STUDY AND PERFORMANCE OF FAIR SCHEDULING ALGORITHMS IN OFDMA

A DISSERTATION

Submitted in partial fulfillment of the requirements for the award of the degree

of

MASTER OF TECHNOLOGY

in

ELECTRONICS AND COMMUNICATION ENGINEERING
(With Specialization in Communication Systems)

By

NITESH KUMAR SRIVASTAVA

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE - 247 667 (INDIA)

JUNE, 2007

CANDIDATE'S DECLARATION

I hereby declare that the work, which is presented in this dissertation report, entitled **"STUDY AND PERFORMANCE OF FAIR SCHEDULING ALGORITHMS IN OFDMA"**, being submitted in partial fulfillment of the requirements for the award of the degree of **Master of Technology** in **Electronics and Communication Engineering** with specialization in **Communication Systems**, in the Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee, Roorkee is an authentic record of my own work carried out from July 2006 to June 2007, under the supervision and guidance of **Mr. S. Chakravorty**, Assistant Professor, Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee, Roorkee.

I have not submitted the matter embodied in this dissertation for the award of any other degree.

Date: 29|06|07

Place: Roorkee

NKSrivastava

(Nitesh Kumar Srivastava)

CERTIFICATE

This is to certify that the statement made by the candidate is correct to the best of my knowledge and belief.

S. Chakravorty

Date: 29|06|07

Place: Roorkee

(Mr. S. Chakravorty)

Assistant Professor,
E & CE Department,
Indian Institute of Technology Roorkee,
Roorkee-247667

ACKNOWLEDGEMENT

I am highly indebted to Mr. S. Chakravorty, Assistant Proffessor, Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee, Roorkee, for encouraging me to undertake this dissertation as well as providing me all the necessary guidance and inspirational support throughout this dissertation work. He has displayed unique tolerance and understanding at every step of progress, without which this dissertation would not have been in the present shape. I deem it my privilege to have carried out the dissertation work under his able guidance.

I would like to thank all of my friends, for their help and encouragement at the hour of need.

Finally, I express my regards to my parents and my brothers, who have been a constant source of inspiration to me.

Date: 29/06/07 **Nitesh Kumar Srivastava**

Place: Roorkee

ABSTRACT

In this dissertation work, different types of fair resource scheduling algorithms for OFDMA system have been discussed and their performances are compared with the algorithms for which fairness is no criterion. Time varying nature of the wireless channel creates a problem in efficient allocation of the subcarriers to the users. Hence, adaptive modulation technique is used in allocation of bits to the subcarriers. In this dissertation, the performance of minimum normalized transmit rate based fair scheduling algorithm has been evaluated which uses the adaptive modulation technique. Using the instantaneous channel gains of each subcarrier corresponding to each user, maximum number of bits that each user can allocate to each subcarrier, is calculated. Then utilization efficiency of each subcarrier corresponding to each user is calculated. Utilization efficiency of a subaccier-user pair is the ratio of the bits that the user can allocate to the subcarrier, to the total number of bits that each user can allocate to the subcarrier. To provide fairness among users, the user with the minimum normalized transmit rate is selected and to achieve high throughput the subcarrier corresponding to this user with the highest utilization efficiency is selected. Simulation results show that a high throughput is achieved while providing fairness.

CONTENTS

Chapter 1

INTRODUCTION

The demand on high rate multimedia information has pushed the development of wireless communication systems in an unprecedented pace. System design of fourth-generation (4G) technology has already begun, which seeks to support a wide range of packet services with highest data rates reaching 20Mbps. With limited bandwidth resources, the use of efficient scheduling algorithm is essential to provide satisfactory services to the users. Recently, intense interest has focused on modulation techniques which can provide broadband transmission over wireless channels for applications including wireless multimedia, wireless internet access, and future generation mobile communication systems. One of the main requirements on the modulation technique is the ability to combat intersymbol interference (ISI), a major problem in wideband transmission over multipath fading channels. Many methods have been proposed to combat ISI. Multicarrier modulation techniques, including orthogonal frequency division multiplexing (OFDM), [1], [2] are among the more promising solutions to this problem. In orthogonal frequency division multiple access (OFDMA), a block schematic of which is shown in Fig. 1.1, different users have to access the same channel and hence there is a need for a scheduling algorithm which will allocate the subcarriers to different users.

1.1 Scheduling [3]

The process of selecting a user out of many users in order to access the common resources of a channel is called scheduling. The function of a scheduling algorithm is to select the user whose data is to be transmitted next. This selection process is based on the QoS requirements of each user.

It is desirable for a scheduling algorithm to possess the following features:

- **Efficient link utilization:** The algorithm must utilize the channel efficiently. This implies that the scheduler should not assign a transmission slot to a user with a currently bad link since the transmission will simply be wasted.

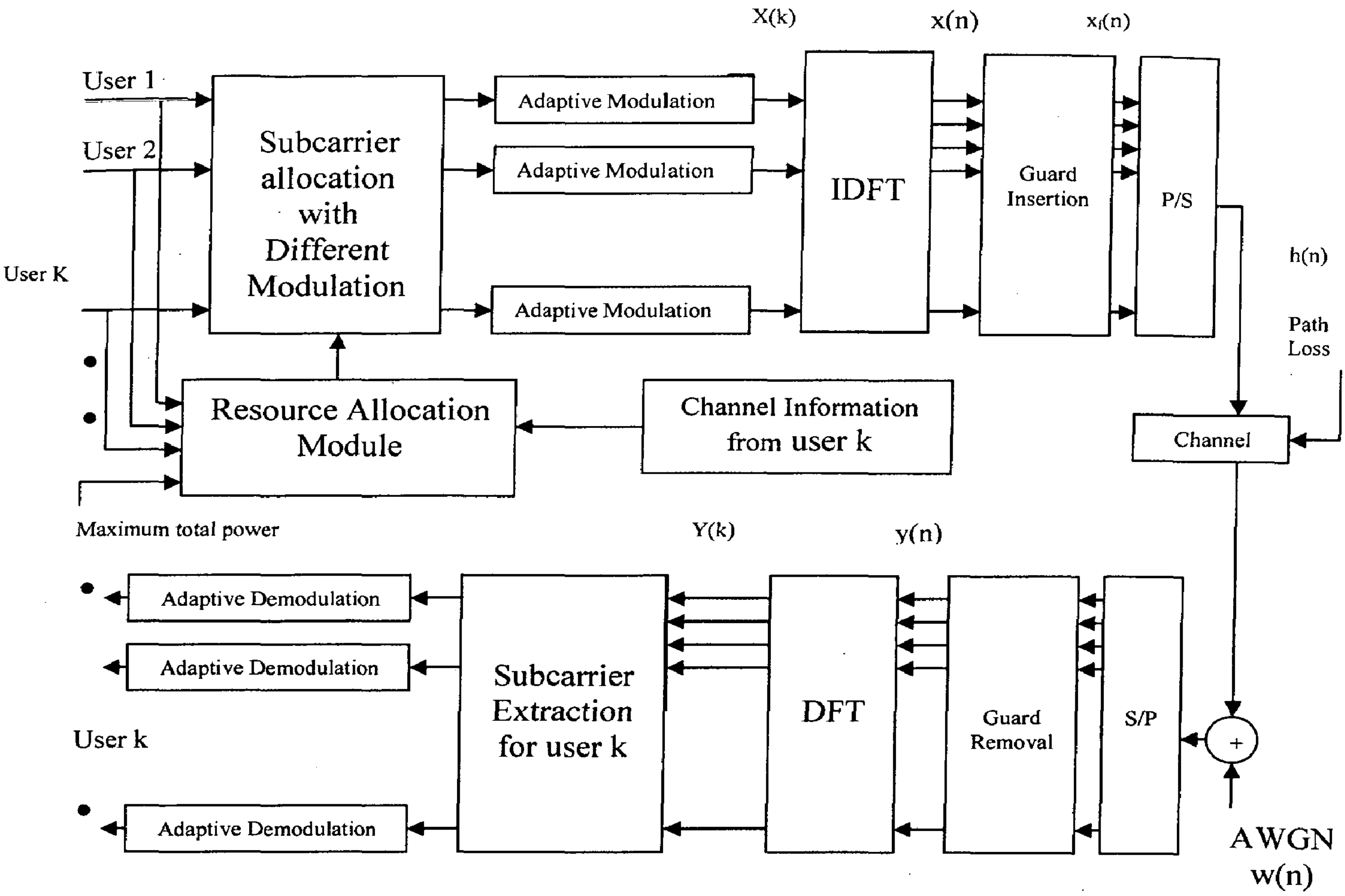


Fig. 1.1 Basic Block Diagram of OFDMA

- **Delay bound:** The algorithm must be able to provide delay bound guarantees for individual users in order to support delay-sensitive applications.
- **Fairness:** The algorithm should redistribute available resources fairly across users. It should provide fairness among error-free users (short-term fairness) and error-prone users (long-term fairness).
- **Throughput:** The algorithm should provide guaranteed short-term throughputs for error free users and guaranteed long-term throughputs for all users.
- **Implementation complexity:** A low-complexity algorithm is a necessity in high-speed networks in which scheduling decisions have to be made very rapidly.
- **Graceful service degradation:** A user that has received excess service at the expense of users whose links were bad should experience smooth service degradation when relinquishing the excess service to lagging users whose links are now good.

- **Isolation:** The algorithm should isolate a user from the ill effects of misbehaving users. The QoS guarantees for a user should be maintained even in the presence of users whose demands are in excess of their reserved values.
- **Energy consumption:** The algorithm should take into account the need to prolong the MS battery life.
- **Delay/bandwidth decoupling:** For most schedulers, the delay is tightly coupled with the reserved rate; that is, a higher reserved rate provides a lower delay. However, some high-bandwidth applications, such as Web browsing, can tolerate relatively large delays.
- **Scalability:** The algorithm should operate efficiently as the number of users sharing the channel increases.

1.2 Classification of Schedulers [3]

Schedulers can be classified as work conserving and non-work conserving.

1.2.1 Work Conserving Scheduler

In this type of scheduler, the scheduler never remains idle if there is data available on any user for transmission. e.g. Generalized Processor Sharing (GPS), Packet-by-Packet GPS also known as Weighted Fair Queuing (WFQ), Virtual Clock (VC), Weighted Round Robin (WRR), Self-clocked Fair Queuing (SCFQ), and Deficit Round Robin (DRR).

1.2.2 Non-Work Conserving Scheduler

In this type of scheduler, the scheduler may be idle even if there is data available on users to transmit, because it may be expecting a data to arrive which may have priority higher than the available data. E.g. Hierarchical Round Robin (HRR), Stop-and-Go Queuing (SGQ), and Jitter-Earliest-Due-Date (Jitter-EDD). These schedulers generally have higher average delays than work conserving schedulers but may be used in applications where time jitter is more important than delay.

1.3 Need of Fair Scheduling [4], [5], [6]

In multi-user OFDM system (OFDMA), the channel conditions of mobile users vary in the time domain and different subcarriers experience different channel gains because of frequency selective fading channels. Wireless link capacity is generally a scarce resource that needs to be used efficiently. Therefore, it is important to find an efficient way of supporting Quality of Service (QoS) for each user. Specifically, it is important to maximize the system throughput while providing fairness among users i.e. the resources must be fairly allocated among the users. The criterion for fairness may be latency, throughput or QoS. To maximize the throughput of the system best subcarrier is allocated to each user. Of course, the procedure is not simple since the best subcarrier of the user may also be the best subcarrier of another user who may not have any other good subcarriers. The overall strategy is to use the peaks of the channel resulting from channel fading. Unlike in the traditional view where channel fading is considered to be impairment, here it acts as a channel randomizer and increases multi-user diversity.

The multi-channel nature of OFDM and the variance of wireless channels make it difficult for good resource scheduling algorithms to maximize the total system throughput and achieve fairness among all users as well. A large number of resource scheduling algorithms have been proposed for OFDMA systems. The goal of some of these algorithms is to achieve either the maximization of system capacity or the minimization of total transmit power, while satisfying the various rate requirements of different users. But, few of them aim to accomplish fair resource sharing among users. As already said, that at all the time the user with the best channel is selected to maximize the throughput. But in this case, if a user is at a great distance from the base station or its channel is in deep fade, then it will never get a chance to access the channel. Hence there is need to provide fairness among users. Some of the fair scheduling algorithms are also proposed which provide fairness among users. In this dissertation, we will discuss some of these algorithms.

1.4 Problem Statements

The 4th generation mobile systems will be required to provide high data rate multimedia services. OFDMA is a multicarrier modulation and access technique which can support high data rates and perform well in wireless environment. However, this can be ensured only through good scheduling algorithm to allocate the common resources. The problem is to allocate resources among users fairly. The study of solution to this problem has been undertaken in this dissertation. Specifically, the following studies have been done:

- Overview of different fair scheduling algorithms for OFDMA.
- Study and simulate the performance of minimum normalized transmit rate based fair resource scheduling algorithm for OFDMA.
- Comparison of the performance of this algorithm with those maximum resource scheduler (MRS) and random resource scheduler (RRS).

1.5 Organization of the Dissertation

Including this introductory chapter, the report is organized in five chapters. In chapter 2, different fair scheduling algorithms for OFDMA are explained. These algorithms provide fairness with respect to QoS, latency and throughput. Brief comparison of these algorithms with non-fair algorithms is also included in this chapter. In chapter 3, the minimum normalized transmit rate based scheduling algorithm has been discussed in detail. This algorithm provides fairness with respect to throughput. The algorithm has been simulated and the results are presented and discussed in chapter 4. Finally, chapter 5 concludes the dissertation. The code is included as Appendix.

Chapter 2

OVERVIEW OF DIFFERENT FAIR SCHEDULING ALGORITHMS

Many fair scheduling algorithms have been proposed in literature. In this chapter, different types of fair resource scheduling algorithms will be discussed and their performances will be compared with each other along with some basic algorithms. These algorithms may be categorized as

1. Fair Scheduling Algorithms based on QoS
2. Fair Scheduling Algorithms based on Latency
3. Fair Scheduling Algorithms based on Throughput

2.1 Fair Scheduling Algorithms based on QoS

QoS fairness is defined as achieving a specified data transmission rate and bit error rate (BER) of each user in each transmission. These algorithms fulfill the QoS requirements of each user.

2.1.1 Iterative Fair Scheduling Algorithm [5]

The GreedyLP and GreedyHungarian methods both first determine the subcarriers and then increment the number of bits on them according to the rate requirements of users. This may not be a good schedule in some certain cases: For instance, consider a user with only one good subcarrier and low rate requirement. The best solution for that user is allocating its good carrier with high number of bits. But if GreedyLP or Greedy-Hungarian is used, user may have allocated more than one subcarrier with lower number of bits and in some cases, its good subcarrier is never selected. Consider another scenario where a user does not have any good subcarrier (i.e. it may have a bad channel or be at the edge of the cell). In this case, rather than pushing more bits and allocating less subcarriers as in GreedyLP and GreedyHungarian, the opposite strategy is preferred since fewer bits in higher number of subcarriers give better result. Another difficulty arises in providing fairness. Since GreedyLP and GreedyHungarian are based on greedy approach, the user in the worst condition usually suffers. In any event, these are complex schemes and simpler schemes are needed to finish the allocation within the coherence time. To cope with these challenges, a simple, efficient and fair subcarrier allocation scheme with iterative improvement is given here. This scheme is composed of two modules named scheduling and improvement modules. In the scheduling section, bits and subcarriers are distributed to the users and passed to the improvement module where the allocation is improved iteratively by bit swapping and subcarrier swapping algorithms.

2.1.1.1 Fair Scheduling Algorithm

A simple and mixed allocation scheme that considers fair allocation among users with adaptive modulation is explained here. The allocation procedure starts with the highest level of modulation scheme. In this way, it tries to find the best subcarrier of a user to allocate the highest number of bits. The strategy can be described by an analogy: "The best strategy to fill a case with stone, pebble and sand is as follows. First filling the case with the stones and then filling the gap left from the stones with pebbles and in the same way, filling the gap left from pebbles with sand. Since filling in opposite direction may leave the stones or pebbles outside". With this strategy more bits can be allocated and the scheme becomes immune to uneven QoS requirements. The fair scheduling algorithm (FSA) runs greedy release algorithms (GRA) if there are unallocated subcarriers after the lowest modulation turn and the rate requirement is not satisfied. GRA decrements one bit of a subcarrier to gain power reduction, which is used to assign higher number of bits to the users on the whole. Before describing the algorithm let us introduce with the terms used in the algorithm and the problem of allocation .

