On Timeslots Scheduling Algorithms of Wireless Ad Hoc Network

CHAU, Wai Shing

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Degree of
Master of Philosophy
in
Information Engineering

October 2007

Abstract of thesis entitled:
On Timeslots Scheduling Algorithms of Wireless Ad Hoc Network
Submitted by CHAU, Wai Shing
for the degree of Master of Philosophy
at The Chinese University of Hong Kong in July 2007

Wireless ad hoc network consists of a set of wireless nodes without a fixed infrastructure. It enables new and exciting application, but also poses a lot of significant technical challenges. How to share the radio spectrum for different nodes is one of the challenges. TDMA is one of the solution to share the spectrum. However, it can be shown that finding the minimal timeslots for scheduling different links in a network with arbitrary topology is a NP Complete problem.

In this thesis, we present some algorithms for timeslots scheduling in ad hoc network. In the first part, we present a simple centralized scheduling algorithm which can produce optimal scheduling in some kinds of ring networks. Such algorithm can facilitate some kinds of applications such as video-streaming or VOIP, in which delay jitter is one of the concerns. In the second part, we propose a distributed heuristic algorithm which can be applied on arbitrary topology for scheduling traffic. This algorithm can produce optimal scheduling in some kinds of special networks.

Acknowledgement

I would like to express my sincere gratitude to my supervisor Prof. Wing Shing Wong for his valuable guidance and advice during my M.Phil. studies. His constant patience and teaching were impressive to me. I admire him for spending so much time in guiding me to do research work.

I am grateful to my co-supervisor Prof. Angela Y. J. Zhang for her valuable comments and advices on my work.

Many thanks go to my labmates in Information and Systems Laboratory, who have created an excellent environment for study and work. It is enjoyable to discuss with them on study, research and life.

Last but not least, I would like to show my thanks to my family for their endless support and unconditional love.

摘要

無線 Ad Hoc 網絡是由一組無線節點組成，沒有固定設施。它帶來很多新的運用，但同時亦帶來了很多技術挑戰。其中一個挑戰是如何令不同的無線節點能有效地共用同一個頻譜。時分多址(TDMA)是一個分配頻譜的辦法，可是在一個任意拓撲的網絡上，怎樣用最少的時段進行調度，卻是是一個 NP Complete 的問題。

本論文旨在提出若干算法以解決在 Ad Hoc 網絡上進行調度的問題。在第一部分，我們提出一個簡單的集中式算法，這種算法可以在某些環形網絡上得到最佳的調度。此外，這種算法對於一些需要低時延抖動的應用有著若干好處，這些應用包括視訊串流和 IP 電話；在第二部分，我們提供一個分佈式的啟發式算法，可以於任意拓撲的網絡上進口調度。我們證明這種算法能在幾種特殊網絡上得到最佳的調度。

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Summary

In this chapter, introduction of the ad hoc network and the outline of the thesis will be given.

1.1 Ad hoc network

Nowadays, wireless networks are commonly used and they have become a part of our life. Notebooks, mobile phones, PDAs, tracking sensors, all make use of the wireless networks. Human beings are able to communicate with each other or collect useful information at anytime and anywhere they want. This brings a lot of convenience to people and greatly improve our quality of life compared to the people living in the past centuries. Besides, new wireless network standards and technologies are appearing everyday to enable the easy deployment of applications.

The deployment of wireless networks without a fixed infrastructure can be difficult. Ad hoc network is proposed to solve this problem. An ad hoc network is a network consisting of a collection of radio units with a wireless interface forming tem-

porary connections. There is no fixed infrastructure involved in the communication. If the signal-to-noise ratio is high enough, two radio units can establish a direct communication link. However, if two radio units are far away, and the signal-to-noise ratio is not high enough, they can still communicate if other nodes in the network are willing to forward packets for them. This kind of network is called multi-hop radio network, which have many applications in military command and control systems, for which it is often not feasible to install any permanent communication infrastructure.

Ad hoc networks are highly appealing for many reasons. They can be deployed and reconfigured quickly. They are highly robust due to its distributed nature and the absense of a single point of failure. So they are suitable for use in situations where an infrastructure is unavailable or it is expensive to deploy a network with traditional infrastructure, such as cellular network. One of the possible uses is to provide crisis management service applications, such as in disaster recovery, in which the communication infrastructure is destroyed and it is crucial to set up the network in a short time. By using ad hoc networks, communication networks can be set up in hours instead of weeks in the case of wired line communication.

Bluetooth provides a platform for implementing ad hoc networks. Users can connect various devices, such as mobile phones, notebooks or printers in a quick and easy way via Bluetooth. The IEEE 802.11 protocol also supports ad hoc mode to let the devices to form an ad hoc network when a fixed wireless access point is unavailable. Recently, there is also a growing interest in other applications using ad hoc networks, such as communication between mobile sensors and peer-to-peer communications.

There are many challenges for designing ad-hoc network [1]. One issue concerns the multiple access control (MAC) scheme which is related to how to schedule traffic demands of the links in an efficient way. Although this is a relatively simple problem to state, it is complicated enough that no efficient solution for a general network topology has been commented yet. In this thesis, we investigate the problem of how to design a good schedule for certain types of ad hoc networks.

1.2 Outline of the thesis

The organization of the thesis is as follows: In Chapter 2, the background of the subject will be given. In Chapter 3, a centralized scheduling algorithm for a special kind of network called a ring network will be discussed. In Chapter 4, a distributed heuristic scheduling algorithm will be presented for the ad hoc network with an arbitrary topology. Lastly, in Chapter 5, a conclusion will be given as a final statement.

□ **End of chapter.**

Chapter 2

Background Study

Summary

In this chapter, background study on scheduling algorithms of ad hoc network is presented.

2.1 Multiple Access Control (MAC)

In a wireless environment, the radio spectrum is one of the scarcest resources. How to utilize the valuable specturm in the most efficient way is one of the critical issues. In order to share the radio spectrum for different nodes in a network, a multiple access control (MAC) is needed to allocate channels for each node. MAC protocols for wireless networks can be classed into 2 categories, namely contention-based and contention-free, depending on the medium access strategy [24].

The contention-based scheme is normally used for transferring sporadic data on the mobile networks where the network topology is random and temporary. Wi-Fi local network in their ad hoc mode employs contention-based MAC protocols. Examples of such protocols include ALOHA, Slotted-ALOHA and

CSMA. In ALOHA, a station accesses the medium as soon as it has a frame to send. If two or more stations are transmitting at the same time, collisions will occur. For Slotted-ALOHA, a station must wait for the pre-defined interval of the time to start its transmission. In CSMA, a station that has data to send would try to detect the current transmission by sensing the medium. If the medium is busy, it will postpone the transmission. If the medium is idle, it will transmit the data immediately.

On the other hand, the contention-free schemes pre-define assignments to allow nodes to transmit information without contending for the medium. Examples include FDMA, CDMA, TDMA[23]. Contention-free schemes are normally employed to provide bounded end-to-end delay and minimum bandwidth, facilitating delay sensitive applications such as video streaming. In this thesis, we will propose algorithms based on Spatial TDMA.

2.1.1 Time Division Multiple Access (TDMA)

TDMA systems divide the radio spectrum into timeslots, and in each slot only one user is allowed to transmit or receive. Each user occupies a cyclically repeating timeslot, so a channel may be thought of a particular timeslot that reoccurs in every frame, where N timeslots comprise a frame.

2.1.2 Spatial TDMA (STDMA)

Although simple to implement, TDMA is very inefficient in view of resource utilization. An extension of TDMA which can increase the network capacity is Spatial TDMA, or STDMA [20]. STDMA takes into the account that the nodes are in fact spread out geographically, so links with enough spatial separation can use the same timeslot for transmission.

STDMA algorithms can be categorized into link scheduling [7] [16] [3] [11] and broadcast scheduling [26] [8] [6] [9] . In link scheduling, the transmission right is assigned to links and both transmitters and receivers should be determined a priori. The algorithm introduced in this thesis belongs to this category. In broadcast scheduling, the transmission right is allocated to nodes, which allows them to transmit to any of their neighbors. In STDMA, the algorithm used to generate the transmission schedule will affect the efficiency of spatial reuse. Thus, scheduling algorithm becomes important for implementing the STDMA scheme. In this thesis, we investigate algorithms for generating the transmission schedules. To introduce our approach, we need to introduce the interference model first.

2.2 Interference Models

2.2.1 Primary and Secondary Interferences

To schedule two links at the same timeslot, we have to ensure that the schedule will avoid interference. There are two types of interference in the literature, namely primary interference and secondary interference [21].

Primary interference is related to a scheduling whereby a single node performs more than one operation in the underlying timeslot, such as transmitting and receiving data at the same time.

Secondary interference occurs under a scheduling in which a receiver R tuning to a particular transmitter is in the interference range of another transmitter, whose intended destination is not R.

If the scheduling does not consider these two kinds of interference, conflicts will likely occur.

2.2.2 Interference-based Model

There are two different ways to decide when a link can be active, depending on the assumptions. The two models are interference-based model and graph-based model [12] [2].

For the interference-based model, signal-to-interference-and-noise ratio (SINR) is used to described the aggregated inteferences in the network as follows:

Let $\{i_k, i_k \in W\}$ be the subset of nodes simultaneously transmitting at a time instant over the channel. Then the transmission from a node i_r, $i_r \in W$, is successfully received by a node j_s, $j_s \notin W$, if and only if

$$\frac{P_{i_r}/d^{\alpha}(i_r, j_s)}{N + \sum_{k \neq r, i_k \in W} P_{i_k}/d^{\alpha}(i_k, j_s)} \geq \beta_c$$

where P_{i_r} is the power of node i_r, $d(i_k, j_s)$ is the distance between nodes i_k and j_s, and N is the thermal noise power level at the receiver. α is the path loss exponent, in which for outdoors environments $2 < \alpha \leq 5$ [25]. β_c is the threshold for reliable communication. This kind of interference model is commonly known as the Physical Interference Model [19].

2.2.3 Graph-based Model

For the graph-based model, the ad hoc network is represented as a directed graph G with a set of nodes V and a set of edges E satisfying the condition

$$(i, j) \in E \text{ if and only if } \Gamma_{i,j} = \frac{P_i/d^{\alpha}(i, j)}{N} \geq \beta_c$$

where P_i is the power of node i, $d(i,j)$ is the distance between nodes i and j, N is the thremal noise level at the receiver and α is the path loss exponent. β_c is the threshold for reliable communication. The schedule is then designed from the graph G. Interference from other nodes are not taken into account. Traditional method of link assignment is that given the set of edges E, the edges (i,j) and (k,l) can be assigned to a same timeslot if and only if

1. The vertices v_i, v_j, v_k, v_l are all mutually distinct,
2. $(i,l) \notin E \wedge (k,j) \notin E$

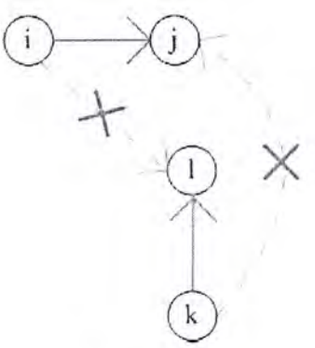

Figure 2.1: Illustration of the conditions in graph-based model.

The first criteria is to avoid the primary interference, that is, a node cannot receive and transmit simultaneously in the same timeslot. The secondary criteria is to avoid the secondary interference, that is a node cannot receive a packet while neighboring links are transmitting.

Centralized algorithms [18] [4] as well as distributed algorithm [10] [13] [15] have been proposed using this model.

2.3 Scheduling in Graph-based Model

In this thesis, we will consider scheduling algorithms assuming the graph-based interference model. In order to model the in-

terference relationships between different links on graph-based model, the concept of a conflict graph [14] will be used.

2.3.1 Conflict Graph

Every link in the connectivity graph $G(V, E)$ is represented by a node in the conflict graph $G'(V', E')$. Two nodes in the conflict graph are connected by an edge if their corresponding links in G are interfering with each other, i.e. the links cannot be active simultaneously. Whether two links are interfering with each other depends on the interference model. In Figure 2.2, we show an example on how to find the conflict graph. The connectivity graph is shown on the left. Assume each node in the connectivity graph has only one radio interface, i.e. each node cannot transmit and receive at the same time. Thus link A will conflict with link B, link B will conflict with link C and link C will conflict with link D. A dotted line is shown in the connectivity graph to represent there is interference between them. Assume there is no other conflict due to sufficient separations between transmission nodes. Then we can obtain the corresponding conflict graph, which is shown on the right. Notice that the link in the connectivity graph will become a node in the conflict graph.

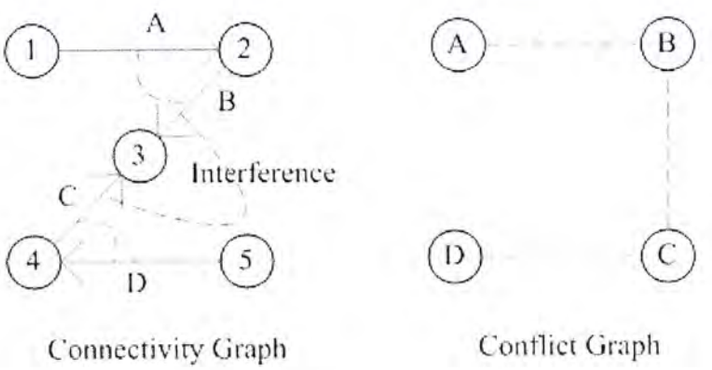

Figure 2.2: Example of conflict graph.

