CTA allocation experiments. We calculated throughput, packet blocking probability, average queue size, and bridge delay as a function of packet arrival rate and traffic locality probability. The EWMA constant for three different experiments are 0.3, 0.5 and 0.7. Simulation time and warm up periods are 10 seconds and 6 seconds respectively. The list of all the parameters are given in Table 8.1

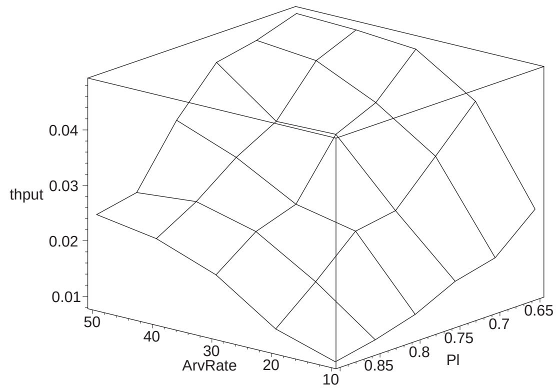
In Figure 8.2 we show the simulation results of algorithm 2 with EWMA 0.3. We measured throughput, packet blocking probability, average queue size, and bridge delay of down link queue of bridge device. All four measurements we took varying packet arrival rate and traffic locality probability. Throughput increases smoothly with the increase of traffic as we expect. We have low packet blocking probability compare to the results of non-adaptive bandwidth allocation and adaptive bandwidth allocation algorithm. We observe slow increasing pattern in average queue size (Fig-

Name	**Type**	**Value**	**Increasing Step**
Packet arrival rate	Variable	10 to 50 packets/s	10
Traffic locality probability	Variable	0.65 to 0.9	0.05
Bandwidth for member devices	Fixed	4 CTAs	-
Bandwidth for bridge uplink	Fixed	3 CTAs	-
Bandwidth for bridge downlink	dynamic	3/4/5/6/7 CTAs	1
Packet size	Fixed	1200 bytes	-
Data rate	Fixed	11 Mbps	-
BER	Fixed	0.0001	-
EWMA constant	Fixed	0.3/0.5/0.7	-

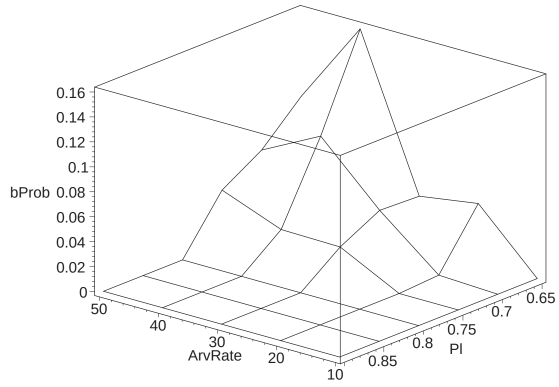
Table 8.1: List of parameters for the experiments of adaptive bandwidth allocation with hysteresis threshold.

ure 8.2(c))and bridge delay (Figure 8.2(d)) graphs compare to sudden increase in adaptive CTA allocation graphs (Figure 7.2(d) and Figure 7.2(c)). The slanted increasing graph indicates that marginalized threshold helps efficient CTA allocation. The last hop delay has significant improvement over non-adaptive and adaptive bandwidth allocation.

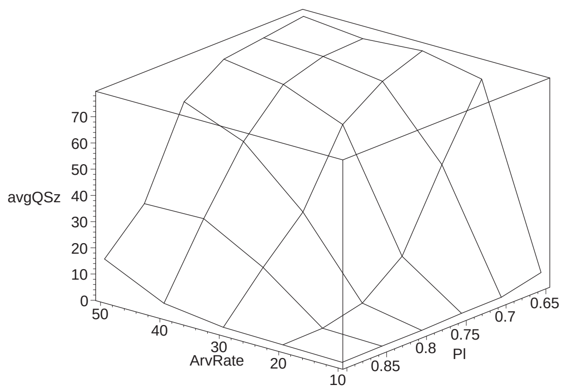
Further we simulated our extended algorithm with EWMA value 0.5. In Figure 8.3 we show the simulation results. We observe better throughput that tends to flatten at maximum value. If we compare this graph with the graph of non-adaptive CTA allocation with a value of five, we see the throughput values are very close. With the increase of packet arrival rate and decrease of traffic locality probability the throughput reaches saturation point with five CTAs for non-adaptive CTA allocation. For hysteresis threshold the throughput graph in Figure 8.3(a) shows a smooth increase