Let $\gamma_{k,n}$ is the indicator of allocating the nth subcarrier to the k^{th} user. If $\alpha_{k,n}$ is the channel gain of subcarrier *n* corresponding to user *k* then the transmission power allocated to the nth subcarrier of the k^{th} user is expressed as

$$P_{k,n} = \frac{f_k(c_{k,n}, BER_k)}{\alpha_{k,n}^2}$$

Where $f_k(c_k,n)$ is the received power with unity channel gain for reliable reception of ck,n bits per symbol. We can formulate the resource allocation problem with an imposed power constraint as

$$\max_{c_{k,n},\gamma_{k,n}} R_k = \sum_{n=1}^{N} c_{k,n}\gamma_{k,n}$$

Subject to

$$P_T = \sum_{k=1}^{K}\sum_{n=1}^{n} \frac{f_k(c_{k,n}, BER_k)}{\alpha_{k,n}^2} \gamma_{k,n} \le P_{MAX}$$

Where the limit on the total transmission power is expressed as P_{MAX} for all $n \in \{1,.....,N\}$, $k \in \{1,.....,K\}$ and $c_{k,n} \in \{1,.....,M\}$. For multiple modulation techniques, the dimension of the indicator is incremented and represented by $\gamma_{k,n,c}$ and defined as $\gamma_{k,n,c} = 1$, if $c_{k,n}=c$ otherwise 0. There K*N*M indicator variables and M power matrices where the entries of each matrix for a given c can be found from

$$P_{k,n}^{c} = \frac{f_k(c, BER_k)}{\alpha_{k,n}^2}$$

The cost function now can be written as

$$P_T = \sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{C=1}^{M} P_{k,n}^{c}\gamma_{k,n,c}$$

And the description of the Integer Programming (IP) problem is

$$\min_{k,n,c} P_T, \text{ for } \gamma_{k,n,c} \in \{0,1\}$$

Subject to

$$R_k = \sum_{n=1}^{N}\sum_{c=1}^{M} c.\gamma_{k,n,c} \quad \text{for all } k,$$

And

$$0 \le \sum_{k=1}^{K}\sum_{c=1}^{M} \gamma_{k,n,c} \le 1, \text{ for all } n.$$

FSA is described as follows;

FS Algorithm

Step 1) Set $c = M$, Select a k, and $P_T=0$;

Step 2) Find $\bar{n} = \arg\min_n P_{k,n}^{c}$;

Step 3) Set $R_k = R_k - c$ and $\rho_{k,\bar{n}} = 1$, Update P_T, Shift to the next k;

Step 4) If $P_T > P_{Max}$, Step Out and Set $c = c - 1$, GOTO STEP 2.

Step 5) If $\forall k$, $R_k < c$, Set $c = c - 1$, GOTO STEP 2.

Step 6) If $\{c == 1\}$, $\sum_{k=1}^{K}\sum_{n=1}^{N}\rho_{k,n} < N$, $P_T > P_{Max}$, Run "Greedy Release" and GOTO STEP 2.

Step 7) Finish.

2.1.1.2 Greedy Releasing Algorithm

The GRA tends to fill the unallocated subcarriers. It releases one of the bits of the most expensive subcarrier to gain power reduction in order to fulfill the QoS requirement. GRA works in the opposite direction of bit loading algorithm (BLA) [5]. GRA is described as follows;

GR Algorithm

Step 1) Find $\{\bar{k}, \bar{n}, c_{\bar{k},\bar{n}}\}$ = arg $\max_{k,n,c} P^{c}_{k,n}\,\rho_{k,n}$ $\forall c$;

Step 2) Set $c_{\bar{k},\bar{n}} = c_{\bar{k},\bar{n}}$ -1, $P_T = P_T - \Delta P_{\bar{k},\bar{n}}(c_{\bar{k},\bar{n}})$;

;

Step 3) Set c = $c_{\bar{k},\bar{n}}$ - 1;

Step 4) Finish.

2.1.1.3 Horizontal Swapping Algorithm

The horizontal swapping algorithm (HSA) aims to smooth the bit distribution of a user. When the subcarriers are distributed, the bit weight per subcarrier can be adjusted to reduce power. One bit of a subcarrier may be shifted to another subcarrier of the same user if there is a power reduction gain. Therefore, variation of the power allocation per subcarrier is reduced and a smoother transmission is performed. HSA is described as follows;

HS Algorithm

Step 1) Set $P_C = \infty$

STEP1a: Find $\{\bar{k}, \bar{n}, c_{\bar{k},\bar{n}}\}$ = arg $\max_{k,n,c} P^{c}_{k,n}\,\rho_{k,n} < P_C$ $\forall c$;

Step 2: Define $n \in S_k$, where $\{\rho_{k,n} == 1\}$ for $\forall n$;

Step 3: Set $\Delta_{\dot{n}} = \max_n \Delta P_{\bar{k}\bar{n}}\left(c_{\bar{k},\bar{n}} - 1\right) - \Delta P_{\bar{k},\dot{n}}\left(c_{\bar{k},\dot{n}}\right)$, $n \in S_k$;

Step 4: Set $P_C = P^{\bar{c}}_{\bar{k},\bar{n}}$;

STEP4a: if $\Delta \dot{n} > 0$, Set $P_T = P_T - \Delta \dot{n}$

STEP4b: Set $c_{\bar{k},\bar{n}} = c_{\bar{k},\bar{n}} - 1$, $c_{\bar{k},\dot{n}} = c_{\bar{k},\dot{n}} + 1$

GOTO Step 1a;

Step 5: if $\{P_C == \min_{k,n,c}(P^{c}_{k,n}\rho_{k,n})\}$,

Finish.

2.1.1.4 Vertical Swapping Algorithm

Vertical swapping is done for every pair of users. In each iteration, users try to swap their subcarriers such that the power allocation is reduced. There is more than one class where each class is defined with its modulation (i.e. number of bits loaded to a subcarrier) and swapping is only within the class. Each pair of user swaps their subcarriers that belong to the same class if there is a power reduction. In this way, adjustment of subcarrier is done across users, to try to approximate the optimal solution. VSA is described as follows;

VS Algorithm

Step 1) $\forall$ pair of user *{i, j}*;

STEP1a: Find $\partial P_{i,j}(n) = P^{\dot{c}}_{i,n} - P^{\dot{c}}_{j,n}$ and $\Delta^{\hat{n}} = \max \partial P_{i,j}(n)$, $\forall n \in S_i$;

STEP1b: Find $\partial P_{j,i}(n) = P^{\dot{c}}_{j,n} - P^{\dot{c}}_{i,n}$ and $\Delta^{\check{n}} = \max \partial P_{j,i}(n)$, $\forall n \in S_j$;

STEP1c: Set $\Omega^{\hat{n},\check{n}} P_{i,j} = \Delta^{\hat{n}} P_{i,j} + \Delta^{\check{n}} P_{j,i}$;

STEP1d: Add $\Omega^{\hat{n},\check{n}} P_{i,j}$ to the $\{\Lambda\}$

list;

Step 2: Select $\Omega = \max_{\left(i,\hat{n}\right),\left(j,\check{n}\right)} \Lambda$;

Step 3: if $\Omega > 0$, Switch subcarriers and $P_T = P_T - \Omega$ GOTO STEP 1a;

Step 4: if $\Omega \leq 0$, Finish.

2.1.1.5 Performance of iterative algorithm

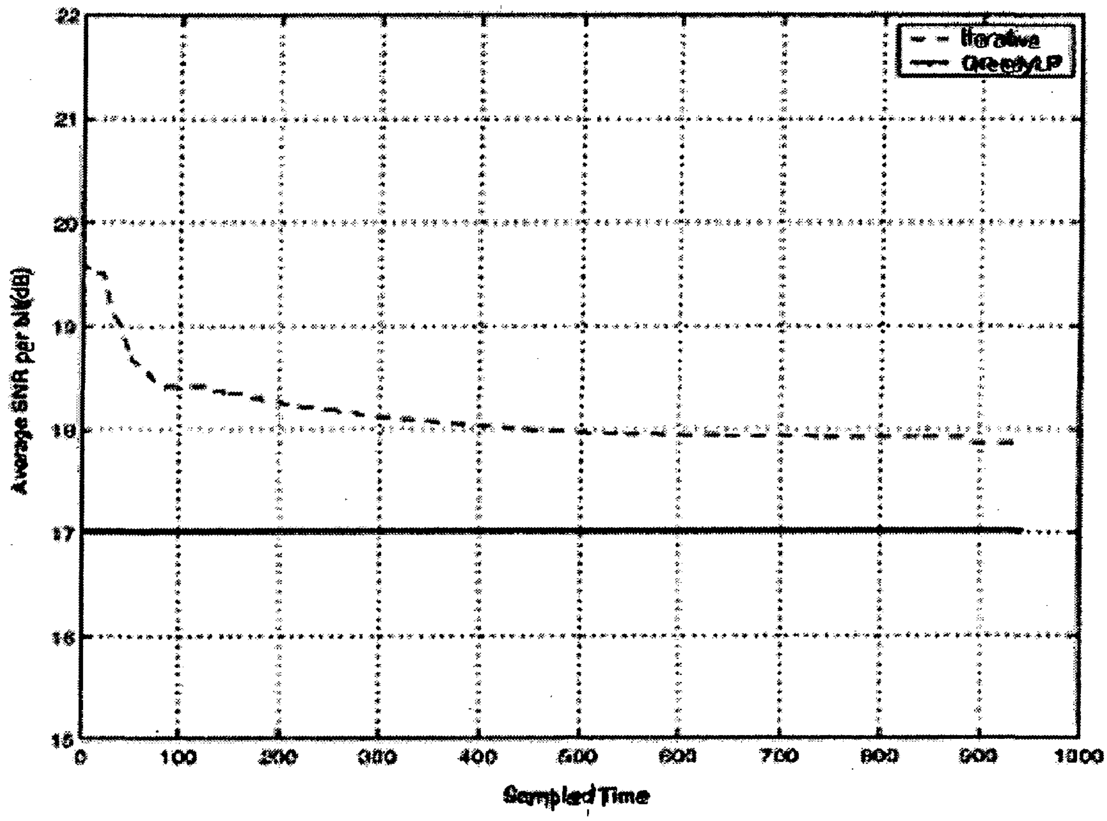
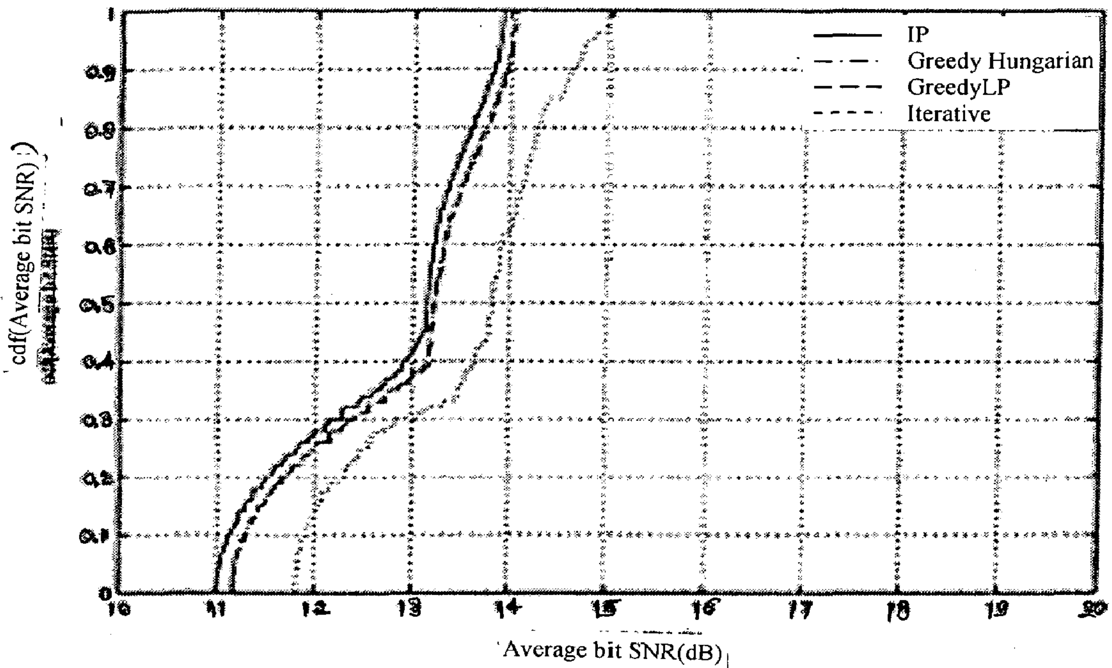
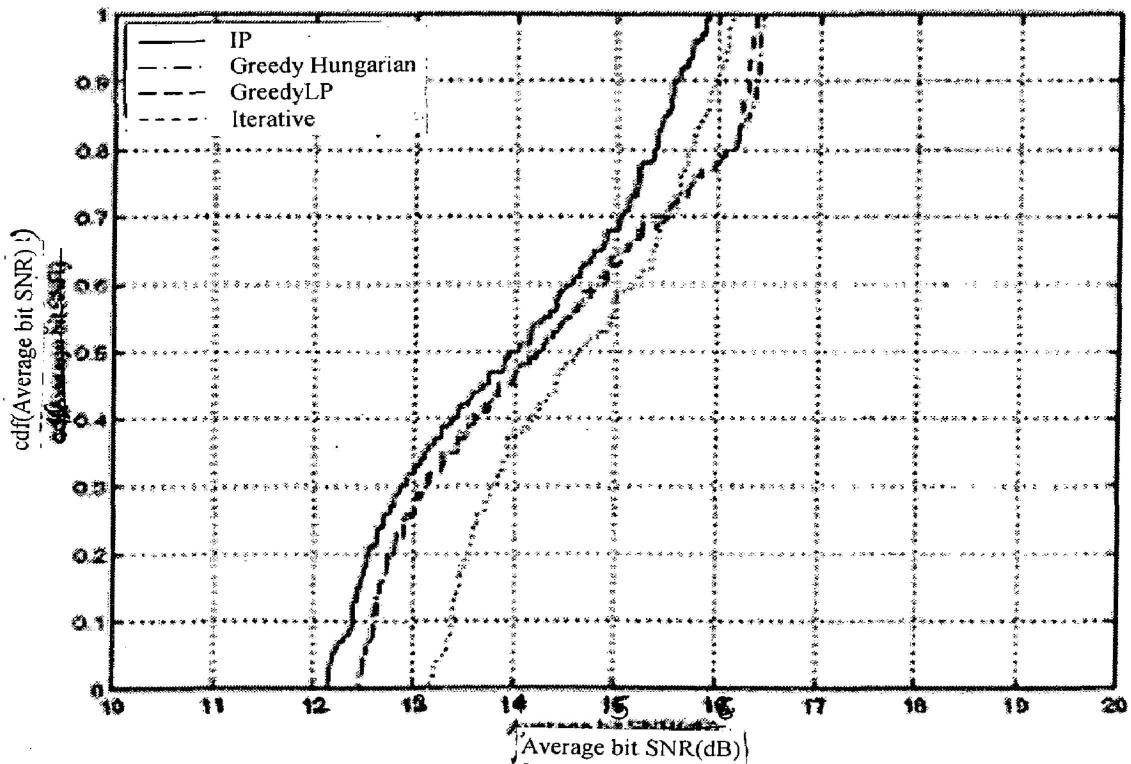


Fig. 2.1 Comparison of convergence of the iterative approach to the GreedyLP one

Fig. 2.1 shows the convergence of the algorithm with iterative betterment. In each iterative step, the power is reduced keeping the total number of bits constant. The steepest decrease is observed in the HSA step since the power reduction in bit swapping is higher than the one in subcarrier swapping because of the exponential growth of the *f(x,y)* function. It can be seen from the figure that iterative solution approximates the GreedyLP with time.



(a)



(b)

Fig. 2.2 Comparison of the cumulative distribution function of the average bit SNR; (a) without power constraint; (b) with power constraint

Fig. 2.2 presents the cumulative distribution functions of the average bit SNR for the cases with and without power constraint. There are four users in two sets of BER requirement and each user has rate requirement of 120 bits/symbol. It can be seen from the Fig. 2.2(a) that iterative approach approximates the optimal solution up to 0.9*dB* when there is no power constraint. When there is a power constraint, as seen in Fig. 2.2(b), the iterative approach outperforms the GreedyHungarian and GreedyLP approach and is close to the IP solution within 0.3*dB*. The reason why iterative solution gives better performance than the suboptimal solution is its tight power control scheme, which allows transmitting more number of bits. GRA is one of the most important modules that decreases the variance of average bit SNR and make the iterative approach perform better at the end by exchanging one high cost bit with more than one low cost bits i.e. lower level modulation.

2.1.2 Heuristic Genetic Algorithm [6]

In this algorithm, the objective is to schedule resources so as to maximize the overall transmitting data rates while satisfying total power constraints and providing fairness among users. This is achieved by allocating the subcarriers to the users and by determining the rates and the power level transmitted on each sub-carrier based on the channel conditions and fairness. This algorithm is based on genetic algorithm [7] and its steps are as follows:

2.1.2.1 Coding

Just like in genetic algorithm there is also a chromosome here in this algorithm. The chromosome is divided into N blocks, the i^{th} block in the chromosome records the information of the channel i. Each block consists of two fields, the first field represents the user index who is assigned to the channel, and the next field is the transmission rate on the channel received by the user, see Fig. 2.3. In the model considered here, the two numbers are integers so the chromosome can be coded as integer array. It is easy to see that each chromosome is an integer array of length 2N, where N is the number of subcarriers. In terms of the user number K and the maximum value C of the numbers of C^n, we can reduce the search space to 2N*max (K, C) where C^n is the maximum rate that can be allocated to subcarrier n.

channel 1		channel 2			channel N	
U_id	rate	U_id	rate		U_id	rate

Fig. 2.3 Coding Scheme

2.1.2.2 Initial Population Creation

For creating initial population a refining process is executed, so that the initial population includes as more feasible and even "good" individuals as possible. The refining process is simple and only need a linear searching time: for the subcarrier, if the magnitude of the channel gain of the subcarrier as seen by the assigned user is below a threshold, we set the assigned rate to be very small or even to 0; otherwise we randomly assigned the transmission rate.

2.1.2.3 Fitness Function

The fitness should response the individual performance: the "good individual" (its utility function is fairly large) has bigger fitness than the "bad one" (its utility function is fairly small). The utility function of a user is its throughput. The fitness function can be just defined as the system utility function which is given as

$$U(Y)=\sum_{k=1}^{K}U_k\left(\sum_{n=1}^{N}y_k^n\right)$$

And the resource allocation problem can be formulated as

$$\max\ U(Y)=\sum_{k=1}^{K}U_k\left(\sum_{n=1}^{N}y_k^n\right)$$

$$\text{such that } \sum_{k=1}^{K}\sum_{n=1}^{N}p_k^n\left(y_k^n\right)\le P_{total}$$

$$0\le y_k^n\le C^n x_k^n$$

$$\sum_{k=1}^{N}x_k^n\le 1,\ x_k^n\le\{0,1\}$$

where y_k^n is the transmit data rate of the user k assigned to the subcarrier n. x_k^n is 1 if the subcarrier n is assigned to user k and 0 otherwise. P_{total} is the total transmit power.

