Delft University of Technology
Master's Thesis in Electrical Engineering
Specialization (Telecommunication)

Optimization on Burst Allocation in D2DWRAN

Zhongxian Pan

embedded software

TU Delft Delft University of Technology

Optimization on Burst Allocation in D2DWRAN

Master's Thesis in Electrical Engineering (Specialization:Telecommunication)

Embedded Software Section
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
Mekelweg 4, 2628 CD Delft, The Netherlands

Zhongxian Pan
z.pan@student.tudelft.nl

11th September 2014

Author
Zhongxian Pan (Z.pan@student.tudelft.nl)
Title
Optimization on Burst Allocation in D2DWRAN
MSc presentation
11th September 2014
Supervisor
Dr.R.V.Prasad
Phd.H.Shi

Graduation Committee

Prof.dr.K.G.Langendoen	Delft University of Technology
Dr.R.V.Prasad	Delft University of Technology
Dr.ir.F.Kuipers	Delft University of Technology

Abstract

With rapid development of digital devices with wireless services, communication networks play a significant role in our daily lives. The explosive data demand makes it urgent to extend the capacity of the existing networks. Cognitive Radio (CR), which can sense and use the frequency band dynamically, can provide an efficient radio frequency usage. IEEE 802.22 working group has developed several standards based on CR technology leading to Wireless Regional Area Networks (WRANs). However, WRANs adopt the cellular topology in which all communication is in a point-to-multi-point (P2M) manner. This P2M communication limits the network capacity because all messages need to be routed by the Base Station (BS) even for intra-cell communication. Therefore, Device-to-Device (D2D) WRAN (D2DWRAN) has been proposed, which employs D2D communication into the WRAN.

A burst allocation is the allocation of resources for wireless communication to the users, and influences the network capacity significantly. With D2D communication, the burst allocation problem in D2DWRAN is different from it in WRAN. This thesis attempts to solve this new burst allocation problem. After introducing the progression from WRAN to D2DWRAN, the new burst allocation problem is stated and formulated mathematically in different scenarios. Then the existing optimization methods are studied and evaluated, such as Greedy Algorithm, Simulated Annealing Algorithm, Genetic Algorithm and Ant Colony Algorithm. Some modifications based on these algorithms are also proposed to solve the burst allocation problem in D2DWRAN. These modified algorithms are also simulated and the advantages and disadvantages of each of them are discussed and analysed.

Contents

Chapter 1

Introduction

In this chapter, the background of IEEE 802.22 and D2DWRAN will be introduced in detail. Further, the research about problem of this project and the structure of this thesis will be persented.

1.1 Background

1.1.1 IEEE 802.22

In recent years, digital devices with wireless services become extremely popular. The strategy to deal with proliferation of wireless service is still the topic of today's digital communication. Not only traditional communication network, such as Wi-Fi, mobile communication devices, and TV broadcast, nearly all electrical equipment need the network. In a home network, all devices are connected and share information. Thus, the capacity of network become critical. The unlicensed bands (e.g., industrial, scientific and medical (ISM) and Unlicensed National Information Infrastructure (U-NII)) play an important role in wireless communications since the deployment of applications in these bands is unused [1]. Under this situation, the regulatory bodies start to consider opening the licensed bands for unlicensed use. At the same time, it has been proved that the occupancy of spectrum in license bands is highly underutilized [2]. Therefore, Cognitive Radio (CR) is taken into consideration to increase the channels utilization [3].

CR is a radio that can be programmed and configured dynamically. It can detect available channels in wireless spectrum automatically, and then adapt its parameters to allow more wireless communications in a given spectrum band at one location. According to this technology, the spectrum can be used more flexibly and efficiently [4].

With all these facts, the TV band Notice of Proposed Rule Making (NPRM) was issued by the U.S. Federal Communications Commission (FCC) in May 2004 [5]. The NPRM proposes to allow unlicensed parties to use the TV broadcast bands if no harmful interference is caused to incumbent services. This proposal can be accomplished by employing CR-based technologies. IEEE 802.22 has been started to work on the usage of TV channels in a CR manner.

IEEE 802.22 proposes Wireless Regional Area Network (WRAN). It can use the same spectrum currently utilized by TV service [6]. The key technology-CR- allows users the standard to use the unused spectrum which is called TV White Spaces (TVWSs) [7] in TV bands on a non-interfering basis. The TV bands are from 54 to 862 MHz [8].

1.1.2 Structure of WRAN

Topology

WRANs network establishes a point-to-multipoint topology with one base station (BS) and multiple customer premise equipment (CPEs) in a cell. The BS controls the whole cellular network and makes the decision which channels are available and what information should be broadcasted. CPEs have an omni-directional antenna for sensing and Geo-location information, and a directional antenna for communication with BS [9].

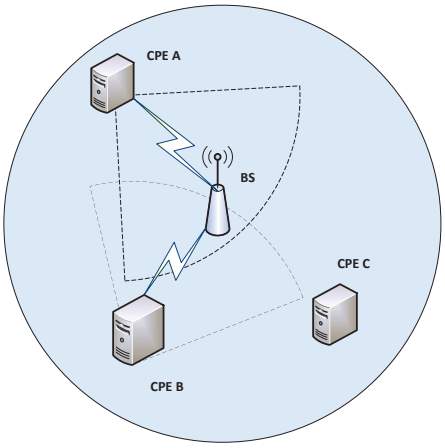


Figure 1.1: WRAN standard cell.

In IEEE 802.22 network, both the BS and CPEs sense the spectrum in quiet period of a superframe, and the information is sent to the BS. Only the BS can decide the usage of channels via a classification mechanism [7]. Figure 1.1 shows a simple standard WRAN cell. In each time slot, only one communication can happen in the same cell. So if CPE A wants to send data to CPE B, there are four steps:

Step 1 CPE A sends a channel request to the BS.

Step 2 The BS allocates slots in upstream subframe and receives the message that CPE A sends to CPE B.

Step 3 BS allocates slots for the messages from CPE A in downstream subframe and sends the message to CPE B.

Each step needs one time slot, so the whole procedure requires at least three time slots to complete. The two key characteristic of standard WRAN network are shown up in this example.

- Direct connection between two CPEs in the same cell are not allowed. All the connections needs to be linked to the BS.
- Very limited connections can be established because the BS can only deal with one connection in order to avoid interference.

However, this kind of network can limit the performance on channel capacity and channel usage.

Superframe Structure

Table I is the list of important concepts and terms regarding the channel allocation problem in this thesis [10]. We will use these terms to explain the superframe structure of WRAN and the research problem of channel allocation in this thesis.

In IEEE 802.22 network, orthogonal frequency-division multiple access (OFDMA) is used to make multiple CPEs able to access to the BS simultaneously in a cell. So a superframe structure and OFDMA is shown in Figure 1.2 [8]. A superframe contains 16 frames. A frame contains downstream (DS) subframe and an upstream (US) subframe. The channel is divided both in frequency domain and time domain. For convenience, one symbol in the channel can be called an OFDMA slot or a slot for short. The DS MAP and US MAP carry the burst allocation information. The upstream channel descriptor (UCD) and downstream channel descriptor (DCD) provide the channel information.

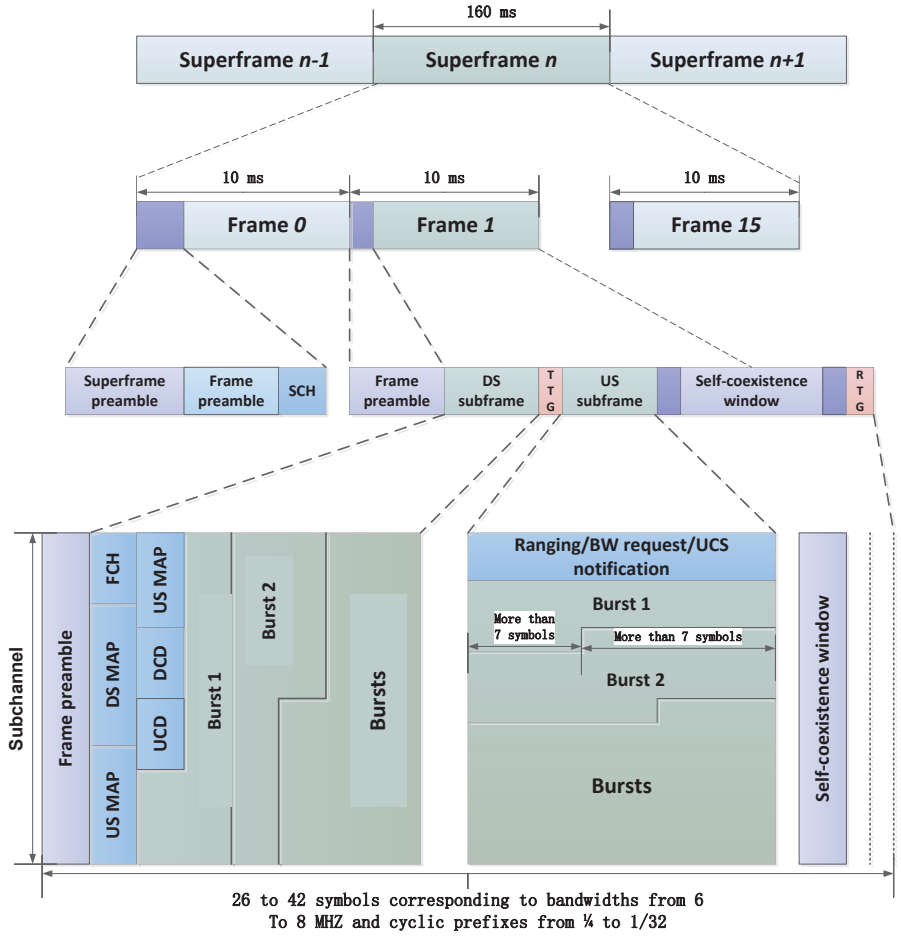


Figure 1.2: The superframe structure and OFDMA system in 802.22 [8].

Burst allocation

Burst allocation is significant for the BS. Using appropriate burst allocation mechanism, the capacity of channel can be enhanced greatly. So, it is one of the most important functions to evaluate the performance of the channel.

Some similar burst allocation mechanisms are discussed in [11]. There are two main patterns: row by row and rectangle shapes. In this thesis, we employ, row by row arrangement as priority is given to the capacity of the channel which utilises all the slots. Another constraint is that for every 7 slots, a pilot carrier is inserted [9]. In order to decrease the latency for CPEs, the downstream bursts are allocated over the frequency domain (vertically) while upstream burst is spread over the time domain (horizontal) to manage the instantaneous transmit power of CPEs [8][9].

Some disadvantages can been seen in the topology and communication in WRAN, which are listed:

1 All packets from CPEs have to be sent to BS, which leads to more delay and intra-cell traffic.

2 Because of the broadcast of BS, only one CPE can be allocated in one slot in the frame.

3 The subchannels can only be operated simultaneously when they are adjacent.

4 Multi-input multi-output is not supported currently [9].

5 Continuously sensing the available channels requires scheduled Quiet Periods [7].

6 Multiple channels scenario needs to consider the channel bonding [12].

Because of the drawback, an enhancement of OFDMA structure and some advanced channel allocation algorithms are studied in [13][12],which is called Device-to-Device WRAN (D2DWRAN), which is introduced in the following section.

1.1.3 D2DWRAN

Device-to-Device (D2D) communication is a technology component for Long Term Evolution-Advanced (LTE-A) [14][15]. In D2D communication, CPEs transmit packages to each other over a direct link instead of transmitting through the BS unlike the former construction. D2D users communicate directly while being controlled by the BS. Therefore, the potential of improving spectral utilization has been brought out in recent years, which shows that D2D can enhance the communicating system performances by reusing channels. As a result, D2D is expected to be a significant feature supported by the next generation cellular networks. D2DWRAN is based on this technology and can extend it to WRAN field.

Topology

In D2DWRAN, multiple CPEs can communicate with other CPEs at the same time. Figure 1.3, shows an example of D2DWRAN. When CPE A wants to communicate with CPE B, it only needs to send request through BS like standard structure. When the connection is established, there is no need to continue involving the BS in the connection. There are also three steps:

Step 1 CPE A sends the connecting request to the BS.

Step 2 The BS sends this request to CPE B.

Step 3 CPE B receives the request, schedules the slots and sends back the response. The connection directly between CPE A and CPE B is set up.

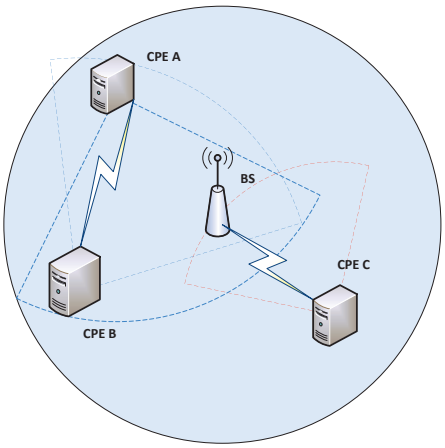


Figure 1.3: Structure of D2DWRAN.

The steps are the same as in the Standard structure except that the advantage is that in this process, the BS only participates in the first step. After establishing the connection, the BS can jump out to proceed to other request instead of transmitting the message. If CPE C wants to communicate with CPE D, these two connections could be carried out simultaneously once the network finds that there is no interference between these two connections. There are three main proposals in D2DWRAN?"

- Direct connection can be established between CPEs.
- Multiple channels can be allocated in one time slot.
- One channel can be reallocated in the same time slot.

According to the advantage above, the efficiency and capacity of channel can be improved better than the standard structure.

Superframe Structure

With the new proposals in D2DWRAN, the original structure of superframe is not suitable any more. However, the DS subframe is still the same because it is used for broadcasting by BS and there is no need to change it. The big change is in the US subframe for D2D communication. Furthermore, because the channel can be reused by multiple D2D links, the slots in US subframe can be shared by different traffic simultaneously. The new structure for the US subframe of D2DWRAN is introduced in [10] and shown in Figure 1.4.

The constraints during in the burst allocations are:

1 A burst allocation should contain at least 7 symbols as well [9].

2 Slots that are allocated in the same burst to one CPE should be adjacent so that the US MAP can use less information to describe a burst.

It is proved that D2DWRAN can achieve higher channel capacity and efficiency than WRAN [8][10]. We discuss the burst allocation problem further to give a whole picture of this problem.

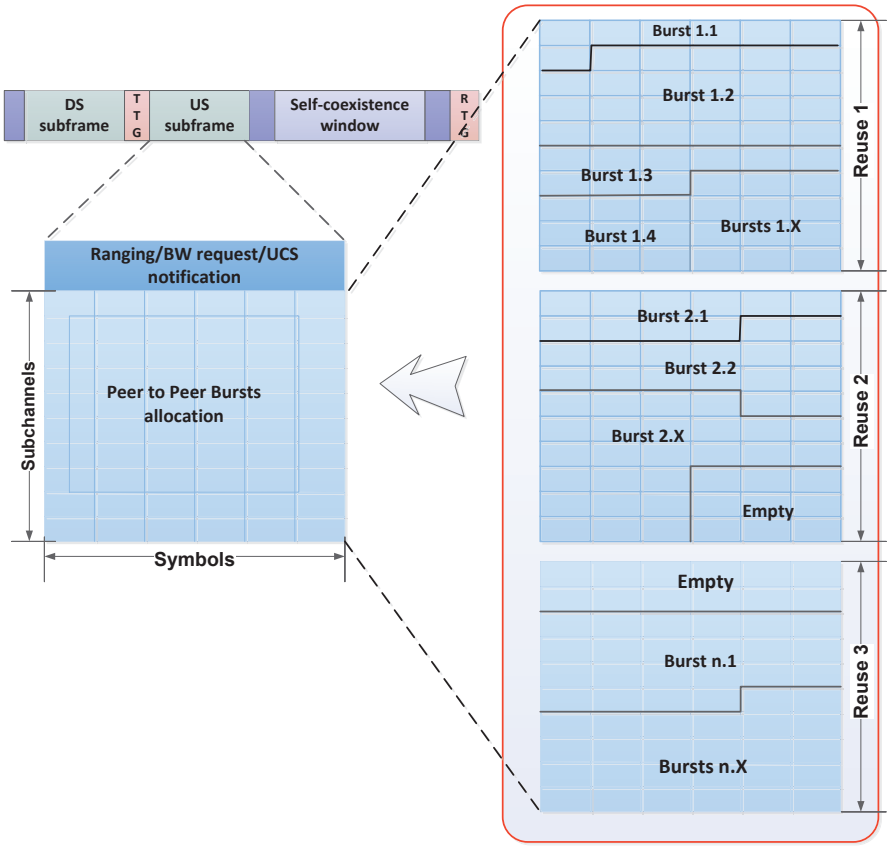


Figure 1.4: The US subframe in D2DWRANs [10].