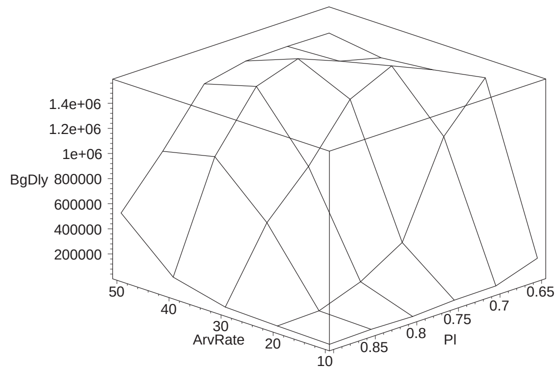
(a) Throughput



(b) Blocking probability



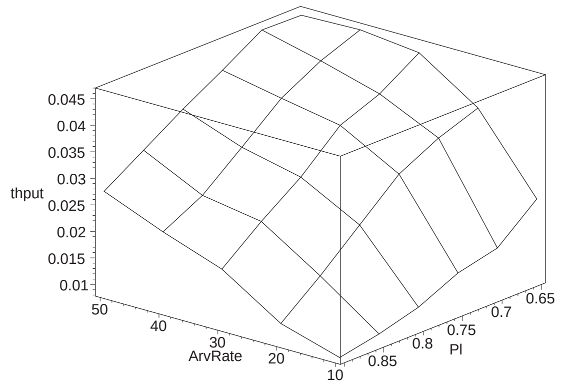
(c) Average queue size



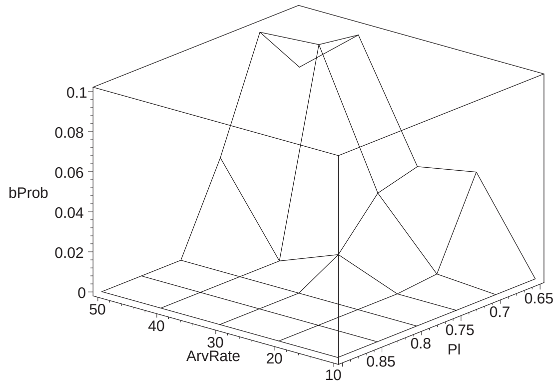
(d) Bridge delay of packets

Figure 8.2: Adaptive bandwidth allocation with hysteresis threshold: throughput, blocking probability, average queue size and bridging delay when locality probability and packet arrival rate are varied. EWMA constant is set to 0.3.

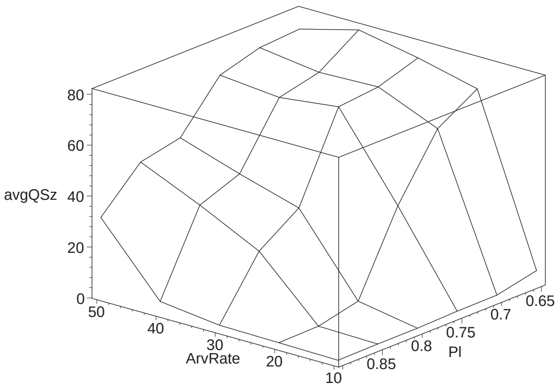
with the increase of traffic and tends to flatten. The average queue size is still large but packet blocking probability and last hop delay has lower value than the results of adaptive simulation. The average queue size flattens with the increase of traffic. Although we observe large average queue size, introduction of hysteresis threshold reduces packet blocking probability and last hop delay. From the above analysis we can say that our extended algorithm increases system utilization and improves the