U(Y) is the utility function where $Y = \left(y_1^1, y_1^2, ..., y_1^N; y_2^1, y_2^2, ..y_2^N; ..; y_K^1, y_K^2, ..y_K^N\right)$ is defined as the rate allocation vector.

2.1.2.4 Selection, Crossover and Mutation Operation

In this genetic algorithm, an elitist model is adopted as the selection operator. By the model, the optimal individuals are selected first and directly copied to the next generation, and then the rest are selected by the proportional model. The crossover scheme is selected as parents to produce an offspring. Because after refining the initial population produces good population, the crossover operations and mutation operation used in this algorithm is not special, and can be selected from the popular algorithm.

2.1.2.5 Performance of Heuristic Genetic Algorithm

The performance of this algorithm is compared with OFDMA-FDMA scheme in which each user is assigned a predetermined band of subcarriers and can only use those subcarriers exclusively in a scheduling period. Fig. 2.4 shows the average received data rate per user as the number of users increases from 2 to 15 in the system. As shown in Fig. 2.4, the genetic algorithm (GA) always has more throughputs because they assign

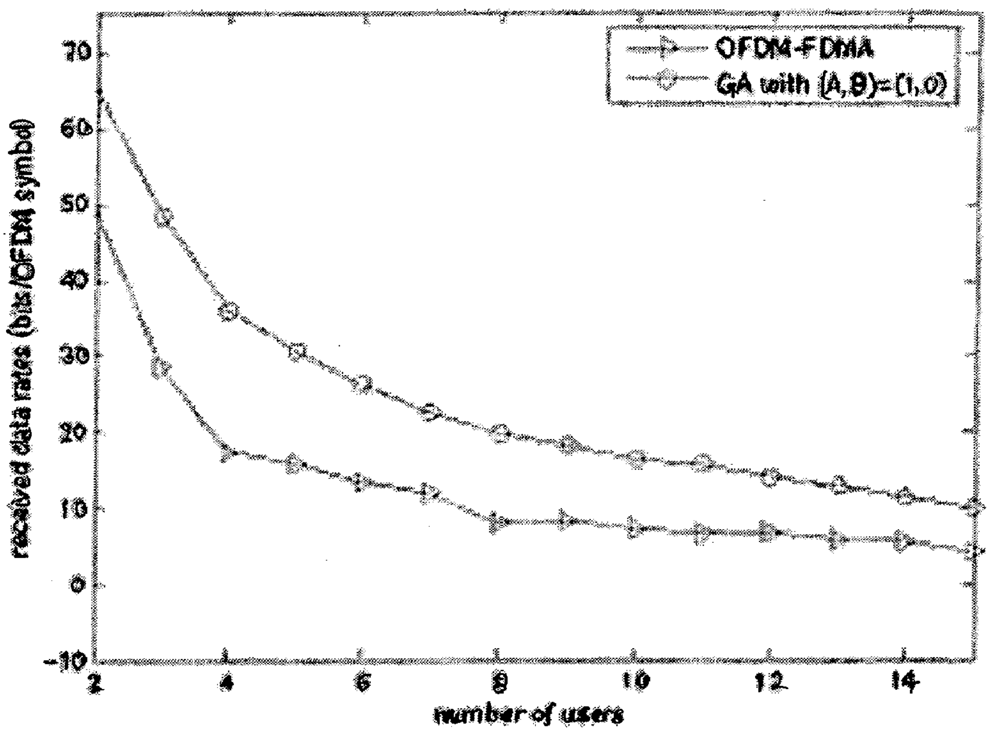


Fig.2.4 Average received data rates per user

subcarriers adaptively based on the channel conditions in the scheduling period. When the number of users is 13 and 2 in the system, the throughputs of this scheme increase 123% and 32%, respectively, compared with OFDM-FDMA.

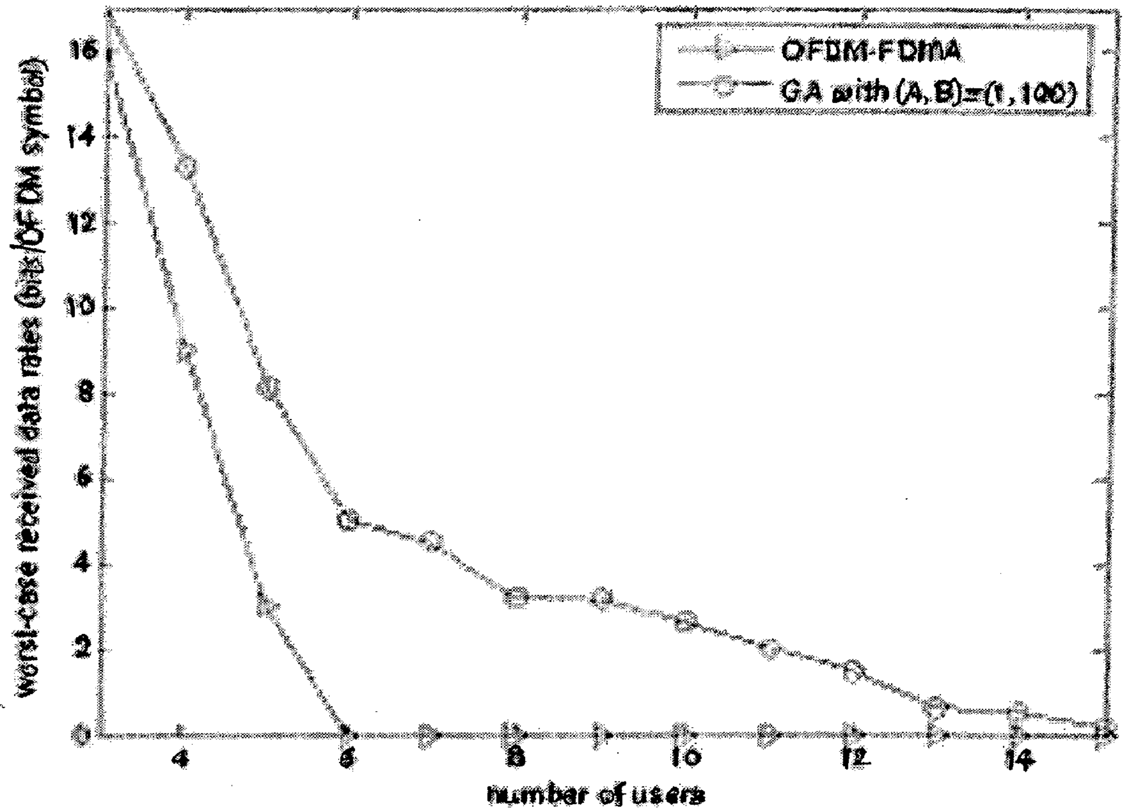


Fig. 2.5 Average throughput of worst-case users

Fig. 2.5 shows the average throughput of worst-case users as the number of users increases from 3 to 15 in the system. As shown in Fig. 2.5, the received data rates of worst-case users in the proposed scheme are higher than that of OFDM-FDMA because the scheme considers fairness among users. Average received data rates of worst-case users in OFDM-FDMA are zero when the number of users in the system is more than 7. However, the average received data rate of this scheme is nonzero even when the number of users is 15.

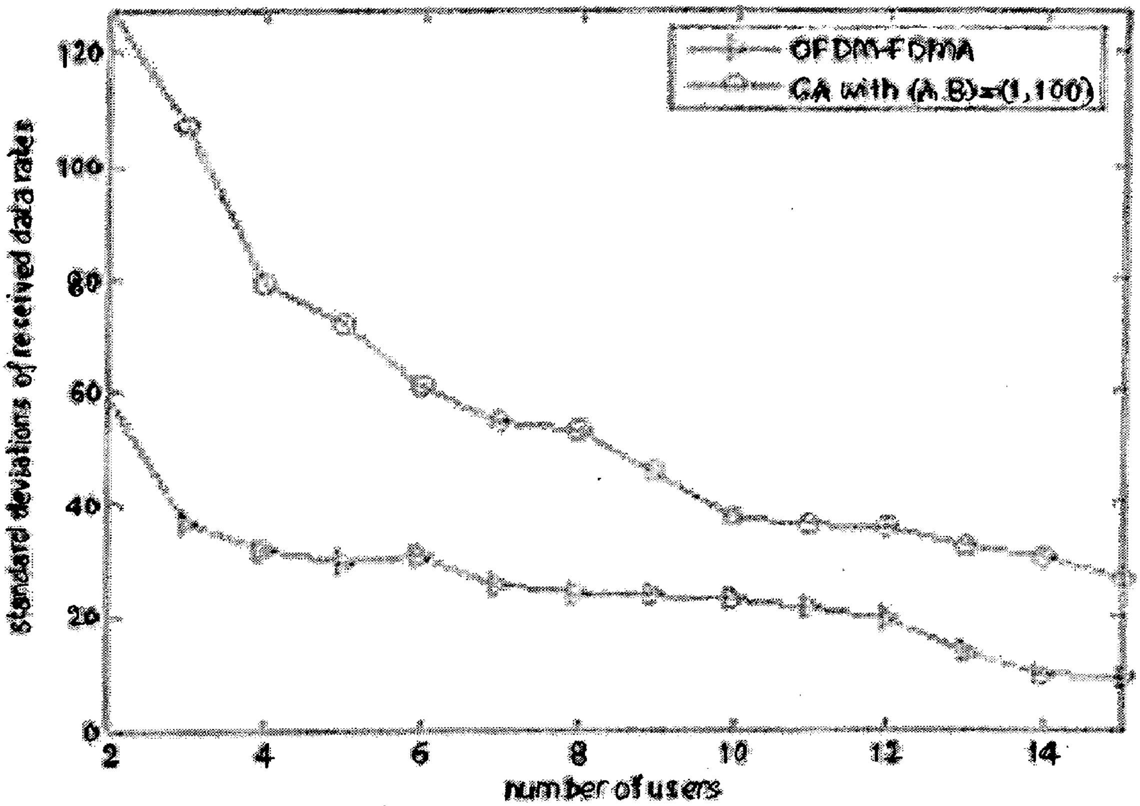


Fig. 2.6 Standard deviations of users' received data rates

Fig. 2.6 shows the standard deviations of each user's received data rates as the number of users increases from 2 to 15 in the system. The standard deviations of OFDM-FDMA are the lowest. However, as shown in Fig. 2.6, the throughputs of worst-case users in OFDM-FDMA are much worse than that of this scheme. In other words, only few users with good channel conditions can receive very high data rate while most users cannot receive a high data rate in OFDM-FDMA. On the other hand, users with good channel conditions in this scheme receive much higher data rate and users with poor channel conditions receive a higher data rate than that of OFDM-FDMA. As a result, the GA is found to perform considerably better than OFDM-FDMA for non real-time data traffic.

2.2 Fair Scheduling Algorithms based on Latency[8]

In this section two algorithms are described which make use of the fact that several users can be multiplexed on a same subcarrier in a time division manner. These algorithms aim at approaching optimal proportional fairness and throughput with lower complexity. The allocation is performed on a per subchannel basis, *i.e.*, a mapping decision made for a subchannel is applied for all the subcarriers which belong to that subchannel.

A scheduler P is said to be proportionally fair (PF)[7] if and only if

$$\sum_{i \in U} \frac{R_i^{(\sigma)} - R_i^{(P)}}{R_i^{(P)}} \leq 0 \qquad (2.1)$$

where U is the user set and σ is any feasible scheduler, and $R_i^{(\sigma)}$ is the average rate of user i by scheduler S.

In other words, any positive change of a user in the allocation must result in a negative average change for a system. A proportional fair allocation P should maximize the sum of logarithmic average user rates *i.e.*

$$P = \arg\max_{\sigma} \sum_{i \in U} \log R_i^{(\sigma)} \qquad (2.2)$$

Let K be the number of users, then the proportional fairness metric of a scheduler σ Genaral Proportional Fairness (GPF) parameter, Γ_{σ}, can be defined as

$$\Gamma_{\sigma}(t) = \sum_{k=1}^{K} \log \ R_k^{\sigma}(t) \qquad (2.3)$$

As explained in [8], this parameter can be decomposed as

$$\Gamma = A + B_N$$

where

$$A = \sum_{k=1}^{K} \log\left[\frac{T-S}{T} R_{k'}\right] \qquad (2.4)$$

and

$$B_n = \sum_{k=1}^{K} \log\left[1 + \sum_{j=1}^{n} \frac{s_{k,j} r_{k,j}}{(T-S) R_k'}\right] \qquad (2.5)$$

where T is the PFS window length equal to a multiple of the scheduling time frame, n_f, i.e., $T=n_f*S$.

2.2.1 PF-User Multiplexing Algorithm (PF-Mux)

Let M be the total number of subchannels. In each subchannel m, all the users are ordered according to decreasing user PFS metric $\rho_{k,m}$. The best PFS users are first allocated in each subchannel. At each step, the number of multiplexed users per subchannel is doubled from a users to $2a$ users if the following two conditions are fulfilled. The first condition is the rate condition expressed as

$$\sum_{k=2^{a-1}+1}^{2^a} r_{k,m} \geq \sum_{k=1}^{2^{a-1}} r_{k,m} \tag{2.6}$$

where $r_{k,m}$ is the achievable rate of user k in subcarrier n. Condition (2.6) ensures that the average rate after sharing $2a$ users will be increased compared to the case with a users. The second condition is the proportional fairness condition expressed as

$$B_m^{(2a)} \geq B_m^{(a)} \tag{2.7}$$

which ensures that the partial general proportional fairness (GPF) metric will be increased when allowing $2a$ users, since the term A, defined in (2.4) is constant during one frame allocation. Considering subchannels, this can be expressed as

$$\sum \log\left[1+\frac{N}{M}\times\sum\frac{s_{k,m}^{2a} r_{k,m}}{(T-S)R'_k}\right] \geq \sum \log\left[1+\frac{N}{M}\times\sum\frac{s_{k,m}^{a} r_{k,m}}{(T-S)R'_k}\right]$$

where the terms $s_{k,m}^{a}$ is equal to $S/2^i$ for the a best PFS users with $i = log_2(a)$ and $s_{k,m}^{2a}$ is equal to $S/2^{i+1}$ for the $2a$ best PFS users. If both proportional fairness and rate conditions are satisfied, the $2a$ users are multiplexed on the subchannel by getting $S/(2a)$ slots each, otherwise only the initial a users are given the S/a slots. The number of sharing users is doubled at each step until one of the conditions is invalidated or when the maximum number of allocable users, e.g. S, is reached.

2.2.2 Rate-user Multiplexing Algorithm (Rate-Mux)

This algorithm works in the same manner as the previous one, except that in the beginning, users are ordered according to their rate metrics instead of PFS metrics. In this algorithm, as well as the previous one, in each subcarrier, the number of slots per multiplexed user is divided equally.

2.2.3 Performance

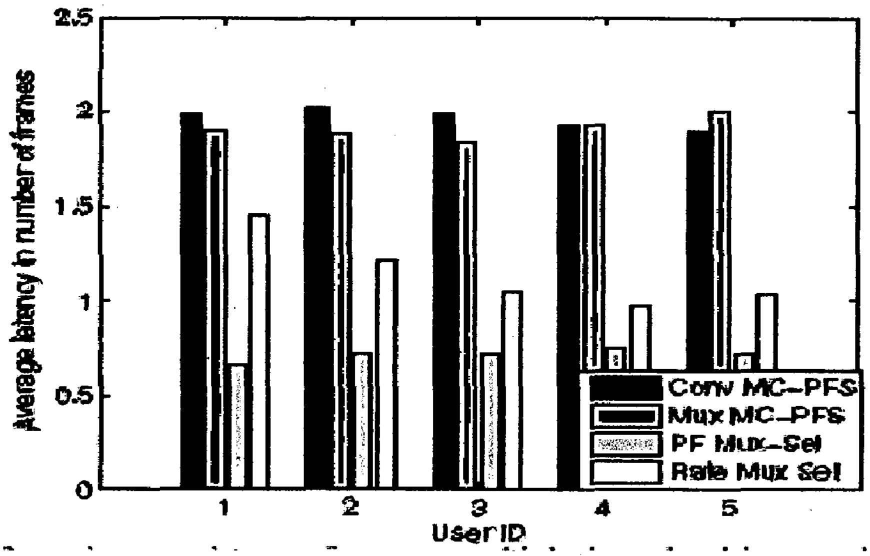


Fig. 2.7 Average latency for user multiplexing algorithms and reference algorithms

The performances of these algorithms are compared with each other along with conventional MC-PFS and Mux MC-PFS on the basis of average latency shown in Fig.2.7. The best performance is achieved by *PF–Mux*, in which latency is less than half of the latencies experienced by Conventional MC– PFS and *Mux MC–PFS*. *Rate–Mux* achieves the second best latency, but quite higher than for *PF–Mux*. Furthermore, compared to Conventional MC–PFS, *PF–Mux* really improves latency while keeping a comparable throughput and *Rate–Mux* improves both metrics. Therefore it can be expected that when assuming real packet transmissions, the delay performance to be also improved, as these results indicate that a higher throughput can be achieved in a reduced amount of time.

2.3 Fair Scheduling Algorithms Based on Throughput

In this section different types of scheduling algorithms based on throughput are discussed.

2.3.1. Dynamic Resource Scheduling Algorithm [9]

In this algorithm, the *M* subcarriers are divided into *N* subbands, which are made up of *M/N* neighboring subcarriers. Algorithm is stated as follows:

Step 1: Determine feedback DRC (Data Rate Control) of each subband by the measured SNR (Signal to Noise Ratio) of that subband and transmitted from the access terminal (AT) to the access point (AP).

