Technical University of Crete

Department of Electronic and Computer Engineering

M.Sc

Design and Performance Evaluation of Sensing Algorithms and Cooperative Relay Selection Protocols for Multichannel Cognitive Radio Networks

Maria Theodorou

Supervisor:	**Professor Michael Paterakis**
Guidance / Examining Committee:	**Professor Athanasios Liavas**
	Assoc. Professor Polychronis Koutsakis

Chania, 2015

Ευχαριστίες

Αρχικά θέλω να ευχαριστήσω θερμά τον επιβλέποντα της μεταπτυχιακής έρευνας καθηγητή κ. Μιχάλη Πατεράκη για τη συνεργασία μας καθόλη τη διάρκεια εκπόνησής της. Πιο συγκεκριμένα, τον ευχαριστώ για τη βοήθεια που μου παρείχε κατά την πρώτη μου επαφή με θέματα γνωστικών ραδιοδικτύων. Με τις συμβουλές, το ενδιαφέρον, τις γνώσεις και την πολύτιμη καθοδήγησή του κατάφερα να υλοποιήσω επιτυχώς τη μεταπτυχιακή έρευνα. Επίσης θέλω να ευχαριστήσω τα μέλη της εξεταστικής επιτροπής κ. Λιάβα και κ. Κουτσάκη. Επιπλέον θα ήθελα να ευχαριστήσω όλους τους φίλους μου που πίστεψαν σε μένα και με ενθάρρυναν σε κάθε στάδιο των σπουδών. Τέλος θέλω να ευχαριστήσω την οικογένειά μου, που στηρίζει πάντα τις επιλογές μου και μου δίνει δύναμη να προχωρώ.

Acknowledgment

This research has been co-financed by the European Union (European Social Fund – ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) - Research Funding Program: Thales, Investing in Knowledge Society through the European Social Fund.

Also the research of Ms. Maria Theodorou has been partially supported by the Project “autoNomous, self-Lerning, OPTImal and compLete Underwater Systems” (NOPTILUS) of the Seventh Framework Programme for Research of the European Commission.

Abstract

Recent years have witnessed a dramatic increase in the demand for radio spectrum. This is partly due to the increasing interest of consumers in wireless services, which in turn is driving the evolution of wireless networks toward high-speed data networks. Cognitive radio has been proposed as a promising technology to improve the spectral efficiency of radio spectrum, and is achieved by allowing unlicensed secondary users (SUs) to coexist with licensed primary users (PUs) in the same spectrum. The primary network owns the spectrum, and has performance guarantees. The secondary network(s) can access the spectrum if no significant degradation on the primary communication is caused.

In this Thesis we start in Chapter 2 by proposing four new transmission algorithms for multichannel homogeneous cognitive radio networks (CRNs). We examine two cases: **(i)** the case where the network's channels are not assigned to the SUs by a centralized entity and **(ii)** the case where a centralized entity exists and assigns the network's channels to the SUs. Our event-driven simulations results demonstrate that the new transmission algorithms we have introduced improve **(i)** the normalized average throughput of SUs, **(ii)** reduce the dropping probability and **(iii)** increase the number of successful transmissions occurring during the system operation, when compared with a popular algorithm proposed in recent work in this area.

Chapter 3 of the Thesis studies new transmission algorithms for multichannel heterogeneous CRNs. As in the first part we examine two cases: **(i)** a distributed CRN and **(ii)** a centralized CRN. For each case and for the same network topology, as in the first part of the work, we propose a new algorithm. Our event-driven simulations results demonstrate that the new transmission algorithms we have introduced considerably improve the average number of Mbits of secondary user traffic transmitted in each time slot, when compared with the corresponding results of the *"γ-persistent strategy"* recently introduced in the literature.

In Chapter 4 of the Thesis the *"Distributed algorithm"* proposed in Chapter 3, in which the SUs select their network's channels in a distributed way without coordination by a centralized entity, is used and evaluated in the case of homogeneous multichannel CRNs. Our simulation results demonstrate that the *"Distributed algorithm"* achieves results close to those achieved by the algorithms proposed in Chapter 2 of the Thesis in which it has been assumed that a centralized entity exists and assigns the network's channels to the SUs.

Finally, in Chapter 5 of the Thesis new cooperative communication protocols are proposed for cognitive radio networks, in which one primary user and multiple SUs cooperate for mutual benefit. We proposed and evaluate two new protocols, the Best Relay Selection Protocol (*BRSP*) and the Stopping Criterion Protocol (*SCP*) which allow cooperation between the PU and the SUs. Our simulation results demonstrate that the proposed protocols decrease the total primary packet transmission time, compared with the time required for direct packet transmission by the PU.