1.2 Problem and Motivation

It has been shown that D2DWRAN can provide better performance with respect to the capacity of channels than WRAN in [10] with a brief algorithm from burst allocation in D2DWRAN. But this is only the beginning. More potentials of D2DWRAN still need to be studied. This thesis focuses on how to enhance the performance of D2DWRAN in solving the burst allocation problem with new ways.

On the other hand, there are optimization methods regarding this problem such as Simulated Annealing, genetic algorithm, Tabu Search. Hence, in this thesis, we focus on one adapting existing optimization algorithms to solve the burst allocation problem in D2DWRAN. Thus, we first list and analyze the existing optimization methods, then several optimization methods have been chosen and adapted into the problem. In order to find out the performance of different algorithms, we have simulated these algorithms and the analysis of the results are presented.

Through this project, the traditional optimization methods are studied, and are applied into a practical problem.

1.3 Thesis Overview

In Chapter 2, we first give a detailed introduction to the burst allocation problem in D2DWRAN. Then this problem is formulated into a NP-hard problem. In [16], it has been approved that the practical NP-hard problem could be solved, but in here the problem cannot fulfil the that

situation. In order to simplify this problem, the problem is considered in two different scenarios: with unlimited subchannels and with limited subchannels in Section 2.2. The details of these two scenarios are discussed separately in Chapter 3 and Chapter 4. In Section 2.3, the detailed mathematical formulation of the problems are presented. In Section 2.4,the existing optimization methods are listed and analyzed.

In Chapter 3, the implementation of the existing methods into this problem are studied with unlimited subchannels. According to the analysis in Chapter 2, we choose Greedy Algorithm, Tabu Search, Simulated Annealing, Genetic Algorithm and Ant Colony Algorithm to solve the burst allocation problem in D2DWRAN. These methods have been simulated with different parameters and the performance of these methods are also compared and analyzed.

In Chapter 4, the implementation of the existing methods are studied with limited subchannel. There are five mechanisms to change unlimited subchannels into limited subchannels in the problem, which are abandoning rule, cutting first 60 subchannels, cut last 60 subchannels, cut random 60 subchannels and selective choosing methods. We have simulated these mechanisms with different optimization methods. We present and analyze the results at the end of this chapter.

We conclude this thesis in Chapter 6. Some future works are also discussed to draw a complete picture of the burst allocation problem in D2DWRAN.

Table 1.1: Explanations and concepts regarding the channel allocation in this thesis. [10]

Terms	Description	In 802.22
Subchannel	The smallest allocation unit in the frequency domain of the system, it is constructed with group of subcarriers	One subchannel contains 28 subcarriers.
Channel	A TV channel that contains multiple subchannels.	TV channels with a bandwidth of 6,7,or 8 MHz. One channel contains 60 subchannels.
Symbol	The smallest allocation unit in the time domain of the system.	There are 26 to 42 symbols in a frame.
Slot	The smallest allocation unit in the system. It constructs symbols an subchannel.	60×26 to 60×42 slots in a frame.
CPE request	Information about the number of slots the CPE need to communicate either to CPE or BS.	CPEs send their requests to the BS.
Burst	A group slots that are allocated to certain communication.	A burst on a subchannel has to cross at least 7 symbols.
Burst allocation	Allocating slots to different connection according to their requests.	The BS manages the burst allocation vertically in the DS subframe and horizontally in the US subframe.
Link	A transceiver pair with a CPE and the BS or two CPEs.	Direct CPE-CPE links are not supported.
Slot (re)use	It means a state that the slot is allocated to one link (use) or multiple links (reuse).	
Slot reuse times	The number of links that use a certain slot simultaneously in a frame.	Maximally one link can be allocated to a slot.
DS	Downstream. Directions of the data flow are from the BS to CPEs.	
US	Upstream. Directions of the data flow are from CPEs to the BS or between two CPEs.	Only the links that from CPEs to the BS are supported.

Chapter 2

D2D Resource Allocation

In this chapter, the main topic of this thesis is proposed - resource allocation in D2DWRAN. We first analyze the existing studies regarding this problem in the literature, then in Section 2.2, a simplified problem for resource allocation of D2DWRAN is introduced. This simplified problem is formulated in Section 2.3.

2.1 OFDMA Resource Allocation

In D2DWRAN, Orthogonal Frequency Division Multiple Access (OFDMA) is employed. It is a technique based on Orthogonal Frequency-Division Multiplexing (OFDM) with multi-user version by assigning subsets of subcarriers to individual users. However, in order to support D2D communication, the OFDMA system and resource allocation problem in D2DWRAN is different from the OFDMA systems [13]. In this section, related work on resource allocation of OFDMA systems in the literature and the resource allocation problem in D2DWRAN are compared and analyzed.

2.1.1 Related Work on Burst Allocation

As introduced in Chapter1, burst allocation is one of the important functions in D2DWRAN. Burst allocation in OFDMA is also an important way to enhance its performance. There are some existing studies of burst allocation in OFDMA systems.

There are two subproblems here: the downlink burst allocation and uplink burst allocation.

In the downlink of OFDMA, there are two main dynamic resource allocations:

The first allocation problem is Margin Adaptive (MA), which tries to achieve the minimum overall transmit power with certain data rate. This problem was first studied in [17], and it is a convex minimization problem. A Multiuser Adaptive OFDM (MAO) scheme is proposed, which can get the Lagrangian of the problems by Vector Space Methods [18].There are drawbacks in this approach such as low sensitivity to channel estimation errors. Then an improvement by using blockwise subcarrier allocation algorithm is proposed in [19][20].

The other allocation problem is Rate Adaptive (RA), which maximizes the users' data rate with a constrained transmit power which is formulated in [21]. A cross-layer optimization is investigated, and algorithms are developed for Dynamic Subcarrier Allocation (DSA), Adaptive Power Allocation (APA) [22][23]. A combination of greedy algorithm and water-filling algorithm, which formulates the problem by maximizing the total throughput differently, is proposed in [24][25]. A bandwidth optimization algorithm is proposed to find the best number and set of subscribers [26] .

In uplink process of OFDMA, resource allocation can be divided into three parts[27].

1. Centralized single cell scheduling which the BS is responsible for the scheduling process. In this part, greedy algorithm is proposed and improved [28][29]. A combination of water-filling for the user on unallocated subcarriers to get optimal allocation [30]. These algorithms are suboptimal. Ergodic resource allocation was investigated [31]. This scheduling introduces a Ergodic which can weight the sum-rate maximization and determine the capacity region [32][33]. It is also enhanced by some other works, e.g., the problem is divided into a convex utility maximization problem in [34][35] and Lagrangian parameters are computed in [36].

The second type is distributed base stations scheduling. This type can be divided into two different categories: scheduling with distributed base stations (DBSs) and scheduling with mobile user participation.

DBS approach is proposed to increase the coverage and capacity of wireless networks [37]. It can enhance both scenarios when a single cell with distributed antennas or a single cell with one central antenna. But the simultaneous transmission in the downlink reduces the performance of this approach [38]. An optimal solution to this problem is by selecting only the best channel to the user is proposed in [39][40].

The distributed uplink scheduling scheme for OFDMA with the collaboration of mobile users is proposed in [41].Channel state information (CSI) are exchanged between users to complete the collaboration. Further, distributed uplink scheduling to the OFDMA system without user cooperation is considered [42]. In order to be fair, the system changes from the scenario that users with high priorities have privilege to a random access transmission. Aloha is one of the first algorithms for random access without user cooperation [43][44]. Based on it, some extensions have been proposed from then on, such as slotted Aloha to make the packets transmissions more regular to reduced collisions [45][46][47] and reservation Aloha to further reduce the collision [48][49][50].

The third one is scheduling in multiple cell scenarios. In this type of problems, the burst allocation within multiple cells is considered together [27]. Static reuse schemes are based on Fractional Frequency Reuse (FFR) where a cell is divided into an inner area [51]. Because we do not discuss further with multiple cell scenarios, this part is left for further research.

2.1.2 Burst Allocation in D2DWRAN

The OFDMA system in D2DWRAN is developed based on the OFDMA system in WRAN. In general, the DS subframe in the OFDMA system of WRAN supports the broadcast from BS to CPEs, and the US subframe is used for communications from CPEs to BS. The DS subframe in D2DWRAN is the same as WRAN. However, the US subframe in D2DWRAN is more complicated in order to support CPE to CPE communication. Hence, the associated burst allocation problem in the US subframe of D2DWRAN is different. There are two main differences in the burst allocation of US subframe of D2DWRAN compared to WRAN:

- Because multiple links can be allocated in the same slots, the interference between these links should be avoided.
- Because of the reuse of slots, it is no longer important to minimize the number of waste slots, which is the main goal in the OFDMA of WRAN. Instead, the burst allocation in the US subframe of D2DWRAN should try to maximize the average reuse times of slots.

Analysis and solutions for this new burst allocation problem are discussed in [10]. The main goal of the problem is to maximize the capacity of the network and there are five constraints:

1 A link request should be only allocated in one burst.

2 The slots allocated in the same CPEs should be adjacent. In DS subframe, US MAP is used to record the positions of all burst allocations. All slots allocated to the same burst should be adjacent so that it only needs four elements to describe one burst.

3 The allocation to a burst on one subchannel should contain at least 7 slots.

4 Algorithms applied should be simple, because a burst allocation decision needs to be made in each frame every 10 ms.

5 It is possible to take other QoS parameters into consideration.

Constraints 1 and 2 guarantee the efficiency of US MAP. Otherwise, if slots allocated to the same burst can be spread into the subchannels separately, the US MAP need much more than 4 elements to describe one burst, and the slots would be underutilized in the DS subframe.

Constraints 3 and 4 are the limitations in terms of the structure and procedure of the network [8]. Constraint 5 indicates other possible QoS requirements in different scenarios.

In order to keep the US MAP small and also have some flexibility during allocation, we need to consider the shape of a burst before the allocation. We discussed the above constraint regarding the shape of a burst more in detail as follows:

1 The slots occupied by the same burst should be adjacent in one subchannel.

2 Burst can occupy several subchannels, which also should be adjacent.

3 Only top or bottom subchannel in one burst can have some space for other bursts. In other words, all subchannels that are allocated to a burst should be fully occupied except the top or bottom subchannel, which is called as an extra subchannel in this thesis. With this assumption, bursts allocation first spreads in the time domain, then in the frequency domain.

4 The minimum length of the burst in one subchannel is 7.

5 The connections which are interfered with each other cannot have overlap slots.

The first three constraints are used to reduce the size of the US MAP. Figure 2.1 is an example of a burst, in which five parameters are needed to indicate the position of one burst. These parameters are the starting subchannel, number of subchannels that are occupied, an indicator to identify whether the extra channel is on top or bottom of a burst, the staring slots on the extra channel, and the number of slots on the extra channel.

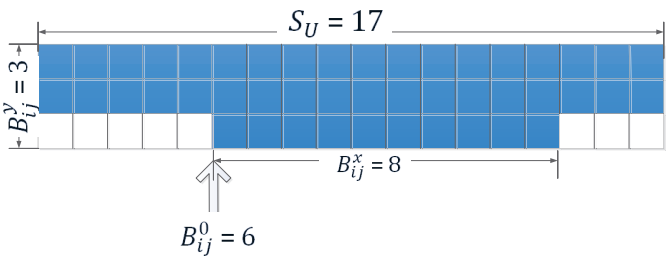


Figure 2.1: Burst allocated in the subframe

2.2 Two Steps to Solve the Problem

The burst allocation problem in the US subframe is a NP-hard problem [52]. To simplify this problem, we solve this problem in two steps:

Step 1. We first assume there would be infinite number of subchannels in the US subframe for allocation. With this assumption, all the requests can be allocated. Therefore, the goal of the problem is to find the minimum number of required subchannels. Note that this model also can correspond to the multiple channels situation in D2DWRAN, but it is not in the scope of this thesis. We mainly discuss the single channel case here.

Step 2. After getting the allocation from Step 1, limited subchannels are considered in the problem. In this step, subchannels allocated in Step 1 are selected and an allocation for limited subchannels will be built. The goal in this step is to maximize the number of requests with limited number of subchannels in the US subframe.

In the following section, we discussed the mathematical models of both of these two steps.

2.3 Problem Formulation

We first define some indicators in this section, then the problems are formulated.

2.3.1 Burst Indicators

There are three main parameters to indicate the position of one burst in the US subframe.

- Number of slots the burst needs.
- The start point in the extra subchannel.
- The shape of the burst, to indicate whether the extra subchannel is on top or bottom.

In the previous model, two more parameters are introduced which can provide more details of the burst: the start slot of the subchannel and the end slot of the subchannel. These two parameters are ignored here because in this model, we only consider whether two interfered requests can be allocated in the same subchannel instead of the exact positions in the US subframe.

- B_{ij} indicates the burst from CPE i to CPE j.
- B_{ij}^{l} indicates the number of slots that B_{ij} contains.
- B_{ij}^{y} indicates the number of required subchannels of B_{ij}.
- B_{ij}^{x} indicates the number of slots in the extra subchannel of B_{ij}.
- B_{ij}^{0} indicates the position of the extra subchannel. There are three values. 1 stands for on bottom, -1 for on top, 0 means the burst only occupies one subchannel and there is no extra subchannel.

Table 2.1: An example of S_s.

	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8	B_9
S_1	1	0	0	0	0	1	0	1	0
S_2	0	1	0	0	0	1	0	0	0
S_3	0	1	0	0	1	0	0	0	0
S_4	0	0	0	0	1	0	0	0	1
S_5	0	0	0	1	1	0	0	0	1
S_6	0	0	0	1	1	0	1	0	0
S_7	0	0	1	1	1	0	1	0	0
S_8	0	0	1	0	0	0	1	0	0

Table 2.2: An example of S_{iex}.

	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8	B_9
S_1	27	0	0	0	0	40	0	27	0
S_2	0	40	0	0	0	17	0	0	0
S_3	0	14	0	0	10	0	0	0	0
S_4	0	0	0	0	40	0	0	0	30
S_5	0	0	0	40	40	0	0	0	40
S_6	0	0	0	40	40	0	17	0	0
S_7	0	0	23	7	40	0	40	0	0
S_8	0	0	40	0	0	0	40	0	0

In the rest of the thesis, to make it more comprehensible, B_n (for example B_1,B_2) is introduced to represent B_{ij}. S_l indicates the length of one subchannel. S_n indicates the total number of subchannel in the whole subframe. S_s is a table to record the allocation in subframe, in which rows are subchannels, and columns are bursts. S_s is defined as,

$$S_s(n,k) = \begin{cases} 1, \ if \ B_n \ is \ allocated \ in \ subchannel \ k, \\ 0, \ otherwise. \end{cases} \tag{2.1}$$

S_{iex} is an extension of S_s. It shows if burst is allocated in the subchannel and also the number of slots that are allocated to this burst on this subchannel. S_{iex} is defined as,

$$S_{iex}(n,k) = n(0 \leq n \leq S_l) \tag{2.2}$$

Table 2.3 and Table 2.2 are two examples of S_s and S_{iex} respectively. In both tables, columns indicate bursts, and rows indicate subchannels. The 1 appears in the first column and first row in Table 2.3 means that the B_1 is allocated in subchannel S_1. Assuming that B_1 requires 54 slots. The length of subchannels is less only 40 slots, so it needs two subchannels to be allocated.

Table 2.3

	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8	B_9
B_1	0	1	1	1	0	1	0	1	0
B_2	1	0	0	1	0	1	0	0	0
B_3	1	0	0	0	0	0	1	1	1
B_4	1	1	0	0	1	0	0	0	1
B_5	0	0	0	1	0	1	1	0	0
B_6	1	1	0	0	1	0	1	0	0
B_7	0	0	1	0	1	1	0	1	1
B_8	1	0	1	0	0	0	1	0	0
B_9	0	0	1	1	0	0	1	0	0

Therefore, S_1 and S_2 are assigned to B_1. Then in table S_s, first two rows of first column becomes 1, which shows that first two subchannels are allocated to B_1. S_s is simpler than S_{iex}. But when the length of S_l is more than 14 slots, there is an opportunity that even though two connections are interfered with each other, they can be allocated into the same subchannel if the total length of their extra subchannels is less than the length of the subchannels.

I_m is an interference map to indicate whether B_1 and B_2 are interfered with each other when they are allocated with the same slots. I_m is defined as,

$$I_m(1,2) = \begin{cases} 1, \ if \ B_1 \ interferes \ with \ B_2 \\ 0, \ otherwise. \end{cases} \tag{2.3}$$

The burst allocation in the US subframe of D2DWRAN can be divided into two different cases: $S_l < 14$ and $S_l \geq 14$. We discuss these cases separately in the following content.