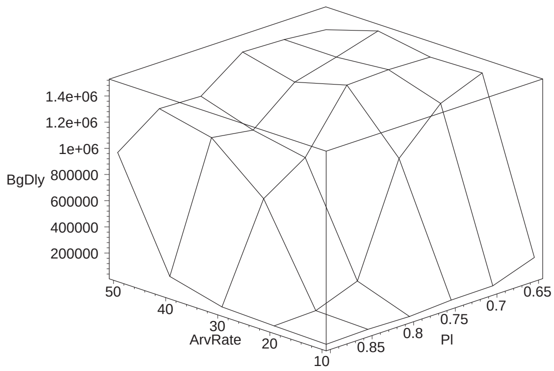
(a) Throughput



(b) Blocking probability



(c) Average queue size

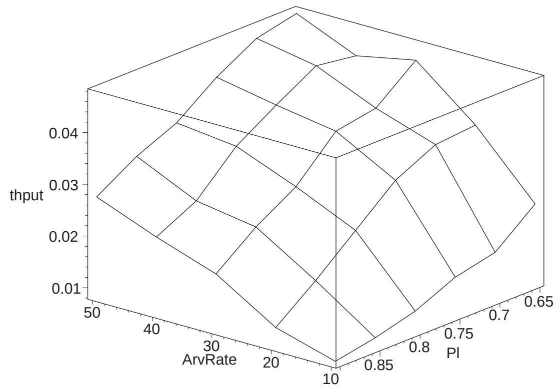


(d) Bridge delay of packets

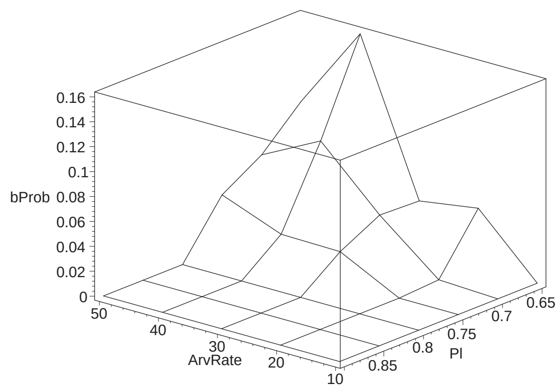
Figure 8.3: Adaptive bandwidth allocation with hysteresis threshold: throughput, blocking probability, average queue size and bridging delay when locality probability and packet arrival rate are varied. EWMA constant is set to 0.5.

over all performance of bridge operation.

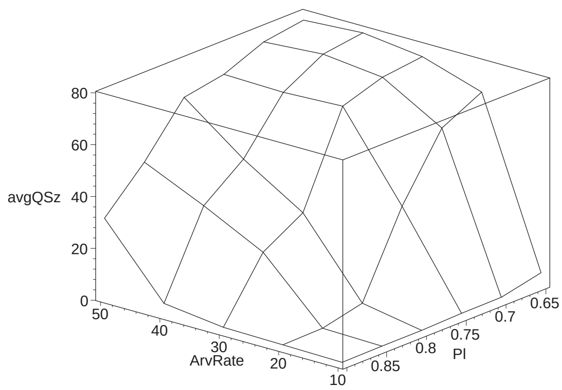
Finally, we simulate the hysteresis threshold algorithm (algorithm 2) with EWMA value 0.7. We observe similar performance improvements, except the bridge delay has little higher value than the results of 0.3 and 0.5. Longer bridge delay with higher weight factor is reasonable because the weight factor will include more history of queue size which has a continuous increasing pattern. As a result our estimated queue size



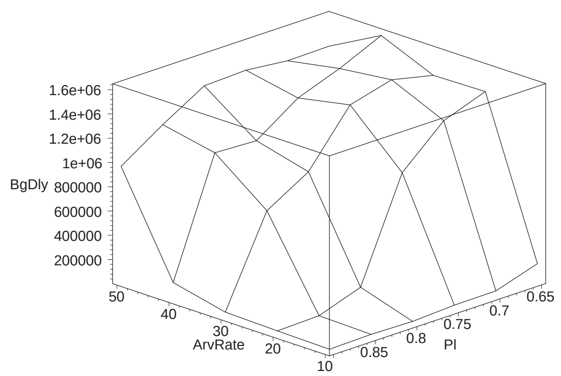
(a) Throughput



(b) Blocking probability



(c) Average queue size



(d) Bridge delay of packets

Figure 8.4: Adaptive bandwidth allocation with hysteresis threshold: throughput, blocking probability, average queue size and bridging delay when locality probability and packet arrival rate are varied. EWMA constant is set to 0.7.

will be larger than the average queue size, hence number of CTA allocation gets less chance of decrease. For large queue size (close to buffer size) maximum number of allowed CTAs are always allocated . In this case the superframe length increases, so is bridge delay. The overall performance shows improvement compared to adaptive CTA allocation results in terms of throughput, packet blocking probability and bridge delay.

Chapter 9

Conclusion

In this thesis, we have evaluated the performance of a personal area network consists of two piconets and built upon IEEE 802.15.3 high data rate WPAN standard. The parent piconet and child piconets are interconnected through a Master-Slave bridge which also acts as the coordinator of the child piconet. The bridge device mainly forwards packets to and forth from parent piconet and child piconet maintaining two queues: uplink queue and downlink queue. We have designed and evaluated the performance of an adaptive bandwidth allocation algorithm for bridge downlink operation. We also introduced threshold hysteresis with our adaptive algorithm which shows improved performance. We have designed and simulated our network model by Artifex simulation tool.