Contents

List of figures

List of Tables

Chapter 1: Introduction

1.1 Introduction to Cognitive Radio Networks

Cognitive radio (CR) technology is envisaged to solve the problems in wireless networks resulting from the limited available spectrum and the inefficiency in the spectrum usage by exploiting the existing wireless spectrum opportunistically [12]. The Federal Communication Commission (FCC) estimates [11] that the variation of use of licensed spectrum ranges from 15% to 85%, whereas according to the Defense Advance Research Projects Agency (DARPA) only 2% of the spectrum is in use in USA at any given moment. It is then clear that the solution to these problems can be found dynamically looking at the spectrum as a function of time and space. This is the base of Cognitive Radios: the paradigm, defined the first time by J. Mitola [10], foresees devices able not only to adapt themselves to spectrum environment and, in general, to external environments, but also to learn from experience, as a biological cognitive process, how to carry out this adaptation [9]. In cognitive radio, new spectrum allocation policies are used, which allow unlicensed users (secondary users - SUs) to opportunistically exploit the spectrum owning to licensed users (primary users - PUs), when the spectrum it is not occupied by the PUs. TV broadcasters, public safety users, cellular operators and point-to-point microwave links are examples of the PUs [20]. Cognitive radio will improve spectrum utilization in wireless communication systems while accommodating the increasing amount of services and applications in wireless networks [14]. The main characteristic of the cognitive ratio is the cognitive capability. The above characteristic refers to the ability of the radio technology to sense the information from its radio environment. Through this capability, the portions of the spectrum that are unused at a specific time or location can be identified. Consequently, the best spectrum and appropriate operating parameters can be selected. The ultimate objective of the cognitive radio is to obtain the best available spectrum through cognitive capability and enable the usage of temporarily unused spectrum. Despite the positive aspects, a cognitive radio network has to face with some novel spectrum management functionalities such as: spectrum sensing, spectrum sharing, spectrum decision and spectrum mobility.

1.2 Main features of spectrum management

As already mentioned in the introduction, a cognitive ratio has new spectrum management functionalities such as: spectrum sensing, spectrum decision, spectrum sharing, and spectrum mobility. Spectrum sensing refers to the procedure in which the SU monitors the available spectrum bands and detects the spectrum holes. When the above procedure is completed, the spectrum decision is invoked. The SU has to select the appropriate band, depending on PU activity, in order to exploit the spectrum for as long as possible. Spectrum sharing provides the capability to share the spectrum resource opportunistically with multiple SUs which includes resource allocation to avoid interference caused to the PUs [12]. Since there may be multiple SUs trying to exploit the same bands of the spectrum, the network has to distribute the SUs among the available bands, to prevent collisions. Due to the fact that the SU is unlicensed, should vacate the spectrum immediately if a PU is detected. The secondary transmission may continue in another vacant portion of the spectrum. Thus, spectrum mobility necessitates a spectrum handoff scheme to detect the link failure and to switch the current transmission to a new spectrum band with minimum quality degradation [12].

1.2.1 Spectrum sensing for cognitive radio

In this section we present the functionalities of spectrum sensing such as PU detection, and cooperation. In case of cooperation the observed information in each SU is exchanged with its neighbors so as to improve sensing accuracy. In case of PU detection, the SU should detect the presence or not of the PU in order to identify the available spectrum. The most important parameters impacting the performance of spectrum sensing are the time available to sense the transmission channels and the strength of the primary signals [19]. Two parameters are generally used to measure the performance of spectrum sensing, namely, detection probability (the probability of detecting the activity of a primary user when the primary user is active) and false alarm probability (the probability of mistakenly claiming that a primary user is active when the primary user is actually idle).

Generally speaking, the duration of the spectrum-sensing part in a slot determines the accuracy of spectrum sensing. A longer spectrum-sensing duration leads to a higher detection probability and a lower false alarm probability. However, it also means less time in the slot for the data-transmission part [16]. The most widely used technique for primary user detection is the transmitter detection. Three schemes can be used for the transmitter detection in spectrum sensing: matched filter detection, energy detection and feature detection [12]. These schemes are shortly presented below.