Case-1(When $S_l < 14$)

In this case, two interfered bursts cannot be allocated in one subchannel. So the extra subchannel can be considered as a complete subchannel and there is no need to consider the S_{iex}. The burst allocation can be formulated as follows:

$$\min(\sum_k max(S_s(n,k))) \tag{2.4}$$

subject to:

$$\sum_k S_s(n,k) \geq B_n^y, \ \forall n. \tag{2.5}$$

$$\sum_k (S_s(n,k) \bigoplus S_s(n,k+1)) \leq 2, \ \forall n. \tag{2.6}$$

$$\sum_{m,n,k(m \neq n)} S_s(m,k) S_s(n,k) I_m(ij,pq) = 0. \tag{2.7}$$

Eq.(2.4) indicates that when the number of the subchannels is unlimited, the one algorithm estimates how less subchannels it requires to fill out all the requests. Eq.(2.5) indicates that

the number of subchannels allocated to one request should be larger the its need. Eq.(2.6) is used to constrain all the subchannels that are allocated to one request are adjacent. Eq.(2.7) constrains if two subchannels are interfered, they can not be allocated in the same subchannel.

Case-2(When $S_l \geq 14$)

This case is more complicated than the previous one. Two scenarios should be considered between any two bursts:

Scenario 1: B_1 and B_2 don't interfere with each other. Then they can be allocated in one subchannel anyway.

Scenario 2: B_1 and B_2 interfere with each other. Then total number of slots in both the extra subchannels should be considered. If the total number is less than the length of one subchannel, then their extra subchannels can be allocated in the same subchannel. Otherwise, the extra subchannels should be allocated to different subchannels.

According to these two scenarios, when considering three or more bursts, the situation will be much more complicated. Therefore, Eq.(2.7) is not suitable anymore in this case. To avoid interference as well as maximize the network capacity, a set $\mathbf{S}$ needs to be introduced, which contains a series of subsets $\mathbf{S}(\mathbf{k}, \mathbf{mn})$. The subset $\mathbf{S}(\mathbf{k}, \mathbf{mn})$ is a maximum set contains all the links that occupy the same subchannels and interfere with others. For example, $\mathbf{S}(\mathbf{k}, \mathbf{123})$ means in subchannel k, request 1, interferes with requests 2 and 3, while request 2 also interferes with requests 1 and 3 in the same subchannels. The formulation is modified as follows:

$$\min(\sum_{k} max(S_s(n,k))) \tag{2.8}$$

subject to:

$$\sum_{k} S_s(n,k) \geq B_n^y, \forall n. \tag{2.9}$$

$$\sum_{k} (S_s(n,k) \bigoplus S_s(n,k+1)) \leq 2, \forall n. \tag{2.10}$$

$$\sum_{n \in \mathbf{S}} \mathbf{S} \leq S_l, \forall n. \tag{2.11}$$

In this formulation, Eq.(2.9) and Eq.(2.10) stay the same as Eq.(2.5) and Eq.(2.6) because the constraints are the same. The significant change is in Eq.(2.11). To fulfil the request that even though two bursts are interfered with each other, if the total length of their extra subchannels is less than the length of the subchannel, they can still be allocated into the same subchannel. But actually, the problem to get all the subsets $\mathbf{S}$ is already a maximal independent set problem, which is NP-hard. Therefore, we simplify this problem again with a new constraint as,

$$\begin{aligned} &\prod_{\forall m} S_s(n,k) S_s(m,k) I_m(n,m) = 1, \\ &if \;\; S_{iex}(n,k) + \sum_{\forall m,} S_{iex}(m,k) \leq S_l, \\ &(\forall n, m \; and \; n \neq m) \end{aligned} \tag{2.12}$$

Note that in Eq.(2.12) once B_n interferes with link B_r and another burst B_s, the total number of the slots of B_n B_r and B_s are considered no matter whether B_r and B_s interfere with each other. It can guarantee a correct final solution (no interference is caused), but maybe not the best solution.

The above constraints are based on the assumption that the number of subchannels is unlimited in the US subframe, which is easy to implement in mathematical model. However, it cannot be used in practical in which the number of subchannels is limited. So in Step 2, the goal is to maximize the number of requests that are allocated with limited number of subchannels. Therefore, Eq.(2.4) and Eq.(2.8) should be replaced by,

$$\max(\sum_{n} \max(S_s(n,k))) \tag{2.13}$$

Additionally, there would be one more constraint to limit the number of subchannels. In Eq.2.14, N_s is the number of subchannels the can be allocated.

$$\sum_{k} \max(S_s(n,k)) \leq N_s \tag{2.14}$$

2.4 Methodology of Optimization

In this section, the existing optimization algorithms for the burst allocation problem in the US subframe of D2DWRAN are listed and analysed before we try to solve the problem in Chapter 3 and Chapter 4.

2.4.1 Classic Algorithm

Divide and Conquer Algorithm

For most of the cases, the D&C yields solutions faster than brute-force method. The computing time is based on the methods of dividing the problem. In general, it searches for the solution in a recursive manner. However, this is not suitable for NP-hard problems because it still needs to go through all the solutions. Therefore, the burst allocation problem cannot be solved in polynomial time by D&C, and we do not consider it further in this thesis.

Dynamic Programming

Dynamic Programming is an enhancement of D&C Algorithm, except that most the sub-problems to be solved should be the same. To make it effective, we need to keep polynomial number of different sub-problems in a limited scale. In solving NP-hard problem, it does not modify the drawbacks of D&C Algorithm. Thus, it is not suitable for the burst allocation problem in the US subframe of D2DWRAN.

Greedy Algorithm

The Greedy Algorithm can quickly get a good solution of problems. However, it can only be used to find the local optimal solutions instead of a global one. As a result, it cannot be considered as a best choice of optimization.

Another problem is that while choosing the best candidate at the moment, sometimes the Greedy Algorithm may lead to a dead end which cannot get a complete solution. In traveling salesman problem, if there are three nodes connected to each other with shortest distances, then the algorithm may go into this loop until other constraints are introduced.

Though not always the optimal solutions are guaranteed by the Greedy Algorithm, because of the low computing time complexity, it can be used in the burst allocation problem in the US subframe of D2DWRAN, which has been studied in [10].

Modification

Backtracking is a general algorithm when searching all solutions of a problem [53]. It incrementally builds candidates to the solutions and abandons each partial candidate that is impossible to appear with a valid solution.

With backtracking, the Greedy Algorithm can avoid invalid solutions. And by setting up the backtracking parameter, it helps the Greedy Algorithm more likely to get to the global optimal solution [54]. However, the Greedy Algorithm combined with backtracking will increase the computation time of the method.

2.4.2 Heuristic Algorithms

Tabu Search

Tabu Search can accept the solution that is not better than the current candidate solution as a new candidate solution, so it has the chance to jump out from the local best solution and get the global optimal solution.

The aspiration criteria is introduced in Tabu Search to improve the efficiency of searching for optimal solution [55]. In each iteration time, the program also checks the possible solutions which are in the Tabu list. If some candidate solution can provide better solution than the current best solution, it can be overridden from Tabu list.

The biggest disadvantage of Tabu Search is that the high computing complexity, which increases sharply with the size of the candidate solutions. To improve the efficiency, it is possible to combine Tabu Search with other Heuristic algorithms. For example, it can be combined with Simulated Annealing [55] or Genetic Algorithm [56].

Tabu Search is a potential method for the burst allocation problem. But with the high computing complexity, it is not a practical method, which will be discussed later in Chapter 3.

Simulated Annealing

The Simulated Annealing is simple and comprehensive to solve parallel processing problem [57]. It is robust. However, the convergence and time consumption could limit the efficiency of the algorithm.

According to its ability of jump out the local optimal solution, it is able to solve most optimization problems including the burst allocation problem in the US subframe of D2DWRAN. Furthermore, some other heuristic algorithms employ Simulated Annealing to improve their performances, for example the Neural networks. A series of new type of optimizations have been generated via combination of these two methods, such as Boltzmann Machine [58][59],Gaussian Machine [60] and Cauchy Machine [61][62]. However, because of the high complexity of Neural network algorithm, it is not suitable for the burst allocation problem. Therefore, it is not in the scope of this thesis, but more information can be found in [63]. In Chapter 3 and Chapter 4, the implementation of Simulated Annealing will with discussed in detail while solving the burst allocation problem in the US subframe of D2DWRAN.

Genetic Algorithm

There are some advantages in Genetic Algorithm. Firstly, it doesn't need the continuity of the objective function. It can realize global search to complex, peaky and non-linear function. Secondly, because of its parallelism, it is suitable to process substantial computation. Thirdly, Genetic Algorithm regards the problem by process characteristic symbols that are independent from specific issues.

However, it cannot effectively solve problems which require the global optimal solution because it is easy to turn up premature convergence [64].

The main difficulty exists in the implementation of Genetic Algorithm is choices of proper control parameters and functions to avoid premature and also to improve the convergent efficiency. It can be combined with other heuristic algorithms too, such as Simulated Annealing [65] and Tabu Search [56].

Genetic Algorithm can be employed to solve the burst allocation problem in the US subframe of D2DWRAN, which will be discussed in detail in Chapter 3 and Chapter 4 including the choice of the initial group size and the design of all the functions.

2.4.3 Ant Colony Algorithm

The Ant Colony Algorithm was designed to find shortest path initially. However, it has become attractive in many other scientific communities, such as combinatorial optimization [66], data analysis [67] and global optimization of random continuous function [68]. It is mainly employed to solve two application fields: NP-hard problems and shortest path problems. The burst allocation problem in the US subframe of D2DWRAN can be solved by Ant Colony Algorithm, and it will be discussed in detail in Chapter 3 and Chapter 4.

One possible difficulty of this algorithm is when they are adapted to the problems with huge number of neighbourhood solutions, the possibility of the ants visiting the same way becomes very small. In this case it requires lot of iterations to get a reasonable solution.

2.5 Summary

This chapter discussed the burst allocation problem in D2DWRAN including the mathematical models and solving steps. Then two main research directions in this thesis were presented: burst allocated with unlimited subchannels and with limited subchannels. In the following sections, we will first analyze the existing solutions for these problems and then adapt these solutions to these two steps. Some optimizations are chosen and analyzed. The chosen optimization is divided into two parts. The first part are classical methods such as Dynamic Programming and Greedy Algorithm. The other one are heuristic algorithms such as Tabu Search, Simulated Annealing, Genetic Algorithm and Ant Colony Algorithm. Because the burst allocation problem is NP-hard problem, we pay more attention on the heuristic algorithms which can control the computation time and performance.

Chapter 3

Algorithm with Unlimited Subchannels

In this chapter, the burst allocation problem with unlimited subchannels has been studied. In Section 3.1, the implementations of the existing algorithms for this problem are discussed, and detailed descriptions of the algorithms are presented. In Section 3.2, the simulation results of all algorithms under different parameters are plotted and discussed.

3.1 Algorithm Implementation

In this Section, we first build a general system model based on the mathematical model proposed in Section 2.2. The difference between this general model and the mathematical model will also be discussed. Then, to adapt this general model in different optimization algorithms, some modifications are introduced to the model for each algorithm.

3.1.1 System Model

One of the basic problems during burst allocation is sequence of requests to be allocated. In our system, we assume all requests are processed according to their priorities, i.e., the request with high priority is considered first during the burst allocation. If this request is successfully allocated, the next request is considered. Once a request is considered for allocation, the system first checks whether it interfered with the existing allocated requests in current subchannel. Each request often needs several continuous subchannels, so all these subchannels do not have interfered situations. The check starts from the first subchannel of the US subframe. If it shows there is interference, the system would check the next subchannel. The rest can be done in the same manner until a non-interference burst is found for this request and allocated to this requests, or such a burst cannot be found in this US subframe and the request would be moved to the waiting list (only with assumption of limited subchannels). For example, when the first request comes into the channel, the program checks the first subchannel whether there is interference. Of course there is no burst in the subchannel right now, so the first request is allocated starting from the first subchannel in the US subframe. Then the second request comes into the channel. If the second request doesn't interfere with the first request, it also can be allocated from the first subchannel; otherwise, it needs to check whether it is possible to be allocated from next subchannels. This process continues until there are no requests in the waiting list. Different considering sequences of requests leads to different allocations and further different slot reuse and network capacity. Therefore, different optimization algorithms

may employ different considering sequence of requests, which will be discussed later in the implementation of each algorithm.

As discussed in Chapter 2, there are two cases to be considered in the system model: the case less than 14 slots in the US subframe and the case with greater than or equal to 14 slots in the US subframe. In case with $S_l < 14$, the system doesn't need to consider the length of extra subchannel, because only one request can be allocated in this subchannel. While in case with $S_l \geq 14$, the program needs to calculate whether two interfered requests can be allocated in the same subchannel. Further, the system needs to decide whether each request puts its extra subchannel on top or on bottom.

According to differences of these two cases, the procedures of burst allocation are also different, which are shown in Algorithm 1 and Algorithm 2. In the following sections, we adapt the

Algorithm 1 The general burst allocation procedure in Case $S_l < 14$.

```
1: if S_l < 14 then
2:   for every request B_ij in the queue do
3:     for every subchannel k do
4:       Deploy B_ij starting from kth subchannel.
5:       for next B_ij^m subchannels in S_s(k, ij) do
6:         Examine whether it interferes with the existing request.
7:         if interference is found then
8:           Withdraw the deployment, try next subchannel.
9:         else
10:          Allocate with the deployment, exit loop and try next request.
11:        end if
12:      end for
13:    end for
14:  end for
15: end if
```

existing algorithms in to this general procedure and also evaluate the performance of these algorithms, including Brute Force Algorithm, Greedy Algorithm, Simulated Annealing, Genetic Algorithm, Tabu Search and Ant Colony Algorithm.

3.1.2 Brute Force Algorithm

Brute Force Algorithm lists all possible sequences of requests and run all the sequences. When compared, it gives the optimal solution. The adapted algorithm of brute force in our system model is described in Algorithm 3. In Algorithm 3, The E is the performance of candidate solution. The E_{best} is the performance of current best solution. The time complexity is $n!$, n is the total number of requests. Because of its high time complexity, it is not suitable for the burst allocation problem in D2DWRANs, even though it is possible to be adapted to this problem. Therefore, this algorithm will not be analyzed and discussed any further in this thesis.

3.1.3 Greedy Algorithm

Greedy Algorithm provides a simple method to get a good solution[54]. It queues the request with a specific rule and gets the allocation result by only considering this queue of request. When adapting to the burst allocation problem in the US subframe, the queueing rule can be the interference degree of each request. The interference degree of a request is the number of requests interfering with it when allocated with the same slot. Therefore, before allocation, all

Algorithm 2 The general burst allocation procedure in Case $S_l \geq 14$.

```
1: if S_l ≥ 14 then
2:   for every request S_ij in the queue do
3:     for every subchannel k, do
4:       Deploy B_ij starting from kth subchannel.
5:       for next B_ij^m subchannels in S_s(k, ij) do
6:         Examine whether it interferes with the existing request.
7:         if interference is found then
8:           Examine whether its available slots is more than the request.
9:           if the slots is enough then
10:            Return a true result and try next subchannels.
11:          else
12:            Return a false result and exit the loop.
13:          end if
14:        end if
15:      end for
16:      examine the result.
17:      if all return is true then
18:        Allocate the deployment, exit the loops and examine next request.
19:      else
20:        if there exist a false result then
21:          Withdraw the subchannel, and try next subchannel.
22:        end if
23:      end if
24:    end for
25:  end for
26: end if
```

the requests need to calculate their interference degrees based on the Interference map. The requests with low interference degree is considered first during the allocation[69]. A description is shown in Algorithm 4. From the above description, the greedy algorithm is much simpler than brutal search. It only considers one queue during the allocation. However, the performance of the results cannot be guaranteed. Since it is very efficient, we still use it to solve the burst allocation problem and put it as a standard to evaluate the performance of other algorithms later in this thesis.

3.1.4 Simulated Annealing

Simulated Annealing iteratively compares different solutions and chooses the better one for the next loop. Each solution needs to compare to its neighbour solutions and the better one is chosen. In order to jump out from local optimal solution, the temperature parameter T is introduced, T is used to control the solution chosen. The detail of algorithm that is adapted in the burst allocation problem is shown in Algorithm 5 when there are unlimited number of subchannels. In the algorithm, T_0 is the initial temperature and T_e is the end temperature. T starts from T_0 and reduces to T_e. T_{down} represents the efficiency of cooling down. A parameter E is introduced to estimate the performance of a solution, which is the total number of occupied subchannels. E_{new} presents the number of subchannel current solution used, E_{best} presents the number of subchannel the best solution until now. P illustrates the possibility that whether the current best solution E_{best} can be replaced by current solution E_{new}. The lower T is, the

Algorithm 3 Brute Force Algorithm when there are unlimited subchannels.