We have shown that an adaptive algorithm for allocating downlink bandwidth to the bridge, utilizing threshold hysteresis, can easily outperform any fixed bandwidth allocation algorithm. First we evaluated the performance of non-adaptive bandwidth allocation to bridge device for downlink operation. The results show that in case of low traffic we have very low blocking probability, smaller queue size and shorter bridge delay. The maximum number of CTA (in our case 7) allocations is not always a good choise. Then we desinged our adaptive bandwidth allocation algorithm that shows slight improvement over nonadaptive bandwidth allocation. Further, we introduced hysteresis threshold with our adaptive bandwidth allocation algorithm. Our simulation results show improvements over adaptive bandwidth allocation algorithm. We have also observed the performance of our algorithms varying smoothing constant (α). The results of varying smoothing constant shows us that medium value i.e. if we consider half of the recent past values of a parameter, it gives us better results comparing to low or high constant values.

The performance of bridge device in IEEE 802.15.3 based scatternet is a new research area and many challenges are unsolved. In our experiments we have considered only two piconets. It will be an interesting work to further extend our model with more than two piconets and observe the bridge performance. In future we will focus on extensions to control the bandwidth allocation for uplink traffic, as well as that of ordinary nodes in the network.

Bibliography

[1] *Wimedia Alliance.* `http://www.wimedia.org`, 2006.

[2] I. F. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: a survey," *Computer Networks*, vol. 47, pp. 445–487, March 2005.

[3] M. Baldassari, R. Benso, and R. Pane, "Modeling WDM protocols and networks using the Artifex environment," in *Proceedings of Society of Photo-Optical Instrumentation Engineers (SPIE) conference on All-Optical Networking 1999: Architecture, Control, and Management Issues*, vol. 3843, August 1999, pp. 136–147.

[4] M. Brokate and J. Sprekels, *Hysteresis and Phase Transitions.* New York, NY: Springer-Verlag, 1991.

[5] M.-G. Di Benedetto, T. Kaiser, A. F. Molisch, I. Oppermann, C. Politano, and D. Porcino, *UWB Communication Systems A Comprehensive Overview.* New York, NY: Hindawi Publishing Corporation, 2006.

[6] IEEE, *IEEE 802.15 WPAN high rate alternative PHY Task Group 3a*

(TG3a). Available online at `http://www.ieee802.org/15/pub/TG3a.html`, August 2006.

[7] "Standard for wireless LAN medium access control (MAC) and physical layer (PHY) specifications," IEEE, New York, NY, IEEE standard 802.11, May 1997.

[8] "Standard for part 15.3: Wireless medium access control (MAC) and physical layer (PHY) specifications for high rate wireless personal area networks (WPAN)," IEEE, New York, NY, IEEE standard 802.15.3, 2003.

[9] "Standard for part 15.4: Wireless medium access control (MAC) and physical layer (PHY) specifications for low rate wireless personal area network (WPAN)," IEEE, New York, NY, IEEE Std 802.15.4, Oct. 2003.

[10] R. K. Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. New York: John Wiley and Sons, 2001.

[11] N. Johansson, U. Korner, and L. Tassiulas, "A distributed scheduling algorithm for a bluetooth scatternet," *Elsevier Science B.V.*, 2001.

[12] J. F. Kurose and K. W. Ross, *Computer Networking: A Top-Down Approach Featuring The Internet*. Boston, MA: Addison-Wesley Longman, 2000.

[13] J. V. Misic, V. B. Misic, and C. J. Fung, "Interconnecting 802.15.4 clusters in master-slave mode using guaranteed time slots and acknowledged transfers: Queueing theoretic analysis," in *IEEE 8th International Symposium on Parallel*

Architectures,Algorithms and Networks, ISPAN 2005, Las Vegas, Neveda, December 2005.

[14] J. Mišić and V. B. Mišić, *Performance Modeling and Analysis of Bluetooth Networks.* Boca Raton, FL: Auerbach Publications, 2006.

[15] RSoft Design, Inc., *Artifex v.4.4.2*, San Jose, CA, 2003.

[16] G. Tan and J. Guttag, "A locally coordinated scatternet scheduling algorithm," in *27th Annual IEEE Conference on Local Computer Networks*, Tampa, FL, November 2002.

[17] A. Torok, L. Vajda, P. Laborezi, Z. Fulop, and A. Vidacs, "Analysis of scatternet formation in high-rate multi-hop wpans," in *Proceedings of 17th Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communication (PIMRC'06)*, Helsinki, Finland, 2006.

[18] Y. Xiao, "MAC layer issues and throughput analysis for the IEEE 802.15.3a UWB," *Dynamics of Continuous, Discrete and Impulsive Systems, Series B: Applications & Algorithms*, vol. 12, pp. 443–462, June 2005.

[19] Z. Yin and V. C. Leung, "Connection data rate optimization of IEEE 802.15.3 scatternets with multi-rate carriers," in *IEEE International Conference on Communications, ICC '06*, Istanbul, Turkey, 2006.