By transforming the connectivity graph into the conflict graph, the timeslot assigning problem in the connectivity graph is equivalent to the coloring problem in the conflict graph. No two nodes connected by an edge in the conflict graph can have the same color. Thus finding the minimal-length schedule for the nodes is equivalent to finding the minimal colors needed for coloring the conflict graph. Such coloring problems are shown to be nondeterminiistic polynomial (NP) complete [22] [17]. Different graph coloring heuristics[5] can be used to solve the problems in a practical way.

In chapter 3 of this thesis, we are focusing on a centralized algorithm which can produce optimal results for ring networks under some conditions, by using a kind of sequences proposed by us, called Regular Sequences. In chapter 4, we propose a distributed heuristic algorithm which is simple to implement for scheduling traffic demands on a general network. Moreover, this algorithm can produce the optimal scheduling in certain kind of special networks.

□ End of chapter.

Chapter 3

Scheduling Algorithms in Ring Networks

Summary

In this part, a new family of sequences called Regular Sequences will be introduced. By using the sequences, we present a centralized algorithm which can produce optimal scheduling for ring networks with even-number of edges and certain type of ring networks with odd-number of edges. The resulting schedules have properties that make them suitable for applications such as video streaming or VOIP in which delay jitter between packets is an important concern.

3.1 Problem Formulation

Network Model

Consider a ring network where the number of links is equal to n as in Figure 3.1. The links are labeled as $l_1, l_2, \ldots l_n$. The capacities of all links are assumed to be the same, which is equal to C. However, the traffic demands r_i are different for different links. Let $r_i = \frac{p_i}{q_i} \cdot C$ where $p_i \in \mathbf{N}$, $q_i \in \mathbf{N}$ and $p_i \leq q_i$. In this part of the thesis, we are dealing with the problem of how to allocate timeslots for different links so that the traffic demands are satisfied. Before going on to introduce our algorithms, it should be stated that following assumptions are made.

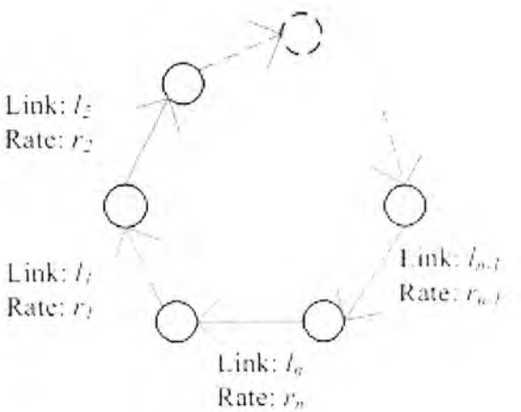

Figure 3.1: Example of a ring network.

Assumptions

1. Primary interference is being considered, that is a node cannot receive data and transmit data at the same time.

2. Each node is equipped with a directional antenna with transmission range equal to one hop. Interference at 2 hops or more is negligible. In other words, it is assumed that, any node will not receive more than one transmission at a time. So there is no secondary interference.

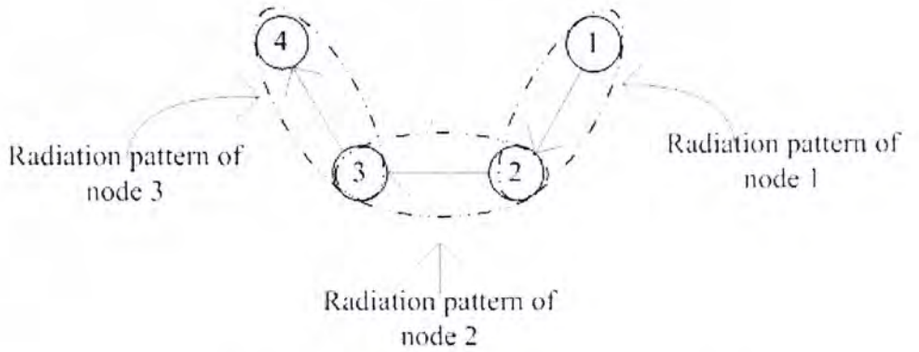

Figure 3.2: Illustration of assumption 2.

3. The rates of all the links are feasible (i.e. $r_i \leq C$).

4. The capacities of all the links are the same, which is equal to 1.

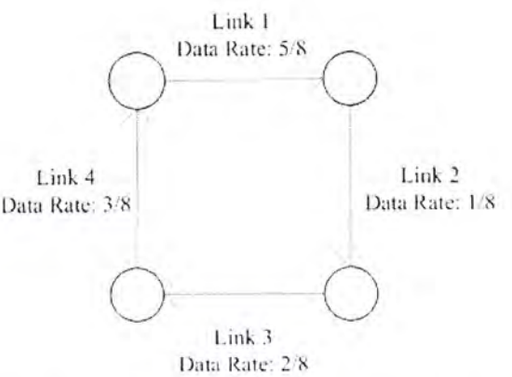

Figure 3.3: Example of scheduling a ring network.

Based on these assumptions, consider the network in Figure 3.3. There are totally four links with required rates of $\frac{5}{8}$, $\frac{1}{8}$, $\frac{2}{8}$ and $\frac{3}{8}$ respectively. Suppose the period of the timeslots is 8. One way to assign the timeslots for Link 1, 2 and 3 is following:

Timeslots	1	2	3	4	5	6	7	8
Link 1	1	1	1	1	1	0	0	0
Link 2	0	0	0	0	0	1	0	0
Link 3	0	0	0	0	0	0	1	1

To schedule Link 4, only timeslot 5 can be used. Other timeslots cannot be used due to the interference from Link 1 and Link 3. By using this timeslot allocation, Link 4 can only achieve the rate of $\frac{1}{8}$ which is smaller than the required rate $\frac{3}{8}$. However, if we assign the timeslots as following:

Timeslots	1	2	3	4	5	6	7	8
Link 1	1	0	1	0	1	0	1	1
Link 2	0	1	0	0	0	0	0	0
Link 3	1	0	0	0	1	0	0	0
Link 4	0	1	0	1	0	1	0	0

The required rates of all four links can be achieved. From this example, we can see that whether the rates of a network can be achieved depends on how the timeslots are assigned. If the timeslots are allocated without a systematic design, network capacity cannot be fully utilized.

3.2 Regular Sequences

To obtain the proposed schedules, we first introduce two sets of binary sequences, $s_{p,q}$ and $t_{p,q}$. $t_{p,q}$ is a shifted version of $s_{p,q}$. By using these two sets of sequences, we can allocate timeslots efficiently according to the rates of the links. We will describe the algorithm in the later section. Here, we discuss how to construct the sequences via the following algorithms:

Algorithm 1 *For a positive integer q, define the binary sequences $s_{p,q}$ where $p \in \{0, 1, 2, \ldots q\}$ in the following ways:*

1. *The length of $s_{p,q}$ is q. Denote the sequence, $s_{p,q}$, as $a_{p,q,1}a_{p,q,2}\ldots a_{p,q,q}$.*
2. $a_{0,q,i} = 0 \ \forall i \in \{1, 2, \ldots q\}$
3. $a_{1,q,1} = 1$ *and* $a_{1,q,i} = 0 \ \forall i \in \{2, 3, \ldots q\}$
4. *for* $m := 2$ *to* $\lfloor \frac{q}{2} \rfloor$ *do*
   (a) $a_{m,q,i} = 0 \ \forall i \in \{i : (2 \leq i \leq q, i = 2m, m \in \mathbf{Z}) \vee (i = q)\}$
   (b) $a_{m,q,i} = 1 \ \forall i \in \{i : a_{m-1,q,i} = 1\}$
   (c) *For all j where $a_{m,q,j}$ have not been allocated in steps (a) and (b), choose one j and let $a_{m,q,j} = 1$ so that the 1's in the sequence $s_{p,q}$ will appear as evenly distributed as possible. (That is, the distances between the newly placed '1' are the nearest '1' to its left and to its right in the newly formed sequence differ at most by 2.) Let $a_{m,q,j} = 0$ for other j.*
5. *for* $m := \lfloor \frac{q}{2} \rfloor + 1$ *to* q *do*
   $a_{m,q,i} = 1 - a_{q-m,q,i} \ \forall i \in \{1, 2, \ldots q\}$

Algorithm 2 *Define the binary sequences $t_{p,q}$ where $p \in \{0, 1, 2, \ldots q\}$ in the following ways:*

1. *The length of $t_{p,q}$ is q. Denote the sequence, $t_{p,q}$, as $b_{p,q,1}b_{p,q,2}\ldots b_{p,q,q}$.*
2. *for* $m := 2$ *to* $\lfloor \frac{q}{2} \rfloor$ *do*
   (a) $b_{m,q,1} = a_{m,q,q}$
   (b) $b_{m,q,i} = a_{m,q,i-1} \ \forall i \in \{2, 3, \ldots q\}$
3. *for* $m := \lfloor \frac{q}{2} \rfloor + 1$ *to* q *do*
   $b_{m,q,i} = 1 - b_{q-m,q,i} \ \forall i \in \{1, 2, \ldots q\}$

We call the binary sequences $s_{p,q}$ and $t_{p,q}$ where $p \in \{0, 1, 2 \ldots q\}$ the Regular Sequences of q. We name the set of binary sequences $s_{p,q}$ as Set S and that of $t_{p,q}$ as Set T.
Tables 3.1 and 3.2 show the Regular Sequences for $q = 7$ and $q = 8$.

Rate	(p, q)	$s_{p,q}$	$t_{p,q}$
0/7	$p = 0, q = 7$	0000000	0000000
1/7	$p = 1, q = 7$	1000000	0100000
2/7	$p = 2, q = 7$	1000100	0100010
3/7	$p = 3, q = 7$	1010100	0101010
4/7	$p = 4, q = 7$	0101011	1010101
5/7	$p = 5, q = 7$	0111011	1011101
6/7	$p = 6, q = 7$	0111111	1011111
7/7	$p = 7, q = 7$	1111111	1111111

Table 3.1: Regular Sequences for $q = 7$.

Rate	(p, q)	$s_{p,q}$	$t_{p,q}$
0/8	$p = 0, q = 8$	00000000	00000000
1/8	$p = 1, q = 8$	10000000	01000000
2/8	$p = 2, q = 8$	10001000	01000100
3/8	$p = 3, q = 8$	10101000	01010100
4/8	$p = 4, q = 8$	10101010	01010101
5/8	$p = 5, q = 8$	01010111	10101011
6/8	$p = 6, q = 8$	01110111	10111011
7/8	$p = 7, q = 8$	01111111	10111111
8/8	$p = 8, q = 8$	11111111	11111111

Table 3.2: Regular Sequences for $q = 8$.

Theorem 1 *Let*
$x = a_1 a_2 \ldots a_n$ *be a Regular Sequence in Set S with rate* $r_x = \frac{a}{q} \leq \frac{1}{2}$,
$y = b_1 b_2 \ldots b_n$ *be a Regular Sequence in Set T with rate* $r_y = \frac{b}{q} \leq \frac{1}{2}$, *for some non-negative integers* a, b *and positive integer* q.
Then,

$$\{i : a_i = 1\} \cap \{j : b_j = 1\} = \emptyset.$$

Proof
By the algorithm constructing the Regular Sequences, if $r_x \leq \frac{1}{2}$, then $\{i : a_i = 1\} \subseteq \{j : 1 \leq j \leq n-1, j = 2m+1, m \in \mathbf{Z}\}$.
If $r_y \leq \frac{1}{2}$, then $\{i : b_i = 1\} \subseteq \{j : 2 \leq j \leq n, j = 2m, m \in \mathbf{Z}\}$.
Since,

$$\begin{aligned}
&\{j : 1 \leq j \leq n-1, j = 2m+1, m \in \mathbf{Z}\} \cap \{j : 2 \leq j \leq n, j = 2m, m \in \mathbf{Z}\} \\
&= \{j : (1 \leq j \leq n-1, j = 2m+1, m \in \mathbf{Z}) \wedge (2 \leq j \leq n, j = 2m, m \in \mathbf{Z})\} \\
&= \{j : 2 \leq j \leq n-1, (j = 2m \wedge j = 2m+1), m \in \mathbf{Z}\} \\
&= \emptyset
\end{aligned}$$

So,

$$\{i : a_i = 1\} \cap \{j : b_j = 1\} = \emptyset$$

■

Theorem 2 *Let*
$x = a_1 a_2 \ldots a_n$ *be a Regular Sequence in Set U with rate* $r_x = \frac{a}{q} \leq \frac{1}{2}$,
$y = b_1 b_2 \ldots b_n$ *be a Regular Sequence in Set U with rate* $r_y = \frac{b}{q} > \frac{1}{2}$ *for some non-negative integers* a, b *and positive integer* q, *where* $U = S$ *or* T, $r_x + r_y \leq 1$.
Then,

$$\{i : a_i = 1\} \cap \{j : b_j = 1\} = \emptyset.$$

Proof

Let $r_c = 1 - r_y$ and $z = c_1 c_2 \ldots c_n$ be the Regular Sequence in Set U with the rate r_c. By the basic property of the constructing algorithm, $\{i : c_i = 1\} = \{i : b_i = 0\}$. Since

$$\begin{aligned} r_x + r_y &\leq 1 \\ r_x + (1 - r_c) &\leq 1 \\ r_x &\leq r_c. \end{aligned}$$

By the way we construct the sequences, $\{i : a_i = 1\} \subseteq \{i : c_i = 1\}$ for $r_x \leq \frac{1}{2}$ and $r_y \leq \frac{1}{2}$.
Thus,

$$\begin{aligned} \{i : a_i = 1\} \cap \{i : b_i = 1\} &\subseteq \{i : c_i = 1\} \cap \{i : b_i = 1\} \\ &= \{i : b_i = 0\} \cap \{i : b_i = 1\} \\ &= \emptyset \end{aligned}$$

■

Next, we introduce the following algorithm for allocating timeslots. We will show that this algorithm can achieve optimal scheduling for a ring network if the number of edges is even.