Step 2: Select user for each subband: The user with the highest ratio of DRC_i^j / R_i, that applies for this subband will be: selected for this subband, where DRC_i^j denotes the DRC of the i^{th} user at the j^{th} subband and R_i is the average rate received by the i^{th} user. A user may obtain more than one resource units and suppose the i^{th} user obtains k subbands in this slot.

Step 3: Determine the transmission schemes for each user: All subbands for the same user are unified in the coding and modulation processes to obtain a larger interleave gain. The modulation parameter is the highest value from k subbands and the overall transmission data size L is the sum of k DRCs.

Step 4: Update Average Rate of each user: For the i^{th} user,

$$R_i = (1 - 1/T_c)R_i + 1/T_c * L \tag{2.8}$$

,where T_c is the maximum time to be starved. For the user that is not served by any subband, its R should be updated as

$$R_i = (1 - 1/T_c)R_i \tag{2.9}$$

2.3.1.1 Performance of Dynamic Resource Scheduling Algorithm

Performance of this multicarrier proportional fair (MCPF) algorithm is compared with FS (Fixed Scheduling) algorithm where the subband is given to a fixed user. When the number of subband is greater than that of user, the j^{th} subband is allocated to the user whose ID equals $j \bmod M$ at any slot. Its modulation and coding parameters are also related to the user's downlink SNR at this subband. Otherwise, each user will be allocated to the fixed subband in a fixed period more than one slot. Fig 2.8 compares the throughputs of FS and MCPF when the number of user ranges from 1 to 8 and T_c equals to 200. This figure indicates that the average throughput of MCPF is greater than that of FS by more than 60 percent when the number of user is greater than two. Further on, MCPF's performance on system throughput increases evidently with the number of user before it reaches 7 but this increase becomes slight afterwards. With more users, there is more possibility for MCPF to find a user with better channel condition at the subband. Therefore, MCPF has throughput advantage in the multi-user systems.

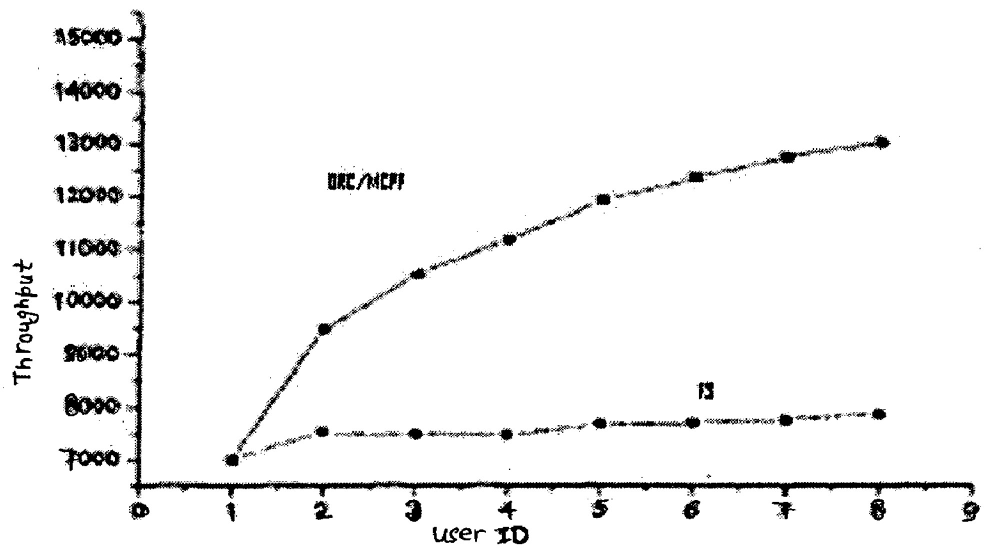


Fig. 2.8 DRC/MCPF vs. FS in average throughput when T_c=200

2.3.2 Multicarrier Proportional Fair (MCPF) Scheduling Algorithm [10]

This is another MCPF algorithm. In this algorithm the power of each carrier is assumed to be equal. The information on the instantaneous transmittable rate of each downlink carrier is reported by the mobile station (MS), which is assumed to be received without error at the base station (BS). At each scheduling epoch, a scheduler is identified by (user, carrier) pair selection for the data transmission over a lot. Let U be the user set and C be the carrier set. The PF scheduling is given below:

$$P = arg\ max_S \prod_{i \in U_S} \left(1 + \frac{\sum_{k \in C_i} r_{i,k}}{(T-1)R_i^{'}} \right) \qquad (2.10)$$

where U_s is the set of selected users by S. C_i is the set of carriers allocated to user $i \in U_S$, and $r_{i,k}$ is the instantaneous transmittable rate of the carrier $k \in C_i$ when transmitted for user i at the current slot. R_i is the average rate of user i at the previous slot, and T is the average window size.

For a simpler implementation, an allocation scheme selects only one user at each scheduling epoch. In this case the number of comparisons is $|U|$, and the scheduling rule becomes

$$p = arg\ max_i \prod_{i \in U_S} \left(1 + \frac{\sum_{k \in C_i} r_{i,k}}{(T-1)R_i^{'}} \right) \qquad (2.11)$$

If PFS is adopted for each carrier, not for all of carriers, the number of comparisons is $|C| \cdot |U|$. In this case, the rule is represented as

$$p_k = \max_i \frac{r_{i,k}}{R'_i}, \quad k = 1,2,\ldots,|C| \qquad (2.12)$$

2.3.2.1 Performance of Multicarrier Proportional Fair Scheduling Algorithm

The performance of this algorithm is compared with SCPF (Single Carrier Proportional Fairness) and RR (Random Resource) scheduling algorithms. SCPF scheduling adopts the proportional fair scheduler in each carrier which is described above. RR allocates all carriers to user $j\%N$ where j is the index of a scheduling epoch (slot) and N is the number of users.

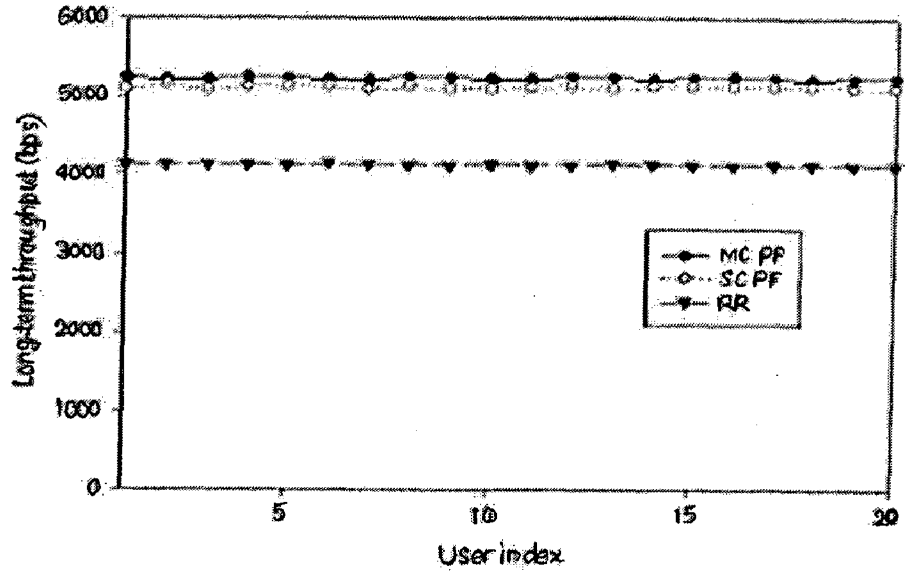


Fig. 2.9 Long-term throughput vs. user index with T of 10

From Fig.2.9, it is clear that the scheduler of MCPF and SCPF take advantages of the variations of the channel quality. Throughput increase is achieved by 26% and 2% in MCPF compared with RR and by SCPF, respectively.

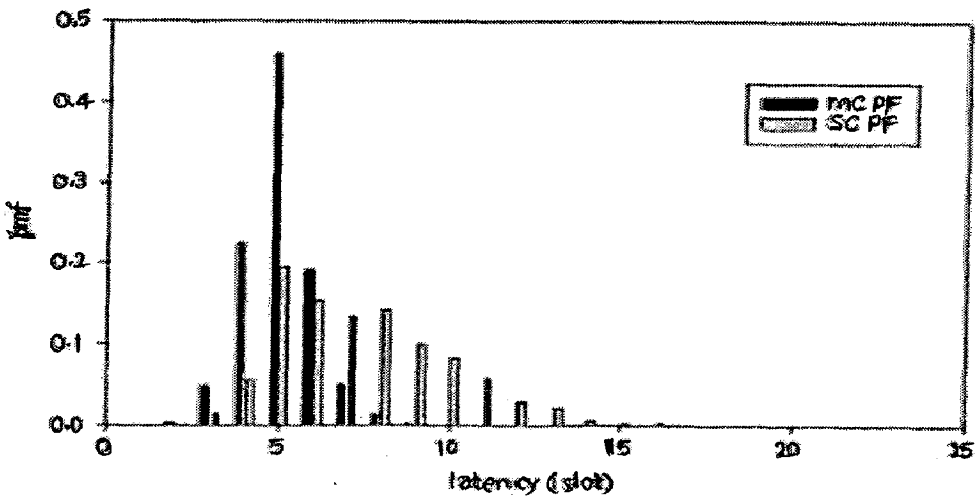


Fig. 2.10 Latency Distribution with T of 10

Fig. 2.10 shows the latency distribution in MCPF and HDR PF of a user when T is 10. By MCPF, the number of allocations is larger than that by SCPF, and the average and the maximum latency by MCPF is smaller than those by SCPF.

2.3.3 Rounding Algorithm, Best-Rate Algorithm, Best-PFS Algorithm [11]

The MC-PFS problem can be written as

$$\begin{aligned} &\text{minimize } -\sum_{k=1}^{K}\log\left(a^{(k)T}x+b^{(k)}\right) \\ &\text{subject to } h^{(n)T}x=1,\ n=1,\ldots..,N \qquad (2.13) \\ &x_i\in\{0,1\},\ i=1,\ldots\ldots,KN \end{aligned}$$

with the variable $x\in R^{KN}$, $a^{(k)}\in R^{KN}$, $h^{(n)}\in R^{KN}$ and $b^{(k)}\in R$ where $a_i^{(k)} = \frac{r_{k,n}}{T}$, for $i=(k-1)\times N+n$; n=1,..., K and 0 otherwise and $b^{(k)}=\frac{T-1}{T}R'_k$. The first constraint expresses that only one user can be allocated to each subcarrier. This is a difficult combinatorial problem, but if we relax the constraints $x_i \in \{0, 1\}$ to $0 \le x_i \le 1$, $i = 1, \ldots, KN$ we get a *convex optimization problem*,

$$\begin{aligned} &\text{minimize } -\sum_{k=1}^{K}\log\left(a^{(k)T}x+b^{(k)}\right) \\ &\text{subject to } h^{(n)T}x=1,\ n=1,\ldots..,N \qquad (2.14) \\ &0\le x_i\le 1,\ i=1,\ldots\ldots,KN \end{aligned}$$

Its solution gives an *upper bound* to (2.13) since the problem is solved over a larger domain. Such an approach is a *convex relaxation.* This optimization is performed every time frame. The average user rates are updated for the next frame according to x_i. GPF and throughput are calculated based on x_i. This can be interpreted as the allocation of a fraction of the subcarrier corresponding to the value x_i. This is not possible to implement since we assume to have a single OFDM symbol per allocation instance.

2.3.3.1 Rounding Algorithm

In Rounding algorithm, the terms of x are mapped to either 1 or 0, by assigning a subcarrier $n*$ to the user $k*$ for which x_{kn*} is maximum: $x_{k*n*} = 1$ and $x_{kn*} = 0$ for $k \neq k*$. Thus, the subcarrier is assigned to the user with the highest fraction.

2.3.3.2 Trade-off between Rate and Fairness

The Rounding Algorithm does not achieve the best throughput. Two other algorithms are proposed which make a trade-off between those two metrics: the Best-Rate algorithm which aims at near-optimal rate while keeping a good PF, and the Best-PFS algorithm, which targets a near-optimal PF, at a small expense of the throughput. These algorithms only differ in step 4) in the description below. The set of empty subcarriers to be allocated is denoted N_E. Initially, the cardinality card (N_E) = N.

1) For $n \in N_E$, order users by:

- order of decreasing rate metric $r_{k,n}$
- order of decreasing user PFS metric $\rho_{k,n}$

If several users have the same rate1, put them in the same level of order.

2) For $n \in \mathbf{N}_E$, determine the best PFS user $k^*_{n,\rho}$ and the best rate user $k^*_{n,r}$ as:

$$k^*_{n,\rho} = \arg\max_k \rho_{k,n}$$

$$k^*_{n,r} = \arg\max_k r_{k,n}$$

In each n, compare $k^*_{n,\rho}$ and $k^*_{n,r}$:

- if $k^*_{n,\rho} \neq k^*_{n,r}$: add n in $\mathbf{N}_{E'}$, the set of empty subcarriers after this matching process, and go to the next subcarrier.
- if $k^*_{n,\rho} = k^*_{n,r}$: allocate n to this user and go to the next subcarrier.

Repeat this procedure for every subcarrier.

3) If there were some matching cases, i.e. if card ($\mathbf{N}_{E'}$) < card ($\mathbf{N}_E$), partially update the average user rates with

$$R^{l}_{k^*}(t) = \left(1 - \frac{1}{T}\right) R'_{k^*}(t) + \frac{1}{T} \sum_{n=1}^{N} c_{k^*,n} r_{k^*,n}(t)$$

and go to step 5). If there were no matching at all, i.e. if card ($\mathbf{N}_{E'}$) = card ($\mathbf{N}_E$), proceed to step 4).

4) The algorithms differ in this step:

4A) *Best-Rate algorithm*: select the best rate user k^*_r among all best rate users $k^*_{n,r}$ and his corresponding subcarrier n^* among $n \in \mathbf{N}_{E'}$:

$$\left(k_r^*, n^*\right) = \arg_{k_{n,r}^*, n \in N_{E'}} \max \rho_{k,n}$$

Allocate user k^*_r to subcarrier n^*.

4B) *Best-PFS algorithm*: select the best PFS user k^*_ρ among all best PFS users $k^*_{n,\rho}$ for $n \in \mathbf{N}_{E'}$:

$$\left(k_\rho^*, n^*\right) = \arg_{k_{n,\rho}^*, n \in N_{E'}} \max \rho_{k,n}$$

Allocate user k^*_ρ to subcarrier n^*.

For 4A) and 4B): partially update the average user rates with (4) and $\mathbf{N}_{E'} = \mathbf{N}_{E'} - \{n^*\}$.

5) PFS metrics are reordered. Re-initialize the set of remaining subcarriers: $\mathbf{N}_E = \mathbf{N}_{E'}$ and the set of empty subcarriers after matching: $\mathbf{N}_{E'} = \phi$. Repeat the algorithm from step 1 if $\mathbf{N}_E \neq \phi$ and end when $\mathbf{N}_E = \phi$.

With the *partial* updates of steps 3) and 4), the allocation utilizes sequence of subcarriers that is directed towards more optimal PF.

2.3.3.3 Performance of Rounding Algorithm, Best-Rate Algorithm, Best-PFS Algorithm

Fig. 2.11 shows that Rounding has a near-optimal PF, as the optimal solution is located between the upper-bound and Rounding. However, Fig. 2.12 shows that the throughput of Rounding is outperformed by Best-PFS and Best-Rate, which approaches Max CSI but with much higher PF. This is because the AMC model allows users with different CSI to achieve identical rate. While Max CSI selects the users with the best SNR, Best-Rate selects among the best rate users, the ones with highest PFS. Moreover, the GPF decrease of Max CSI at $K = 20$ is due to the increased number of unscheduled users for constant $N = 16$: after several frames, their average rates decrease and become less than 1, leading to a negative contribution to GPF. The decrease of the throughput of Rounding for more than 15 users is due to the fact that, the more users, the more

Rounding diverges from the upper bound as the difference between the allocated subcarrier fractions increases.