1. Matched filter detection

The matched filter is the linear optimal filter used for coherent signal detection to maximize the signal-to-noise ratio (SNR) in the presence of additive stochastic noise. It is obtained by correlating a known original PU signal $s(t)$ with a received signal $r(t)$. Then the output of the matched filter is sampled at the synchronized timing. If the sampled value is greater than a threshold λ, the spectrum is determined to be occupied by the PU transmission. The matched filter necessitates not only a priori knowledge of the characteristics of the PU signal but also the synchronization between the PU transmitter and the SU. If this information is not accurate, then the matched filter performs poorly. Furthermore, a SU needs to have different multiple matched filters dedicated to each type of the PU signal, which increases the implementation cost and complexity.

2. Energy detection

The energy detector is optimal to detect the unknown signal if the noise power is known. In the energy detection, the SU senses the presence/absence of the PU's activity based on the energy of the received signals. An seen in Fig. 1.1, the measured signal $r(t)$ is squared and integrated over the observation interval T. The output of the integrator is compared with a threshold λ to decide if a PU is present.

While the energy detector is easy to implement, it has several shortcomings. The energy detector requires O($1/SNR^2$) samples for a given detection probability [12]. Thus, if a SU needs to detect weak PU signals (SNR:-10 dB to -40 dB), the energy detection suffers from longer detection time compared to the matched filter detection. Furthermore, since the energy detection depends only on the SNR of the received signal, its performance is susceptible to uncertainty in noise power. If the noise power is uncertain, the energy detector will not be able to detect the signal reliably as the SNR is less than a certain threshold, called an SNR wall. In addition, while the energy detector can only determine the presence of the signal it cannot differentiate signal types. Thus, the energy detector often results in false detection triggered by the unintended CR signals. For these reasons, in order to use energy detection the CRN needs to provide the synchronization over the sensing operations of all neighbors, i.e., each SU should be synchronized with the same sensing and transmission schedules. Otherwise, a SU cannot distinguish the received signals from primary and SUs, and hence the sensing operations of the SU will be interfered by the transmissions of its SU neighbors.

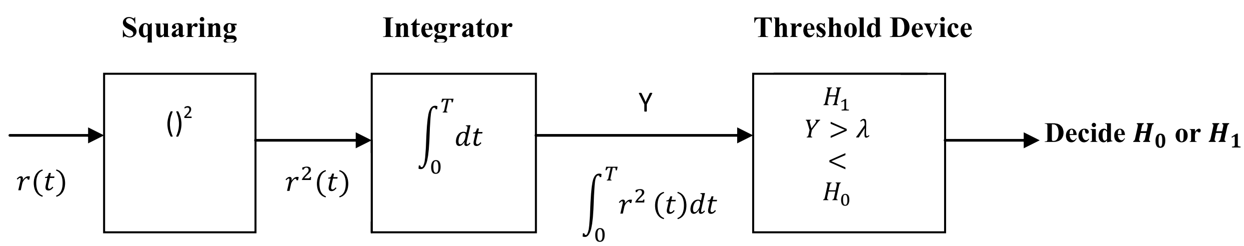


Figure 1.1: Block diagram of energy detection.

3. Feature detection

Feature detection determines the presence of PU signals by extracting their specific features such as pilot signals, cyclic prefixes, symbol rate, spreading codes, or modulation types from its local observation. The main advantage of the feature detection is its robustness to the uncertainty in noise power. Furthermore, it can distinguish the signals from different networks. This method allows the SU to perform sensing operations independently of those of its neighbors without synchronization. Although feature detection is most effective for CRN, it is computationally complex and requires significantly long sensing time.

1.2.2 Spectrum decision for cognitive radio

Spectrum decision is closely related to the channel characteristics and the operations of PUs. Spectrum decision usually consists of two steps: First, each spectrum band is characterized based on not only local observations of a SU but also statistical information of primary networks. Then, based on this characterization, the most appropriate spectrum band can be chosen. The following are the main functionalities required for spectrum decision:

Spectrum characterization: Based on the observation, the SUs determine not only the characteristics of each available spectrum but also its PU activity model. Closely related with the spectrum characterization are the radio environment and the primary user activity, which are explained below.

a. **Radio environment:** Since the available spectrum holes exhibit different characteristics, which vary over time, each spectrum hole should be characterized by considering both the time varying radio environment and the spectrum parameters such as operating frequency and bandwidth.

b. **Primary user activity:** We need a new metric to capture the statistical behavior of the primary networks, called primary user activity. Since there is no guarantee that a spectrum band will be available during the entire communication of a SU, the estimation of the PU activity is a very crucial issue in spectrum decision. Most of SU research assumes that PU activity is modeled by exponentially distributed inter-arrivals.

c. **Spectrum selection:** The SU finds the best spectrum band on the determined end-to-end route so as to satisfy end-to-end QoS requirements. Based on user QoS requirements and the spectrum characteristics, the data rate, acceptable error rate, delay bound, the transmission mode, and the bandwidth of the transmission can be determined. Then, according to a spectrum selection rule, the set of appropriate spectrum bands can be chosen.