Algorithm 3

1. *Starting from link l_1, assign the sequence from Set S or T with the rate r_1.*

2. *for $i := 2$ to n do*
   if $r_{i-1} \leq \frac{1}{2}$ and $r_i \leq \frac{1}{2}$
   then assign the sequence from a set, which is different from the set used in the link l_{i-1}, to link l_i.
   else assign the sequence from a set, which is the same as the set used in the link l_{i-1}, to link l_i.

A chain network is a network consisting of a chain of nodes, such as the one shown in Figure 3.4.

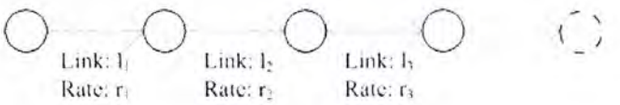

Figure 3.4: A chain network.

Theorem 3 *Consider a chain network with n edges, where n is an even number. If $r_i + r_{i+1} \leq 1$ $\forall i \in \{1, 2, ...n-1\}$ and r_i is in the form $\frac{a_i}{q}$ for some non-negative integers a_i and positive integer q. Apply the scheduling using the Regular Sequences by Algorithm 3. Let the sequence assigned to link n belong to the Set α_n. The following statements hold:*

1. *If $r_1 \leq \frac{1}{2}$ and $r_n \leq \frac{1}{2}$, then, $\alpha_1 \neq \alpha_n$,*
2. *If $r_1 \leq \frac{1}{2}$ and $r_n > \frac{1}{2}$, then, $\alpha_1 = \alpha_n$,*
3. *If $r_1 > \frac{1}{2}$ and $r_n \leq \frac{1}{2}$, then, $\alpha_1 = \alpha_n$,*
4. *If $r_1 > \frac{1}{2}$ and $r_n > \frac{1}{2}$, then, $\alpha_1 \neq \alpha_n$.*

Proof
We prove this by mathematical induction.
Let $S(n)$ be the statement stated in the theorem where n is an even number.
When $n = 2$, there are four different cases:

$$\begin{aligned}
&\text{Case(1): } r_1 \leq \tfrac{1}{2} \text{ and } r_2 \leq \tfrac{1}{2},\\
&\text{Case(2): } r_1 \leq \tfrac{1}{2} \text{ and } r_2 > \tfrac{1}{2},\\
&\text{Case(3): } r_1 > \tfrac{1}{2} \text{ and } r_2 \leq \tfrac{1}{2},\\
&\text{Case(4): } r_1 > \tfrac{1}{2} \text{ and } r_2 > \tfrac{1}{2}.
\end{aligned}$$

For case (1), by Theorem 1, we can schedule the links with the sequences from two different sets, i.e. $\alpha_1 \neq \alpha_2$.
For case (2) and (3), by Theorem 2, we can schedule the links using the sequences from the same set, i.e. $\alpha_1 = \alpha_2$.
For case (4), this case is not possible for $n = 2$ as it violates the assumption $r_1 + r_2 \leq 1$.
So $S(2)$ is true.

Assume $S(k)$ is true where k is an even number. For a chain network with number of edges $= k + 2$
Consider four different cases:

$$\begin{aligned}
&\text{Case(1): } r_1 \leq \tfrac{1}{2} \text{ and } r_k \leq \tfrac{1}{2},\\
&\text{Case(2): } r_1 \leq \tfrac{1}{2} \text{ and } r_k > \tfrac{1}{2},\\
&\text{Case(3): } r_1 > \tfrac{1}{2} \text{ and } r_k \leq \tfrac{1}{2},\\
&\text{Case(4): } r_1 > \tfrac{1}{2} \text{ and } r_k > \tfrac{1}{2}.
\end{aligned}$$

In each of the above cases, consider three subcases:

$$\begin{aligned}
&\text{Case(a): } r_{k+1} \leq \tfrac{1}{2} \text{ and } r_{k+2} \leq \tfrac{1}{2},\\
&\text{Case(b): } r_{k+1} \leq \tfrac{1}{2} \text{ and } r_{k+2} > \tfrac{1}{2},\\
&\text{Case(c): } r_{k+1} > \tfrac{1}{2} \text{ and } r_{k+2} \leq \tfrac{1}{2}.
\end{aligned}$$

Note that the case $r_{k+1} > \frac{1}{2}$ and $r_{k+2} > \frac{1}{2}$ is not possible as it violates the assumption $r_{k+1} + r_{k+2} \leq 1$.

Case(1a): By the assumption, $\alpha_1 \neq \alpha_k$. By Theorem 1, we have to let $\alpha_{k+1} = \alpha_1$ and $\alpha_{k+2} = \alpha_k$. So $\alpha_1 \neq \alpha_{k+2}$ for $r_1 \leq \frac{1}{2}$ and $r_{k+2} \leq \frac{1}{2}$.

Case(1b): By the assumption, $\alpha_1 \neq \alpha_k$. By Theorem 1 and 2, we have to let $\alpha_{k+1} = \alpha_1$ and $\alpha_{k+2} = \alpha_1$. So $\alpha_1 = \alpha_{k+2}$ for $r_1 \leq \frac{1}{2}$ and $r_{k+2} > \frac{1}{2}$.

Case(1c): By the assumption, $\alpha_1 \neq \alpha_k$. By Theorem 2, we have to let $\alpha_{k+1} = \alpha_k$ and $\alpha_{k+2} = \alpha_k$. So $\alpha_1 \neq \alpha_{k+2}$ for $r_1 \leq \frac{1}{2}$ and $r_{k+2} \leq \frac{1}{2}$.

Case(2a): By the assumption, $\alpha_1 = \alpha_k$. By Theorem 1 and 2, we have to let $\alpha_{k+1} = \alpha_k$ and $\alpha_{k+2} \neq \alpha_k$. So $\alpha_1 \neq \alpha_{k+2}$ for $r_1 \leq \frac{1}{2}$ and $r_{k+2} \leq \frac{1}{2}$.

Case(2b): By the assumption, $\alpha_1 = \alpha_k$. By Theorem 2, we have to let $\alpha_{k+1} = \alpha_1$ and $\alpha_{k+2} = \alpha_1$. So $\alpha_1 = \alpha_{k+2}$ for $r_1 \leq \frac{1}{2}$ and $r_{k+2} > \frac{1}{2}$.

Case(2c): This case is not possible as it violates the assumption $r_k + r_{k+1} \leq 1$.

Case(3a): By the assumption, $\alpha_1 = \alpha_k$. By Theorem 1, we have to let $\alpha_{k+1} \neq \alpha_k$ and $\alpha_{k+2} = \alpha_1$. So $\alpha_1 = \alpha_{k+2}$ for $r_1 > \frac{1}{2}$ and $r_{k+2} \leq \frac{1}{2}$.

Case(3b): By the assumption, $\alpha_1 = \alpha_k$. By Theorem 1 and 2, we have to let $\alpha_{k+1} \neq \alpha_k$ and $\alpha_{k+2} \neq \alpha_1$. So $\alpha_1 \neq \alpha_{k+2}$ for $r_1 > \frac{1}{2}$ and $r_{k+2} > \frac{1}{2}$.

Case(3c): By the assumption, $\alpha_1 = \alpha_k$. By Theorem 2, we have to let $\alpha_{k+1} = \alpha_k$ and $\alpha_{k+2} = \alpha_1$. So $\alpha_1 = \alpha_{k+2}$ for $r_1 > \frac{1}{2}$ and $r_{k+2} \leq \frac{1}{2}$.

Case(4a): By the assumption, $\alpha_1 \neq \alpha_k$. By Theorem 1 and 2, we have to let $\alpha_{k+1} = \alpha_k$ and $\alpha_{k+2} \neq \alpha_k$. So $\alpha_1 = \alpha_{k+2}$ for $r_1 > \frac{1}{2}$ and $r_{k+2} \leq \frac{1}{2}$.

Case(4b): By the assumption, $\alpha_1 \neq \alpha_k$. By Theorem 2, we have to let $\alpha_{k+1} = \alpha_k$ and $\alpha_{k+2} = \alpha_k$. So $\alpha_1 \neq \alpha_{k+2}$ for $r_1 > \frac{1}{2}$ and $r_{k+2} > \frac{1}{2}$.

Case(4c): This case is not possible as it violates the assumption $r_k + r_{k+1} \leq 1$.

So $S(k+2)$ is ture.

Since the statement $S(2)$ clearly holds, by mathematical induction, the statement $S(n)$ is true for all positive even intergers n.

■

3.3 Scheduling in Ring Networks with Even-number of Edges

Definition 1 *A* **feasible schedule** *is a schedule in which each node in the network would not transmit and receive at the same time.*

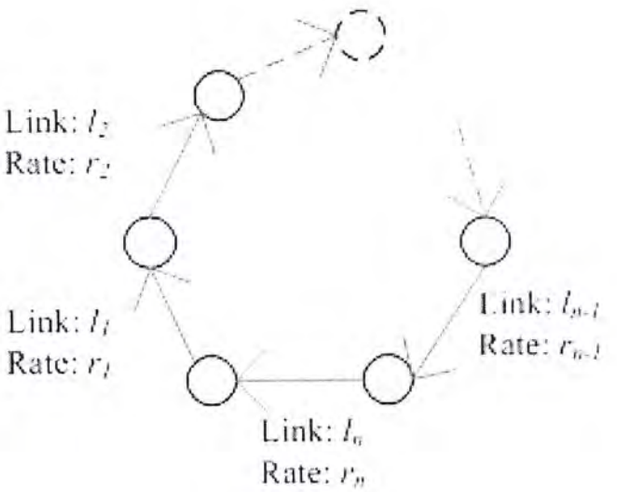

Figure 3.5: Example of a ring network.

Theorem 4 *For a ring network with n edges, where n is an even number, the scheduling with Regular Sequences obtained from Algorithm 3 does not contain any conflicting timeslots (i.e. a feasible schedule) provided that*

$$r_i + r_{i+1} \leq 1 \ \forall i \in \{1, 2, ...n-1\}$$

and

$$r_1 + r_n \leq 1$$

where r_i is the rate of link i and r_i is in the form $\frac{a_i}{q}$ for some non-negative integers a_i and positive integer q.

Proof

Finding the schedules of a ring network with n edges, where n is even, $r_i + r_{i+1} \leq 1 \ \forall i \in \{1, 2, \ldots n-1\}$ and $r_1 + r_n \leq 1$, is the same as finding the schedules of a chain network with number of edges $= n$, where (1) $r_i + r_{i+1} \leq 1 \ \forall i \in \{1, 2, \ldots n-1\}$, and (2) $r_1 + r_n \leq 1$.

To satisfy Condition (1), we assign the sequences in the same way as a chain network with an even number of edges, n. By Theorem 3,

1. If $r_1 \leq \frac{1}{2}$ and $r_n \leq \frac{1}{2}$, then, $\alpha_1 \neq \alpha_n$,
2. If $r_1 \leq \frac{1}{2}$ and $r_n > \frac{1}{2}$, then, $\alpha_1 = \alpha_n$,
3. If $r_1 > \frac{1}{2}$ and $r_n \leq \frac{1}{2}$, then, $\alpha_1 = \alpha_n$,
4. If $r_1 > \frac{1}{2}$ and $r_n > \frac{1}{2}$, then, $\alpha_1 \neq \alpha_n$.

For case (1), since $\alpha_1 \neq \alpha_n$, by Theorem 1, link 1 and n can be scheduled together provided that $r_1 + r_n \leq 1$.
For case (2), since $\alpha_1 = \alpha_n$, by Theorem 2, link 1 and n can be scheduled together provided that $r_1 + r_n \leq 1$.
For case (3), since $\alpha_1 = \alpha_n$, by Theorem 2, link 1 and n can be scheduled together provided that $r_1 + r_n \leq 1$.
For case (4), this case is not possible since $r_1 + r_n > 1$ violates the assumption.
So Algorithm 3 can assign feasible sequences(i.e. sequences with no conflicts) on ring networks with an even-number of edges.