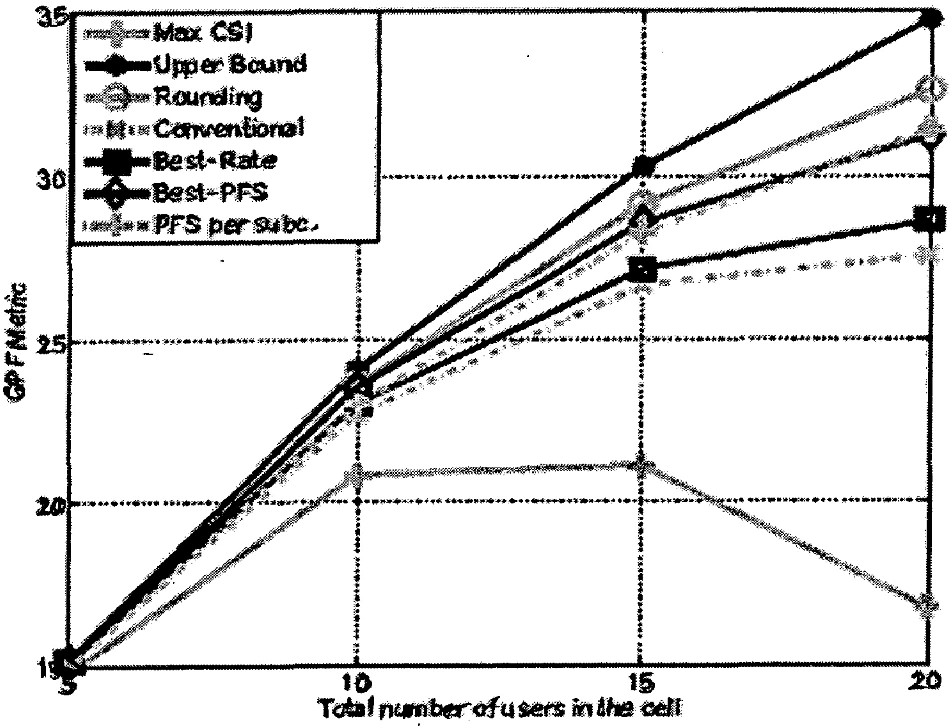


Fig. 2.11 Performance in terms of users in the cell

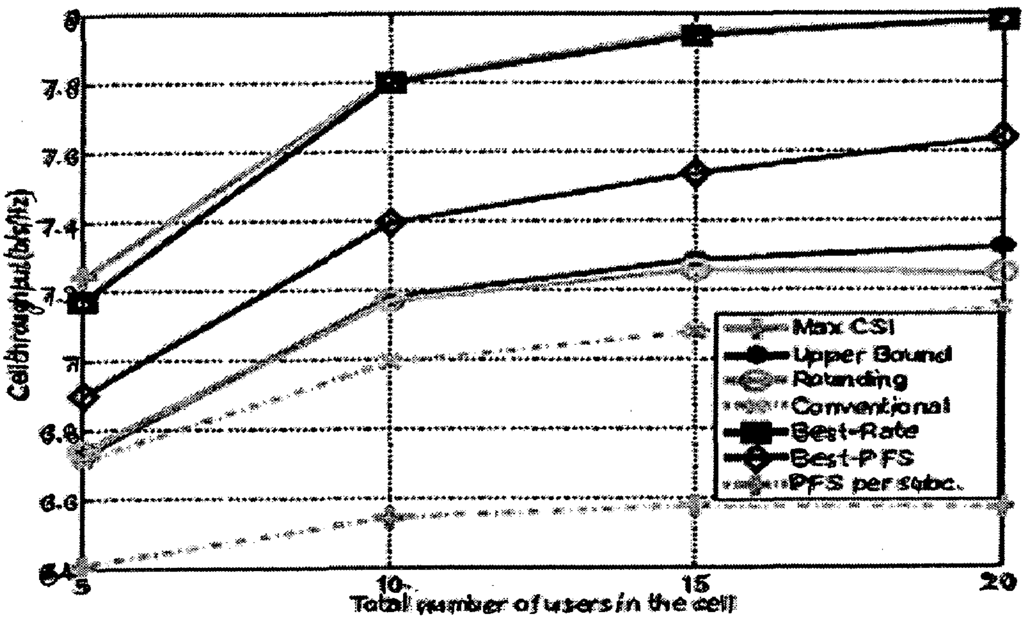


Fig. 2.12 Cell throughput in [b/s/Hz]

Compared to Best-Rate, Best-PFS improves PF, at the expense of throughput but still outperforms existing algorithms. Besides, Best-Rate and Best-PFS algorithms delimit a region for each metric. Any point of this region can be achieved by a combined algorithm which selects randomly step 4A) or 4B).

Finally, a complexity analysis shows that Conventional MC-PFS grows as $O(2KN)$, PF upper-bound at least as $O((2KN)^3)$, and Rounding algorithm as $O((2KN)^3+KN)$. For α iterations, our algorithms grow as $O(2\alpha KN)$, which equals in the worst case to $O(2KN2)$ for $\alpha = N$. Both algorithms achieve a better performance than Conventional MC-PFS with a reasonable complexity, much lower than the one required by the optimal solution.

2.3.4 Minimum Normalized Transmit Rate Based Fair Resource Scheduling Algorithm [4]

In this algorithm the user with the minimum normalized transmit rate is selected first which ensures the fairness among users and then the subcarrier with the highest utilization efficiency corresponding to this user is selected. In this dissertation report the results of this algorithm are simulated on MATLAB. So, this algorithm will be discussed in the next chapter in detail.

<u>Chapter 3</u>

MINIMUM NORMALIZED TRANSMIT RATE BASED FAIR SCHEDULING ALGORITHM

In this chapter, a throughput based resource scheduling algorithm has been discussed. The simulation part of the dissertation is based on this algorithm, so before going through the algorithm we will discuss the mathematical background for the algorithm in brief.

3.1 Square QAM Approximation [12]

QAM constellation may take a variety of forms. When these points are arranged regularly within a square boundary, the constellation is said to be a QAM square. The distance between points in the constellation is denoted by d, and all points are assumed equally likely. The constellation is centered at the origin and has zero mean value. Energy per two dimensional symbols is given by

$$E = (M-1)d^2/6 \tag{3.1}$$

where $M=2^b$ represents the number of points in the constellation, and b is the number of bits represented by single QAM symbol. In practice, when $b \geq 2$ and odd or non-integer, then (3.1) is not exact; but still an accurate approximation to the average transmit energy.

The problem of symbol error in QAM is closely approximated by,

$$P_e \leq 4Q(d_{\min}/(2\sigma)) \tag{3.2}$$

where σ^2 is noise variance and d_{min} is the minimum distance between constellation points at the channel output and if h is the channel gain and assuming the channel has no ISI,

$$d_{\min}{}^2 = d^2 . |h|^2 \tag{3.3}$$

where d = minimum distance between constellation points at the channel input in an uncoded input constellation. The Q-function in (3.2) is defined by

$$Q(x) = \frac{1}{\sqrt{2\Pi}} \int_x^\infty e^{-\frac{z^2}{2}} dz \tag{3.4}$$

For a given P_e, the term $\left(\frac{d_{\min}{}^2}{4\sigma^2}\right)$ is calculated and SNR gap (Γ), is defined as

$$3.\Gamma = \left(\frac{d_{\min}^{2}}{4\sigma^{2}} \right) \tag{3.5}$$

(3.1) can be written as

$$M = 1 + \frac{6E|h|^{2}}{d_{\min}^{2}} \tag{3.6}$$

By taking log base 2 of (3.6) and substituting for d_{min} from (3.5) in (3.6) number of bits can be computed as

$$b = \log_2(M) = \log_2\left(1 + \frac{SNR}{\Gamma}\right) \tag{3.7}$$

where SNR for a given channel is given by

$$SNR = E.|h|^{2} / 2\sigma^{2} \tag{3.8}$$

Based on the channel SNR and P_e requirement one can decide on the number of bits that can be transmitted on the channel using the above equations.

3.2 Multicarrier Analysis [12]

In case of multicarrier transmission such as OFDMA where number of user is M and the number of subcarriers is K. Let SNR_{mk} be the SNR corresponding to m^{th} user and k^{th} subcarrier then from (3.7)

$$b_{mk} = \log_2(M) = \log_2\left(1 + \frac{SNR_{mk}}{\Gamma}\right) \tag{3.9}$$

Let p be the transmit power of each carrier and g_{mk} be the channel gain corresponding to m^{th} user and k^{th} subcarrier. Then

$$SNR_{mk} = \frac{g_{mk}\,p}{\sigma^{2}} \tag{3.10}$$

Substituting (3.10) in (3.9), we get

$$b_{mk} = \log_2(M) = \log_2\left(1 + \frac{g_{mk}\,p}{\Gamma\sigma^{2}}\right) \tag{3.11}$$

The above equation tells maximum how many bits a user can allocate on a subcarrier.

Here it is assumed that P_e is maintained constant in all the subcarriers and hence SNR gap is the same for all the subcarriers. The total number of bits transmitted in one OFDM symbol is given by,

$$b = \sum_{k=1}^{K} b_{mk} \tag{3.12}$$

3.3 Mathematical Model for Algorithm [4]

Let $a_{mk}=1$ denotes that the subcarrier k is allocated to user m and $a_{mk}=0$ if subcarrier k is not allocated to user m. Assume that a subcarrier can not be used by more than one user, thus

$$\sum_{m=1}^{M} a_{mk} \text{ , } \forall k \tag{3.13}$$

For each user the total granted transmit rate is calculated as

$$W_m = \sum_{k=1}^{K} a_{mk} b_{mk} \tag{3.14}$$

and the normalized transmit rate is W_m / ϕ_m, where ϕ_m is the rate weight of user m.

Service Fairness Index (SFI), the maximum difference between the normalized transmit rates, is usually used to measure the fairness of scheduling algorithm, which is represented as:

$$SFI = \max_{m,n} \left| W_m / \phi_m - W_n / \phi_n \right| \tag{3.15}$$

SFI = 0 indicates the ideal fairness, which is unfeasible in real systems. For real systems there an upper bound B is used to measure the fairness, i.e.

$$\max_{m,n} \left| W_m / \phi_m - W_n / \phi_n \right| \le B \tag{3.16}$$

An efficient fair resource scheduling algorithm should maximize the total throughput while satisfying the fairness requirement. The optimization problem can be expressed as:

$$\max_{a_{mk} \in \{0,1\}} \sum_{m=1}^{M} \sum_{k=1}^{K} a_{mk} b_{mk} \tag{3.17}$$

subjected to:

$$\max_m\left(\frac{\sum_{k=1}^{K} a_{mk}b_{mk}}{\phi_m}\right)-\min_n\left(\frac{\sum_{k=1}^{K} a_{nk}b_{nk}}{\phi_n}\right)\leq B \tag{3.18}$$

and

$$\sum_{m=1}^{M} a_{mk}\leq 1. \tag{3.19}$$

3.4 Minimum Normalized Transmit Rate Based Fair Scheduling Algorithm [4]

In this algorithm, the user with the least normalized transmit rate is allowed to choose a subcarrier. This guarantees the fairness among all users. Then, the subcarrier with the highest utilization efficiency to that user is selected. Thus, a high efficiency can be achieved. This algorithm is divided into K steps where K is the number of subcarriers. The main algorithm is given below:

Step 1 : Initialize $a_{mk} = 0$, W_m=0 for m=1,2, ..., M and for k=1,2,...,K. Set S = {1,2 ,.., K}.

Step 2: Select m*=arg min$\{W_m / \varphi_m\}$. For all $k\in S$, calculate the utilization efficiency $u_{m^*k}=b_{m^*k}/\sum_{m=1}^{M} b_{mk}$. Let $k^*=\arg_k \max\{u_{m^*k}\}$ and set $a_{m^*k^*}=1$. Update $W_{m^*}=W_{m^*}+b_{m^*k^*}$ and $S=S\setminus\{k^*\}$.

Step 3: If $S=\phi$, then stop. Otherwise, go to Step 2.

3.4.1 Performance

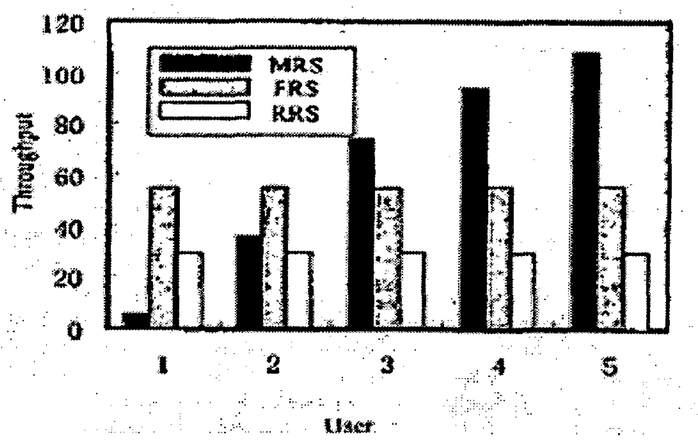


Fig. 3.1 Achieved throughput of each user

The performance of this algorithm is compared with random resource scheduler (RRS) where a subcarrier is selected randomly while the FRS selects a subcarrier with the largest utilization efficiency and maximum rate scheduler (MRS) which allocates each subcarrier to the user with the largest number of loaded bits. From Fig. 4.1 it is clear that although a higher total throughput is achieved in the MRS algorithm, it is not a fair scheduling algorithm since the algorithm always allocates more subcarriers to users with higher average channel gain. In both the FRS algorithm and the RRS algorithm, each user gets almost the equal throughput. But a higher throughput is achieved in FRS than in RRS, since each user in the FRS algorithm selects a "best" subcarrier instead of a "random" subcarrier in RRS case.

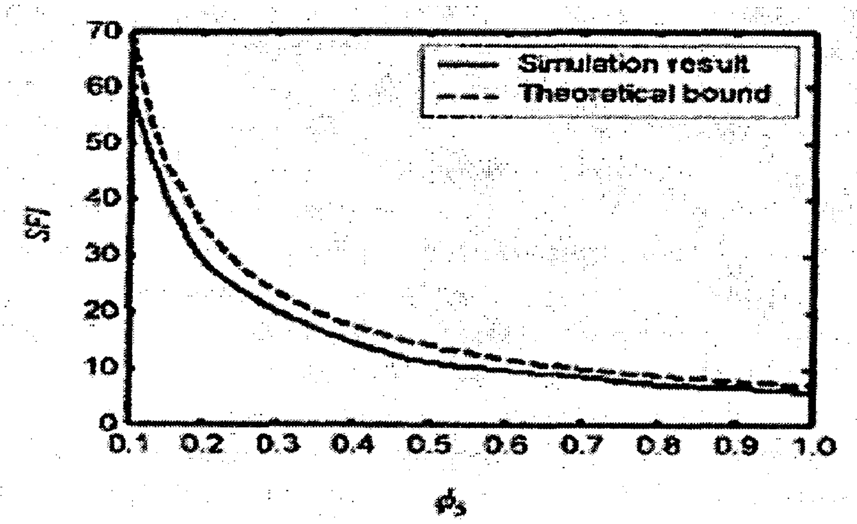


Fig. 3.2 Fairness of the FRS algorithm

Fig. 3.2 shows that the service fairness index of the algorithm is within the theoretical bound.

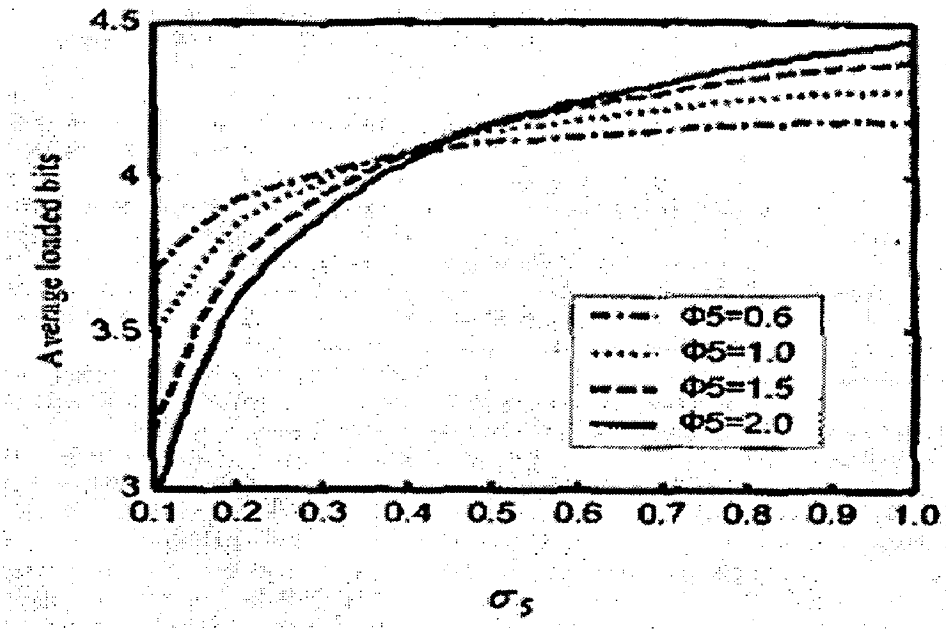


Fig. 3.3 Average loaded bits per subcarrier versus σ_5

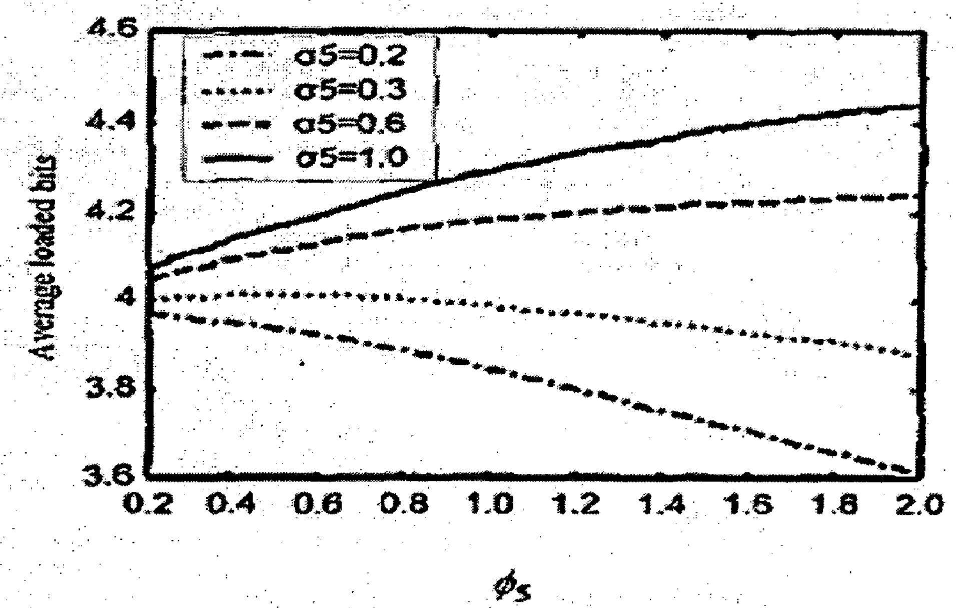


Fig. 3.4 Average loaded bits per subcarrier versus ϕ_5

From Fig. 3.3 we can see that average no. of loaded bits per subcarrier increases as average channel gain of the 5th user increases, and the larger the rate weight is, the more the rate improves. This is reasonable because with the increase of the fifth user's average channel gain, it's beneficial for the system throughput if more bandwidth is allocated to the fifth user.