1.2.3 Spectrum sharing for cognitive radio

A spectrum sharing cognitive radio network allows cognitive radio users (secondary users) to share the spectrum bands of the licensed-band users. However, the cognitive radio users must restrict their transmit power so that the interference caused to the licensed-band users is kept below a certain threshold. Spectrum sharing techniques are generally focused on two types of solutions, i.e., spectrum sharing inside a CR network (intra-network spectrum sharing), and among multiple coexisting CR networks (inter-network spectrum sharing) [12]. The following are the main functionalities required for spectrum sharing:

1. **Resource allocation:** Based on the QoS monitoring results, the SUs select the proper channels (channel allocation) and adjust their transmission power (power control) so as to

achieve QoS requirements as well as resource fairness and to avoid the interference to the primary network.

2. **Spectrum access:** It enables multiple SUs to share the spectrum resource by determining who will access the channel or when a user may access the channel. This is (most probably) accomplished via a random access method due to the difficulty in synchronization.

1.2.4 Spectrum mobility for cognitive radio

Secondary users are generally regarded as ‘visitors’ to the spectrum. Hence, if the specific portion of the spectrum in use is required by a PU, the communication needs to be continued in another vacant portion of the spectrum. This notion is called spectrum mobility. Spectrum mobility gives rise to a new type of handoff in CRN, the so-called spectrum handoff, in which, the users transfer their connections to an unused spectrum band. In the following, the main functionalities required for spectrum mobility in the CRN are described:

1. **Spectrum Handoff:** The SU switches the spectrum band physically and reconfigures the communication parameters for an RF front-end (e.g. operating frequency, modulation type). Spectrum handoff can be implemented based on two different strategies:

    a. **Reactive spectrum handoff**: the SUs perform spectrum switching after detecting link failure due to spectrum mobility. This method requires immediate spectrum switching without any preparation time, resulting in significant quality degradation in on-going transmissions.

    b. **Proactive spectrum handoff**: the SUs predict future activity in the current link and determine a new spectrum while maintaining the current transmission, and then perform spectrum switching before the link failure happens.

    Since proactive spectrum handoff can maintain current transmissions while searching a new spectrum band, the spectrum switching is faster but requires more complex algorithms for these concurrent operations. Depending on the events that trigger the spectrum mobility, different handoff strategies are needed. While reactive spectrum handoff is generally used in the event of a PU appearance, proactive spectrum handoff is suitable for the events of user mobility or spectrum quality degradation. These events do not require immediate spectrum switching, and can be predicted. Even in the PU appearance event, the proactive spectrum handoff may be used instead of the reactive scheme, but requires an accurate model for PU activity to avoid an adverse influence on communication performance.

2. **Connection management:** The SU sustains the QoS or minimizes quality degradation during the spectrum switching by interacting with each layering protocols. When the current operational frequency becomes busy in the middle of a communication by a SU, then applications running in this node have to be transferred to another available frequency band. However, the selection of new operational frequency may take time. An important requirement of connection management protocols is the information about the duration of a

spectrum handoff. Once the latency information is available, the SU can predict the influence of the temporary disconnection on each protocol layer, and accordingly preserve the ongoing communications with only minimum performance degradation through the reconfiguration of each protocol layer and an error control scheme.

1.3 Related work on Cognitive Ratio Networks

A SU may coexist with the incumbent PUs either on a non-interfering basis [1]–[24] or an interference-tolerant basis [25]-[47], [49]-[51]. The former case guarantees the exclusive frequency occupancy for PUs, and SUs can only operate in the unused frequency bands, also known as *spectrum holes* or *white spaces.* On the other hand, the interference-tolerant case works such that the SUs are allowed to operate on the frequency band assigned to the PU as long as the total interference power received at the primary receiver/transmitter remains below a certain threshold [51].

One of the key affecting factors of the CRN is the spectrum sensing sequences of the SUs. To find the transmission opportunities appropriately and to protect the PUs from interference, the SUs need to sense the channels regularly using local or cooperative sensing, and start a spectrum handoff procedure, if the current channel is busy. In this Thesis, we propose new algorithms that increase the throughput of SUs via appropriate selection of the SUs sensing sequences, referred to as SS. In order to design high performance CRN, research efforts have been undertaken in two directions: **a)** the SUs cooperate with each other in order to select their transmission channels, and **b)** the SUs select their transmission channels without cooperation among them. The first approach is expected to perform more efficiently, since the SUs exchange information about the primary traffic load on the networks channels, their activities and as a result the collisions are avoided in many cases. However, that approach is expected: *(i)* to consume more power and *(ii)* increase the implementation cost and complexity, compared to the no cooperation approach. In the next sub-sections recently work in CRNs is presented.