■

If the denominators of the rates are not the same, we can normalize all the rates to a common denominator. The following is an example showing this:

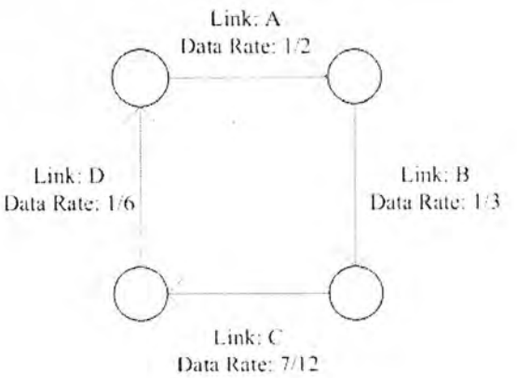

Figure 3.6: Example of a ring network with even-number of edges.

Example 1 *Consider the network in Figure 3.6. Suppose the rates of the four links are* $\frac{1}{2}$, $\frac{1}{3}$, $\frac{7}{12}$ *and* $\frac{1}{6}$ *respectively. To construct feasible schedules, we first normalize the rates to* $\frac{6}{12}$, $\frac{4}{12}$, $\frac{7}{12}$, $\frac{2}{12}$. *Construct Regular Sequences with* $q = 12$.

Rate	(p, q)	$s_{p,q}$	$t_{p,q}$
0/12	$p = 0, q = 12$	000000 000000	000000 000000
1/12	$p = 1, q = 12$	100000 000000	010000 000000
2/12	$p = 2, q = 12$	100000 100000	010000 010000
3/12	$p = 3, q = 12$	101000 100000	010100 010000
4/12	$p = 4, q = 12$	101000 101000	010100 010100
5/12	$p = 5, q = 12$	101010 101000	010101 010100
6/12	$p = 6, q = 12$	101010 101010	010101 010101
7/12	$p = 7, q = 12$	010101 010111	101010 101011
8/12	$p = 8, q = 12$	010111 010111	101011 101011
9/12	$p = 9, q = 12$	010111 011111	101011 101111
10/12	$p = 10, q = 12$	011111 011111	101111 101111
11/12	$p = 11, q = 12$	011111 111111	101111 111111
12/12	$p = 12, q = 12$	111111 111111	111111 111111

Table 3.3: Regular Sequences for $q = 12$.

By Algorithm 3, we schedule link A first, choose a sequence from Set S. For link B, since rate of B is smaller than $\frac{1}{2}$, we choose the sequence from different Set, i.e. Set T. For link C, since the rate is larger than $\frac{1}{2}$, we choose the sequence from the same set, i.e. Set T. For the last link, as the rate is smaller than $\frac{1}{2}$, we choose the sequence from the same set too, i.e. Set T. So the schedules of those 4 links will be:

Link	Rate	Set	Schedules
A	$\frac{1}{2}$	$s_{6,12}$	101010 101010
B	$\frac{1}{3}$	$t_{4,12}$	010100 010100
C	$\frac{7}{12}$	$t_{7,12}$	101010 101011
D	$\frac{1}{6}$	$t_{2,12}$	010000 010000

Table 3.4: Schedules for example 1.

3.4 Scheduling in Ring Networks with an Odd-number of Edges

In this section, we deal with the case where the number of edges in a ring network is odd. For a ring network with n edges, where n is odd, the following conditions

$$r_i + r_{i+1} \leq 1 \ \forall i \in \{1, 2, \ldots n-1\}$$

and

$$r_1 + r_n \leq 1$$

where r_i is the rate of link i and r_i is in the form $\frac{a_i}{q}$ for some non-negative integers a_i and positive integer q

are not sufficient to guarantee there exists a feasible schedule. Following is an example showing this.

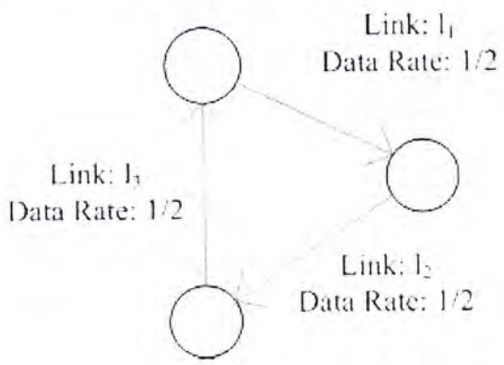

Figure 3.7: Example of a ring network with three edges.

Example 2 *Consider a ring network with three links as in Figure 3.7. Let the required rates of links (l_1, l_2, l_3) be $(\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$. Thus, it satisfies the conditions*

$$r_1 + r_2 \leq 1,\ r_2 + r_3 \leq 1 \text{ and } r_3 + r_1 \leq 1.$$

Since each link will interfere with another two links, at most one link can be active at a time. So the total rates of all the three

links must be less than 1. However, the sum of the required rates is $\frac{1}{2}+\frac{1}{2}+\frac{1}{2}=\frac{3}{2}$ *which is greater than 1. So there must not exist a feasible schedule to achieve the required rates.*

In this section, we will provide extra conditions such that the ring network with odd-number of edges can be scheduled using the Regular Sequences.

3.4.1 Scheduling by Reducing a Ring Network with an Odd-number of Edges into a Ring Network with an Even-number of Edges

One way to obtain a feasible schedule is to group two neighbouring links into one imaginary link. Then the ring network with odd-number of edges can be reduced into a ring network with even-number of edges. Thus, we can do the scheduling using Regular Sequences just like in section 3.3. So the imaginary link can get a Regular Sequence. By spliting that regular sequence using Algorithm 4 into two sequences(which are not Regular Sequences), the two neighboring links can get the schedules.

Here we introduce Algorithm 4.

Algorithm 4

Input:

x: *Regular sequence of link* l_c

$\frac{p_1}{q_1}$: *Rate of link* l_j

$\frac{p_2}{q_2}$: *Rate of link* l_{j-1}

Output:

w_1: *Sequence of link* l_j

w_2: *Sequence of link* l_{j-1}

Process:

1. *Let* p *be the number of 1's in* x, q *be the length of the sequence* x.
2. *Let* m *be the L.C.M. (Least Common Multiple) of* q, q_1 *and* q_2.
3. *Let* y *be the sequence by concatenating* x *for* m/q_1 *times.*
   (i.e. If $x = a_1a_2 \ldots a_n$,
   then $y = \underbrace{a_1a_2 \ldots a_na_1a_2 \ldots a_n \ldots\ldots a_1a_2 \ldots a_n}_{\frac{m}{q_1} \text{ consecutive } a_na_1a_2 \ldots a_n}$.*)*

4. *Convert any* $(p \cdot \frac{m}{q} - p_1 \cdot \frac{m}{q_1})$ *1's into 0's. Let the converted sequence be* w_1. *Note that the rate of* $w_1 = \frac{p_1}{q_1}$. *(Number of 1's in* $y_1 = p \cdot \frac{m}{q}$. *By converting* $(p \cdot \frac{m}{q} - p_1 \cdot \frac{m}{q_1})$ *1's into 0's, number of 1's remains* $= p \cdot \frac{m}{q} - (p \cdot \frac{m}{q} - p_1 \cdot \frac{m}{q_1}) = p_1 \cdot \frac{m}{q_1}$. *So the rate of* $w_1 = (p_1 \cdot \frac{m}{q_1})/m = \frac{p_1}{q_1}$.

5. *Let* $w_2 = y - w_1$. *Note that the rate of* $w_2 = \frac{p_2}{q_2}$.*(Number of 1's in* $y = p \cdot \frac{m}{q}$ *while number of 1's in* $w_1 = p_1 \cdot \frac{m}{q_1}$. *So number of 1's in* $w_2 = p \cdot \frac{m}{q} - p_1 \cdot \frac{m}{q_1} = m(\frac{p}{q} - \frac{p_1}{q_1}) = m \cdot \frac{p_2}{q_2}$. *So rate of* $w_2 = (m \cdot \frac{p_2}{q_2})/m = \frac{p_2}{q_2}$.

6. *Output the sequence* w_1, w_2.

Remarks:

If we want the '1's distributed on the output sequences more evenly, step 4 of the algorithm can be replaced by the following:

4a Let z *be the Regular Sequence* $s_{p\prime,q\prime}$
where $p\prime = p_1 \cdot \frac{m}{q_1}$ *and* $q\prime = p \cdot \frac{m}{q}$.
Note that rate of $z = r_z < 1$.
(Since $r_z = \frac{p_1 \cdot \frac{m}{q_1}}{p \cdot \frac{m}{q}} = \frac{p_1}{q_1}/\frac{p}{q} < 1$*).*

4b Construct the Regular Sequence $z = z_1 z_2 \ldots z_{q\prime}$, *denote* $K = \{i : z_i = 0\}$. *So,* $|K| = q\prime - p\prime$.

4c Convert the i-th '1' of sequence y into '0' where $i \in K$. *Let the modified sequence be* w_1. *Note that rate of* $w_1 = \frac{p_1}{q_1}$, *(Number of 1's in* $y = q\prime$. *By converting* $(q\prime - p\prime)$ *1's into 0's, number of 1's in* $w_1 = q\prime - (q\prime - p\prime) = p\prime$.
So rate of $w_1 = \frac{p\prime}{m} = \frac{p_1 \cdot \frac{m}{q_1}}{m} = \frac{p_1}{q_1}$.

Theorem 5 *When the number of edges in the ring network is odd, one can construct a feasible schedule provided that the following two conditions hold:*

1. $r_{i-1} + r_i \leq 1 \ \forall \ i \in \{1, 2, \ldots n-1, n\}$ *and*
2. *There exists* $j \in \{1, 2, \ldots n-1, n\}$ *such that*
   (a) $r_{j-2} + r_{j-1} + r_j \leq 1$ *and*
   (b) $r_{j-1} + r_j + r_{j+1} \leq 1$

where we define $l_{-1} = l_{n-1}$, $l_0 = l_n$, $l_{n+1} = l_1$, $r_{-1} = r_{n-1}$, $r_0 = r_n$ *and* $r_{n+1} = r_1$

Proof
If there exists a j such that the Condition (2) is hold, then we can group the link l_{j-1} and l_j to form an imaginary link l_c with rate $r_c = r_{j-1} + r_j$. Thus, the network $(l_1, l_2, \ldots l_{j-1}, l_j, l_{j+1}, \ldots, l_n)$ can be reduced to $(l_1, l_2, \ldots l_{j-2}, l_c, l_{j+1} \ldots l_n)$ which is an even-edge ring network. By Theorem 4, if condition 1 holds, then we can do the scheduling by the Regular Sequences in the same way described in Section 3.3. Thus link l_c will be assigned a regular sequence. By spilting the sequence assigned to l_c into 2 sequences(which are not regular sequences) using Algorithm 4, we can get the schedules of links l_{j-1} and l_j.

■

Example 3 *Consider a ring network with the number of edges equal to 5. Let the rates for links* $(l_1, l_2, l_3, l_4, l_5)$ *be* $(\frac{1}{2}, \frac{3}{24}, \frac{5}{24}, \frac{7}{12}, \frac{1}{6})$. *To get a schedule, we can group* l_2 *and* l_3 *to form an imaginary link* l_c *with rate* $r_c = \frac{3}{24} + \frac{5}{24} = \frac{1}{3}$. *Thus, by previous section, we can have the following schedules: To split the sequences of* l_c, *we first calculate the L.C.M. (Least Common Multiple) of 12,24 and 24, which is equal to 24. Then we construct the*

Link	Rate	Set	Schedules
l_1	$\frac{1}{2}$	$s_{6,12}$	101010 101010
l_c	$\frac{1}{3}$	$t_{4,12}$	010100 010100
l_4	$\frac{7}{12}$	$t_{7,12}$	101010 101011
l_5	$\frac{1}{6}$	$t_{2,12}$	010000 000000

Table 3.5: Schedules for example 3.

sequence y by concatenating 2 regular sequences of l_c. Thus, $y = 010100010100010100010100$. *There are eight 1's in the sequence y. To split the sequence, we can just take any 5 1's to obtain the schedule for l_2 and take the remaining 3 1's for l_3. If we want to make the 1 's distributed more evenly, we can do the following. Construct the regular sequence z with the rate $\frac{5}{8}$. So* $z = 01010111$. *The first, third, fifth position of z is 0. Convert the first, third and fifth '1' of y into '0'. Thus the new sequence* $w_1 = 000100000100000100010100$ *would be the schedule of link l_2.* $w_2 = y - w_1 = 010000010000010000000000$ *would be the schedule of link l_3.*

3.4.2 Scheduling by Shifting the Regular Sequences

For some kinds of ring network with odd number of edges, although there exists a feasible schedule for the nodes in the network, we may not able to obtain the schedule by using the method stated in section 3.4.1. Following is an example showing this.

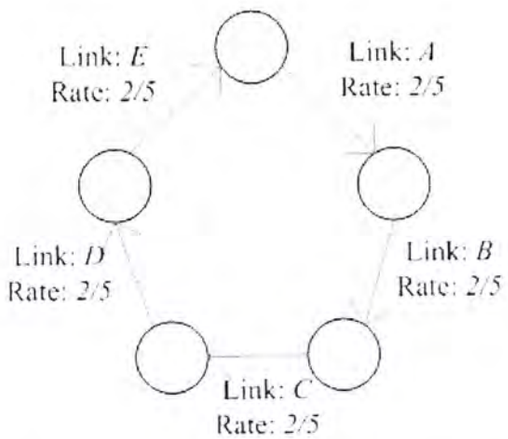

Figure 3.8: Example of a ring network with five links.