1: List all possible sequences in inserting order of request.
2: **for** every sequence in the list **do**
3: Call the general burst allocation procedure to reorder the request under new sequence.
4: Count the number of subchannels that the sequence occupies and record it to E.
5: **if** $E < E_{best}$ **then**
6: $E_{best} \leftarrow E$.
7: Record the current sequence as the best solution.
8: **end if**
9: **end for**

Algorithm 4 Greedy Algorithm when there are unlimited subchannels.

1: Count the interference degrees of requests.
2: Sort the requests in the ascending order and call the general burst allocation procedure to allocate them in the order of the queue.
3: Count the total number of subchannels that the requests occupied.

lower P.

3.1.5 Genetic Algorithm

Genetic Algorithm needs a group of solutions as the first generation. The adapted algorithm is shown in Algorithm 6. The fitness function evaluates whether it is a better solution, which also means the performance of the solution. While solving the burst allocation problem, we define the fitness function as the square of reciprocal of the number of subchannels that the solution occupies, it is shown as in Eq.(3.1),

$$E_{GA} = 1/(N_{GA})^2 \tag{3.1}$$

E_{GA} is the fitness function, N_{GA} is the occupied subchannels number by one solution. So, if one solution occupies less subchannels, it can obtain higher fitness function. Fitness value $F(i)$ is the percentage value compared to the highest fitness function shown as Eq.(3.2),

$$F(i) = E_{GA}/E_{GAbest} \tag{3.2}$$

The program generates a random value range between 0 and 1 for each solution, if the percentage is higher than this value, the solution is retained otherwise, it is abandoned. In this way, a solution with high fitness value has low possibilities to be abandoned. Then the solution with highest fitness value reproduces itself to keep the size of a generation.

There also needs over-cross function and mutation function to make variance of the generation. In over-cross function, two solutions are picked up randomly. Then they change the number in a random position. At the same time, the same number in the sequence needs to be changed to make sure there is not a same number in one sequence. Through this way, two new solutions are generated. In mutation function, if one solution is chosen, it changes its own two positions of number and generates a new solution.

3.1.6 Tabu Search

The detailed description of the adapted Tabu Search algorithm is shown in Algorithm 7. In Tabu Search Algorithm, the main idea is to create a Tabu list to avoid falling into a local

Algorithm 5 Simulated Annealing when there are unlimited subchannels.

```
1: Set up T_0 and T_e.
2: Set up circle time n.
3: while T > T_e do
4:   for every circle i do
5:     Generate a random sequence.
6:     Call the general burst allocation procedure to reorder the requests with the sequence.
7:     Count the number of subchannels which the sequence occupies and record it to E_SAnew.
8:     if E_SAnew < E_SAbest then
9:       B_SAbest ← B_SAnew.
10:      S_SAbest ← S_SAnew.
11:    else
12:      if P(E_SA, E_SAnew, K) > random(0, 1) then
13:        B_SAbest ← B_SAnew.
14:        E_SAbest ← E_SAnew.
15:      end if
16:    end if
17:  end for
18:  T ← T * T_down.
19: end while
```

optimal solution. Therefore, we assume Ta is a Tabu list array and the length of this array is set Ta_l. When the algorithm starts, Ta is empty, and one solution is chosen randomly. The solutions that have only two differences with the chosen solution are called neighbour solutions. During each iteration, the program searches all the possible neighbour solutions of the chosen one, and records the best one of these. Then it checks the Ta to see whether the change that leads to best solution is in the list. If not, the best solution will be taken as the current best solution, the pair which makes this solution will be recorded into the Ta. If the pair has been recorded by Ta, the best solution will be discarded and the second best choice is considered, and so on. The program stops when the Ta is full. At that time the best solution is the output of this algorithm.

There is an aspiration criterion in Tabu Search. With this criterion, when the program finds a change that already exists in Ta, it will check whether this change can get a better solution than the current best solution. If so, the aspiration criterion takes effect and the change will be allowed and it will replace the current best solution.

In burst allocation problem, the neighbour solutions are produced by changing the two sequences of the initial solution. After comparison, the change that can produce the best solution will be recorded in Ta.

3.1.7 Ant Colony Algorithm

In Ant Colony Algorithm, we need to simulate ants finding route through all nodes. If ants are at random nodes, they start to find out which is the best way to go through all the nodes. The method of deciding the route for ants is after calculating all the distances to each node which have not been, and then ants find the shortest one. This procedure takes the new node as a starter to find the next node. When an ant goes to a new node, a parameter called pheromone will be left for this choice, which will help other ants to choose the route when they are in the same position. The strength of pheromone decreases over time. However, it can help other ants

Algorithm 6 Genetic Algorithm when there are unlimited subchannels.

Set up P_s and P_t.
Set up probability P_c and P_m.
Generate P_s random sequence.
while circle time $T_{GA} < P_t$ **do**
  for every circle T_{GA} **do**
    Get the sequence from P_s.
    Call the general burst allocation procedure to reorder the requests with the sequence.
    Count the number of subchannels which the sequence occupies and record it to E_{GA}.
    if $E_{GA} < E_{GAbest}$ **then**
      $E_{GAbest} \leftarrow E_{GA}$.
    end if
    for every solution i **do**
      $F(i) \leftarrow \frac{E_{GA}}{E_{GAbest}}$.
      Generate a random value m between 0 and 1.
      if $F(i) < m$ **then**
        Discard this sequence i, and copy the best sequence i_b.
      else
        Keep the sequence.
      end if
    end for
    Generate random value $P1$ and $P2$ between 0 and 1.
    if $P_c < P1$ **then**
      Pick up two solutions from the group and change their number in one position.
      Change the same number in the sequence.
      Re-evaluate the fitness of this two solutions.
    end if
    if $P_m < P2$ **then**
      Pick up one solution form the group and change its positions of two numbers.
      Re-evaluate the fitness of this solution.
    end if
  end for
  $T_{GA} \leftarrow T_{GA} + 1$;
end while

to make decision more effectively.

When this algorithm is used in the burst allocation problem in the US subframe problem, we can assume ants get many requests and try to go through all the requests. Ants may pass different sequences and we can also assume the distances are the numbers of occupied subchannels. In scenarios with unlimited subchannels, the goal is to achieve the lowest total distance. So in the next iteration time, ants intend to go to the sequence with short distance. The follow Algorithm 8 shows the implementation of Ant Colony Algorithm in the burst allocation problem in the US subframe of D2DWRAN.

3.2 Simulations and Results

In this section, simulation results of different algorithms within a D2DWRAN system are plotted and discussed. We assume this D2DWRAN cell has a radius of 40km and up to 200 CPEs. In

Algorithm 7 Tabu Search Algorithm when there are unlimited subchannels.

Set up Ta and Ta_l.
Generate a random sequence.
while circle time $T_{TB} < Ta_l$ **do**
  List all the possible change in Ta_c in the sequence.
  Get the sequence after changing.;
  Call the general burst allocation procedure to reorder the links with the sequence.
  Count the subchannels that have be occupied and record it to E_{TB}.
  if $E_{TB} < E_{TBtemp}$ **then**
    Check the current change in the Tabu list Ta_l.
    if the change exists in Ta_l **then**
      if $E_{TB} < E_(TBbest)$ **then**
        Take the current sequence as the best solution.
        $E_(TBbest) \leftarrow E_{TB}$
      else
        Discard the change.
      end if
    else
      $E_{TBtemp} \leftarrow E_{TB}$.
    end if
  end if
  Add the change which lead to E_{TBtemp} to Ta.
  $T_{TB} \leftarrow T_{TB} + 1$.;
end while

the upstream, some requests are sent to the BS and the rest are CPE-CPE communication within transmission radius. Each CPE generates a new request as soon as the previous request is allocated. The situations of channels are fixed. The size of requests, length of the US subchannels and the interference map between CPEs are given in the Table 3.1. All other parameters regarding different algorithms are discussed separately later in this chapter. Finally, we also compare the performance of these algorithms.

Table 3.1: Simulation parameters

Parameters	Values
Number of requests	200
Length of US subframe	40
Number of US subframe	unlimited
Size of request	200 fixed requests are given, 30% is CPE-BS 70% is CPE-CPE
Interference map	fixed and provided by previous research

We consider two main performances to evaluate the algorithm: computation time and total number of subchannels to allocate all the requests.

3.2.1 Brute Force Algorithm

With Brute force algorithm, the computation time is 200! in this simulation scenario. The number of solutions goes up to $7.88 \exp 374$. Hence, it is difficult to use Brute Force algorithm

Algorithm 8 Ant Colony Algorithm when there are unlimited subchannels.

```
1: Set up Pheromone and T_AC.
2: Assign the dots to nodes randomly.
3: while Circle time T_AC < T_max. do
4:   for each ant do
5:     List all the possible choice to other nodes in current node.
6:     Calculate the possibility to different choice.
7:     Choose one node and update the pheromone.
8:     Call the general burst allocation procedure to reorder the links with the sequence.
9:     Count the subchannels that have be occupied and record it to E_AC.
10:    if E_AC < E_ACtemp then
11:      if the change exists in Ta_l then
12:        if E_AC < E_ACbest then
13:          Take the current sequence as the best solution.
14:          E_ACbest ← E_AC.
15:        else
16:          Discard the change.
17:        end if
18:      else
19:        E_ACtemp ← E_AC.
20:      end if
21:    end if
22:  end for
23:  T_AC ← T_AC + 1.
24: end while
```

in the burst allocation problem.

3.2.2 Greedy Algorithm

In the Greedy algorithm, the requests are considered in a definite order. One possible parameter can be considered, that is how to arrange the order of the requests with the same interference degrees. There are three ways to deal with this problem that may influence the results. The first one is after ordering the requests by interference degree without any further action. The second one is when the requests are ordered by interference degree, the requests with same interference degree are ordered by the number of occupied subchannels in an accent order. While the third one is the opposite of second, it orders requests that have the same interference degree by the number of occupied subchannels in a descending order. All of these ways are tested and the results are plotted in Figure 3.1 shows the results under three methods and different interference degree arrangement. The blue bars show the performance with ascending order of interference degrees, while the red bars show the one with descending order of interference degrees. The results show evidently that the ascended interference degree can use less subchannels to be allocated all the requests, which means the better performance.

When considering both results with ascending order and descending order together, it shows another conclusion. When the interference degree is in ascending order, the subchannels number in descending order shows the best performance. When the interference degree is in descending order, the result without further ordering gets the best solution. So it is hard to conclude that the ordering the subchannels' number in descending order can give best performance, although it often shows a better solution than the one with ascended order.

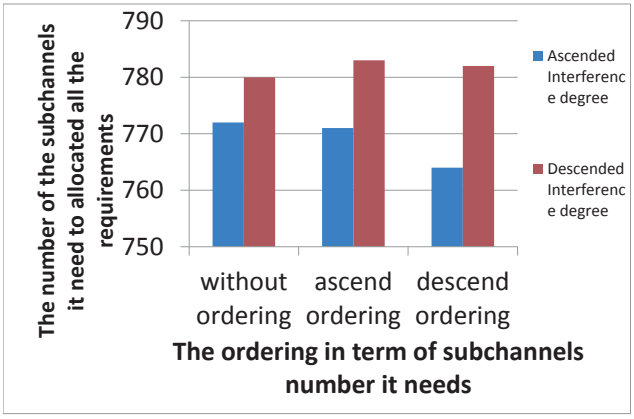


Figure 3.1: Using different ordering sequence in greedy algorithm.

3.2.3 Simulated Annealing

The main parameters in Simulated Annealing are the decrease speed k and iteration time m. k is determined by the temperature range that decides how many times to change in Simulated Annealing and how better solution it is. In Simulated Annealing, the total computation time is $O(m * k)$. m and k can be decided manually, so the computation time can be very small, in which case the performance is not reasonable.

In this simulation, two different ranges of temperature are shown in the result, $0.1 \sim 1$ and $0.01 \sim 1$. In general, with the temperature range extending, there is more time cost to get the results meanwhile, it gives a better performance to the problem. Iteration time also influences the performance of results and the time to get the result.

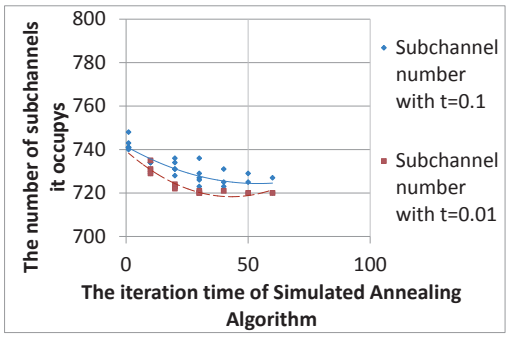


Figure 3.2: Occupied subchannels when using SA algorithm with different parameters.

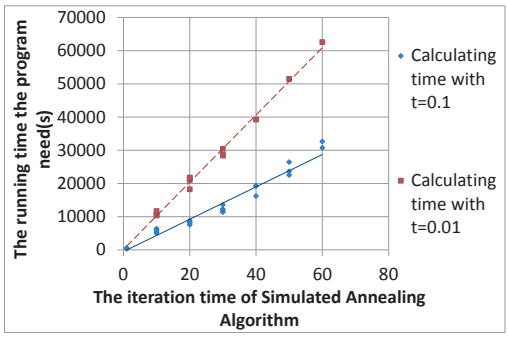


Figure 3.3: Running time when using SA algorithm with different parameters.

In Figure 3.2, there is comparison between the number of occupied subchannels when the range of temperature is $1 \sim 0.1$ and $1 \sim 0.01$. The x-axis is the circle time under each temperature, and the y-axis is the number of occupied subchannels. It can be seen that when the range of temperature increases, the number of occupied subchannels decreases. And when the circle time goes up, the number of occupied subchannels also represents a trend of descending.

So it can be concluded that the performance of Simulated Annealing greatly depends on the iteration time, which is affected both by range of temperature and the circle time. When the circle time goes up to 60, the result does not show better performance than when circle time is 50. It can be concluded that when getting closer to the optimal solution, it will be more difficult to get a further improvement.

Figure 3.3 shows the computation time of the same scenarios as in Figure 3.2. Obviously, it is more time consuming when the range of temperature is getting larger. When the range of temperature is $1 \sim 0.01$, the computation time is almost linearly increasing with the temperature.

From Figure 3.2 and Figure 3.3, it can be concluded that if the Simulated Annealing is expected to be a better performance, it needs more time to operate more iterations. However, when the solution gets close to the optimal solution, it will need much more iterations and more time consumption to achieve a better solution.

3.2.4 Genetic Algorithm

We consider four parameters in the Genetic Algorithm: Generation group size G, generation time G_m, crossover probability P_c and mutation probability P_m. The total time consumption is $O(G + G_m * G * (P_c + P_m))$

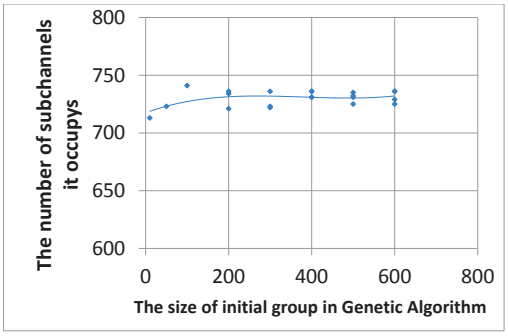


Figure 3.4: Occupied subchannels number when using GA with different initial group.

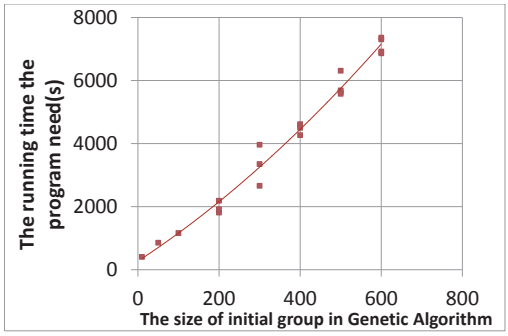


Figure 3.5: Running time when using GA with different initial group.

Figure 3.4 and Figure 3.5 show the performance when Generation time is set to 10. The x-axis is the generation group size range from 100 to 600. The y-axis in Figure 3.4 is the number of occupied subchannels, and in Figure 3.5 it is the time consumption.