1.3.1 p-persistent random access (PPRA) scheme

The authors in [1] proposed the p-persistent random access (*PPRA*) algorithm, for operation over a multichannel CRN. The channels are assumed slotted and the SUs in each mini-slot of a slot try to randomly find an idle channel. The proposed algorithm's operation is modeled by a Markov chain with several states (each state corresponds to a mini-slot within a time slot), for the details we refer to [1].

Each SU has a fixed number of opportunities to find an idle channel within the duration of a time slot. The authors have assumed that each PU has a unique transmission channel and is either absent or present for the entire time of a slot. During a particular mini-slot of each slot, the SUs may randomly sense one channel from the total N_p channels, with probability $\frac{1}{N_p}$ or may skip the sensing procedure with a pre-defined sensing probability. In this algorithm the likelihood of false alarm was considered, where a SU incorrectly senses an idle channel as busy and as a result the secondary transmission opportunity is not exploited. Also the likelihood of missed detection was taken into account, where a busy channel incorrectly is sensed as idle and a collision occurs between the SUs transmissions on that channel or between the SU and the PU transmissions on the same channel.

The authors in [1] did not take into account the case of heterogeneous channels, but they considered that all the channels have the same transmission rates. As already mentioned, the SUs in each mini-slot of a slot are trying to randomly find an idle channel without taking into account the primary traffic load on the channels or the transmission rates. Clearly, *"PPRA"* is not expected to be efficient in the case of heterogeneous CRNs.

1.3.2 γ-persistent strategy

In [7] an efficient sensing order selection strategy for a distributed CRN was proposed, where two or more independent SUs sense the channels sequentially (in some sensing order) for spectrum opportunities. According to the proposed in [7] algorithm, the SUs independently select the sensing orders in which they visit the network channels, without coordination by a centralized entity. The sensing order comes from a common predefined Latin Square, i.e., a N_p by N_p matrix of N_p channel indices in which every channel index occurs exactly once in each row and column of the matrix, where N_p corresponds to the number of channels. The selected order corresponds to the SS of the corresponding SU. If the number of SUs is less or equal to the number of PUs, the proposed strategy enables the SUs to converge to collision-free channel sensing orders. Collision-free sensing orders are those in which two or more SUs never simultaneously sense the same channels and therefore never collide with one another.

At the beginning of the system operation each SU chooses randomly one row from the predefined Latin Square. If the SU successfully transmits or found all the channels from its SS as busy then it persists in using the specific sensing sequence. If the SU collides with another SU, then the SU reduces the selection probability of the specific sensing sequence. After experiencing a collision, the SU chooses a sensing sequence from the Latin Square based on the collision probabilities, which have been computed during the system operation. The details of the *"γ-persistent strategy"* are presented in Fig 1.2.

The authors of [7] assumed that the probability of missed detection is equal to 0 and as a result collisions among the SUs and PUs do not occur. In our work we propose a new transmission algorithm which uses the Latin Square, as does the *"γ-persistent strategy"*, and we demonstrate the weakness of the *"γ-persistent strategy"* in case the probability of missed detection is positive. The weakness of the *"γ-persistent strategy"* is significant, even when we have more transmission channels than SUs, because that strategy does not exploit the 'best' channels for secondary transmissions, e.g. the channels with low primary user traffic and high transmission rates.

γ-persistent strategy

1) Initialize $p = \frac{1}{|S|}$, where $|s|$ is number of channels and set the binary flag and the success counter $b = SC = 0$

2) Toss a weighted coin to select a sensing order, with p_i the probability of choosing sensing order i. Sense the channels sequentially in the order given in the selected sensing order.