From Fig. 3.4 we can see that, when the average channel gain of the fifth user is small (σ_5 =0.2, 0.3), the average loaded bits decrease as the ϕ_5 increases. The reason is that when a user's channel condition is poor, the larger the proportion of bandwidth allocated to that user, the more the bandwidth is wasted. On the contrary, when the average channel gain of the fifth user is large (σ_5 =0.6, 1.0), it's better to allocate more subcarriers to the fifth user.

3.5 Fairness Bound [4], [13]

Proposition: The service fairness index of the algorithm is upper bounded by

$$B = b_{\max} / \min_{m}(\phi_m) \tag{3.20}$$

where $b_{\max}$ is the maximum loaded bits in a subcarrier.

Proof:

After the k[th] (k=1, 2,...., K-1) subcarrier is allocated, suppose the normalized transmit rate of user m and user n is the largest one and the smallest one among all users, respectively. Then, the SFI [12] after the k[th] step is:

$$SFI(k) = W_m / \phi_m - W_n / \phi_n \tag{3.21}$$

At the (k+1)[th] step, user n will select a subcarrier. Let k* be the index of subcarrier selected by user n. Thus, if $(W_n + b_{nk^*})/\phi_n \geq W_m / \phi_m$, the SFI after the (k+1)th step is $SFI(k+1) \leq b_{nk^*} / \phi_n \leq B$. Otherwise, SFI(k+1) is smaller than SFI(k), and will be bounded by B if SFI(k)≤B. Obviously, at the first step, SFI(0)=0<B, so SFI(k) can be proved to be within B inductively.

Chapter 4

SIMULATION AND RESULTS

In this dissertation, a throughput based fair scheduling algorithm [4] has been simulated in MATLAB R2006b considering only downlink (base station to users) resource scheduling in OFDM cellular system.

4.1 System Model

In this dissertation, the simulation model considered in [10] is used in simulating the downlink resource scheduling for OFDMA. There are 5 users sharing the total bandwidth, which is divided into 64 subcarriers. A three-path Rayleigh fading channel is considered for simulation.

4.1.1 Simulation Parameters

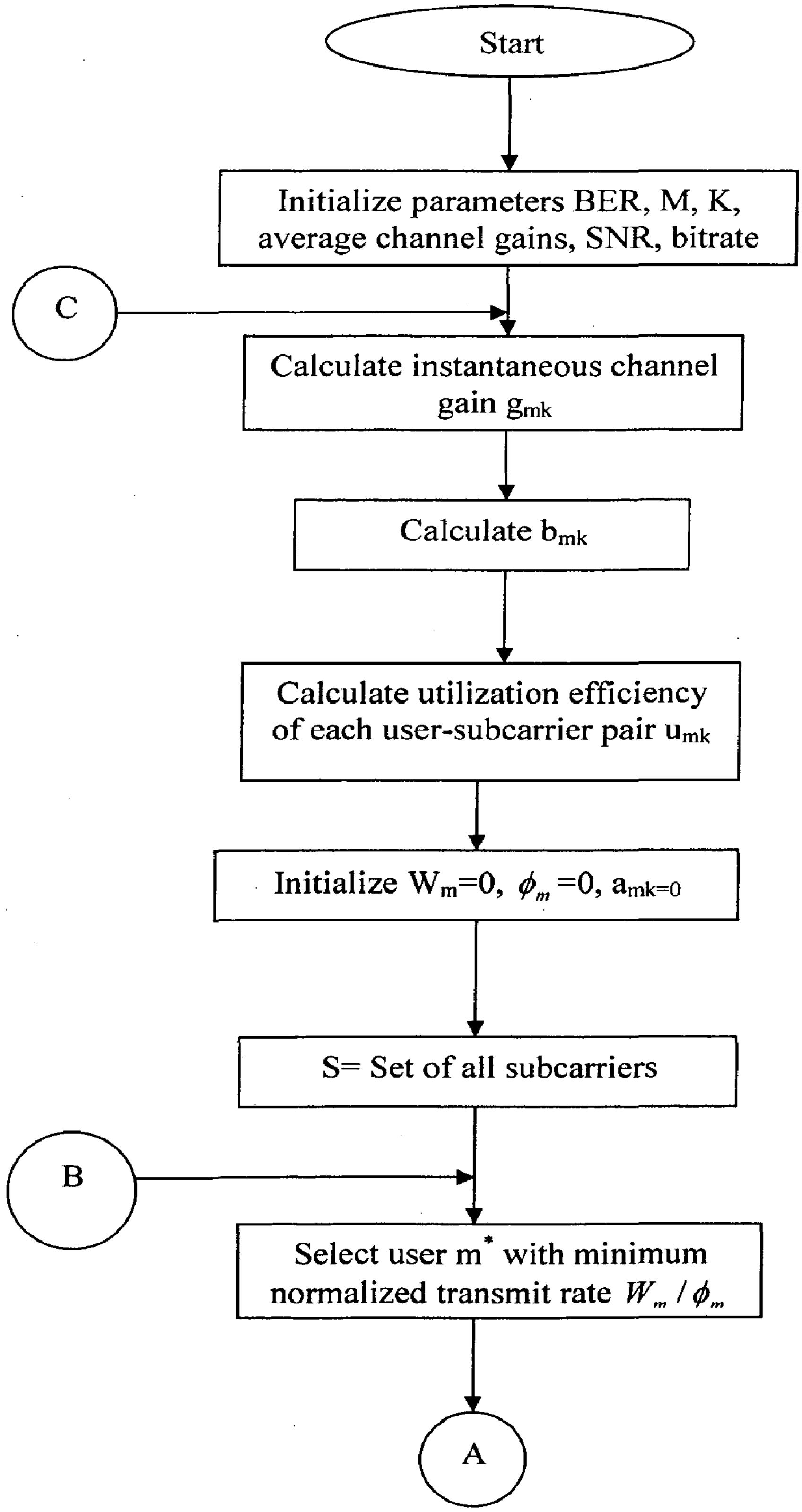
7 modulation schemes, viz. BPSK, QPSK, 8 PSK, 16 QAM, 32 QAM, 64 QAM and 128 QAM, are employed in this simulation. The simulation parameters are given in Table 5.1.

Table 5.1 Simulation Parameters

Parameter	Value
No. of users (M)	5
No. of subcarriers (K)	64
Carrier frequency (f_c)	2 GHz
Bit rate	10^5 bps
SNR (P/σ^2)	20 dB
BER	10^{-3}
Doppler frequency (f_d)	100 Hz

4.1.2 Simulation Process

The simulation methodology used to compute the performance measures is described below in an algorithm form:


Start
Initialize parameters BER, M, K, average channel gains, SNR, bitrate
C
Calculate instantaneous channel gain g_{mk}
Calculate b_{mk}
Calculate utilization efficiency of each user-subcarrier pair u_{mk}
Initialize W_m=0, ϕ_m=0, $a_{mk=0}$
S= Set of all subcarriers
B
Select user m^* with minimum normalized transmit rate W_m / ϕ_m
A

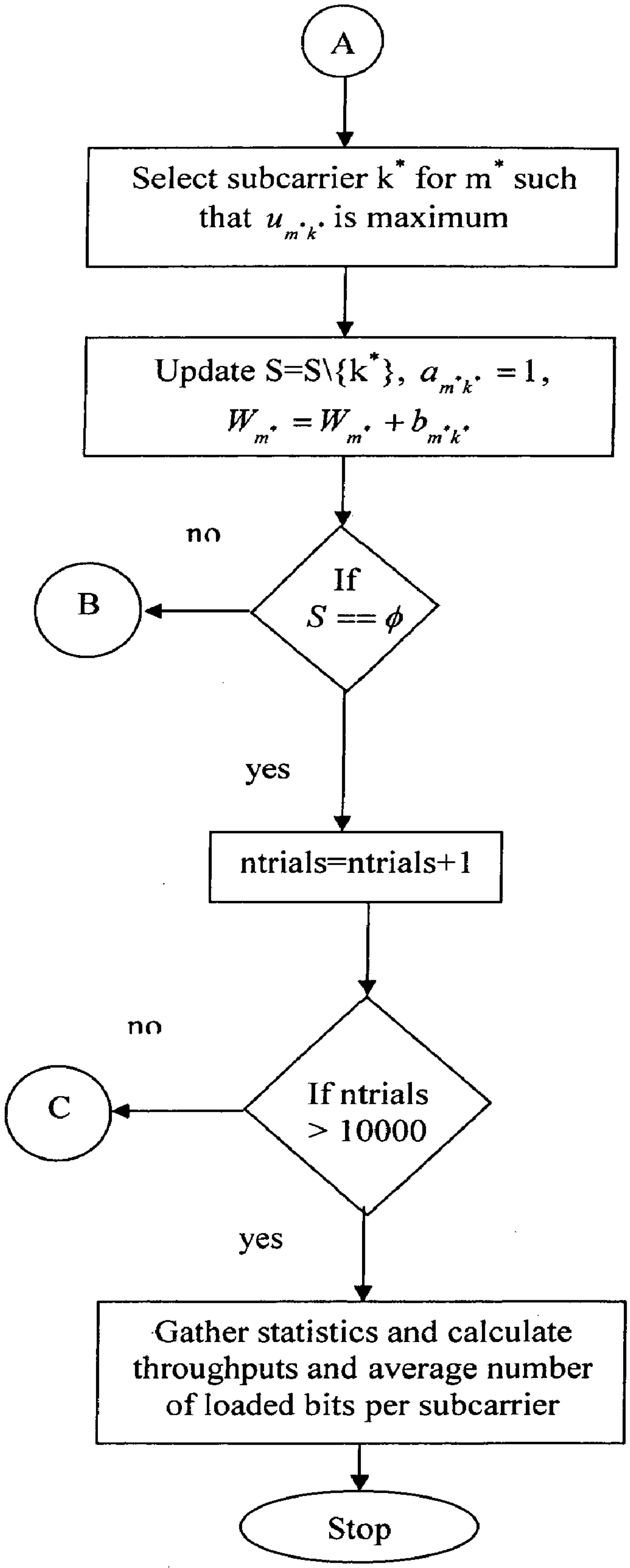


Fig. 4.1 Flow chart for the minimum normalized transmit rate based fair scheduling algorithm

Step 1: Initialize all the parameters: number of users (M) and number of subcarriers (K), carrier frequency (f_c), bit rate, signal to noise ratio (SNR), bit error rate (BER), average channel gains, and number of trials.

Step 2: Calculate the channel gains of each user corresponding to each subcarrier and form a channel gain matrix $[g_{mk}]_{M\times K}$.

Step 3: Calculate the maximum number of bits b_{mk} that can be loaded to each subcarrier k by each user m using equation (3.11).

Step 4: Calculate the utilization efficiency u_{mk} by section 3.4.

Step 5: Apply minimum normalized transmit rate based fair scheduling algorithm to calculate the number of allocated bits to each subcarrier by each user.

Step 6: Gather statistics to measure performance.

The detailed flow chart for the simulation of minimum normalized transmit rate based fair scheduling algorithm is shown in Fig. 4.1.

4.2 Results and Discussion

In the simulation, for the same model, throughputs of minimum normalized transmit rate based fair resource scheduling algorithm (FRS), maximum resource scheduling algorithm (MRS) and random resource scheduling algorithm (RRS) have been measured. For FRS, the average number of loaded bits per subcarrier is also measured first by varying average channel gain of the 5^{th} user keeping its rate weight constant and then by varying rate weight of the 5^{th} user keeping its average channel gain constant. This performance is also measured for different SNR.

Fig. 4.2 shows the throughput of each user in case of FRS, MRS and RRS. The average channel gain and rate weight of 5^{th} user is set to 1. It is clear from the figure that maximum throughput is achieved in case of MRS because it allocates most of the subcarriers to the best user but it is unfair because less throughputs are achieved for the users whose gains are less or who are in deep fade. In case of RRS, although the throughput of each user is same and, therefore, it is a fair algorithm, the throughput is less because it allocates the subcarriers to each user randomly. FRS provides a compromise between the above two algorithms. In this case, throughput of each user is same and it is higher than that in the case of RRS.

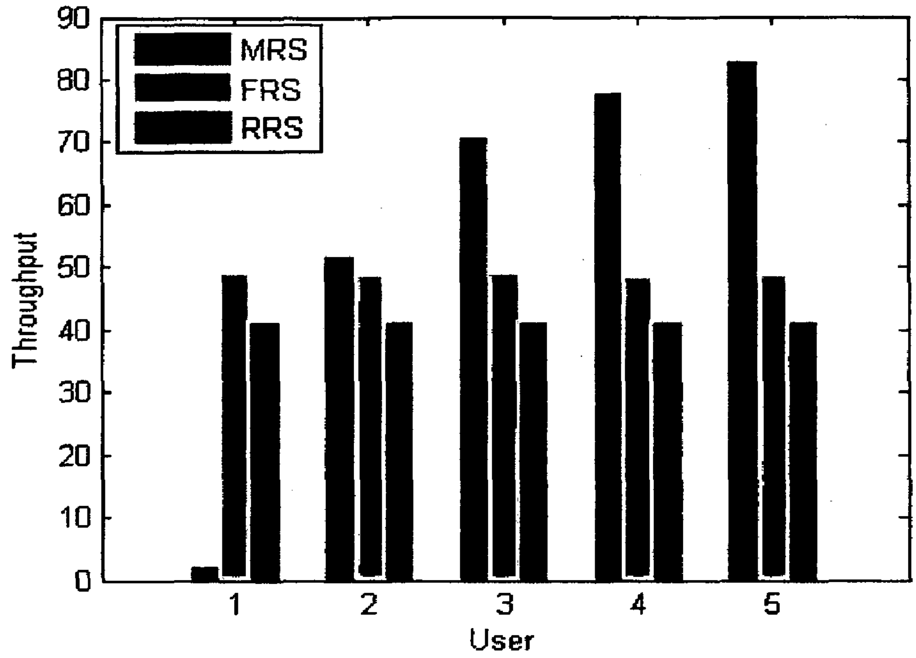


Fig. 4.2 Achieved throughput of each user

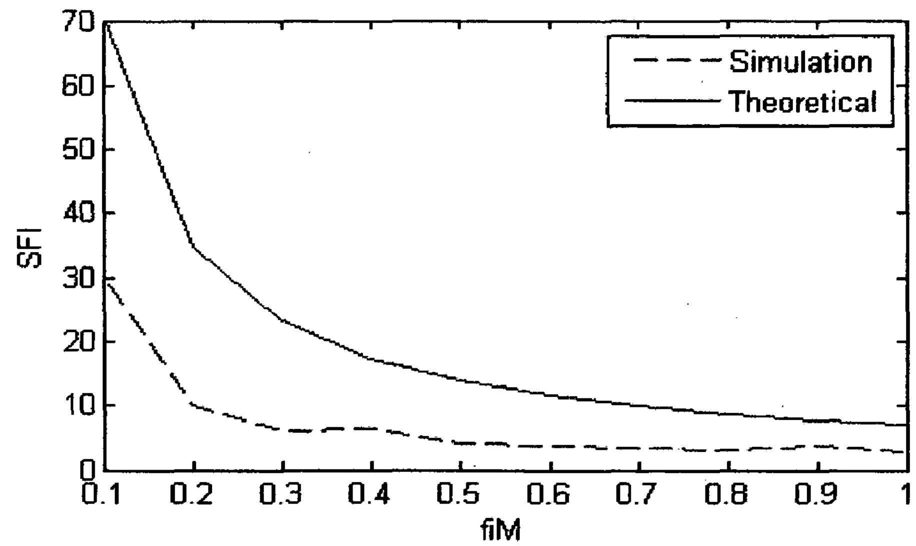


Fig. 4.3 Fairness of FRS algorithm

Fig. 4.3 shows the comparison of service fairness index of FRS and the theoretical bound discussed in section 4.5. It is clear from the figure that the service fairness index is within the theoretical bound as expected.

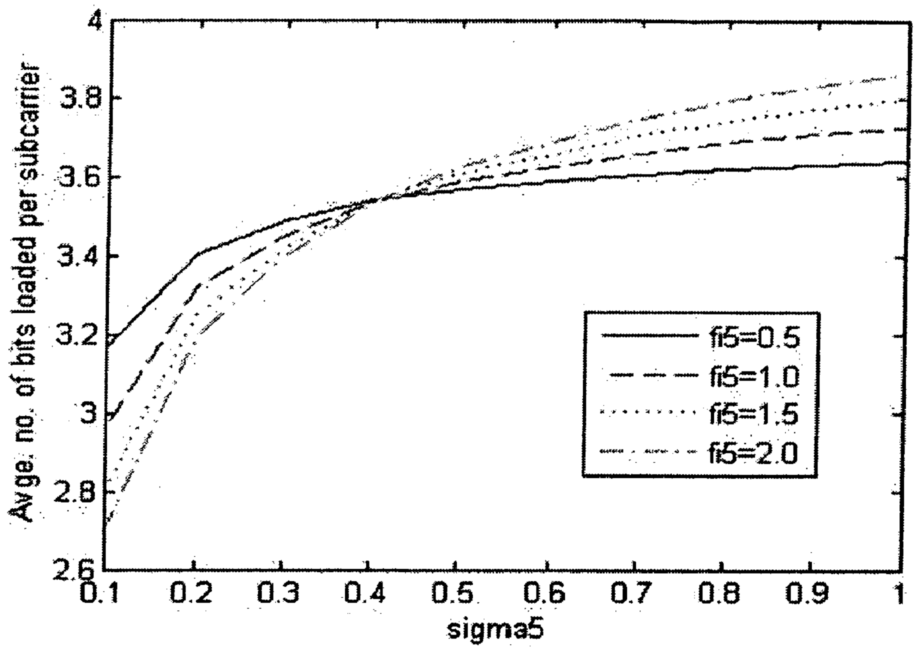


Fig. 4.4 Average loaded bits per subcarrier versus σ_5 for SNR=20dB

Fig. 4.4 shows the relationship between the average number of loaded bits per subcarrier $\bar{b}$ versus average channel gain of the 5th user, σ_5, for different rate weights of the 5th user, ϕ_5. As seen from the figure, $\bar{b}$ increases with the increase in σ_5 because as σ_5 increases more number of bits can be allocated to the subcarriers and hence overall $\bar{b}$ increases. It is also clear that this rate of increase is higher for higher ϕ_5, because if the rate weight is higher then the normalized transmit rate will be less and the probability of selection of the user will be high.