In Figure 3.4 the number of occupied subchannels is almost a horizontal line. It is hard to say there is some relationship between the group size and the result. The result shows some randomness. So it can be concluded that increasing the generation group size cannot improve the performance of the Genetic Algorithm but significantly increase the cost of it.

From Figure 3.5 it can be seen the time consumption related to the generation group size almost linearly. The reason is that the initial group is the main part that needs to be calculated. When the program needs to proceed to a new generation, only two or three new lists need to be reconsidered.

In Figure 3.6 and Figure 3.7, the generation group size is 600, and we test different generation times. The x-axis represents the generation times range from 10 to 500. The y-axis in Figure 3.6 is the number of occupied subchannels, while in Figure 3.7 shows the time consumption.

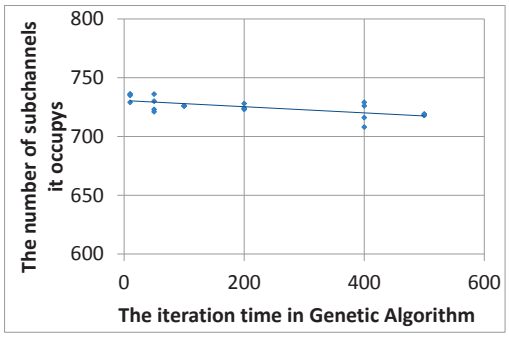


Figure 3.6: Occupied subchannels number when using GA with different generation times.

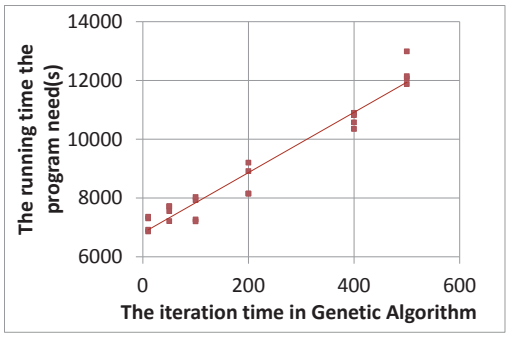


Figure 3.7: Running time when using GA with different generation times.

From Figure 3.6, It can be seen that as the generation times increases, there is a decline in the performance. It also shows that same tendency when the result is getting close to the optimal solution. But In Figure 3.7, the time consumption keeps increasing, and the tendency is much more obvious than the tendency of the number of subchannels going down. And the time already goes up to 12000sec when the iteration time goes up to 500, so it is difficult to get a further solution.

Compared to the Simulated Annealing, the Genetic Algorithm shows a higher performance. When the generation group size is 600, its result can stay behind 736, and its computation time is only 12000s, which is one fifth of the Simulated Annealing.

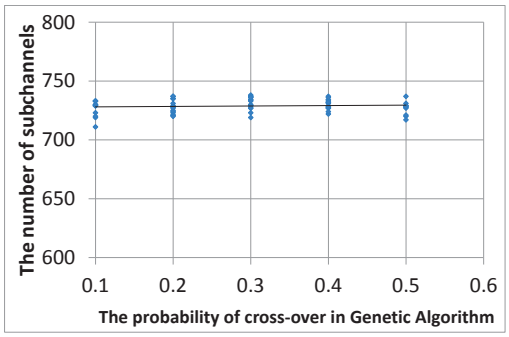


Figure 3.8: Occupied subchannels number when using GA with different cross over probability.

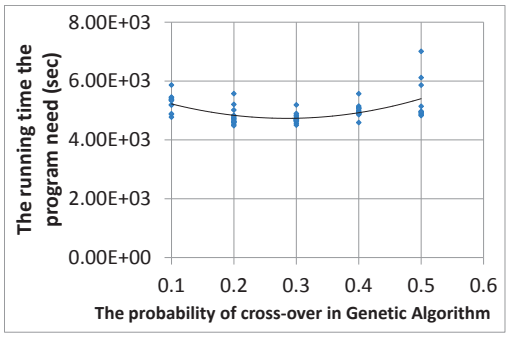


Figure 3.9: Running time when using GA with different cross over probability.

Figure 3.8 to Figure 3.11 show the results when the size of initial group is 600 and the generation time is 100. Figure 3.8 and Figure 3.9 show the occupied subchannels number and time consumption when the cross over probability ranges from 0.1 to 0.5. Figure 3.10 and Figure 3.11 show the occupied subchannels number and time consumption when the mutation probability ranges from 0.1 to 0.5.

The results show both of the probabilities can influence the occupied subchannels number

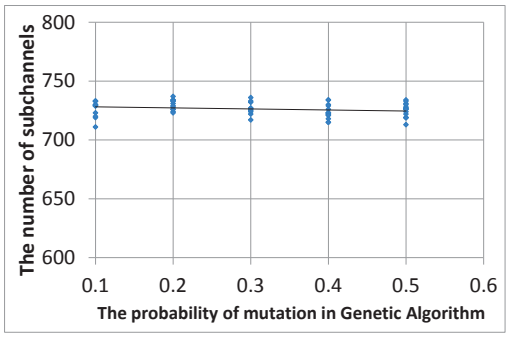


Figure 3.10: Occupied subchannels number when using GA with different mutation probability.

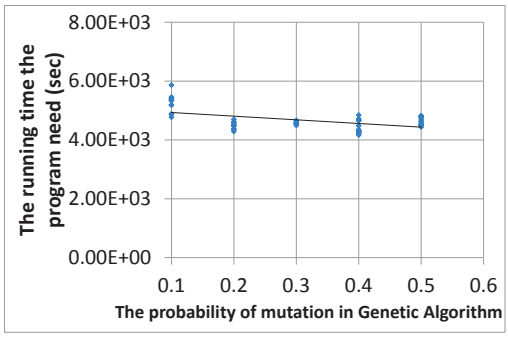


Figure 3.11: Running time when using GA with different mutation probability.

and time consumption limitedly. The performance is nearly linear as either of them changes. So in our other simulations later in this thesis, both of them are assigned with a specific value 0.5.

From the last four simulations it can be seen that the simulation results are different from the theoretical results. The crossover probability and mutation probability cannot influence the results as expected. The most efficient way to improve the performance of Genetic Algorithm is by increasing the iteration time in each generation. While going up to a certain size, the initial group can influence the results further.

3.2.5 Tabu Search

The Tabu Search mainly depends on the length of Tabu list T_l. The time consumption of the algorithm is $O(\binom{200}{2} * T_l)$. If the Tabu list is set to 15, then there will be about 30000 solutions to be calculated. Compared to Genetic Algorithm and Simulated Annealing, it is not efficient. Therefore, we do not discuss it further later in this thesis similar to Brutal Search.

3.2.6 Ant Colony Algorithm

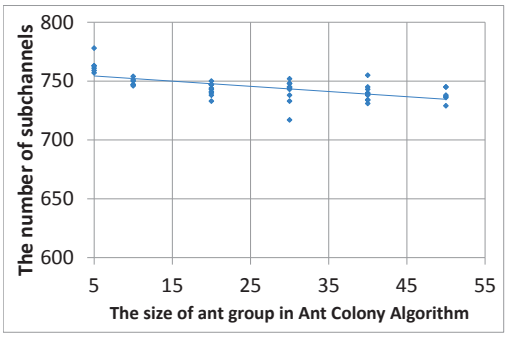
Ant Colony Algorithm also has four parameters: the number of ants N_a, circle time m, attractiveness α and trail level β. Among them, later two can be determined by the probability of one move chosen by ants. The formulation can be shown as,

$$p = \frac{\alpha\beta}{\sum_{all\ allowed\ route} \alpha\beta} \tag{3.3}$$

The formulation of these two basic factors are changed to get different results under different scenarios. The time complexity is about $O(N_a * m)$.

In general, Ant Colony Algorithm can give a good solution when the iteration time increases. The detailed results are shown below.

There are eight figures to provide the results of Ant Colony Algorithm. Figure 3.12 and Figure 3.13 show when the other parameters are fixed, the performance and consuming time with the size of ants group. In Figure 3.12 and Figure 3.13, the iteration time is set to 20, and the factors *alpha* and *beta* are set to 2 and 5 respectively. In Figure 3.14 and Figure 3.15, the



z

Figure 3.12: Occupied subchannels number when using AC algorithm with different group size.

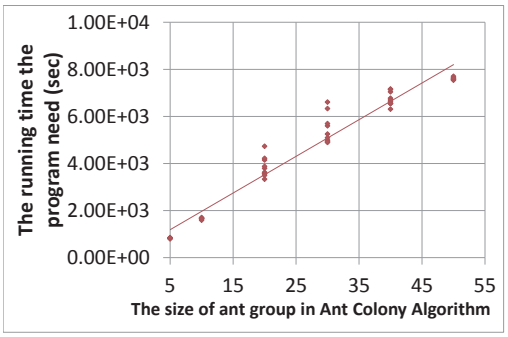


Figure 3.13: Running time when using AC algorithm with different group size.

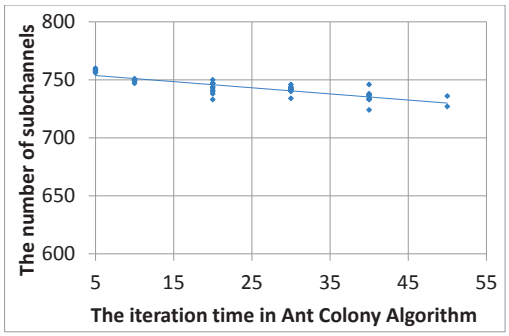


Figure 3.14: Occupied subchannels number when using AC algorithm with different circle time.

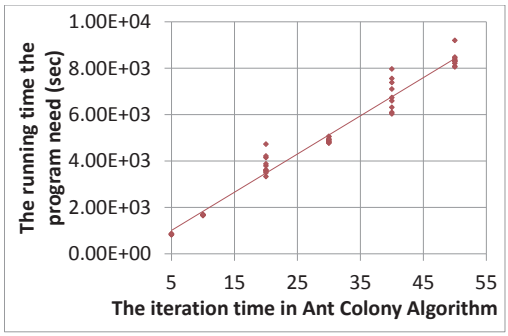


Figure 3.15: Running time when using AC algorithm with different circle time.

size of ants group is also 20, and factors *alpha* and *beta* are set to 2 and 5 respectively. The x-axis is the iteration times from 1 $\sim$ 50. In Figure 3.16 and Figure 3.17, the only changed parameter is *alpha*, from 1 to 5. While in Figure 3.18 and Figure 3.19, it is the result from different value of *beta*.

The result shows with either the number of ants or circle time increasing, the cost time increases significantly. However, the performance does not get better as expected. There may be the problem of this design of program. Ant Colony Algorithm is a graphic algorithm that needs the distance between two nodes. In this program, the distance is defined as the number of subchannels required. As the result showing, most of the solutions have less differences than 30 subchannels, and the total number of the subchannels goes up to 800 subchannels. Compared to the cardinal number, the change is very limited, which make ants hard to choose a better path.

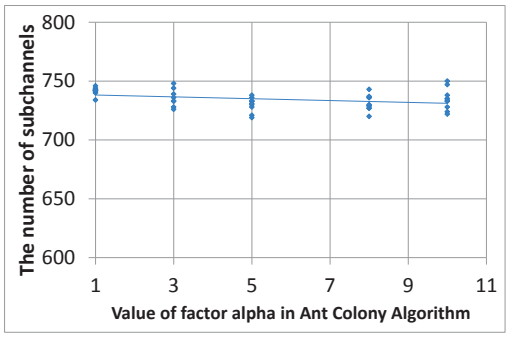


Figure 3.16: Occupied subchannels number when using AC algorithm with different factor α.

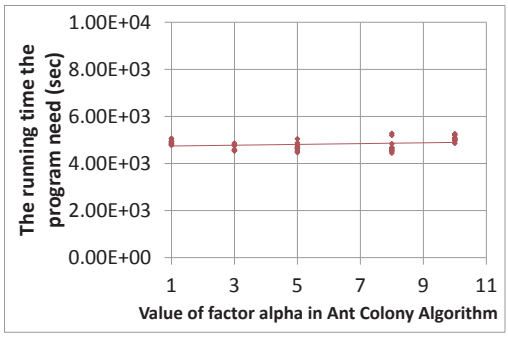


Figure 3.17: Running time when using AC algorithm with different factor α.

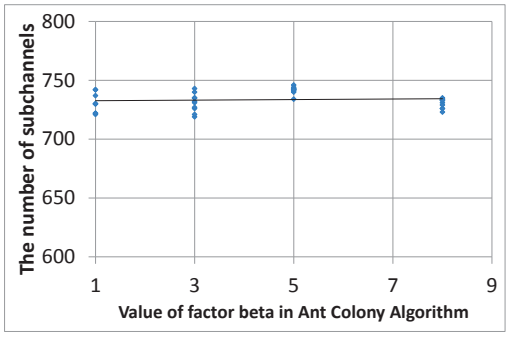


Figure 3.18: Occupied subchannels number when using AC algorithm with different factor β.

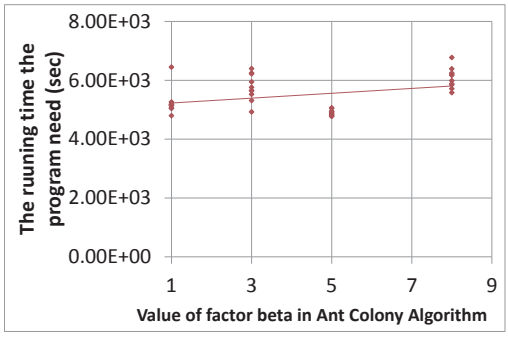


Figure 3.19: Running time when using AC algorithm with different factor β.

3.3 Summary

In this chapter, the implementations of existing optimizations under unlimited subchannels scenarios are discussed and simulated. It can be seen that it is hard to the implement the Brute force algorithm and Tabu Search because of the high computational complexity. Greedy algorithm can provide a quick result, but the performance is not guaranteed. Simulated Annealing and Genetic Algorithm can control the time and performance. Both of them need much more time than Greedy algorithm, but also can achieve better performance. Simulated Annealing needs more time to get the same performance as Genetic Algorithm. Ant Colony Algorithm does not show the performance as expected, we have to analyse the problem and it needs to be modified in further works.

Chapter 4

Algorithm with Limited Subchannels

In this chapter, the burst allocation problem in the US subframe of D2DWRAN with limited subchannels is studied based on the problem with unlimited subchannels. In Section 4.1, we propose three methods to fulfil the requirement of limited subchannels. In Section 4.2 and Section 4.3, the implementation and simulation results are discussed.

4.1 Implementation of Different Methods

When limited subchannels are considered, the formulation of problem becomes the Eq.(2.13) and Eq.(2.14) as discussed in Chapter 2. After getting the results under unlimited subchannels, we need to modify the solutions with unlimited subchannels into solutions with limited subchannels.

A simple way is to choose the required number of subchannels in the solutions with unlimited subchannels. From the simulation results in Chapter 3, we know that there are about 800 subchannels to allocate all requests, therefore the total choices are $\binom{800}{60}$. However, the strategy to choose the limited subchannels (60 subchannels in the OFDMA system of D2dWRAN) is critical to the performance of solutions. We propose three methods based on the optimization of algorithms in this section for the problem with unlimited subchannels: abandoning rule, cutting rule and selective choosing. The abandoning rule method is used while allocating each request and the cutting rule method is considered at the end of each allocation iteration time. Different from the previous two, the selective choosing only processes the final output solution to get the solution with limited subchannels. The details of these methods are discussed below.

4.1.1 Method One: Abandoning Rule

This method is based on setting up the limited number of subchannels initially. So before solving the problem, the number of subchannels is set to be a definite number directly, in this simulation it is 60, then it uses existing algorithm to process the request. While allocating a new request, this request is first allocated in the same manner as the scenarios with unlimited subchannels. Then the program will check whether the total number of subchannels are beyond the number of subchannels. If it is not beyond the number of subchannels, the new request will be kept in the channel. Otherwise, the new request is discarded directly.

From the algorithm description it can be seen that the number limitation is set up at the beginning of the program, and it will check the total number of subchannels during each iteration. We also consider two cases regarding the length of the US subframe, and the detailed description of this method is shown in Algorithm 9 and Algorithm 10.