3) One of three possibilities occurs:

a) ***Successful transmission:*** On a successful transmission using the current sensing order i, the SU updates p_i and p_j as $p_i = 1$ and $p_j = 0$, $\forall\, j \neq i$, i.e., it utilizes the same sensing order to visit the channels in the next slot. The SU then sets $SC = SC + 1$

b) ***SU finds all channels busy:*** On using sensing order i in the current slot if all the channels visited by the SU are found to be currently occupied by either a PU or another SU, the SU updates p_i and p_j as $p_i = 1$ and $p_j = 0$, $\forall\, j \neq i$, i.e., it utilizes the same sensing order to visit the channels in the next slot. The SU then sets *b* = 1.

c) ***SU collides with another SU:*** On experiencing a collision in the current slot using sensing order i, the the SU updates p_i as

$$p_i = \begin{cases} \frac{1}{|S|}, \text{if } SC = 0 \text{ } and \text{ } b = 1 \\ \gamma p_i \text{ }, otherwise \end{cases}$$

and updates p_j as

$$p_j = \begin{cases} \frac{1}{|S|}, \text{if } SC = 0 \text{ } and \text{ } b = 1 \\ \gamma p_i + \frac{1-\gamma}{|S|-1} \text{ }, otherwise \end{cases}$$

, where γ is the persistent factor and is assumed equal to $\gamma = 1 - \frac{1}{SC - \log_2(P_{fa})}$, and P_{fa} denotes the probabilty of false alarm

i.e., on experiencing a collision in the current slot the SU randomly selects a sensing order whenever $SC = 0$ and $b = 1$; otherwise the SU multiplicatively decreases the probability of picking sensing order *i* redistributing the probability evenly across the other sensing orders. The SU then sets $b = SC = 0$.
4) Return to 2.

Figure 1.2: γ-persistent strategy for sensing order selection

1.3.3 Primary-secondary user cooperation policies for cognitive radio networks

The authors in [14] assumed that the SUs may cooperate with the PU, so that the probability of success of PU transmissions is improved, while SUs obtain more transmission opportunities. In fact, SU cooperation aims precisely at increasing the PU transmission rates, thus emptying the PU queue at a faster rate. Hence, SU cooperation has the potential to increase the range of PU traffic arrival rate for which stability of PU queues can be guaranteed. Thus, SUs have to take intelligent decisions on whether to cooperate or not and with what power, in order to maximize their throughput subject to average power constraints. Cooperation policies in this framework usually require the solution of a constrained Markov decision problem with infinite state space. The authors in [14] restrict attention to the class of stationary policies, that take randomized decisions in every time slot based only on spectrum sensing. The proposed class of policies is shown to achieve the same set of SU rates as the more general policies, and enlarge the stability region of the PU queue.

A network with one PU and multiple SUs was considered. The PU is the licensed owner of the channel and transmits whenever it has data to send. On the other hand, the SUs do not have any licensed spectrum and seek transmission opportunities on the primary channel. It is assumed that one of the SUs can cooperate with the PU in order to improve the success probability of the latter's transmissions. This can be achieved by allocating some of SU's power resources for relaying the primary traffic. Furthermore, the transmission of the SUs is coordinated so that after sensing the PU channel, it is decided which SU will cooperate or not and at what power (if the primary channel is busy) or which SU will transmit and at what power (if the primary channel is idle).

In every time slot, the policy acts as follows:

- When $Q_p > 1$ (primary queue) or equivalently the channel is sensed busy, select secondary user s to cooperate at mini-slot i with a probability $q(s, \frac{i}{b})$.
- When $Q_p = 0$ or equivalently the channel is sensed idle (empty), select secondary user s to transmit its own data at mini-slot i with probability $q(s, \frac{i}{e})$.

1.3.4 Sensing Matrix Setting Algorithm (SMSA)

In [3] a cooperation scheme among the SUs was proposed. A fully synchronized time slotted network comprised of N_s SUs, equipped with narrowband sensing capability, and N_p PUs was assumed. Each SU senses the channels sequentially based on its SS provided by the CRN *coordinator*, i.e., the SU senses the first channel assigned in its SS for a predetermined channel sensing time duration, and then changes its sensing circuitry, which takes a constant time t_s, and senses the second channel if and only if the first channel was sensed busy. This procedure will be continued until a transmission opportunity is found. The structure of a channel time slot is comprised of several mini-slots corresponding to different stages of spectrum sensing and packet transmission.

In the proposed in [3] algorithm, the *coordinator* assigns channels to the SUs in a circular manner, starting with the channel that has the highest reward. The channel with the higher reward was considered as the channel that it will maximize the secondary throughput.

1.3.5 Greedy search algorithm and incremental algorithm

The paper in [17] investigates the sensing-order problem in a two-user multichannel cognitive medium access control. Although brute-force search can be used to find the optimal sensing-order setting of the two users, it has large computational complexity. Accordingly, they proposed two suboptimal algorithms, namely, the *greedy search algorithm* and the *incremental algorithm*, which have comparable performance with that of brute-force search and have much less computational complexity.