Example 4 *Consider the network in Figure 3.8. The desired rate of each link is* $\frac{2}{5}$. *The sum of rates of any three consecutive links is equal to* $\frac{2}{5}+\frac{2}{5}+\frac{2}{5}=\frac{6}{5}$. *Thus, it violates the condition*

There exists $j \in \{1, 2, \ldots n-1, n\}$ *such that*

1. $r_{j-2} + r_{j-1} + r_j \leq 1$ *and*
2. $r_{j-1} + r_j + r_{j+1} \leq 1$

stated in Theorem 5. So we cannot obtain a feasible schedule using the method described in section 3.4.1. However there exists a solution by shifting the Regular Sequences. We will show the method in this section.

First, we define the cyclic shifting function $f(x)$.
Let $f(x)$ be the right-shifting function which operates on a binary sequence $x = a_1 a_2 \ldots a_n$ in following way:

$$f(a_1 a_2 a_3 ... a_n) = a_n a_1 a_2 ... a_{n-1}.$$

Define $f^{(k)}(x)$ as follows:

$$f^{(k)}(x) = \underbrace{f \circ f \circ \ldots f}_{k \ terms}(x).$$

e.g. If $x = 10100$, then $f(x) = 01010$, $f^{(3)}(x) = 10010$.
Define $f^{(0)}(x) = x$ and $f^{(1)}(x) = f(x)$.
Define $a_0 = a_n$ and $a_{n+1} = a_1$.

Lemma 1 *Let the binary sequence $x = a_1 a_2 ... a_n$ and $f(x) = b_1 b_2 ... b_n$. If*

$$a_{i+1} = 0 \ \forall i \in \{i : a_i = 1\}.$$

then

$$\{i : a_i = 1\} \cap \{j : b_j = 1\} = \emptyset.$$

Proof
Since

$$f(x) = b_1 b_2 b_3 ... b_n = a_n a_1 a_2 ... a_{n-1}.$$

So,

$$b_i = a_{i-1}.$$

Thus,

$$\begin{aligned} \{i : a_i = 1\} \cap \{j : b_j = 1\} &= \{i : a_i = 1\} \cap \{j : a_{j-1} = 1\} \\ &= \{i : (a_i = 1) \wedge (a_{i-1} = 1)\} \\ &= \{i + 1 : (a_{i+1} = 1) \wedge (a_i = 1)\} \\ &= \emptyset \ (\text{Since } a_{i+1} = 0 \ \forall i \in \{i : a_i = 1\}). \end{aligned}$$

Lemma 2 *Let the binary sequences* $x = a_1a_2...a_n$ *and* $f(x) = b_1b_2...b_n$. *If*

$$a_{i+1} = 0 \ \forall i \in \{i : a_i = 1\}$$

then

$$b_{i+1} = 0 \ \forall i \in \{i : b_i = 1\}.$$

Proof
Since

$$f(x) = b_1b_2b_3...b_n = a_na_1a_2...a_{n-1}.$$

So,

$$a_i = b_{i+1}.$$

Thus,

$$\begin{aligned} a_{i+1} = 0 \ \forall i \in \{i : a_i = 1\} &\Rightarrow a_i = 0 \ \forall i \in \{i : a_{i-1} = 1\} \\ &\Rightarrow b_{i+1} = 0 \ \forall i \in \{i : b_i = 1\}. \end{aligned}$$

■

Lemma 3 *Let the binary sequences* $x = a_1a_2...a_n$ *and* $f^{(n-1)}(x) = c_1c_2...c_n$. *If*

$$a_{i+1} = 0 \ \forall i \in \{i : a_i = 1\}$$

then

$$\{i : a_i = 1\} \cap \{j : c_j = 1\} = \emptyset.$$

Proof
Since

$$f^{(n-1)}(x) = c_1c_2...c_{n-1}c_n = a_2a_3...a_na_1$$

So,

$$c_i = a_{i+1}$$

Thus,

$$\begin{aligned}\{i : a_i = 1\} \cap \{j : c_j = 1\} &= \{i : a_i = 1\} \cap \{j : a_{j+1} = 1\} \\ &= \{i : (a_i = 1) \wedge (a_{i+1} = 1)\} \\ &= \emptyset \text{ (Since } a_{i+1} = 0 \ \forall i \in \{i : a_i = 1\})\end{aligned}$$

■

Theorem 6 *For a ring network with n edges where n is an odd number, if the rate of each link is equal to $\frac{\lfloor \frac{n}{2} \rfloor}{n}$, we can schedule the links by assigning the sequence x to the first link, $f(x)$ to the second link, and $f^{(i-1)}(x)$ to the i-th link $\forall i \in \{3, 4, \ldots n\}$ where x is the Regular Sequence with the rate $\frac{\lfloor \frac{n}{2} \rfloor}{n}$.*

Proof
By the algorithm we first construct the regular sequence, with rate $\frac{\lfloor \frac{n}{2} \rfloor}{n}$. That is $\underbrace{101010...10}_{\lfloor \frac{n}{2} \rfloor \text{ consecutive '10'}} 0$. Let $y = y_1 y_2 \ldots y_n$ represent this sequence.
Let $S(m)$ be the statement that

$$\{i : a_{m-1,i} = 1\} \cap \{j : a_{m,j} = 1\} = \emptyset$$

and

$$a_{m-1,i+1} = 0 \ \forall i \in \{i : a_{m-1,i} = 1\},$$

where

$$\begin{aligned}f^{(m-1)}(x) &= a_{m-1,1} a_{m-1,2} \ldots a_{m-1,n} \text{ and} \\ f^{(m)}(x) &= a_{m,1} a_{m,2} \ldots a_{m,n}.\end{aligned}$$

As $y_{i+1} = 0 \ \forall i \in \{i : y_i = 1\}$, by Lemma 1, $S(1)$ is true.

Assume $S(k-1)$ is true, i.e.

$$\{i : a_{k-2,i} = 1\} \cap \{j : a_{k-1,j} = 1\} = \emptyset$$

and

$$a_{k-2,i+1} = 0 \ \forall i \in \{i : a_{k-2,i} = 1\}$$

where

$$\begin{aligned} f^{(k-2)}(x) &= a_{k-2,1}a_{k-2,2}\ldots a_{k-2,n} \quad \text{and} \\ f^{(k-1)}(x) &= a_{k-1,1}a_{k-1,2}\ldots a_{k-1,n} \end{aligned}$$

for some integers k.
As

$$a_{k-2,i+1} = 0 \ \forall i \in \{i : a_{k-2,i} = 1\}$$

by Lemma 2,

$$a_{k-1,i+1} = 0 \ \forall i \in \{i : a_{k-1,i} = 1\}.$$

Together with Lemma 1,

$$\{i : a_{k-1,i} = 1\} \cap \{j : a_{k,j} = 1\} = \emptyset$$

where $f^{(k)}(x) = a_{k,1}a_{k,2}\ldots a_{k,n}$.
So $S(k)$ is true.
By mathematical induction, $S(m)$ is true for all $m \in \{1, 2, \ldots n-1\}$.

By Lemma 3,

$$\{i : y_i = 1\} \cap \{j : b_j = 1\} = \emptyset$$

where $y = y_1y_2\ldots y_n$ and $f^{(n-1)}(y) = b_1b_2\ldots b_n$.
Thus, by assigning the sequence y to the first link, $f(y)$ to the second link, and $f^{(i-1)}(y)$ to the i-th link $\forall i \in \{3, 4, \ldots n\}$, the schedules of any two neighbouring links would not have conflict and the required data rate can be achieved.

■

Note that for a ring network with n edges, if the rates of all the links are the same, the maximum achievable throughput of each link would be $\frac{\lfloor \frac{n}{2} \rfloor}{n}$. It is because at any one time, we can only activate at most $\lfloor \frac{n}{2} \rfloor$ links. So the maximum throughput of each links would be $\frac{\lfloor \frac{n}{2} \rfloor}{n}$. By using the Regular Sequences, we can achieve that upper bound.

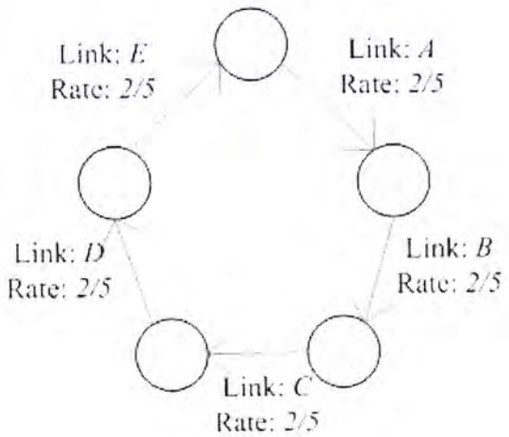

Figure 3.9: Example of a ring network with odd-number of edges and equal rate on each link.

Example 5 *Consider the network in Figure 3.9. The desired rate of each link is* $\frac{2}{5}$. *The Regular Sequence of* $\frac{2}{5}$ *is* 10100. *By shifting the sequences* $x = 10100$, *we would have the following schedules.*

Link	Schedule
A	$x = 10100$
B	$f(x) = 01010$
C	$f^{(2)}(x) = 00101$
D	$f^{(3)}(x) = 10010$
E	$f^{(4)}(x) = 01001$

Table 3.6: Schedules for example 5.

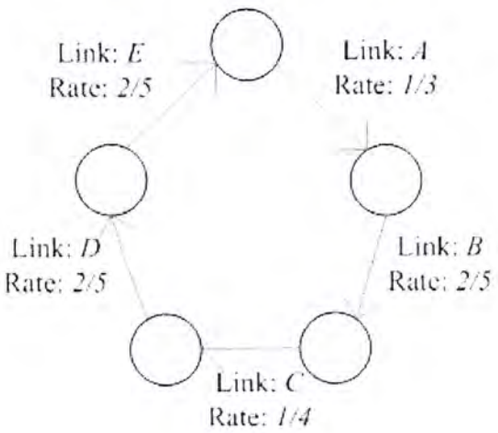

Figure 3.10: Example of a ring network with odd-number of edges and different rates on different links.

Example 6

Consider the network in the Figure 3.10. The rates of links are $\frac{1}{3}$, $\frac{2}{5}$, $\frac{1}{4}$, $\frac{2}{5}$ *and* $\frac{2}{5}$ *respectively. To construct the schedules, we assign the sequences* x, $f(x)$, $f^{(2)}(x)$, $f^{(3)}(x)$, $f^{(4)}(x)$ *to the first, second, third, fourth and fifth link respectively where* x *is the sequence 10100. Since the third link only require the rate of* $\frac{1}{4}$, *we have to convert some of the '1' of the sequence into '0'. First we calculate the L.C.M.(Least Common Multiple) of 5 and 4,* m, *which is equal to 20. Then we construct the sequence* y *by concatenating 4 regular sequences which was assigned previously. Thus,* $y = 00101001010010100101$. *There are eight 1's in the sequence* y. *To reduce the rate from* $\frac{2}{5}$ *to* $\frac{1}{4}$, *we have to remove*

three of the 1's in the sequence. We can choose any three of the them. If we want to remove the 1's more evenly, consider the subsequence of y consisting only the 1's. The length is 8. Construct a regular sequence z of rate $\frac{5}{8}$. *So* $z = 01010111$, *that means the second, fourth, sixth, seventh and eighth position of the sequence z is '1'. Thus, the first, third and fifth '1' of the sequence y should convert to '0' while the second, fourth, sixth, seventh and eighth '1' of the sequence y should remain '1'. The new sequence* $w = 00001000010000100101$ *would be the scheduling sequence of the third link, where the rate would equal to* $\frac{1}{4}$ *which would be our desired rate.*

Link	A	C
Desired Rate	$\frac{1}{3}$	$\frac{1}{4}$
m	15	20
y	10100 10100 10100	00101 00101 00101 00101
Rate of z	$\frac{5}{6}$	$\frac{5}{8}$
z	011111	01010111
w	00100 10100 10100	00001 00001 00001 00101

Table 3.7: Construction of the schedules for links A and C in example 6.

Link	Schedule
A	$\underbrace{00100101001010}\underbrace{0}\underbrace{01001010010100}\ldots$
B	$\underbrace{01010}\underbrace{01010}\ldots$
C	$\underbrace{00001000010000100101}\underbrace{0000100001000010010}\underbrace{1}\ldots$
D	$\underbrace{00101}\underbrace{00101}\ldots$
E	$\underbrace{10010}\underbrace{10010}\ldots$

Table 3.8: Schedules for example 6.

Here, we introduce Algorithm 5 which is used to construct a sequence(not a Regular Sequence) with rate $\leq \frac{\lfloor \frac{n}{2} \rfloor}{n}$ from the Regular Sequence with rate $= \frac{\lfloor \frac{n}{2} \rfloor}{2}$.