In these algorithms, the first is for the case that the length of US subframe is less than 14, and the other one is for the case that length of US subframe is more than 14. It can be seen

Algorithm 9 Abandoning rule in case $S_l < 14$.

```
Set up the subchannels to N_s.
if S_l < 14 then
  for Every request B_ij in the queue. do
    for every subchannel k do
      Deploy B_ij starting from kth subchannel.
      for next B_ij^m subchannels in S_s(k, ij) do
        Examine whether it interferes with the existing request.
        if interference is found then
          Withdraw the deployment, try next subchannel.
        else
          Check the whether number of subchannels k > N_s.
          if k > N_s then
            Abandon the request directly, exit loop and try next request.
          else
            Allocate with the deployment, exit loop and try next request.
          end if
        end if
      end for
    end for
  end for
end if
```

that the main body of the program doesn't change much compared to the program which is under unlimited subchannels, except that some constraints are added. Because this method needs to check all the requests every iteration time, the computational complexity is the same as the problem with unlimited subchannels and the solutions with unlimited subchannels are not used directly in this method.

4.1.2 Method Two: Cutting Rule

This method is also implemented during each iteration of the optimization algorithms, and it is called at the end of each iteration time. When a temporary solution is built in an iteration of the algorithms for the problem with unlimited subchannels, a cutting rule is inserted to pick up some certain subchannels from the unlimited subchannels to form the temporary solution with limited subchannels. Therefore, the main consideration in this method is to select N_s subchannels from the allocation with unlimited subchannels. We choose the first N_s subchannels in this method. Because allocation algorithms allocates starts from the third subchannels of a US subframe and goes to the next subchannel one by one. So it has a probability that the first N_s subchannels contain more requests than other subchannels. Even though there may be more requests assembling on other parts of the channel, the performance of the algorithms are not influenced. That is because the cutting rule is called during iteration and the best solution may be kept. The algorithm description is shown in Algorithm 11.

This method can directly use the results from the scenarios with unlimited subchannels, hence, it does not add extra computation. However, the first N_s subchannels does not necessarily contain the most requests compared to other subchannels and the performance of this method cannot be guaranteed. Therefore, when we implement this method, we also consider other two selecting strategies: the last N_s subchannels and random N_s subchannels. The simulation results are discussed and compared later in this chapter.

Algorithm 10 Abandoning rule in case $S_l \geq 14$.

```
if S_l ≥ 14 then
  for every request S_ij in the queue do
    for every subchannel k, do
      Deploy B_ij starting from kth subchannel.
      for next B_ij^m subchannels in S_s(k, ij) do
        Examine whether it interferes with existing requests.
        if interference is found then
          Check whether available slots is more than the request.
          if the slots is enough then
            Return a true result and try next subchannels.
          else
            Return a false result and exit the loop.
          end if
        end if
      end for
      Examine the result.
      if all return is true then
        Check the whether number of subchannels k > N_s.
        if k > N_s then
          Abandon the request directly, exit loop and try next request.
        else
          Allocate with the deployment, exit loop and try next request.
        end if
      else
        if there exist a false result then
          Withdraw the deployment, and try next subchannel.
        end if
      end if
    end for
  end for
end if
```

4.1.3 Method Three: Selective Choosing

In this method, a simple cutting rule is used similar to the greedy algorithm. The final output solution of the optimization algorithms under unlimited subchannels situation is recorded based on the algorithms in Chapter 3. Then the selective choosing method is called to process the solution with unlimited subchannels into a solution with limited subchannels. Similar to greedy algorithm, all the subchannels are sorted in descending order first by the request number in subchannels. Then subchannels containing the most requests are selected first. However, a request often occupies multiple subchannels, and the neighbouring subchannels which contain the same the requests should also be selected. The selective choosing method is described in Algorithm 12:

This method is more intelligent than other two methods. On the contrary, it will require more time to calculate all the results to get the final solution.

Algorithm 11 Cutting rule

1: Start each algorithm.
2: Set up the subchannels to unlimited.
3: Run the main part of each algorithm.
4: In each end of iteration,
5: Cutting the result to N_s.
6: Count the number of requests N_a.
7: **for** each row **do**
8: Count the subchannels N_s and $N_s + 1$ by rows.
9: **if** the count equal to 2 **then**
10: $N_b \leq N_b + 1$.
11: **end if**
12: go to the next row;
13: **end for**
14: Final subchannels number is $N_a - N_b$ in this iteration.
15: Go to the next iteration.
16: Compare the result of each iteration and get the better solution.

4.2 Methods Implementation

In this section, we implement and simulate the above three methods in Simulated Annealing, Genetic Algorithm and Ant Colony Algorithm. The three methods are considered in five sub-methods, including the abandoning rule, the cutting rule when selecting the first 60 subchannels, the cutting rule when selecting the last 60 subchannels, the cutting rule when randomly selecting 60 subchannels, and selective choosing.

In Simulating Annealing Algorithm, the temperature range is set from 100 to 1, the iteration time in each temperature is 10, and the declined efficiency is 0.05. In Genetic Algorithm, the initial group is set to 100, and the generation time is also 100 times, the probability of mutation and cross over is set to 0.3. In Ant Colony Algorithm, the number of ants is set to 30, the circle time is set to 50, the factor alpha is set to 5, and the factor beta is set to 3.

All algorithms use the same parameters of channels to make the comparison convenient. 200 established requests are provided with constant length and interference map. The length of subchannels is 40, and there are 60 subchannels.

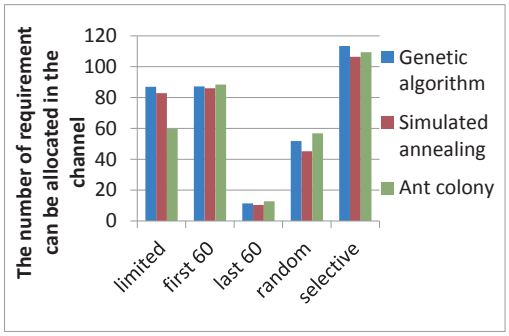


Figure 4.1: Comparison of allocated requests.

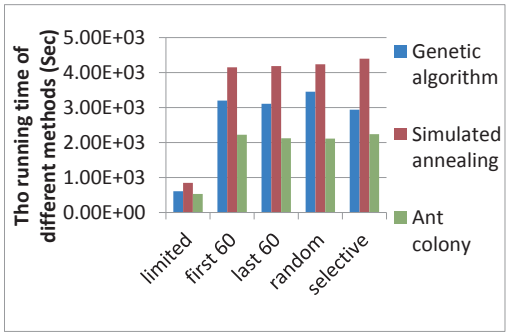


Figure 4.2: Comparison of cost time.

Algorithm 12 Selective choosing

```
Set up the subchannels to unlimited.
Run each algorithm under unlimited subchannels situation.
Get the final result of subchannels.
Sort all the subchannels with the request number in descending order.
Set S_f = 0.
for every subchannels S_n in this new order do
    Count its entire neighbor subchannels which contain some same request.
    S_t <= S_f + S_n.
    if S_t <= N_s then
        S_f = S_t.
    else
        Discard the S_t and quit the loop.
    end if
end for
Count all the requests in the selected subchannels.
```

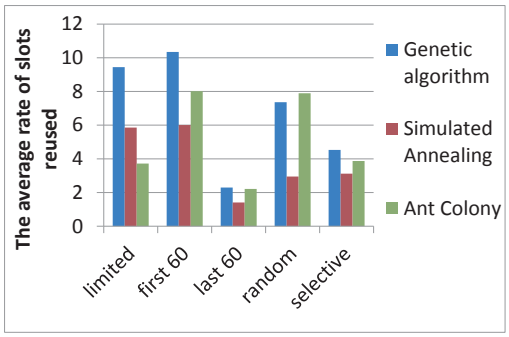


Figure 4.3: Comparison of slots usage.

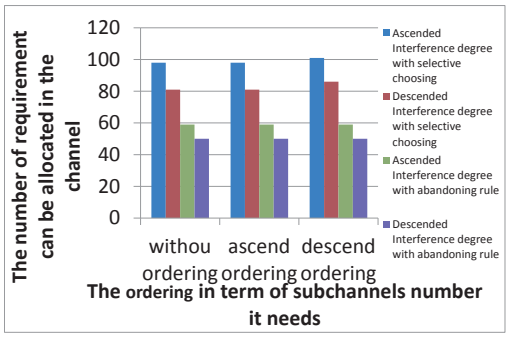


Figure 4.4: Comparison with three different parameters in Greedy algorithm.

Figure 4.1 and Figure 4.2 show the results of allocated requests and running time in different algorithms while using different methods. In Figure 4.1, results of 5 scenarios mentioned above in three algorithms are presented. The x-axis represents the results with five different methods. From left to right, is the abandoning rule, keeping first 60 subchannels, keeping last 60 subchannels, keeping random 60 continuous subchannels and selective choosing method. The y-axis represents the number of requests that can be allocated in the channels. The bars show the results with three algorithms separately, from left to right, they are results with Genetic Algorithm, Simulated Annealing and Ant Colony Algorithm. In Figure 4.2, the running time of different methods is given. Similarly in Figure 4.1, the x-axis and bars represents different methods and different algorithms. The y-axis represents the running time.

From the results it can conclude that as analysed, the method that keep first 60 subchannels can perform better than the method keeping other subchannels. However, if the subchannels are limited initially, the results also show a good performance as the first 60 subchannels, at the same time, it shows shorter time than other methods. Selective method shows a better performance than other methods in requests' capacity, while the time taken is the same as in

the method of keeping first 60 subchannels.

From the results in Figure 4.3, we can conclude that the main efficiency of the algorithm is based on the number of subchannels. So from this point of view, fewer subchannels lead to less time consumption. Meanwhile, fewer subchannels also lead to less capacity. In this case, selective choosing can achieve better performance than others.

In Figure 4.3, another performance evaluation of the solutions is used, which calculates the slot reuse. This result can directly show us how many times the slots are reused in program. It can give us another way to evaluate the methods used in solving the burst allocation problem. However, this evaluation is different from the expectation we initially proposed, so it can be a reference to results of this project. We can see from the figures that the trends of slot reuse are different from request capacity. For example, although selective choosing method shows better performance than other methods in request capacity, the slot reuse of selective choosing is not better than the rest of the methods. Its average slot reuse time is even half of result of using abandoning rule. Meanwhile, the method that just keeps first 60 subchannels achieves the best slot reuse.

Figure 4.3 also shows the performance of three different optimization algorithms in terms of slot reuse. It shows that Genetic Algorithm can usually give the best performance of all. On the contrary, Simulated Annealing achieves the worst performance in the point of view of the slot reuse. In the Ant Colony Algorithm, the abandoning rule method cannot give the same performance compared to the performance of abandoning rule in other algorithms.

In the following simulations and results, the slot reuse performance is also examined. Note that slot reuse is not the main goal of our problem, but the main goal of our problem is to allocate as many requests as possible in the limited subchannels. However, it is still an important reference to evaluate the channel utilization.

4.3 Simulation and Results

According to the analysis in Section 4.2, two main methods are chosen for further simulation and examination: the abandoning rule and selective choosing. The reason for choosing abandoning rule is because that it can get a solution efficiently, i.e., its running time is no more than one third of other methods. Although the performance of the abandoning rule may be not as good as other methods, the loss in request capacity is less than 10% compared to the best methods. Thus, we study the abandoning rule further because of its time efficiency and comparable performance. The reason to study selective choosing further is because it doesn't spend more time than cutting rule, but it shows nearly 20% better performance than other methods. From the points of views of time consumption and performance, it is not worthwhile to study the cutting rule anymore and we only focus on the abandoning rule and selective choosing in the following discussion.

In next step, combination of abandoning rule and selective choosing and different optimization algorithms are simulated and the results are presented. Four optimization algorithms are considered: Greedy algorithm, Simulated Annealing, Genetic Algorithm and Ant Colony Algorithm.

4.3.1 Greedy Algorithm

In this part, we also consider the three parameters. They are interference degree order, subchannels number order, method to modify the unlimited situation into limited one.

Figure 4.4 shows the similar situation as Figure 3.1. The simulation with ascending interference degree and descending number of subchannels shows the best performance. It can be

confirmed that Greedy algorithm is more suitable with ascending interference degree.

On the other hand, when Greedy Algorithm is combined with the abandoning rule and selective choosing, although the abandoning rule can save about 30% running time, it brings out nearly 50% less performance than when selective choosing method is used.

So in greedy algorithm, there is no doubt that selective choosing method should be used to make the algorithm more efficient.

4.3.2 Simulated Annealing

In Simulated Annealing algorithm, there are still two parameters to change, the temperature range and iteration time.

The results with different methods are showed in Figure 4.5 and Figure 4.6. In Figure 4.5, the x-axis represents the iteration time of Simulated Annealing. The y-axis represents the requests capacity of the different algorithms. There are four lines in the figure, each line shows different combination of range and methods. The two upper lines show range from 1 0.1 and 1 0.01 with Selective choosing method. The two under lines show the same range with abandoning rule. Figure 4.6 shows the results of running time of different parameter values in Simulated Annealing.

Figure 4.5 gives the number of requests that can be allocated in the channel. As we can see the selective choosing method shows 20% better performance than the abandoning rule method. When the iteration time increases to 50, it is extremely time consuming. While the abandoning rule does not only show drawback in request capacity, it appears instable when the iteration time is small. This instability especially appears when range is small such as from 1 to 0.1.

Figure 4.6 shows the running time with different methods and different parameters in Simulated Annealing. It shows when the iteration time is small, the difference of running time between selective choosing and abandoning rule is not obvious. But as the iteration time increases, the difference grows. When the iteration time goes up to 50, the running time of selective choosing is almost five times to the one of abandoning rule.

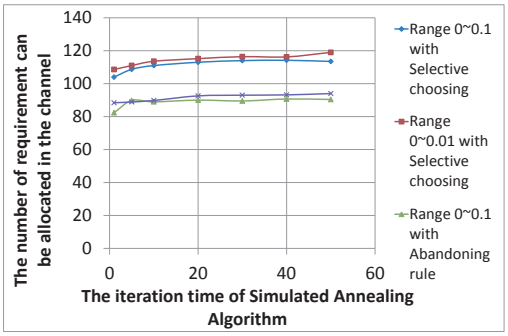


Figure 4.5: Results of allocated requests in SA.

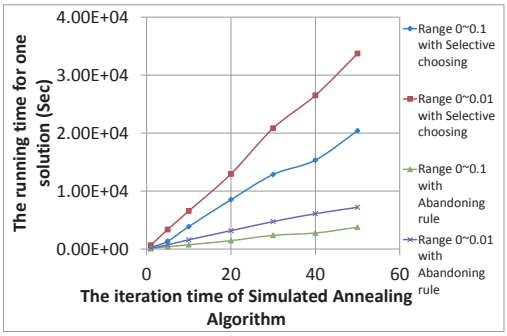


Figure 4.6: Results of running time in SA.

Figure 4.7 shows the average slot reuse time of different combinations in Simulated Annealing. The x-axis and lines are still the same as in Figure 4.5. The y-axis shows the average rate of slots reused. It can be seen that the average slot reuse are quite unstable. The reason is still that the program doesn't consider the rate of slots reused, so the result represents a fluctuation that can be considered as a common pheromone. It is different from Figure 4.5 and Figure 4.6, and the performance of abandoning rule absolutely beyond selective choosing method.

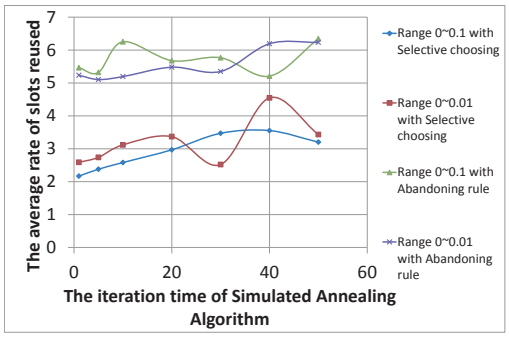


Figure 4.7: Different average rates of slots reused in SA.

So it can be concluded that if the iteration time is small, we can adapt selective choosing method in Simulated Annealing to obtain high performance. As to large iteration time, the time consumption increases much faster than the burst allocation performance. In this case, the only choice is to sacrifice some performance to save the running time by using abandoning rule.

4.3.3 Genetic Algorithm

In this simulation different group sizes and iteration time are considered.

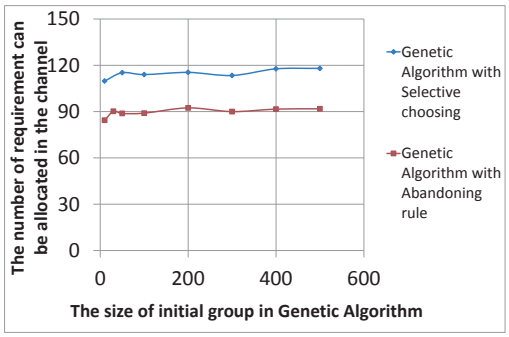


Figure 4.8: Number of allocated requests in GA with different initial group size.