1.3.6 Greedy algorithm

The authors in [18] proposed the priority (PRP) M/G/1 queuing network model and evaluated the total service time for various target channels selections. The problem of selecting the target channels in order to minimize the total service time with multiple spectrum handoffs, was examined. Then, they suggested a low-complexity greedy algorithm to select the target channels. Numerical results show that a spectrum handoff scheme based on the greedy selection strategy can reduce the total service time compared to the random selection scheme. The authors focused on finding the optimal target channel sequences for the *proactive-decision spectrum handoff* in CRN.

They formulated a **Total Service Time Minimization Problem** for spectrum handoff as follows. Given the default channel as well as the arrival and departure models for both the primary and secondary customers, ***find an optimal target channels sequence (denoted by Θ*) to minimize the total service time S.*** Formally,

$$\Theta^* = \underset{\forall\, \Theta}{\arg\min}\, S(\Theta) \tag{1.1}$$

Some important properties for PRP M/G/1 queuing network model are listed below:

- PUs have preemptive priority and can interrupt the transmission of SUs.
- The interrupted SU is assumed to resume its unfinished transmission, instead of required to retransmit the whole packet.
- The interrupted SU's target channel can be different from its current operating channel.
- The first-come-first-served (FCFS) scheduling discipline is adopted to arrange the channel access schedule among all secondary customers.

All of the strategies discussed in Section 1.3 considered the case of homogeneous multichannel CRNs. However, in this Thesis we will examine the cases of both homogeneous and heterogeneous multichannel CRNs and we will examine two cases: (i) the case where the network's channels are not assigned to the SUs by a centralized entity and (ii) the case where a centralized entity exists and assigns the network's channels to the SUs.

1.4 Thesis Outline

The remainder of the Thesis is organized as follows. Chapter 2 presents the first contribution of our work. Section 2.2 presents the network topology. Section 2.3 describes our proposed new algorithms for homogeneous multichannel CRNs. Section 2.4 presents our event-driven simulation model, system parameters and performance results. Finally, section 2.5 contains the discussion of these results.

Chapter 3 presents the second contribution of our work. Section 3.2 presents the topology of the CRN and introduces new transmission algorithms for heterogeneous multichannel CRNs. Section 3.3 describes in detail our proposed algorithms, while section 3.4 presents our event-driven simulation model, system parameters, the examined scenarios and performance results. Finally, section 3.5 contains the discussion of these results.

In Chapter 4 we present the third contribution of our work. Section 4.2 presents the algorithmic description. Section 4.3 presents the performance metric, system model and representative simulation results. Finally, section 4.4 contains the discussion of these results.

Chapter 5 presents the fourth contribution of our work. Section 5.2 presents the network topology. Section 5.3 describes our proposed new protocols that allow cooperation among the PU and the SUs. Section 5.4 presents the performance metric, system model and representative simulation results. Finally, section 5.5 contains the discussion of these results.

Finally, Chapter 6 contains our concluding remarks, a discussion about the research contribution of the Thesis and some ideas for future work.

Chapter 2: Design and Performance Evaluation of Sequential Channel Sensing Algorithms for Multichannel Homogeneous CRNs

2.1 Introduction

The most important challenge for a secondary user is to decide which channel(s) to sense and access, and how they are sensed and accessed [52]. In this chapter, sequential channel sensing problems for multichannel Cognitive Radio Networks are studied. More specifically, a CRN with multiple channels is considered, new transmission algorithms are introduced and their performance is evaluated via simulations.

The PUs are assumed to be oblivious to the presence of the SUs and transmit whenever they have data to send. We consider a homogeneous multichannel CRN and we examine two cases: **(i)** the case where the network's channels are not assigned to the SUs by a centralized entity and **(ii)** the case where a centralized entity exists and assigns the network's channels to the SUs. We introduce four new algorithms, using the network topology, proposed in [1]. The structure of a channel time slot is comprised of several mini-slots corresponding to different stages of spectrum sensing and packet transmission; therefore within the duration of a time slot the SUs have a fixed number of opportunities to find an idle channel.

The limited number of possible observations and the dynamic nature of observed signals lead to imperfect sensing which is usually described by false alarm and miss detection probabilities [1]. In evaluating the performance of the proposed algorithms, we considered the likelihood of false alarm, where an idle channel is sensed incorrectly as busy, and as result the transmission opportunity is not exploited by the SU. We also considered the likelihood of missed detection, which causes interference to a primary or secondary transmission, because the busy channel is sensed incorrectly as idle. Our simulations results show that the new proposed algorithms: **(1)** improve the normalized average throughput of SUs, **(2)** reduce the dropping probability and **(3)** improve the number of successful transmissions occurring during the system operation.