Algorithm 5

Input:

n: *Number of edges in the odd-edge ring network*

s: *Order of the link in the ring network*

$\frac{p}{q}$: *Desired rate of the output sequence, where* $\frac{p}{q} \leq \frac{\lfloor \frac{n}{2} \rfloor}{n}$

Output:

w: *Sequence with rate* $= \frac{p}{q}$

Process:

1. *Construct the regular sequence of* x, *which is equal to* $\underbrace{1010\ldots10}_{\lfloor \frac{n}{2} \rfloor \text{ consecutive '10'}}0$.

2. *Let* m *be the L.C.M. (Least Common Multiple) of* n *and* q.

3. *Let* y *be the sequence by concatenating* $f^{(shift)}(x)$ *for* m/n *times. (i.e. If* $f^{(s)}(x) = a_1a_2\ldots a_n$, *then* $y = \underbrace{a_1a_2\ldots a_na_1a_2\ldots a_n\ldots\ldots a_1a_2\ldots a_n}_{\frac{m}{n} \text{ consecutive '}a_na_1a_2\ldots a_n\text{'}}$.)

4. *Convert any* $(\lfloor \frac{n}{2} \rfloor \cdot \frac{m}{n} - p \cdot \frac{m}{q})$ *1's into 0's. Let the converted sequence be* w. *Note that the rate of* $w = \frac{p}{q}$. *(Number of 1's in* $y = \lfloor \frac{n}{2} \rfloor \cdot \frac{m}{n}$. *By converting* $(\lfloor \frac{n}{2} \rfloor \cdot \frac{m}{n} - p \cdot \frac{m}{q})$ *1's into 0's, number of 1's remains* $= \lfloor \frac{n}{2} \rfloor \cdot \frac{m}{n} - (\lfloor \frac{n}{2} \rfloor \cdot \frac{m}{n} - p \cdot \frac{m}{q}) = p \cdot \frac{m}{q}$. *So the rate of* $w = (p \cdot \frac{m}{q})/m = \frac{p}{q}$.

5. *Output the sequence* w

Remarks:

If we want the '1's distributed on the output sequence more evenly, step 4 of the algorithm can be replaced by the following:

4a *Let z be the Regular Sequence $s_{p\prime,q\prime}$ where $p\prime = p \cdot \frac{m}{q}$ and $q\prime = \lfloor \frac{n}{2} \rfloor \cdot \frac{m}{n}$. Note that rate of $z = r_z < 1$. (Since $r_z = \frac{p \cdot \frac{m}{q}}{\lfloor \frac{n}{2} \rfloor \cdot \frac{m}{n}} = \frac{p \cdot n}{\lfloor \frac{n}{2} \rfloor \cdot q} < \frac{p/q}{(\frac{n}{2}+1) \cdot \frac{1}{n}} = \frac{p/q}{\frac{1}{2}+\frac{1}{n}} < \frac{p}{q} \leq 1$).*

4b *Construct the Regular Sequence $z = z_1 z_2 \ldots z_{q\prime}$, denote $K = \{i : z_i = 0\}$. So, $|K| = q\prime - p\prime$.*

4c *Convert the i-th '1' of sequence y into '0' where $i \in K$. Let the modified sequence be w. Note that rate of $w = \frac{p}{q}$, (Number of 1's in $y = q\prime$. By converting $(q\prime - p\prime)$ 1's into 0's, number of 1's in $w = q\prime - (q\prime - p\prime) = p\prime$. So rate of $w = \frac{p\prime}{m} = \frac{p \cdot \frac{m}{q}}{m} = \frac{p}{q}$).*

Theorem 7 *For a ring network with n edges where n is an odd number, if all the rates of the links are less than or equal to $\frac{\lfloor \frac{n}{2} \rfloor}{n}$, one can obtain a feasible schedule by shifting the Regular Sequences and using Algorithm 5.*

Proof

By Theorem 6, we can do the scheduling using the Regular Sequences if all the rates $= \frac{\lfloor \frac{n}{2} \rfloor}{n}$. By converting some of the 1's of the sequence into 0's, we can achieve the desired rate which is less than or equal to $\frac{\lfloor \frac{n}{2} \rfloor}{n}$. Algorithm 5 is used to do the conversion.

■

3.5 Discussion

From the previous sections, we can see that by using the Regular Sequences, a feasible schedule can be obtained on ring network with even-number of edges. For ring network with odd-number of edges, we can obtain a feasible schedule with Regular Sequences and their splitted versions. Besides, due to the nature of the Regular Sequences, there is at least a '1' within the first two digits in any sequence. That means the nodes in the the ring network can start to transmit the data to their neighbours within the first two timeslots.

Also, the '1' are spread quite evenly on the sequences, which means the delay jitter between any two packets on a link is small. This is important for some internet applications such as video streaming or VOIP. Thus, scheduling using the Regular Sequences can provide some benefits in terms of low delay jitter.

3.6 Conclusion

In this chapter, we provide a scheduling algorithm on ring networks. We can see that such algorithm can utilize the network capacity by just using two sets of Regular Sequences. Furthermore, it can provide benefits on some internet applications in which packet delay jitter is one of the important considerations.

□ **End of chapter.**

Chapter 4

Distributed Scheduling Algorithm for Ad Hoc Network

Summary

Scheduling in ad hoc network with arbitrary topology is a NP Complete Problem, even with a centralized scheduler. In this part, a distributed heuristic algorithm is presented for scheduling, which would provide optimal results in some kinds of special networks.

4.1 Problem Formulation

Consider an ad hoc network with an arbitrary topology. Due to the interferences from other links, when one link is active, neighboring links cannot be active at the same time. Suppose each link has specific traffic demands. How to allocate the minimal number of timeslots to satisfy the traffic demands is an open problem. In this section of the thesis, a distributed algorithm is proposed to allocate the timeslots efficiently.

4.2 Distributed Scheduling Heuristic Algorithm

4.2.1 Weight functions

First, we introduce a metric which is going to be used in the proposed algorithm. For the conflict graph $G(V, E)$, define the weight funciton $W(i)$ for each node $i \in V$. Let r_i be the traffic demands of node $i \in V$. We propose three ways of calculating the weight functions:

1. $W_1(i) =$ a random number $\in [0, 1]$
2. $W_2(i) = r_i$
3. $W_3(i) = r_i + \sum_{j:(i,j)\in E} r_j$.

Note that for these three weight functions, the values are all finite real numbers.

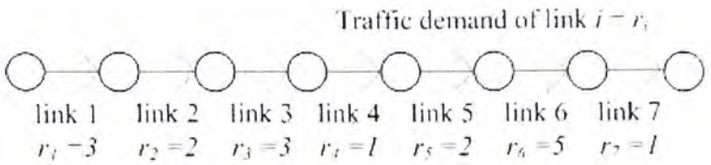

Figure 4.1: Connectivity graph of a chain network.

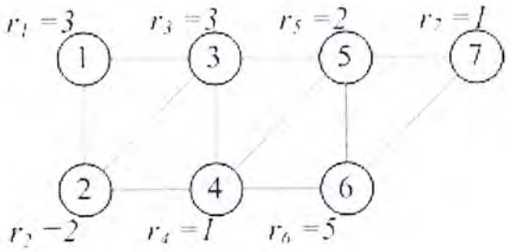

Figure 4.2: Conflict graph of the chain network.

Example 7 *Consider the chain network in Figure 4.1. First, transform the graph into the conflict graph (Figure 4.2). Next we can calculate the weights function* $W(i)$ *for* $i = 6$ *as follows:*

1. $W_1(6) = 0.8355$
2. $W_2(6) = 5$
3. $W_3(6) = 5 + 1 + 2 + 1 = 9.$

4.2.2 Main Algorithm

Here we present our algorithm:

Algorithm 6

1. *For the source node of each link l_i in the connectivity graph, construct a local conflict graph, which is a conflict graph only with nodes (links in connectivity graph) interfering with link l_i, according to the interference model and the network topology. Assign a unique ID with each node in the conflict graph.*
2. *For each node in the conflict graph(i.e. the source node of the link) calculates their weights, which is a finite number, according to the formula described in the previous section. (Note that a node in the conflict graph corresponds to an edge in the original network topology.)*
3. *Repeat the following until all traffic demands are satisfied For Each timeslot,*
   (a) *Each node in the conflict graph exchanges its weight with their neighbors.*
   (b) *Each node in the conflict graph compares its weight with their neighbors. If the node finds that its weight is the largest among others, the timeslot is reserved for this node. If there is a tie, the unique ID of the node will be compared instead.*
   (c) *The nodes which have been allocated the timeslot will send one unit of data.*
   (d) *One unit of traffic demand will be deducted for those nodes which have been allocated the timeslot.*
   (e) *Update the traffic demands and the weights of each node.*

(f) If the traffic demand of a node is equal to zero, the neighboring nodes will remove that node from their local conflict graphs.

By using Algorithm 6, the following results hold:

Theorem 8 *In each timeslot, at least one traffic demand will be satisfied. Hence, the algorithm terminates in a finite number of steps for a network with finite amount of traffic.*

Proof
Since the weights of the nodes in the conflict graph are finite numbers, there must exist a node with weight larger than or equal to its neighbours. If the weight of a node is larger than its neighbours, the node can gain the timeslot. If the weights are equal, the unique ID will be the tie breaker. So at least one unit of traffic demand will be satisfied at each timeslot. As the weights of the nodes are finite numbers, after finite number of timeslots, all the traffic demand in the network will be satisfied. As a result, the proposed algorithm will be terminated.

■

Lemma 4 *Let $G(V, E)$ be the conflict graph. If $(i, j) \in E$, then timeslot will not be allocated for i and j at the same time.*

Proof
Let the weights of i, j where $(i, j) \in E$, be $W(i)$ and $W(j)$ respectively. If two contending links in the network topology are activated at the same time, that means $W(i) > W(j)$ and $W(j) > W(i)$. This leads to contradiction.

■

Theorem 9 *If the conflict graph $G(V, E)$ is a complete graph, i.e. $(i \in V \wedge j \in V) \Rightarrow (i, j) \in E$, by using the Algorithm 6, the number of timeslots used is minimal.*

Proof

For a complete graph, each node is the neighbour of all other nodes. Theoretically, in each timeslot, at most one unit of traffic demand of a node can be satisfied. So minimum number of timeslots required $= T_{ideal} = \sum_{i \in V} r_i$ where r_i is the traffic demand of node i. For the proposed distributed algorithm, by Theorem 8, at least one traffic demand will be satisfied in each timeslot. Since G is a complete graph, by Lemma 4, at most one traffic demand will be satisfied in each timeslot. As a result, one and only one traffic demand will be satisfied in each timeslot. Thus, the number of timeslots required by the proposed distributed algorithm $= \sum_{i \in V} r_i = T_{ideal}$. So the Algorithm 6 will use the minimal number of timeslots.

■

4.3 Centralized algorithm on a chain network

Here, we compare the proposed distributed algorithm with the optimal one. Generally speaking, scheduling the traffic for an ad hoc network with an arbitrary topology is an NP Complete Problem. However, for a chain network, with the following interference model, we can find the minimal timeslots required to satisfy the traffic demands of each link.

1. The nodes are separated in equal distance d.
2. Each node is equipped with one radio interface with omni-directional antenna.
3. Each node will transmit data to the node in their right-hand-side which is one hop away.
4. The transmission range is equal to the interfering range, which is a circle with radius equal to d.

Let r_i be the traffic demands of link i. From [27], if the transmission topology is a forest, i.e. there is no cycles in it, there has a centralized algorithm to schedule the traffic demands of the links in an optimal way. The minimal number of timeslots required is equal to the magnitude of the maximum contention clique (mcc), which is denoted as $|mcc|$. (Contention clique is a set of links such that any two links within the set interfere with each other; the contention clique with maximum size is called the mcc, its size is denoted as $|mcc|$). Consider the conflict graph of a chain network in the figure:

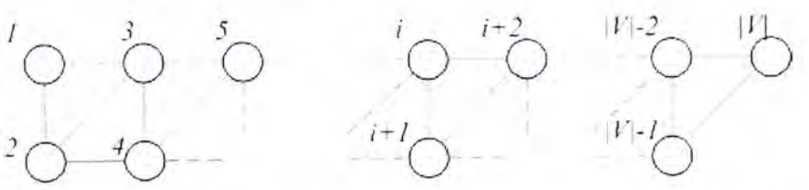

Figure 4.3: Conflict graph of the chain network used in algorithm 7.