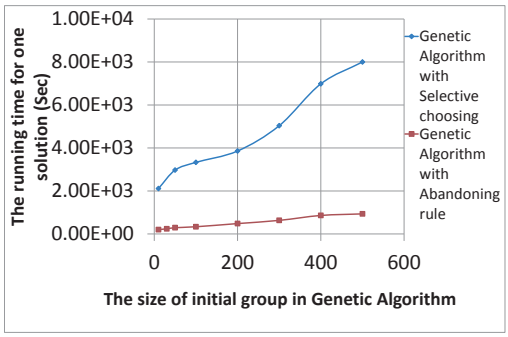


Figure 4.9: Running time in GA with different initial group size.

Figure 4.8 and Figure 4.9 are results in Genetic Algorithm with different initial group sizes. The iteration time is set to 200, and the occurring probability of cross-over and mutation are both set to 0.5. Figure 4.8 shows the request capacity with different initial group sizes and methods. The y-axis shows the number of requests allocated in the channel. The two lines are different results of two methods: abandoning rule and selective choosing. The upper one is the result with selective choosing method, while the under one is the result with abandoning rule. Figure 4.9 shows the running time.

We can see from Figure 4.8 that the initial group size influences the performance less than in the unlimited subchannel scenarios in Figure 3.4. As the size of initial group increases, the

performance is nearly the same. It has a sharp increasing tendency when the size goes up to 50. Because Genetic Algorithm needs a big enough group size at first. When the group size can fulfil the requirement that provides sufficient sample, the increasing group cannot continue to enhance the performance of the algorithm. While solving the burst allocation problem in limited subchannels, the group size can shrink to 50 because it is not helpful to improve the performance of algorithm.

In Figure 4.9, the running time with selective choosing is more significant than Figure 4.6. In general, the running time by using selective choosing method is much more obvious than by using abandoning rule. As the size of initial group increasing, the running time by using selective choosing method increases much more than by using abandoning rule. When the size of initial group goes up to 400, the running time of selective choosing method is almost seven times to the one of abandoning rule. Even when the size of initial group is small, the running time of selective choosing method is still much more than abandoning rule.

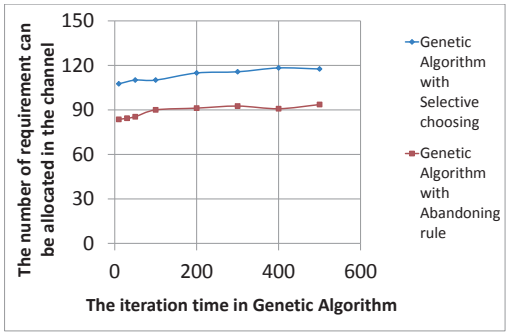


Figure 4.10: Number of allocated requests in GA with different initial group size.

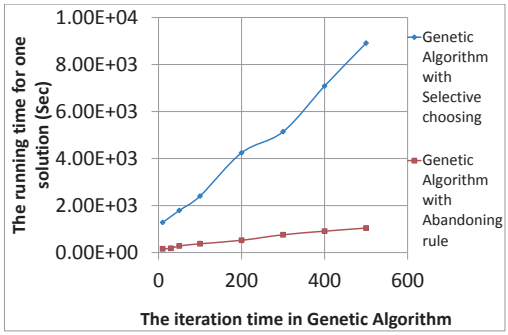


Figure 4.11: Running time in GA with different iteration time.

Figure 4.10 and Figure 4.11 are the results with two methods and different iteration time in Genetic Algorithm. In this simulation, the initial group is set to 200, and the occurring probability of cross-over and mutation are also set to 0.5.

In Figure 4.10, the selective choosing method shows an improved performance when iteration time increasing. While the abandoning rule displays the same instability as in Simulated Annealing. So if the abandoning rule is chosen, low iteration time is enough for the performance. As to selective choosing method, it needs to consider to choose a point in low value of parameters to ensure the performance of the method. In Figure 4.11, the slope of the selective choosing method is much larger than the one in Figure 4.9. It can be concluded that in general, the abandoning rule is more suitable than selective choosing method. If better performance is needed, the modification with selective choosing needs sacrificing much more time, which is not desirable in the burst allocation problem.

In Figure 4.12 and Figure 4.13, the average of slots reuse time with different values of parameters are plotted. Figure 4.12 shows the results with different initial group sizes of Genetic Algorithm and Figure 4.13 shows results with different iteration time of Genetic Algorithm. Figure 4.12 and Figure 4.13, it can be concluded that abandoning rule is still better than selective choosing method. Compared to Figure 4.7, it shows that the stability is better than the latter. Moreover, the reusing time is improved obviously, increasing from about 5.5 to 9.

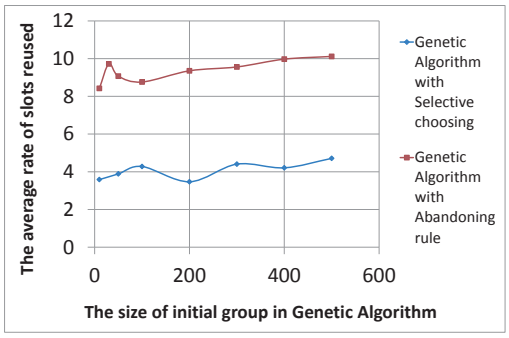


Figure 4.12: The average rate of slots reused in GA with different initial group size.

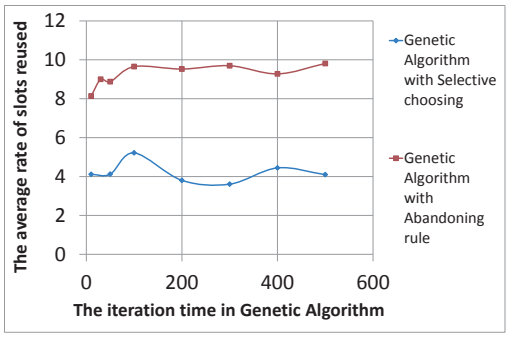


Figure 4.13: The average rate of slots reused in GA with different iteration time.

4.3.4 Ant Colony Algorithm

In Ant Colony Algorithm, there are only two parameters to change in order to assess the performance of two methods in different situations: the size of ant group and the iteration time.

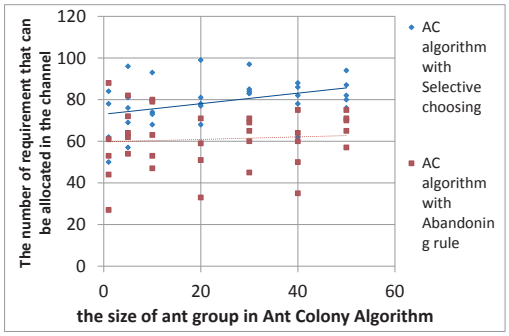


Figure 4.14: Number of allocated requests in AC with different initial group size.

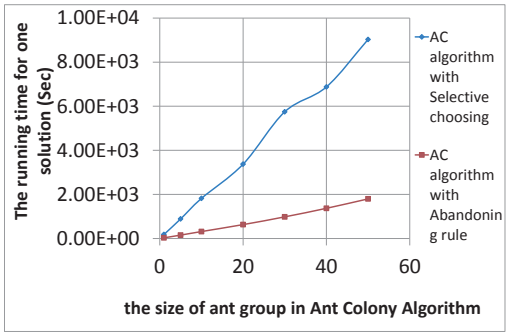


Figure 4.15: Running time in AC with different ant group size.

Figure 4.14 to Figure 4.17 are the results of two methods in Ant Colony Algorithm with different values of ant group size and iteration time separately. In Figure 4.14 and Figure 4.15, the iteration time is set to 20, the two factors in AC α and β are set to 2 and 5 separately, the only changing parameter is the size of ant group and method. While in Figure 4.16 and Figure 4.17 the changing parameter changes from ant group size to iteration time. The ant group size is set to 20.

In Figure 4.14, the x-axis is the size of ant group in Ant Colony Algorithm. The y-axis is the capacity of the requests. Different lines means different methods. In Figure 4.15, the y-axis is changed to the running time of the algorithm, then others are kept the same as in Figure 4.14. Figure 4.16 and Figure 4.17 have the similar measurement with Figure 4.14 and Figure 4.15. The detailed data are shown in the figures instead of the average value. We also draw a tendency line to show the trend of the performance with different methods.

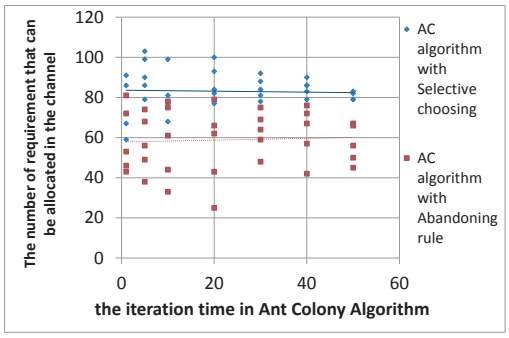


Figure 4.16: Number of allocated requests in AC with different iteration time.

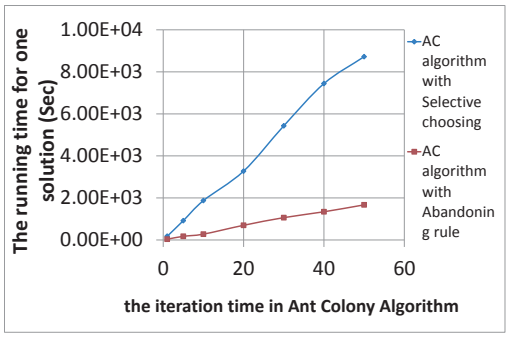


Figure 4.17: Running time in AC with different ant group size.

In Figure 4.14 and Figure 4.16, it can be seen that the results are not like other algorithms before which show regulatory. On the contrary, the data shows a complete irregularity. From the tendency line, we still can see that the selective choosing method shows better performance than abandoning rule method.

In Figure 4.15 and Figure 4.17, the running time of two methods with different parameters in Ant Colony Algorithm are quite similar to the one in Genetic Algorithm. But considering the complication of performance in AC, more efforts are needed to get further results.

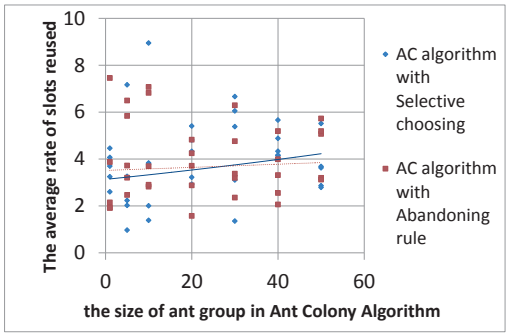


Figure 4.18: The average rate of slots reused with different ant group size.

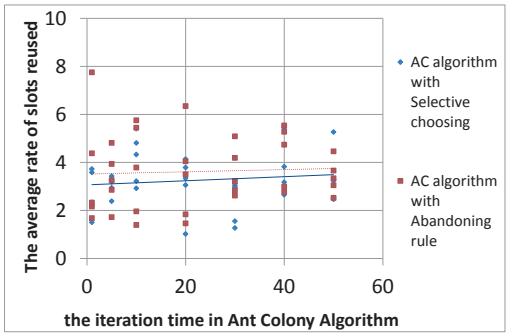


Figure 4.19: The average rate of slots reused with different iteration time.

Figure 4.18 and Figure 4.19 show the average slot reused time in Ant Colony Algorithm with different values of parameters. Figure 4.18 is the performance with different group size of ants, while Figure 4.19 is the performance with different iteration time. The results have some randomness, therefore, in Figure 4.18 and Figure 4.19, we only show all the results instead of the average value. The lines drawn in the figures are the average of the scattered results to see its tendency. Compared to other results in Ant Colony Algorithm, it is difficult to see the differences between the two methods. The results are similar with different methods or with different values of parameters.

4.4 Summary

In this chapter, we extend the problem from unlimited subchannels into limited subchannels by three methods: abandoning rule, cutting rule and selective choosing. Among these three methods, the abandoning rule has the least running time, while the selective choosing method has the best performance. Then these two methods are combined and examined with different optimizations algorithms with different values of parameters. The results show that the Simulated Annealing method and Genetic Algorithm method can both allocate nearly 120 requests when using selective choosing method, which is better than when using abandoning method. However, regarding time consuming, the selective choosing method performs worse than the abandoning rule. Therefore, when the time consumption is acceptable, it is better to choose selective choosing with higher performance rather than the abandoning rule. While the size of the problem grows, the selective choosing scarifies too much time to get a good performance. As to Ant Colony Algorithm, it shows highly randomness in the problem with limited subchannels compared to the problem with unlimited subchannels. The results cannot provide conclusion whether the abandoning rule or the selective choosing is better when Ant Colony Algorithm is adapted.

Chapter 5

Conclusions and Future Work

In this chapter, we draw some conclusions and list future work in D2DWRAN. In Section 5.1, all the discussed optimizations algorithms and methods regarding the burst allocation problem in D2DWRAN are reviewed with some final conclusions. In Section 5.2, the future work regarding the burst allocation problem is discussed.

5.1 Conclusions

The burst allocation problem in D2DWRAN is a computationally hard problem, which is the main topic of this thesis. In Chapter 2, we introduced D2DWRAN and brought out the burst allocation problem. This problem was formulated mathematically in Chapter 2 too as a basis of the following discussions in this thesis. In order to solve this problem, we divided it into two different scenarios: with unlimited subchannels and with limited subchannels. In the unlimited subchannel scenario, we assumed that there are always enough subchannels to allocate all the requests in the US subframe. This is a simplified problem compared to the scenarios with limited subchannels, and we solved this problem first in Chapter 3. Moreover, it can be considered as multiple channels situations that can be useful in further work of D2DWRAN since multiple operating channels are one of the major proposals of D2DWRAN. The limited subchannels scenario was based on the unlimited subchannels scenario with some other methods. This problem is solved in Chapter 4. Therefore, the main research content in this thesis contains two parts: implementation of optimization under unlimited subchannels scenario and implementation of optimization under limited subchannels scenario.

Before solving the problem in Chapter 3 and Chapter 4, we introduced the existing algorithms that had the potential to solve the burst allocation problem in Chapter **??**. These algorithms includes classic optimization: D&C, Greedy Algorithm and dynamic programming; and heuristic algorithm: Simulated Annealing, Tabu Search, Genetic Algorithm and Ant Colony Algorithm. With in-depth analysis and discussion, the Greedy Algorithm, Simulated Annealing algorithm, Tabu Search, Genetic Algorithm and Ant Colony algorithm were chosen to be investigated further in solving the burst allocation problem in Chapter 3 and Chapter 4.

In Chapter 3, the implementation of each algorithm in the burst allocation problem with unlimited subchannels was studied and analyzed. As a conclusion of this chapter, Tabu Search and Brute force algorithms are not suitable for the burst allocation problem because of their extremely high time complexity. Therefore, four other algorithms are simulated and discussed further in this chapter, which are the Greedy Algorithm, Simulated Annealing, Genetic Algorithm and Ant Colony Algorithm. Among these four optimization algorithms, even though the performance of the output solution cannot be guaranteed, Greedy Algorithm has the lowest time complexity compared to other algorithms. Therefore, it is still possible to be used to solve

this problem. Heuristic algorithms show higher performances than Greedy Algorithm, but their running time becomes very long. Meanwhile, one can control heuristic algorithm's running time and performance by playing with the parameters. For example, Simulated Annealing can control the temperature range and circle time in one temperature, both of which can strongly influence the running time and performance of the Simulated Annealing program. Genetic Algorithm achieves better performance than Simulated Annealing as well as shorter running time. It has four parameters to control: size of initial group, generation time, cross-over and mutation probability. The size of initial group can strongly influence the running time of program, but its affects is very limited to the performance, while iteration time can improve the performance efficiently. Both of the probabilities cannot influence the results much. Therefore, it is possible to set a small initial group and select a certain value to iteration time to get desired results. Ant Colony Algorithm also has four parameters to control its performance and running time: ant group, iteration time, factor alpha and factor bate. However in this thesis, the result of Ant Colony Algorithm showed there is some problems existing in the simulation which cannot lead to the right results. So it needs more study to get to the right results. The simulation results showed that the running time of the Genetic Algorithm is accepted but performance of the output solution is worse than the Simulated Annealing.

In Chapter 4, we focused on taking advantages of the studies in Chapter 3 to solve the burst allocation problem with limited subchannels. Three methods were proposed: abandoning rule, cutting rule and selective choosing. After comparing these three methods, abandoning rule and selective choosing method were chosen to be employed in the optimization algorithms so solve the burst allocation with limited subchannels. Greedy Algorithm was also implemented and simulated in this chapter as a reference substance. It showed that compared to Greedy Algorithm, the performance of heuristic algorithms were about 20% better, which is worthwhile for the extra time consumption. However, Ant Colony Algorithm also showed serious instability in the limited subchannel problem than in the unlimited subchannels scenario. Hence, it is less desirable to be used to solve this problem.