Fig. 4.5 shows the relationship between $\bar{b}$ with ϕ_5 for different values σ_5. It is clear from the figure that when σ_5 is less than 0.4, $\bar{b}$ decreases with increase in ϕ_5. When the user is in bad channel condition and its rate weight increases, more bandwidth is allocated to this user and the bandwidth is wasted so $\bar{b}$ decreases. In contrast, when the average channel gain of the 5th user is greater than 0.4, $\bar{b}$ increases with increase in ϕ_5, because in this case more bandwidth is allocated to the user which has good channel condition. The results obtained through simulation is not exactly matching as the results in [4], but the nature of the results is same.

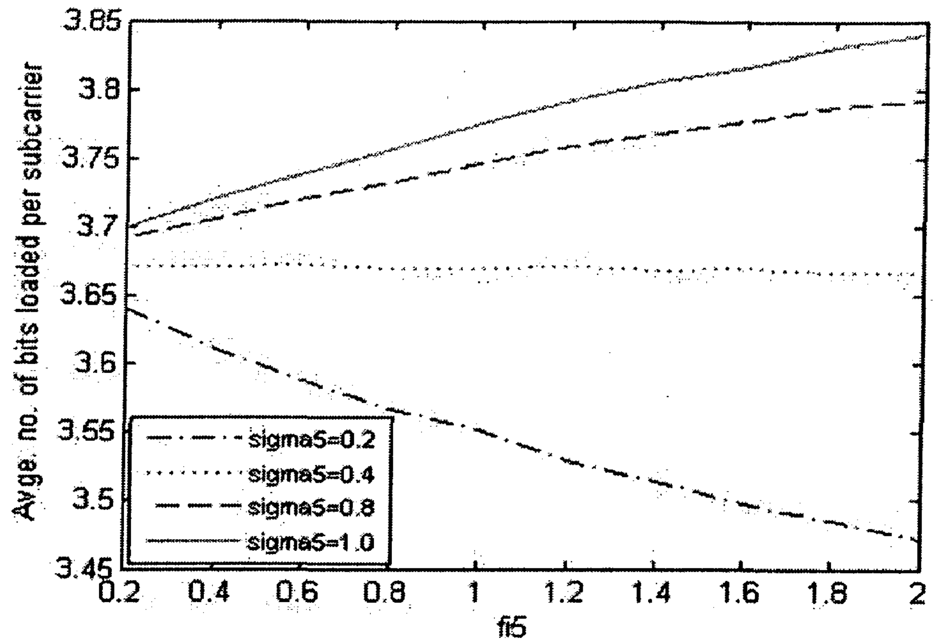


Fig. 4.5 Average loaded bits per subcarrier versus ϕ_5 for SNR=20dB

Fig. 4.6 and Fig. 4.7 show the relationship between $\bar{b}$ and σ_5 for different values of ϕ_5 for SNR =25dB and SNR=30dB respectively. Fig. 4.8 and Fig. 4.9 show the relationship between $\bar{b}$ and σ_5 for different values of ϕ_5 for SNR=25dB and SNR=30dB respectively. Figures show that as the SNR increases, average number of bits loaded per subcarrier increases. Thus, as expected, the performance of the algorithm improves with increasing SNR.

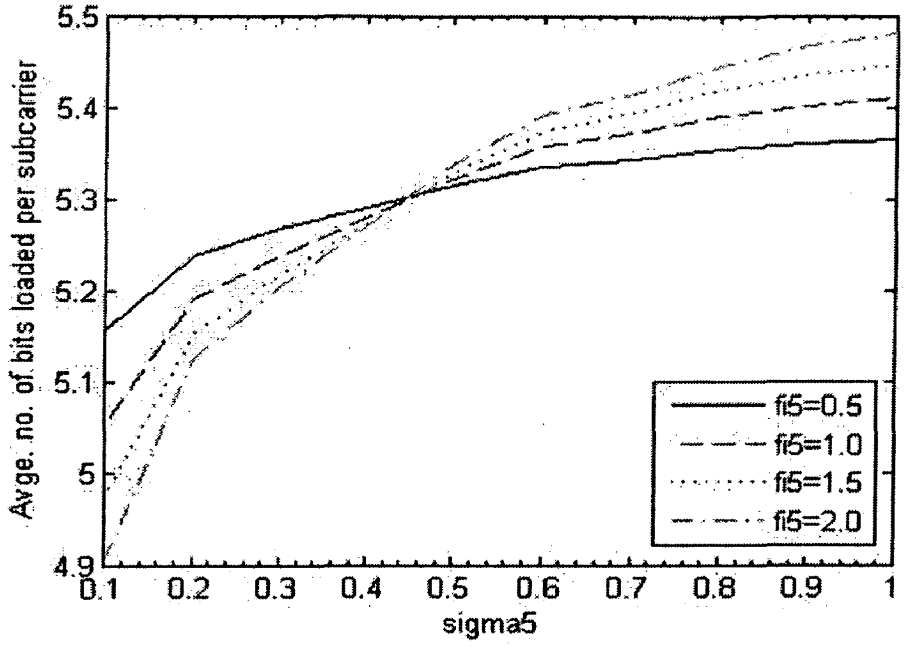


Fig. 4.6 Average loaded bits per subcarrier versus σ_5 for SNR=25dB

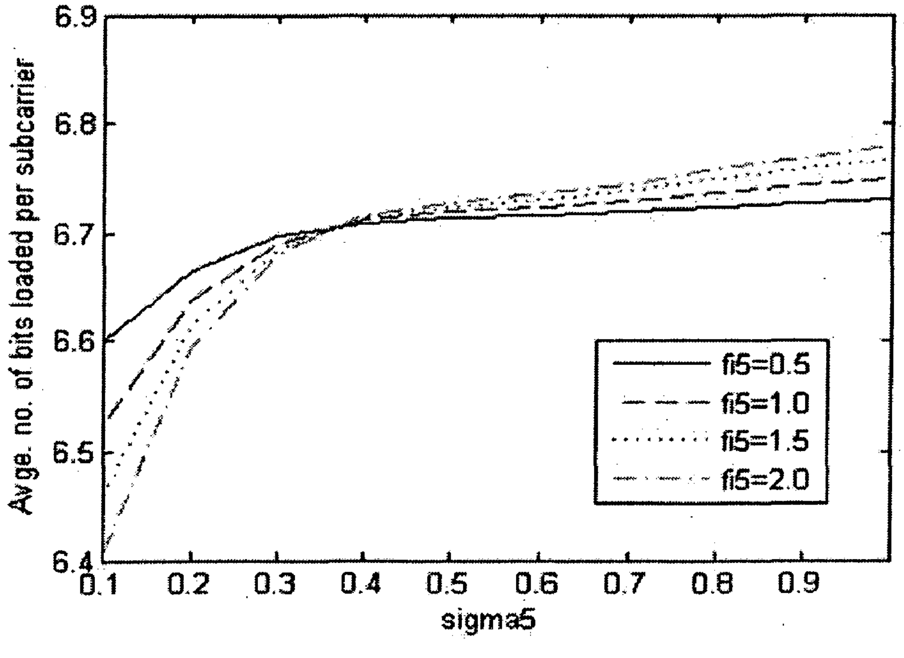


Fig. 4.7 Average loaded bits per subcarrier versus σ_5 for SNR=30dB

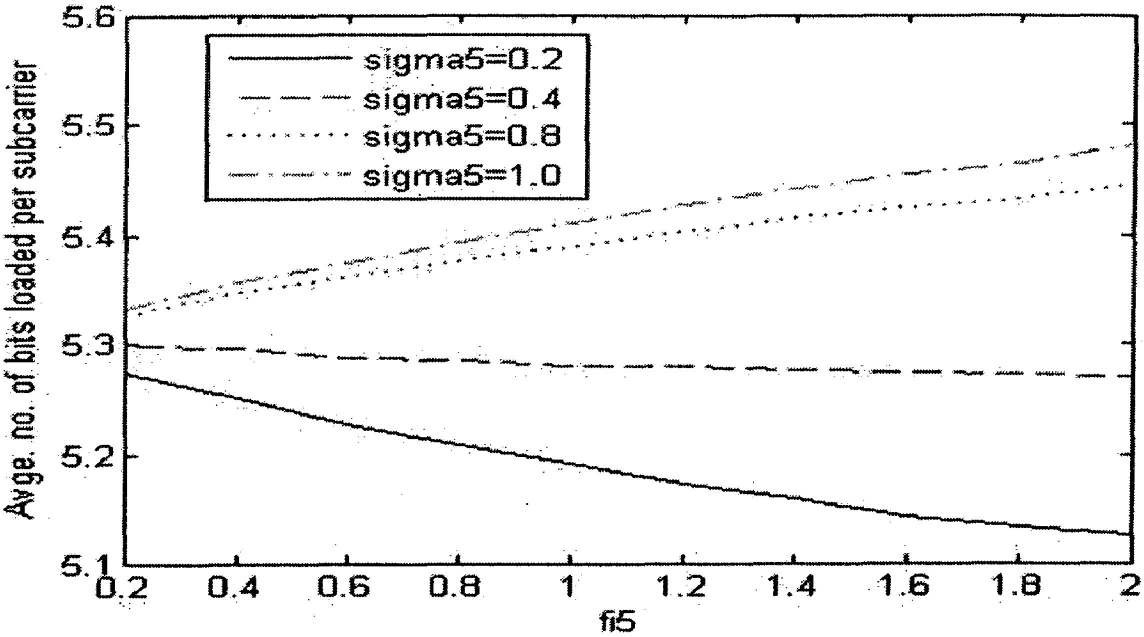


Fig. 4.8 Average loaded bits per subcarrier versus ϕ_5 for SNR=25dB

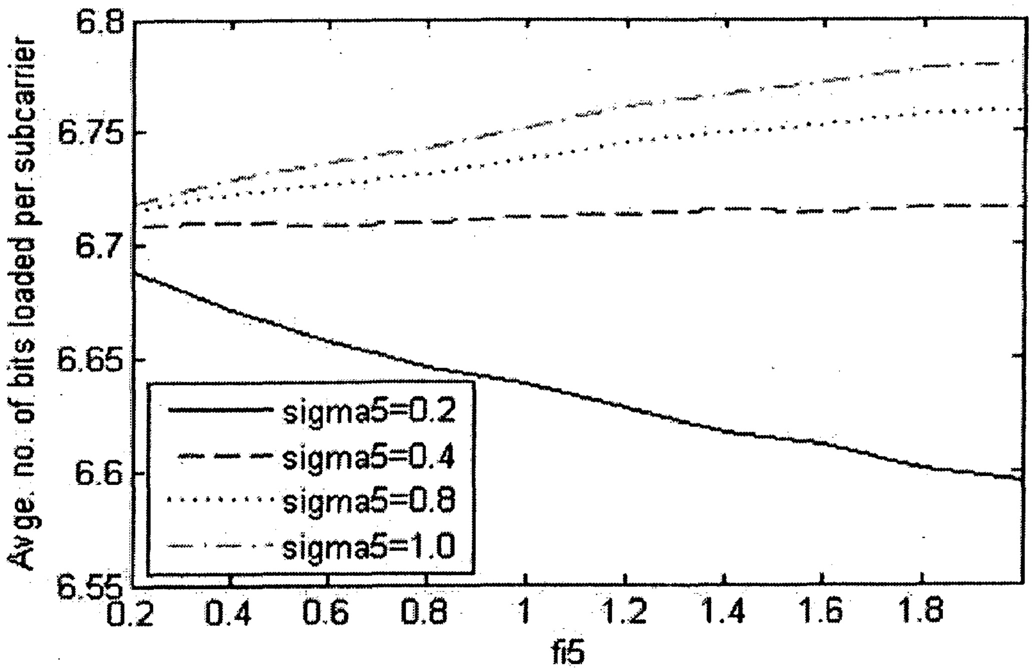


Fig. 4.9 Average loaded bits per subcarrier versus ϕ_5 for SNR=30dB

Chapter 5

CONCLUSION

In this dissertation work, different fair scheduling algorithms have been discussed. Some of these algorithms fulfills the QoS requirements of each user such as BER, bit rate, maximum total transmit power e.g. Iterative Fair Scheduling Algorithm, Heuristic Genetic Algorithm. Some of these algorithms provide equal latency to each user e.g. Proportional Fairness-User Multiplexing Algorithm, Rate-User Multiplexing Algorithm and some algorithms provide equal throughputs to each user e.g. Dynamic Resource Scheduling Algorithm, Multicarrier Proportional Fairness Algorithm, etc. the minimum normalized transmit rate based fair scheduling algorithm is one of the fair scheduling algorithms which provide fairness with respect to throughput. The performance of minimum normalized transmit rate based fair scheduling algorithm has been evaluated through simulation. The performance of this algorithm is compared with that of maximum resource scheduler (MRS) and random resource scheduler (RRS). It is evident from the simulation results that although the fair scheduling algorithm does not provide maximum throughput however the throughput is high enough. It provides fairness among users with respect to throughput.

As a future work, throughput based other fair scheduling algorithms can be compared with minimum normalized transmit rate based fair scheduling algorithm through simulation for the same model.

REFERENCES

1. S. Coleri, M. Ergen, A. Puri, and A. Bahai, "Channel Estimation Techniques Based on Pilot Arrangement in OFDM Systems", in Proc. VTC 2002, vol. 4, p.p. 2449-2453, 2002.

2. S. Barbarossa, M. Pompili, and G. B. Giannakis, "Time and Frequency Synchronization of Orthogonal Frequency Division Multiple Access Systems", in ICC 2001, vol. 6, p.p. 147-151, 2001.

3. H. Fattah, C. Leung, "An Overview of Scheduling Algorithms in Wireless Multimedia Networks", IEEE Wireless Communications, p.p. 76-83, 2002.

4. Y. G. Ding, Z. Zhao-yang, Q. Pei-liang, C. Peng, "Fair Resource Scheduling Algorithm for Wireless OFDM Systems", International Conference on Communications, Circuits and Systems, 2005 Proceedings 2005, vol. 1, p.p. 374-377, 27-30 May 2005.

5. M.Ergen, S.Coleri, and P.Varaiya, "QoS Aware Adaptive Resource Allocation Techniques for Fair Scheduling in OFDMA Based Broadband Wireless Access Systems", in IEEE Transactions On Broadcasting, vol. 49, no. 4, p.p. 362-370, December 2003.

6. E. Zhao, J. Yao, "Data Traffic Fair Scheduling for Multi-user OFDM System Based on Heuristic Genetic Algorithm", International Conference on Computational Intelligence and Security, vol. 2, p.p. 1078-1083, 3-6 Nov., 2006.

7. R.L.Haupt, "An Introduction to Genetic Algorithms for Electromagnetics", IEEE Antennas and Propagation Magazine, vol. 37, Issue 2, p.p. 7 – 15, April 1995.

8. M. Kaneko and P. Popovski, "Heuristic Subcarrier Allocation Algorithms with Multi-Slot Frames in Multi-user OFDM Systems", IEEE International Conference on Communications, vol. 10, p.p. 4391-4396, June 2006.

9. L. Xiao, A. Wang, S. Zhou, Y. Yao, "A Dynamic Resource Scheduling Algorithm for OFDM System", the 9th Asia-Pacific Conference on Communications, vol. 2, p.p. 444-447, 21-24 Sept. 2003.

10. H. Kim, K. Kim, Y. Han and S. Yun, "A Proportional Fair Scheduling for Multicarrier Transmission Systems", 60th Vehicular Technology Conference, vol. 1, p.p. 409-413, 26-29 Sept. 2004.

11. M. Kaneko, P. Popovski, and J. Dahl, "Proportional Fairness in Multi-Carrier System: Upper Bound and Approximation Algorithms", IEEE Communication Letters, vol. 10, no. 6, p.p. 462-464, June 2006.

12. A. R. S. Bahai and B. R. Saltzberg, "Multi-Carrier Digital Communications Theory and Applications of OFDM", Kluwer Academic Publishers, New York, 2002.

13. V. Bharghavan, S Lu, T. Nandagopal. "Fair queuing in wireless networks: issues and approaches", IEEE Personal Communications. vol.6, p.p. 44-53, Feb. 1999.