Algorithm 7 *[27]*
Here we present the optimal centralized algorithm to allocate the timeslots in a chain network according to the idea stated in [27]:

1. *Locate the maximum contention clique* $(i, i+1, i+2)$. *i.e. find the* i *such that* $r_i + r_{i+1} + r_{i+2}$ *is maximum. If there is a tie, just choose one arbitrary.*

2. *Assign timeslots to nodes* i, $i+1$ *and* $i+2$. *Let the number of timeslots used* $= T'$

3. *for* $j := i+3$ *to* $|V|$ *do*
   Assign r_j *out of* T' *timeslots to node* j. *(Note that it is always possible as* $r_{j-2} + r_{j-1} + r_j \leq T'$. *If* $r_{j-2} + r_{j-1} + r_j > T'$, *it will contradict with step 1 that* $(i, i+1, i+2)$ *is the maximum contention clique.)*

4. *for* $j := i-1$ *downto 1 do*
   Assign r_j *out of* T' *timeslots to node* j. *(Note that it is always possible as* $r_j + r_{j+1} + r_{j+2} \leq T'$. *If* $r_j + r_{j+1} + r_{j+2} > T'$, *it will contradict with step 1 that* $(i, i+1, i+2)$ *is the maximum contention clique.)*

The timeslots used $= T'$, *which is minimal as we cannot allocate timeslots less than* T' *for contention clique* $(i, i+1, i+2)$. *So this algorithm can produce the optimal scheduling.*

4.4 Performance of the Algorithm on Chain Network

In this section, we compare the performance of the proposed distributed algorithm with the centralized algorithm.

Figure 4.4: A chain network with 20 edges.

4.4.1 Comparison 1

Consider a long chain network with 20 edges as in Figure 4.4. Suppose r_i is the traffic demand of link l_i. We performed the following simulation:

1. For each link i, assign r_i an integer uniformly distributed from 1 to 10.
2. Calculate the $|mcc|$ of that network.
3. Use the distributed algorithm to find the total number of timeslots, T, that is required for satisfying all the traffic demands the network.
4. Compute the efficiency, eff, of proposed the distributed algorithm.
5. Simulate the network for 5000 times with different r and get the average eff.

Here the efficiency,eff, is calculated as follows:

$$\begin{aligned} eff &= \frac{\text{throughout achieved by the proposed distributed algorithm}}{\text{throughput achieved by the centralized algorithm}} \\ &= \frac{\sum_{i=1}^{20} r_i/T}{\sum_{i=1}^{20} r_i/|mcc|} \\ &= \frac{|mcc|}{T}. \end{aligned}$$

The following table summarizes the simulation results: From

Weight Function	Efficiency
W_1	0.7581
W_2	0.7879
W_3	0.8254

Table 4.1: Simulation results of comparison 1.

the result, we can see that by using W_3, a higher utilization of the bandwidth can be achieved.

4.4.2 Comparsion 2

Let r_x be the vector
[10 10 10 10 10 10 10 10 10 10 (10+x) 10 10 10 10 10 10 10 10 10],
where the elements of the vector are representing the traffic demands of the links. For example, if $x = 3$, $r_x = r_3 =$
[10 10 10 10 10 10 10 10 10 10 13 10 10 10 10 10 10 10 10 10],
which means the traffic demands of link 11 is equal to 13 while the traffic demand of other links = 10.

For each x, we simulate the network for 1000 times with different unique IDs for the links and get the average value of eff. We plot the graph of eff against x in Figure 4.5.

From the result, we can see that by using W_3, a higher utilization of the bandwidth can be achieved.

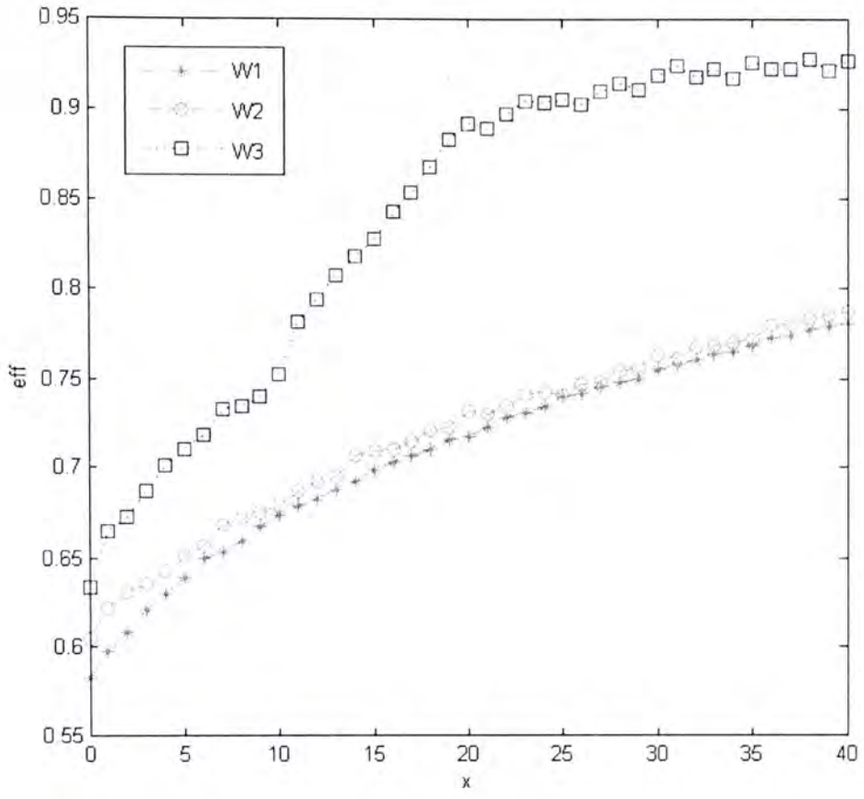

Figure 4.5: Comparsion 2: Plot of eff against x.

4.4.3 Comparsion 3

Let r_x be the vector
[10 10 10 10 10 10 (10+x) 10 10 10 10 10 10 (10+x) 10 10 10 10 10 10],
where the elements of the vector are representing the traffic demands of the links. For example, if $x = 3$, $r_x = r_3 =$
[10 10 10 10 10 10 13 10 10 10 10 10 10 13 10 10 10 10 10 10],
which means the traffic demands of links 7 and 14 are equal to 13 while the traffic demand of other links = 10.

For each x, we simulate the network for 1000 times with different unique IDs for the links and get the average value of eff.

We plot the graph of eff against x in Figure 4.6.

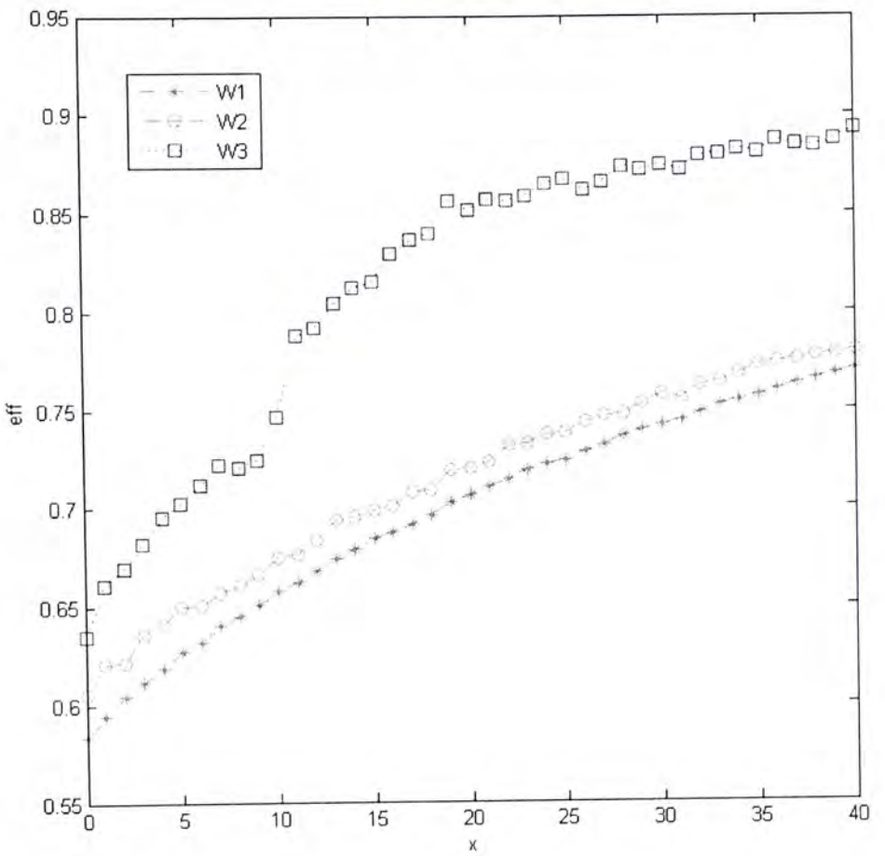

Figure 4.6: Comparsion 3: Plot of eff against x.

From the result, again we can see that by using W_3, a higher utilization of the bandwidth can be achieved.

4.5 Performance of the Algorithm on Random Conflict graph

In this section, we present results comparing the performance obtained using different weight functions in a random conflict graph.
We performed the following simulation:

1. Consider a conflict graph $G' = (V', E')$ with 20 nodes.
2. $\forall i, j \in V'$, $(i, j) \in E'$ with probability p
3. The traffic demand r_i is equal to 10 for all i
4. Calculate the number of timeslots required to satisfy all the traffic demands in the network if we use the proposed distributed algorithm with weight w_1, denoted it as T_1
5. Calculate the number of timeslots required if weight w_2 is used, denoted it as T_2
6. Calculate the number of timeslots required if weight w_3 is used, denoted it as T_3
7. Compute the relative efficiency $re_2 = \frac{T_2}{T_1}$
8. Compute the relative efficiency $re_3 = \frac{T_3}{T_1}$
9. For each p, simulate the network for 1000 times and get the average value of re_1 and re_2

Simulation result is shown in Figure 4.7.

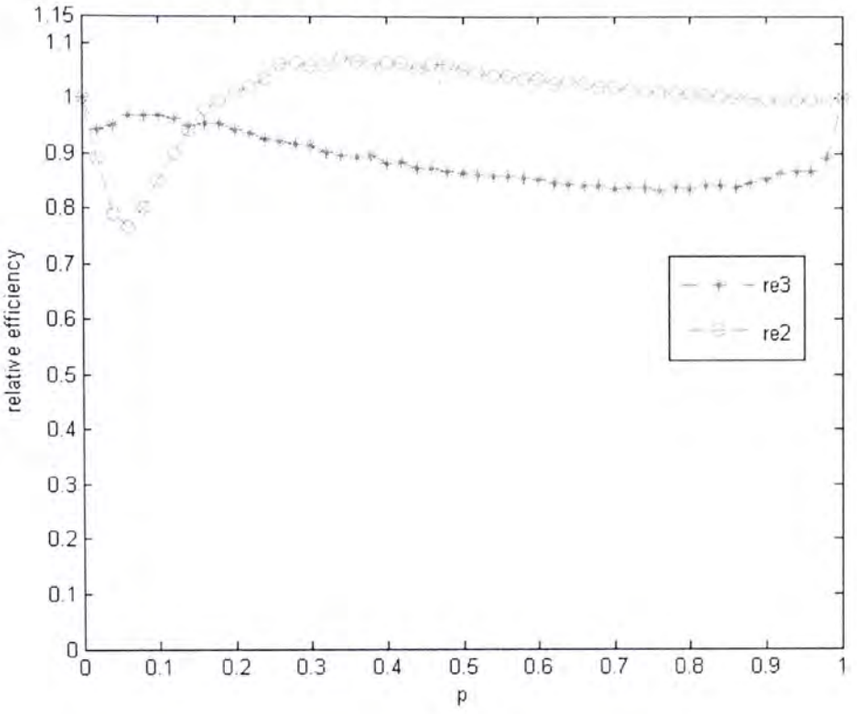

Figure 4.7: Plot of re_2 and re_3 against x.

From the graph, we can see that for a conflict graph which is medium or highly connected (2 nodes in the conflict graph are connected with probability > 0.18, using weight function $W_3(i)$ can have better performance than using $W_2(i)$.

When the traffic demand r_i of above simulation was changed to an integer uniformly distributed from 1 to 10, following results were obtained:

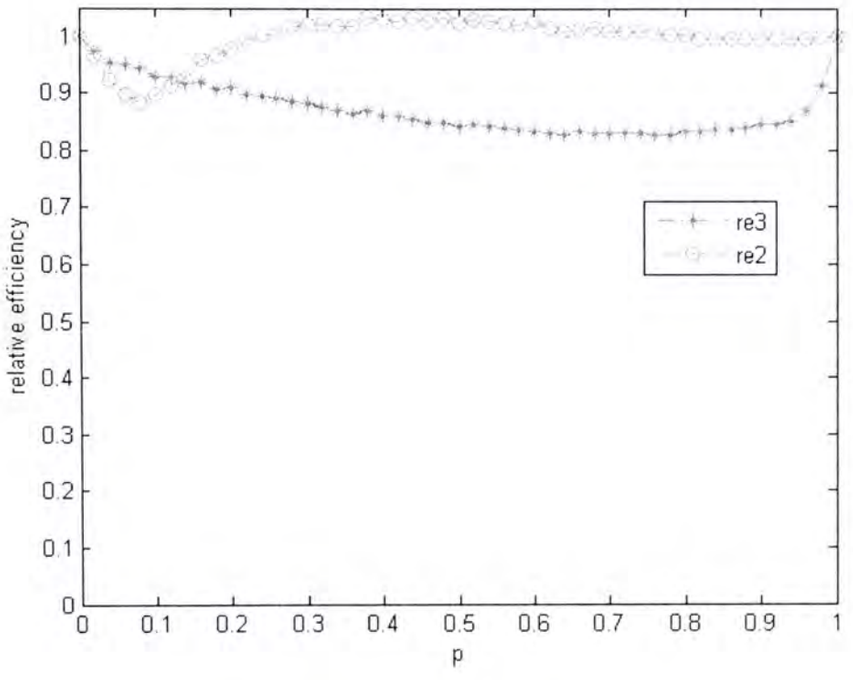

Figure 4.8: Plot of re_2 and re_3 against x.