5.2 Future Work

The optimization of the burst allocation in D2DWRAN is studied in this thesis including the existing algorithms and implementation of these algorithms. However, there are still many aspects regarding this topic need be done in future work before it can be implemented in D2DWRAN. For example, the studies of optimization in this thesis is based on the model established in Chapter 2, and it is possible to modify the model and apply different formulation to simulate the problem. Accordingly, the performance of the optimization algorithms might be different.

Further, we only discussed the possibility of several optimizations in burst allocation problem. There are some other algorithms that might be suitable for this problem, such as combination of several algorithm, and some other Artificial Intelligence algorithm. Moreover, the combination of two or more optimizations is also an option to enhance the performance. In this project, the main problem with the optimization algorithm is the high time complexity that might not be different in a real D2DWRAN system. The algorithms should be implemented into the real experiments and examine the performance. Additionally, there are still some problems in Ant Colony Algorithm especially in solving the problem with unlimited scenarios. To have stable Ant Colony Algorithm, more research needs to be done. Last but not least, we have mentioned that the scenario under unlimited subchannels can be used in multiple channels, but the implementation of multiple channel resource allocation is a different problem, and more research work is certainly needed.

Finally at the end, we have to remember that the burst allocation problem studied in this thesis is only one of the research topics in D2DWRANs. There is still a long way to go before using D2DWRANs in daily life.

Bibliography

[1] M Calabrese and JH Snider. The nation in numbers up in the air. *ATLANTIC MONTHLY*, 292(2):46–49, 2003.

[2] Carlos Cordeiro, Kiran Challapali, Dagnachew Birru, and Nov Sai Shankar. Ieee 802.22: the first worldwide wireless standard based on cognitive radios. In *New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005. 2005 First IEEE International Symposium on*, pages 328–337. IEEE, 2005.

[3] Joseph Mitola and Gerald Q Maguire Jr. Cognitive radio: making software radios more personal. *Personal Communications, IEEE*, 6(4):13–18, 1999.

[4] Joseph Mitola. Cognitive radio—an integrated agent architecture for software defined radio. 2000.

[5] Paul Kolodzy and Interference Avoidance. Spectrum policy task force. *Federal Commun. Comm., Washington, DC, Rep. ET Docket*, (02-135), 2002.

[6] Dusit Niyato, Ekram Hossain, and Zhu Han. Dynamic spectrum access in ieee 802.22-based cognitive wireless networks: a game theoretic model for competitive spectrum bidding and pricing. *Wireless Communications, IEEE*, 16(2):16–23, 2009.

[7] Gwangzeen Ko, A Antony Franklin, Sung-Jin You, Jin-Suk Pak, Myung-Sun Song, and Chang-Joo Kim. Channel management in ieee 802.22 wran systems. *Communications Magazine, IEEE*, 48(9):88–94, 2010.

[8] C Stevenson, Gerald Chouinard, Zhongding Lei, Wendong Hu, S Shellhammer, and Winston Caldwell. Ieee 802.22: The first cognitive radio wireless regional area network standard. *Communications Magazine, IEEE*, 47(1):130–138, 2009.

[9] Apurva N Mody and Gerald Chouinard. Ieee 802.22 wireless regional area networks. *Enabling Rural Broadband Wireless Access Using Cognitive Radio Technology*, 2010.

[10] Huaizhou Shi, R Venkatesha Prasad, Vijay S Rao, and IGMM Niemegeers. Adapting ieee 802.22 ofdma system for p2pwrans. In *Global Communications Conference (GLOBECOM), 2013 IEEE*, pages 974–979. IEEE, 2013.

[11] Juan I Del-Castillo, Francisco M Delicado, Jesús Delicado, and Jose M Villalón. Ofdma resource allocation in ieee 802.16 networks: A performance comparative. In *Wireless and Mobile Networking Conference (WMNC), 2010 Third Joint IFIP*, pages 1–6. IEEE, 2010.

[12] Huaizhou Shi, R Venkatesha Prasad, and IGMM Niemegeers. Channel allocation in peer to peer ieee 802.22 networks. In *Proceedings of the 4th International Conference on Cognitive Radio and Advanced Spectrum Management*, page 22. ACM, 2011.

[13] Huaizhou Shi, R Venkatesha Prasad, and IGMM Niemegeers. An intra-cell peer to peer protocol in ieee 802.22 networks. In *GLOBECOM Workshops (GC Wkshps), 2011 IEEE*, pages 533–537. IEEE, 2011.

[14] Brett Kaufman and Behnaam Aazhang. Cellular networks with an overlaid device to device network. In *Signals, Systems and Computers, 2008 42nd Asilomar Conference on*, pages 1537–1541. IEEE, 2008.

[15] Chia-Hao Yu, Olav Tirkkonen, Klaus Doppler, and Cássio Ribeiro. On the performance of device-to-device underlay communication with simple power control. In *Vehicular Technology Conference, 2009. VTC Spring 2009. IEEE 69th*, pages 1–5. IEEE, 2009.

[16] Fernando A Kuipers and Piet FA Van Mieghem. Conditions that impact the complexity of qos routing. *IEEE/ACM Transactions on Networking (TON)*, 13(4):717–730, 2005.

[17] Cheong Yui Wong, Chi-Ying Tsui, Roger S Cheng, and Khaled Ben Letaief. A real-time sub-carrier allocation scheme for multiple access downlink ofdm transmission. In *Vehicular Technology Conference, 1999. VTC 1999-Fall. IEEE VTS 50th*, volume 2, pages 1124–1128. IEEE, 1999.

[18] Cheong Yui Wong, Roger S Cheng, K Ben Lataief, and Ross D Murch. Multiuser ofdm with adaptive subcarrier, bit, and power allocation. *Selected Areas in Communications, IEEE Journal on*, 17(10):1747–1758, 1999.

[19] Liang Xiaowen and Zhu Jinkang. An adaptive subcarrier allocation algorithm for multiuser ofdm system. In *Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th*, volume 3, pages 1502–1506. IEEE, 2003.

[20] Hujun Yin and Hui Liu. An efficient multiuser loading algorithm for ofdm-based broadband wireless systems. In *Global Telecommunications Conference, 2000. GLOBECOM'00. IEEE*, volume 1, pages 103–107. IEEE, 2000.

[21] Guocong Song and Ye Li. Utility-based joint physical-mac layer optimization in ofdm. In *Global Telecommunications Conference, 2002. GLOBECOM'02. IEEE*, volume 1, pages 671–675. IEEE, 2002.

[22] Guocong Song and Ye Li. Cross-layer optimization for ofdm wireless networks-part i: theoretical framework. *Wireless Communications, IEEE Transactions on*, 4(2):614–624, 2005.

[23] Guocong Song and Ye Li. Cross-layer optimization for ofdm wireless networks-part ii: algorithm development. *Wireless Communications, IEEE Transactions on*, 4(2):625–634, 2005.

[24] Wonjong Rhee and John M Cioffi. Increase in capacity of multiuser ofdm system using dynamic subchannel allocation. In *Vehicular Technology Conference Proceedings, 2000. VTC 2000-Spring Tokyo. 2000 IEEE 51st*, volume 2, pages 1085–1089. IEEE, 2000.

[25] Zukang Shen, Jeffrey G Andrews, and Brian L Evans. Adaptive resource allocation in multiuser ofdm systems with proportional rate constraints. *Wireless Communications, IEEE Transactions on*, 4(6):2726–2737, 2005.

[26] Peter S Chow and John M Cioffi. Bandwidth optimization for high speed data transmission over channels with severe intersymbol interference. In *Global Telecommunications Conference, 1992. Conference Record., GLOBECOM'92. Communication for Global Users., IEEE*, pages 59–63. IEEE, 1992.

[27] Elias Yaacoub and Zaher Dawy. A survey on uplink resource allocation in ofdma wireless networks. *Communications Surveys & Tutorials, IEEE*, 14(2):322–337, 2012.

[28] Keunyoung Kim, Youngnam Han, and Seong-Lyun Kim. Joint subcarrier and power allocation in uplink ofdma systems. *Communications Letters, IEEE*, 9(6):526–528, 2005.

[29] Long Gao and Shuguang Cui. Efficient subcarrier, power, and rate allocation with fairness consideration for ofdma uplink. *Wireless Communications, IEEE Transactions on*, 7(5):1507–1511, 2008.

[30] Cho Yiu Ng and Chi Wan Sung. Low complexity subcarrier and power allocation for utility maximization in uplink ofdma systems. *Wireless Communications, IEEE Transactions on*, 7(5):1667–1675, 2008.

[31] David N. C. Tse and Stephen V Hanly. Multiaccess fading channels. i. polymatroid structure, optimal resource allocation and throughput capacities. *Information Theory, IEEE Transactions on*, 44(7):2796–2815, 1998.

[32] Lifang Li and Andrea J Goldsmith. Capacity and optimal resource allocation for fading broadcast channels. i. ergodic capacity. *Information Theory, IEEE Transactions on*, 47(3):1083–1102, 2001.

[33] Andrea J Goldsmith and Michelle Effros. The capacity region of broadcast channels with intersymbol interference and colored gaussian noise. *Information Theory, IEEE Transactions on*, 47(1):219–240, 2001.

[34] Wei Yu and John M Cioffi. Fdma capacity of gaussian multiple-access channels with isi. *Communications, IEEE Transactions on*, 50(1):102–111, 2002.

[35] Louise MC Hoo, Bijit Halder, José Tellado, and John M Cioffi. Multiuser transmit optimization for multicarrier broadcast channels: asymptotic fdma capacity region and algorithms. *Communications, IEEE Transactions on*, 52(6):922–930, 2004.

[36] Juan-Andrés Bazerque and Georgios B Giannakis. Distributed scheduling and resource allocation for cognitive ofdma radios. *Mobile Networks and Applications*, 13(5):452–462, 2008.

[37] CPRI Specification. V3. 0 common public radio interface (cpri); interface specification, oct. 20, 2006, 89 pages, ericsson ab, huawei technologies col ltd, nec corporation, nortel networks sa and siemens networks gmbh & co. *Ltd, NEC Corporation, Nortel Networks SA and Siemens Networks GmbH & Co. KG*, 2006.

[38] Wonil Roh and Arogyaswami Paulraj. Outage performance of the distributed antenna systems in a composite fading channel. In *Vehicular Technology Conference, 2002. Proceedings. VTC 2002-Fall. 2002 IEEE 56th*, volume 3, pages 1520–1524. IEEE, 2002.

[39] Wan Choi, Jeffrey G Andrews, and Chaehag Yi. Capacity of multicellular distributed antenna networks. In *Wireless Networks, Communications and Mobile Computing, 2005 International Conference on*, volume 2, pages 1337–1342. IEEE, 2005.

[40] Wan Choi and Jeffrey G Andrews. Downlink performance and capacity of distributed antenna systems in a multicell environment. *Wireless Communications, IEEE Transactions on*, 6(1):69–73, 2007.

[41] Elias Yaacoub and Zaher Dawy. Achieving the nash bargaining solution in ofdma uplink using distributed scheduling with limited feedback. *AEU-International Journal of Electronics and Communications*, 65(4):320–330, 2011.

[42] Elias Yaacoub and Zaher Dawy. Distributed probabilistic scheduling in ofdma uplink using subcarrier sensing. In *Wireless Communications and Networking Conference, 2009. WCNC 2009. IEEE*, pages 1–5. IEEE, 2009.

[43] Lawrence G Roberts. Dynamic allocation of satellite capacity through packet reservation. In *Proceedings of the June 4-8, 1973, national computer conference and exposition*, pages 711–716. ACM, 1973.

[44] Leonard Kleinrock and Simon Lam. Packet switching in a multiaccess broadcast channel: Performance evaluation. *Communications, IEEE Transactions on*, 23(4):410–423, 1975.

[45] Dongxu Shen and Victor OK Li. Stabilized multi-channel aloha for wireless ofdm networks. In *Global Telecommunications Conference, 2002. GLOBECOM'02. IEEE*, volume 1, pages 701–705. IEEE, 2002.

[46] Young-June Choi, Suho Park, and Saewoong Bahk. Multichannel random access in ofdma wireless networks. *Selected Areas in Communications, IEEE Journal on*, 24(3):603–613, 2006.

[47] Yunghsiang S Han, Jing Deng, and Zygmunt J Haas. Analyzing multi-channel medium access control schemes with aloha reservation. *Wireless Communications, IEEE Transactions on*, 5(8):2143–2152, 2006.

[48] Ghurumuruhan Ganesan and Ye Li. A simple reservation scheme for multicarrier channel aware aloha. In *Global Telecommunications Conference, 2007. GLOBECOM'07. IEEE*, pages 1451–1455. IEEE, 2007.

[49] Dandan Wang, Hlaing Minn, and Naofal Al-Dhahir. A distributed opportunistic access scheme and its application to ofdma systems. *Communications, IEEE Transactions on*, 57(3):738–746, 2009.

[50] Zhang Yumei and Sheng Yu. Analysis of channel-aware multichannel aloha in ofdma wireless networks. , 3(2), 2009.

[51] Hiromasa Fujii and Hitoshi Yoshino. Theoretical capacity and outage rate of ofdma cellular system with fractional frequency reuse. In *Vehicular Technology Conference, 2008. VTC Spring 2008. IEEE*, pages 1676–1680. IEEE, 2008.

[52] Jia-Ming Liang, Jen-Jee Chen, You-Chiun Wang, and Yu-Chee Tseng. A cross-layer framework for overhead reduction, traffic scheduling, and burst allocation in ieee 802.16 ofdma networks. *Vehicular Technology, IEEE Transactions on*, 60(4):1740–1755, 2011.

[53] Paul E Black. Greedy algorithm. *Dictionary of Algorithms and Data Structures, US National Institute of Standards and Technology*, 2005.

[54] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. Introduction to algorithms, 2009. *Possíveis Questionamentos*.

[55] Miroslaw Malek, Mohan Guruswamy, Mihir Pandya, and Howard Owens. Serial and parallel simulated annealing and tabu search algorithms for the traveling salesman problem. *Annals of Operations Research*, 21(1):59–84, 1989.

[56] Fred Glover, James P Kelly, and Manuel Laguna. Genetic algorithms and tabu search: hybrids for optimization. *Computers & Operations Research*, 22(1):111–134, 1995.

[57] HW Leong, DF Wong, and CL Liu. A simulated annealing channel router. In *Proc. Int. Conf. on Compututer-Aided Design*, pages 226–228, 1985.

[58] Emile Aarts and Jan Korst. Simulated annealing and boltzmann machines. 1988.

[59] Emile HL Aarts and Jan HM Korst. Boltzmann machines for travelling salesman problems. *European Journal of Operational Research*, 39(1):79–95, 1989.

[60] Yutaka Akiyama, A Yamashita, M Kajiura, and H Aiso. Combinatorial optimization with gaussian machines. In *Neural Networks, 1989. IJCNN., International Joint Conference on*, pages 533–540. IEEE, 1989.

[61] Harold Szu and Ralph Hartley. Fast simulated annealing. *Physics letters A*, 122(3):157–162, 1987.

[62] Harold H Szu and Ralph L Hartley. Nonconvex optimization by fast simulated annealing. *Proceedings of the IEEE*, 75(11):1538–1540, 1987.

[63] John J Hopfield and David W Tank. "neural" computation of decisions in optimization problems. *Biological cybernetics*, 52(3):141–152, 1985.

[64] Darrell Whitley. A genetic algorithm tutorial. *Statistics and computing*, 4(2):65–85, 1994.

[65] Feng-Tse Lin, Cheng-Yan Kao, and Ching-Chi Hsu. Applying the genetic approach to simulated annealing in solving some np-hard problems. *Systems, Man and Cybernetics, IEEE Transactions on*, 23(6):1752–1767, 1993.

[66] Marco Dorigo, Eric Bonabeau, and Guy Theraulaz. Ant algorithms and stigmergy. *Future Generation Computer Systems*, 16(8):851–871, 2000.

[67] Nicolas Monmarché, Gilles Venturini, and Mohamed Slimane. On how¡ i¿ pachycondyla apicalis¡/i¿ ants suggest a new search algorithm. *Future Generation Computer Systems*, 16(8):937–946, 2000.

[68] Thomas Stützle and Holger H Hoos. Max–min ant system. *Future generation computer systems*, 16(8):889–914, 2000.

[69] Andreas Brandstädt, Jeremy P Spinrad, et al. *Graph classes: a survey*, volume 3. Siam, 1999.