Appendix

Code for plot between throughput versus users

```
clc;
clear all;
M=5; % No. of users
K=64; %No. of subcarriersHistory = true;
bitrate=1e5;%Bitrate
ntrials=10000;%No. of trials to take average
fc=2e9;%carrier frequency is 2GHz
x=[1];
%Design of Rayleigh Channel for users
c1 = rayleighchan(1/bitrate,100,[0 0.5/bitrate
0.8/bitrate],[-7 -10 -15]);
c1.NormalizePathGains=0;
c1.StoreHistory = true;
c2 = rayleighchan(1/bitrate,100,[0 0.5/bitrate
0.8/bitrate],[-4 -8 -12]);
c2.NormalizePathGains=0;
c2.StoreHistory=true;
c3 = rayleighchan(1/bitrate,100,[0 0.5/bitrate
0.8/bitrate],[-2.2 -6 -9]);
c3.NormalizePathGains=0;
c3.StoreHistory = true;
c4 = rayleighchan(1/bitrate,100,[0 0.5/bitrate
0.8/bitrate],[-0.1 -4 -8]);
c4.NormalizePathGains=0;
c4.StoreHistory = true;
c5 = rayleighchan(1/bitrate,100,[0 0.5/bitrate
0.8/bitrate],[0 -3 -5]);
c5.NormalizePathGains=0;
c5.StoreHistory = true;
```

```
%Initialization of constants
SNR=100;%SNR is 20dB
BER=0.001;
gamma=-log(5*BER)/1.5;
%Rate weights for the four users is 1
for m=1:M
    fi(m)=1;
    userb(m)=0;%No. of bits to each user
    sum(m)=0;
    sumrrs(m)=0;
    sumfrs(m)=0;
end
for l=1:ntrials
    userb=[0 0 0 0 0];
    %Generation of channel gaian matrcix
    for k=1:K
        y1=filter(c1,x);
        y2=filter(c1,x);
        y3=filter(c1,x);
        y4=filter(c1,x);
        y5=filter(c1,x);

        g(1,k)=abs(c1.PathGains(1,1)+c1.PathGains(1,2)+c1.Pa
        thGains(1,3));

        g(2,k)=abs(c2.PathGains(1,1)+c2.PathGains(1,2)+c2.Pa
        thGains(1,3));

        g(3,k)=abs(c3.PathGains(1,1)+c3.PathGains(1,2)+c3.Pa
        thGains(1,3));
```

```
    g(4,k)=abs(c4.PathGains(1,1)+c4.PathGains(1,2)+c4.Pa
    thGains(1,3));

    g(5,k)=abs(c5.PathGains(1,1)+c4.PathGains(1,2)+c4.Pa
    thGains(1,3));
end
%Allocation of bits on each subcarrier by all users
    for m=1:M
        for k=1:K
            var(m,k)=log2(1+g(m,k)*SNR/gamma);
            b(m,k)=var(m,k)-mod(var(m,k),1);
            if b(m,k)>7
               b(m,k)=7;
            end
            userb(m)=userb(m)+b(m,k);
        end
    end
    %Algorithm for Maximum Resource  Schedulre MRS
    max=userb(1);
    index=1;
    for i=1:M
        if userb(i)>max;
           max=userb(i);
           index=i;
        end
    end
    sum(index)=sum(index)+userb(index);
    %Algorithm for Randam Resource Scheduler (RRS)
    for k=1:64
        r = ceil(5.*rand(1,1));
        sumrrs(r)=sumrrs(r)+b(r,k);
```

```
 end
%Algorithm for Fair Resource Scheduler (FRS)
 for k=1:K
      sumb(k)=0;
      for m=1:M
           sumb(k)=sumb(k)+b(m,k);
      end
 end
 for m=1:M
      for k=1:K
           if sumb(k)==0
               u(m,k)=0;
           else
               u(m,k)=b(m,k)/sumb(k);
           end
      end
end
count=K;
for m=1:M
     for k=1:K
          a.(m,k)=0;
     end
     w(m)=0;
end
for count=K:-1:1
mstar=1;
     for m=1:M
          if (w(mstar)/fi(mstar))>(w(m)/fi(m))
               mstar=m;
          end
     end
     kstar=1;
```

```
    for k=1:K
        for n=1:K-count
                if k==S(n)
                   continue;
                end
            end
            if u(mstar,kstar)<u(mstar,k)
                kstar=k;
            end
        end
        sumfrs(mstar)=sumfrs(mstar)+b(mstar,kstar);
        S(K-count+1)=kstar;
        a(mstar,kstar)=1;
        w(mstar)=w(mstar) + b(mstar,kstar);
        count=count-1;
    end
end
sum=sum/ntrials;
sumrrs=sumrrs/ntrials;
sumfrs=sumfrs/ntrials;
sum
ind=1:5;
for m=1:5
    com(m,1)=sum(m);
    com(m,2)=sumfrs(m);
    com(m,3)=sumrrs(m);
end
bar(com);
xlabel('User');
ylabel('Throughput');
```

Code for plot showing the SFI of the algorithm

```
clc;
clear all;
M=5; % No. of users
K=64; %No. of subcarriers
bitrate=1e5;%Bitrate
ntrials=10000;%No. of trials to take average
x=[1];
%Design of Rayleigh Channel for the four users
c1 = rayleighchan(1/bitrate,100,[0 0.5/bitrate
0.8/bitrate],[-7 -10 -15]);
c1.NormalizePathGains=0;
c1.StoreHistory = true;
c2 = rayleighchan(1/bitrate,100,[0 0.5/bitrate
0.8/bitrate],[-4 -8 -12]);
c2.NormalizePathGains=0;
c2.StoreHistory = true;
c3 = rayleighchan(1/bitrate,100,[0 0.5/bitrate
0.8/bitrate],[-2.2 -6 -9]);
c3.NormalizePathGains=0;
c3.StoreHistory = true;
c4 = rayleighchan(1/bitrate,100,[0 0.5/bitrate
0.8/bitrate],[-0.1 -4 -8]);
c4.NormalizePathGains=0;
c4.StoreHistory = true;
c5 = rayleighchan(1/bitrate,100,[0 0.5/bitrate
0.8/bitrate],[0 -3 -5]);
c5.NormalizePathGains=0;
c5.StoreHistory = true;
%Initialization of constants
```

```
SNR=100;
BER=0.001;
gamma=-log(5*BER)/1.5;
%Rate weights for the four users is 1
for m=1:M-1
    fi(m)=1;
end
fiM=0.1:0.1:1.0;%Rate weight of the 5th user
for i=1:10
    fi(M)=fiM(i);
    sfi=0;
    for l=1:ntrials
        %Generation of channel gain matrix
        for k=1:K
            y1=filter(c1,x);
            y2=filter(c1,x);
            y3=filter(c1,x);
            y4=filter(c1,x);
            y5=filter(c1,x);

          g(1,k)=abs(c1.PathGains(1,1)+c1.PathGains(1,2)+c1
          .PathGains(1,3));

          g(2,k)=abs(c2.PathGains(1,1)+c2.PathGains(1,2)+c2
          .PathGains(1,3));

          g(3,k)=abs(c3.PathGains(1,1)+c3.PathGains(1,2)+c3
          .PathGains(1,3));

          g(4,k)=abs(c4.PathGains(1,1)+c4.PathGains(1,2)+c4
          .PathGains(1,3));
```

```
  g(5,k)=abs(c5.PathGains(1,1)+c4.PathGains(1,2)+c4
  .PathGains(1,3));
end
%Allocation of bits on each subcarrier by all users
for m=1:M
    for k=1:K
        var(m,k)=log2(1+g(m,k)*SNR/gamma);
        b(m,k)=var(m,k)-mod(var(m,k),1);
        if b(m,k)>7
            b(m,k)=7;
        end
    end
end
for k=1:K
    sumb(k)=0;
    for m=1:M
        sumb(k)=sumb(k)+b(m,k);
    end
end
%Calculation of utilization efficiency
for m=1:M
    for k=1:K
        if sumb(k)==0
            u(m,k)=0;
        else
            u(m,k)=b(m,k)/sumb(k);
        end
    end
end
count=K;
for m=1:M
```

```
        for k=1:K
            a(m,k)=0;
        end
        w(m)=0;
end
for count=K:-1:1
    mstar=1;
    for m=1:M
        if (w(mstar)/fi(mstar))>(w(m)/fi(m))
            mstar=m;
        end
    end
    kstar=1;
    for k=1:K
        for n=1:K-count
            if k==S(n)
                continue;
            end
        end
        if u(mstar,kstar)<u(mstar,k)
            kstar=k;
        end
    end
    S(K-count+1)=kstar;
    a(mstar,kstar)=1;
    w(mstar)=w(mstar) + b(mstar,kstar);
    count=count-1;
end
wmax=w(1);
wmin=w(1);
indmax=1;
indmin=1;
```

```
        for m=1:M
            if (w(m)/fi(m))>(w(indmax)/fi(indmax))
               indmax=m;
            end
            if (w(m)/fi(m))<(w(indmin)/fi(indmin))
               indmin=m;
            end
        end
        sfi=sfi + w(indmax)/fi(indmax)-
       w(indmin)/fi(indmin);
    end
    sfisim(i)=sfi/ntrials;
    bmax=7;
    min=1;
    for m=1:M
        if fi(m)<fi(min)
           min=m;
        end
    end
    sfitheo(i) = bmax/fi(min);
end
plot(fiM,sfisim,fiM,sfitheo);
xlabel('fi5');
ylabel('SFI');
legend('Simulation','Theoretical');
```

Code for plot between average number of loaded bits per subcarrier versus average channel gain

```
clc;
clear all;
M=5; % No. of users
K=64; %No. of subcarriers
bitrate=1e5;%Bitrate
ntrials=10000;%No. of trials to take average
x=[1];
%Design of Rayleigh Channel for the four users
c1 = rayleighchan(1/bitrate,100,[0 0.5/bitrate
0.8/bitrate],[-7 -10 -15]);
c1.NormalizePathGains=0;
c1.StoreHistory = true;
c2 = rayleighchan(1/bitrate,100,[0 0.5/bitrate
0.8/bitrate],[-4 -8 -12]);
c2.NormalizePathGains=0;
c2.StoreHistory = true;
c3 = rayleighchan(1/bitrate,100,[0 0.5/bitrate
0.8/bitrate],[-2.2 -6 -9]);
c3.NormalizePathGains=0;
c3.StoreHistory = true;
c4 = rayleighchan(1/bitrate,100,[0 0.5/bitrate
0.8/bitrate],[-0.1 -4 -8]);
c4.NormalizePathGains=0;
c4.StoreHistory = true;
%Initialization of constants
SNR=100;%Signal to noise ratio 20dB
BER=0.001;
gamma=-log(5*BER)/1.5;
%Rate weights for the four users is 1
```

```
for m=1:M-1
    fi(m)=1;
end
sigmaM=0.1:0.1:1.0;%Avge channel gain for the 5th user
pd=10*log10(sigmaM);
for t=0.5:0.5:2.0 %Rate weight of the 5th user
    fi(M)=t;
    for i=1:10
        sum(i)=0;
        %Design of Rayleigh Channel for the fifth user
        c5 = rayleighchan(1/bitrate,100,[0 0.5/bitrate
         0.8/bitrate],[pd(i) -3+pd(i) -9+pd(i)]);
        c5.NormalizePathGains=0;
        c5.StoreHistory = true;
        %Generating the channel gain matrix g(m,k)
        for l=1:ntrials
            for k=1:K
                y1=filter(c1,x);
                y2=filter(c2,x);
                y3=filter(c3,x);
                y4=filter(c4,x);
                y5=filter(c5,x);

              g(1,k)=abs(c1.PathGains(1,1)+c1.PathGains(1,2)
              +c1.PathGains(1,3));

              g(2,k)=abs(c2.PathGains(1,1)+c2.PathGains(1,2)
              +c2.PathGains(1,3));

              g(3,k)=abs(c3.PathGains(1,1)+c3.PathGains(1,2)
              +c3.PathGains(1,3));
```

```
  g(4,k)=abs(c4.PathGains(1,1)+c4.PathGains(1,2)
  +c4.PathGains(1,3));

  g(5,k)=abs(c5.PathGains(1,1)+c5.PathGains(1,2)
  +c5.PathGains(1,3));

end
%Allocation of bits on each subcarrier by all
users
for m=1:M
    for k=1:K
        var(m,k)=log2(1+g(m,k)*SNR/gamma);
        b(m,k)=var(m,k)-mod(var(m,k),1);
        if b(m,k)>7
            b(m,k)=7;
        end
    end
end
for k=1:K
    sumb(k)=0;
    for m=1:M
        sumb(k)=sumb(k)+b(m,k);
    end
end
%Calculating the utiliztion efficiency
for m=1:M
    for k=1:K
        if sumb(k)==0
            u(m,k)=0;
        else
            u(m,k)=b(m,k)/sumb(k);
```

```
            end
        end
    end
    %Start of algorithm
    count=K;
    for m=1:M
        for k=1:K
            a(m,k)=0;
        end
        w(m)=0;
    end
    for count=K:-1:1
        mstar=1;
        for m=1:M
            if (w(mstar)/fi(mstar))>(w(m)/fi(m))
                mstar=m;
            end
        end
        kstar=1;
        for k=1:K
            for n=1:K-count
                if k==S(n)
                    continue;
                end
            end
            if u(mstar,kstar)<u(mstar,k)
                kstar=k;
            end
        end
        S(K-count+1)=kstar;
        a(mstar,kstar)=1;
        w(mstar)=w(mstar) + b(mstar,kstar);
```

```
                    sum(i) = sum(i) + b(mstar,kstar);
                    count=count-1;
                end
                %End of algorithm
            end
            avge(i)=sum(i)/(K*ntrials);
        end
        switch(t)
            case 0.5,
                av1=avge;
            case 1.0,
                av2=avge;
            case 1.5,
                av3=avge;
            case 2.0,
                av4=avge;
        end
end
plot(sigmaM,av1,sigmaM,av2,sigmaM,av3,sigmaM,av4);
xlabel('sigma5');
ylabel('Avge. no. of bits loaded per subcarrier');
legend('fi5=0.5','fi5=1.0','fi5=1.5','fi5=2.0');
```

Code for plot between average number of loaded bits per subcarrier versus normalized transmit rate

```
clc;
clear all;
M=5; % No. of users
K=64; %No. of subcarriers
bitrate=1e5;%Bitrate
fc=2e9;%Carrier Frequency
```

```
ntrials=10000;%No. of trials to take average
x=[1];
%Design of Rayleigh Channel for the four users
c1 = rayleighchan(1/bitrate,100,[0 0.5/bitrate
0.8/bitrate],[-7 -10 -15]);
c1.NormalizePathGains=0;
c1.StoreHistory = true;
c2 = rayleighchan(1/bitrate,100,[0 0.5/bitrate
0.8/bitrate],[-4 -8 -12]);
c2.NormalizePathGains=0;
c2.StoreHistory = true;
c3 = rayleighchan(1/bitrate,100,[0 0.5/bitrate
0.8/bitrate],[-2.2 -6 -9]);
c3.NormalizePathGains=0;
c3.StoreHistory = true;
c4 = rayleighchan(1/bitrate,100,[0 0.5/bitrate
0.8/bitrate],[-0.1 -4 -8]);
c4.NormalizePathGains=0;
c4.StoreHistory = true;
%Initialization of constants
SNR=100;
BER=0.001;
gamma=-log(5*BER)/1.5;
%Rate weights for the four users is 1
for m=1:M-1
    fi(m)=1;
end
fiM=0.2:0.2:2.0;%Avge channel gain for the 5th user
for t=0.2:0.2:1.0 %Rate weight of the 5th user
    sigmaM=t;
    pd=10*log10(sigmaM);
    for i=1:10
```

```
fi(M)=fiM(i);
sum(i)=0;
%Design of Rayleigh Channel for the fifth user
c5 = rayleighchan(1/bitrate,100,[0 0.5/bitrate
 0.8/bitrate],[pd -3+pd -9+pd]);
c5.NormalizePathGains=0;
c5.StoreHistory = true;
y5=filter(c5,x);
%Generation of channel matrix
for l=1:ntrials
    for k=1:K
        y1=filter(c1,x);
        y2=filter(c1,x);
        y3=filter(c1,x);
        y4=filter(c1,x);
        y5=filter(c1,x);

      g(1,k)=abs(c1.PathGains(1,1)+c1.PathGains(1,2)
      +c1.PathGains(1,3));

      g(2,k)=abs(c2.PathGains(1,1)+c2.PathGains(1,2)
      +c2.PathGains(1,3));

      g(3,k)=abs(c3.PathGains(1,1)+c3.PathGains(1,2)
      +c3.PathGains(1,3));

      g(4,k)=abs(c4.PathGains(1,1)+c4.PathGains(1,2)
      +c4.PathGains(1,3));

      g(5,k)=abs(c5.PathGains(1,1)+c4.PathGains(1,2)
      +c4.PathGains(1,3));
    end
```

```
%Allocation of bits on each subcarrier by all
users
for m=1:M
    for k=1:K
        var(m,k)=log2(1+g(m,k)*SNR/gamma);
        b(m,k)=var(m,k)-mod(var(m,k),1);
        if b(m,k)>7
            b(m,k)=7;
        end
    end
end
for k=1:K
    sumb(k)=0;
    for m=1:M
        sumb(k)=sumb(k)+b(m,k);
    end
end
%Calculation of utilization efficiency
for m=1:M
    for k=1:K
        if sumb(k)==0
            u(m,k)=0;
        else
            u(m,k)=b(m,k)/sumb(k);
        end
    end
end
%Start of algorithm
count=K;
for m=1:M
    for k=1:K
        a(m,k)=0;
```

```
            end
            w(m)=0;
        end
        for count=K:-1:1
            mstar=1;
            for m=1:M
                if (w(mstar)/fi(mstar))>(w(m)/fi(m))
                    mstar=m;
                end
            end
            kstar=1;
            for k=1:K
                for n=1:K-count
                    if k==S(n)
                        continue;
                    end
                end
                if u(mstar,kstar)<u(mstar,k)
                    kstar=k;
                end
            end
            S(K-count+1)=kstar;
            a(mstar,kstar)=1;
            w(mstar)=w(mstar) + b(mstar,kstar);
            sum(i) = sum(i) + b(mstar,kstar);
            count=count-1;
        end
        %End of algorithm
    end
    avge(i)=sum(i)/(K*ntrials);
end
switch(t)
```

```
            case 0.2,
                 av1=avge;
            case 0.4,
                 av2=avge;
            case 0.8,
                 av4=avge;
            case 1.0,
                 av5=avge;
    end
end
plot(fiM,av1,fiM,av2,fiM,av4,fiM,av5);
xlabel('fi5');
ylabel('Avge. no. of bits loaded per subcarrier');
legend('sigma5=0.2','sigma5=0.4','sigma5=0.8','sigma5=1.0')
;
```