Again, we can see that $W_3(i)$ can have a better performance than $W_2(i)$.

4.6 Discussion

From the simulations above, we can see that, by using the weight function W_3, the number of timeslots used is smaller than using W_1 and W_2. In a chain network, the efficiency can even go to 0.9. So we believe that the proposed distributed algorithm with the weight function W_3 is a good choice for scheduling timeslots in an wireless ad hoc network.

4.7 Special Graphs

In this section, we investigate on certain special graphs in which the proposed distributed algorithm can achieve the optimal schedule if weight W_3 is used.

Theorem 10 *Suppose the conflict graph of the ad hoc network consists of two complete graphs $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$, if $V_1 \cap V_2 \notin \emptyset$, then the proposed distributed algorithm achieves the same throughput as the optimal algorithm.*

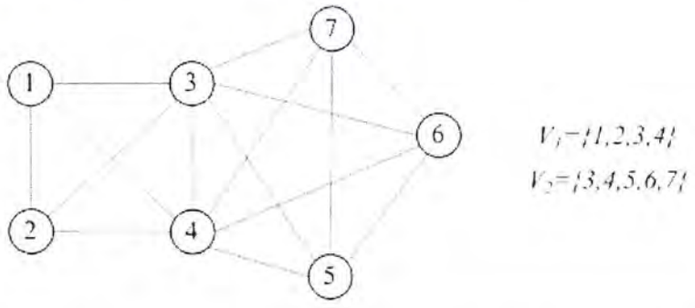

Figure 4.9: Two cliques with some nodes intersected.

Proof
Let A be the set containing the nodes $V_1 \cap V_2$. By the construction of the proposed distributed algorithm, the weights of the nodes in set A will be larger than the weights of the nodes in set $(V_1 \cup V_2) \setminus A$. So the traffic demands of the nodes in set A will be satisfied first. As the nodes in set A will form a complete graph, $\sum_{i \in A} r_i$ timeslots are required for satisfying the traffic demands in set A. After that, the graph will be broken into two separate complete graphs. The timeslots required for the node in these two complete graph will be $\sum_{i \in V_1 \setminus A} r_i$ and $\sum_{i \in V_2 \setminus A} r_i$ respectively. As timeslots can be allocated for these two complete graphs at the same time without any conflict, timeslots required in this stage $= max\left\{\sum_{i \in V_1 \setminus A} r_i, \sum_{i \in V_2 \setminus A} r_i\right\}$. So total

timeslots required for the distributed algorithm would be

$$\sum_{i \in A} r_i + max \left\{ \sum_{i \in V_1 \setminus A} r_i, \sum_{i \in V_2 \setminus A} r_i \right\}$$

where $A = V_1 \cap V_2$

The number of timeslots required is the same as the optimal one. Since each node in set A will conflict with all other nodes in the graph, if a timeslot is allocated for a node in set A, all other nodes cannot use this timeslot. So $\sum_{i \in A} r_i$ timeslots are required for allocating the traffic demands in set A. After allocating the timeslots in set A, the graph will be broken into two complete graphs. Extra timeslots required $= max \left\{ \sum_{i \in V_1 \setminus A} r_i, \sum_{i \in V_2 \setminus A} r_i \right\}$. So the total timeslots for the proposed distributed algorithm is the same as the optimal one.

■

Theorem 11 *Suppose the conflict graph of the ad hoc network consists of three complete graphs $G_1(V_1, E_1)$, $G_2(V_2, E_2)$ and $G_3(V_3, E_3)$. Let $A = V_1 \cap V_2$, $B = V_2 \cap V_3$ and $C = V_1 \cap V_3$. If $A \notin \emptyset$, $B \notin \emptyset$ and $C = \emptyset$. Then the proposed distributed algorithm achieves the optimal result if the following condition holds:*

$$max \left\{ \sum_{i \in V_1 \setminus A} r_i, \sum_{i \in V_3 \setminus B} r_i, \sum_{i \in V_2 \setminus (A \cup B)} r_i \right\} = \sum_{i \in V_2 \setminus (A \cup B)} r_i$$

Proof

To get the centralized optimal scheduling for such kind of graph, suppose the nodes in the set A and B get the timeslots first. While assigning timeslots to nodes in set A, we can

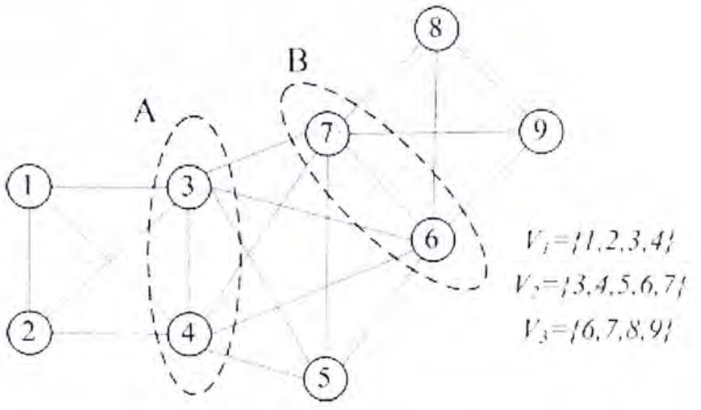

Figure 4.10: Three cliques with some nodes intersected.

schedule the timeslots for nodes in $V_3 \setminus B$ as well. For the same reason, while we are assigning timeslots to nodes in set B, timeslots can be scheduled to nodes in $V_1 \setminus A$ as well. After assigning the timeslots in A and B, the graph will become 3 separate cliques. And timeslots can be assigned to nodes in the 3 cliques in the same time. So if
$max\left\{\sum_{i\in V_1\setminus A} r_i, \sum_{i\in V_3\setminus B} r_i, \sum_{i\in V_2\setminus(A\cup B)} r_i\right\} = \sum_{i\in V_2\setminus(A\cup B)} r_i$,
then the total number of timeslots required =

$$\sum_{i\in A} r_i + \sum_{i\in B} r_i + \sum_{i\in V_2\setminus(A\cup B)} r_i$$

For the distributed algorithm, nodes either in the sets A and B will get the highest weight, so timeslots will be assigned to them first. While assigning timeslots in sets A and B, some nodes in $V_1 \setminus B$ and $V_2 \setminus A$ may get the timeslots. After all nodes in sets A and B have gained the timeslots, the graph will be separated into 3 cliques. So timeslots can be assigned to them in a parallel way. However, if
$max\left\{\sum_{i\in V_1\setminus A} r_i, \sum_{i\in V_3\setminus B} r_i, \sum_{i\in V_2\setminus(A\cup B)} r_i\right\} = \sum_{i\in V_2\setminus(A\cup B)} r_i$,
the total timeslots required using the proposed distributed algorithm =

$$\sum_{i\in A} r_i + \sum_{i\in B} r_i + \sum_{i\in V_2\setminus(A\cup B)} r_i$$

Thus, the distributed algorithm can produce the optimal result if the condition in the theorem holds.

■

So by using Theorems 10 and 11, the proposed distributed algorithm can produce the optimal scheduling in networks with certain special conflict graphs.

4.8 Conclusion

In this chapter, we propose a distributed heurisitic scheduling algorithm which works for networks with an arbitrary topology. Comparisions with the optimal centralized scheduling algorithm in chain networks are studied. Finally, we have studied certain kinds of special graphs in which the optimal scheduling can be achieved by using the proposed distributed algorithm.

□ **End of chapter.**

Chapter 5

Conclusion

Summary

In this part, we give a conclusion for the thesis.

To conclude, we have given the background study about scheduling in ad hoc network in chapter 2. Since scheduling on the network with an arbitrary topology is an NP Complete Problem, an optimal solution is very difficult to obtain.

In chapter 3, we introduce a new family of sequences, called regular sequence. By using the sequences, a simple centralized algorithm is presented which can produce optimal scheduling in ring networks if the number of edges is even. For the odd-edge ring network, if some conditions are hold, optimal scheduling can also be found. Due to the nature of the regular sequences, the schedule obtained by the algorithm can facilitate some kinds of applications such as video streaming or VOIP where the delay jitter between any two packets are required to be small in order to have a smooth data traffic.

In chapter 4, we have presented a distributed heuristic

algorithm which can do scheduling on an ad hoc network with an arbitrary topology. Comparisons with the optimal algorithm for chain networks are given. Moreover, we prove that the algorithm can produce optimal results for certain type of special network topologies.

□ **End of chapter.**

Bibliography

[1] A.J. Goldsmith, S.B. Wicker. Design challenges for energy-constrained ad hoc wireless networks. *IEEE Transactions on Wireless Communications*, Aug. 2002.

[2] Arash Behzad and Izhak Rubin. On the performance of graph-based scheduling algorithms for packet radio networks. In *Proc. IEEE GLOBECOM*, 2003.

[3] Arash Behzad and Izhak Rubin. Power controlled multiple access control for wireless access nets. In *Proc. IEEE VTC*, 2003.

[4] B. Hajek and G. Sasaki. Link scheduling in polynomial time. *IEEE Trans. Inform. Theory*, 34(5), Sept. 1988.

[5] Berge, C. *Graphs and Hypergraphs.* New York: Elsevier, 1973.

[6] C. Zhu and M. S. Corson. A five-phase reservation protocol (fprp) for mobile ad hoc networks. *Wireless Networks*, 7(4), July 2001.

[7] C.G. Prohazka. Decoupling link scheduling constraints in multihop packet radio networks. *IEEE Transactions on Computers*, 38, 1989.

[8] F.N. Ali, P.K. Appani, J.L. Hammond, and V.V. Mehta. Distributed and adaptive tdma algorithms for multi-hop mobile networks. In *Proc. IEEE MILCOM*, 2002.

[9] I. Chlamtac and A. Farago. Making transmission schedules immune to topology changes in multi-hop packet radio networks. *IEEE/ACM Trans. Networking*, Feb. 1994.

[10] I. Chlamtac and S. Pinter. Distributed nodes organization algorithm for channel access in a multihop dynamic radio network. *IEEE Trans. Comput.*, 1987.

[11] J. Gronkvist. Assignment methods for spatial reuse tdma. In *Proc. ACM MOBIHOC*, 2000.

[12] J. Gronkvist and A. Hansson. Comparison between graph-based and interference-based stdma scheduling. In *Proc. ACM MOBIHOC*, 2001.

[13] Ji-Her Ju, Victor O. K. Li. Tdma scheduling design of multihop packet radio networks based on latin squares. *IEEE Journal on Selected Areas in Communications*, 17(8), Aug. 1999.

[14] K. Jain, J. Padhye, V.N. Padmanabhan. Impact of interference on multi-hop wireless network performance. In *Proceedings ACM Mobicom 2003*, San Diego, CA, Sept. 2003.

[15] Lichun Bao. Mals: Multiple access scheduling based on latin squares. In *Proc. IEEE MILCOM*, 2004.

[16] M. Sekhar and K.N. Sivarajan. Routing and scheduling in packet radio networks. In *Proc. IEEE ICPWC*, 2000.

[17] M.R. Garey, D.S. Johnson. *Computers and Intractability: A Guide to the Thoery of NP-Completeness.* New York: Freeman, 1979.

[18] N. Funabiki and Y. Takefuji. A parallel algorithm for broadcast scheduling problems in packet radio networks. *IEEE Trans. Commun*, 41(6), 1993.

[19] P. Gupta and P.R. Kumar. The capacity of wireless networks. *IEEE Trans. Inform. Theory*, Mar. 2000.

[20] R. Nelson, L. Kleinrock. Spatial-tdma: A collision-free multihop channel access control. *IEEE Transactions on Communications*, 33(9), Sept. 1985.

[21] R. Ramanathan and E. L. Lloyd. Scheduling algorithms for multi-hop radio networks. *IEEE/ACM Trans. Networking*, Apr. 1993.

[22] R. Ramaswami and K. K. Parhi. Distributed scheduling of broadcasts in a radio network. In *Proc. of IEEE Conference on Computer Communications (INFOCOM)*, Ottawa, Ont., Canada, Apr. 1989.

[23] Rappaport T.S. *Wireless Communications: Principles and Practices, Second Edition.* Prentice Hall PTR, 2002.

[24] S. Kumar, V.S. Raghavan, and J. Deng. Medium access control protocols for ad hoc wireless networks: A survey. *Ad Hoc Networks*, 4(3), May 2006.

[25] T.S. Rappaport, K. Blankenship, and H. Xu. Propagation and radio system design issues in mobile radio systems for the glomo project. 1997.

[26] X. Ma and E.L. Lloyd. An incremental algorithm for broadcast scheduling in packet radio networks. In *Proc. IEEE MILCOM*, 1998.

[27] Zongpeng Li and Baochun Li. Improving throughput in multihop wireless networks. *IEEE Transactions on Vehicular Technology*, 55(3), May 2006.