2.2 Network System Model

As in [1], we consider a time slotted (synchronous) CRN with N_s SUs, which attempt to opportunistically transmit on the channels each dedicated to one of the N_p PUs. The primary transmissions can start only at the beginning of a slot whenever a PU has data for transmission.

Therefore, the SUs sense the channels at the beginning of each time slot, to protect the PUs transmissions from harmful interference. At the end of the sensing procedure the channel can be established as occupied or vacant. The secondary network is considered saturated, meaning that the SUs always have packets to transmit; however they will start their transmissions when an opportunity is found. The structure of a channel time slot is comprised of several mini-slots corresponding to different stages of spectrum sensing and packet transmission. The variable δ is the maximum number of allowed mini-slots within a slot, and it is defined at the beginning of the network operation. Each SU senses the channels sequentially according to its sensing sequence (SS), i.e., the SU senses the first channel that is assigned to its SS for a predetermined time duration t (channel sensing time), and if the first channel was sensed busy it then starts sensing the second channel. This procedure is continued until a transmission opportunity is found. In order to switch to a new channel, each secondary device needs a constant time duration t_s to prepare its sensing circuitry. After sensing $i-1$ occupied channels, if the SU finds the ith channel free, the SU will transmit data on that channel until the end of the current slot. In this case, the wasted time, i.e., the time spent on the sensing and handover, is equal to $t+(i-1)(t+t_s)$. Therefore, when a SU starts transmitting on the ith channel of its SS the time left in the slot for the SU transmission is:

$$RT_i = T - t - (i-1)(t+t_s) \text{ , where } T \text{ is the time slot duration.} \qquad (2.1)$$

The ith channel is sensed free if **(a)** the ith PU has no packet to transmit (an event which is assumed to occur with probability $(1-\lambda_i)$) and the SU correctly detects this transmission opportunity, this case occurs with probability $(1-\lambda_i)(1-P_{fa,i})$, or **(b)** the channel i is occupied by the PU transmission (occurs with probability λ_i) but the SU mistakenly senses this channel as free, this case occurs with probability $\lambda_i(1-P_{d,i})$, where $P_{fa,i}$ and $P_{d,i}$ denote respectively the false alarm and detection probabilities of the sensing process of the ith channel and λ_i denotes the packet arrival rate for the ith PU. If none of the above two cases occur, the request will be routed to the next mini-slot (an event which is assumed to occur with probability $P_{fa,i}(1-\lambda_i)+\lambda_i P_{d,i}$). If we have a collision between a SU and a PU or between two or more SUs, then no packet is transmitted successfully. This procedure is continued until the maximum number of admissible handovers, δ, is reached. We assumed that the transmission of each PU or SU packet takes one time slot. We further assumed that the packets of the SUs can be segmented in smaller sizes depending on the length of the remaining time slot.

The authors in [1] considered that in case of collision between the SUs transmissions or between a SU and a PU transmission, a small transportation of the packet takes places. More specifically, they considered that in case of no collision the available capacity is $C_0 = \log_2(1-\gamma_s)$, where γ_s denotes the received SNR due to the secondary users' signals at the SU receiver and in case of collision the available capacity is $C_1 = \log_2(1+(\frac{\gamma_s}{1+\gamma_p}))$, where γ_p is the received SNR due to the primary users' signals at the SU receiver. However, our simulation model does not take into consideration the capacities C_0 and C_1, but only the percentage of the remaining time of the slot that is available for transmission is considered. Fig. 2.1 presents the structure of a channel time slot. We note that a similar channel slotted model has been studied in [1], [3]-[7] , [15] and [24].

The maximum number of admissible handovers is limited by two constraints. First, the number of sensed channels cannot exceed the number of the PUs. Second, the elapsed time for both sensing and handover procedures cannot exceed the time slot duration T. So, the maximum number of sensed channels is computed:

$$\delta = 1 + \text{maximum number of admissible handovers} = 1 + \min\left(\left\{\frac{T-t}{t+ts}, N_p - 1\right\}\right) \quad (2.2)$$

Note that if one or more SUs choose the same idle channel, we must consider two different cases. First, only one SU senses the channel as idle and its segmented packet is successfully transmitted during the remaining time of the specific slot. Second, more than one SUs sense the channel as idle; as a result a collision occurs between the SUs transmissions and no packet is transmitted successfully.

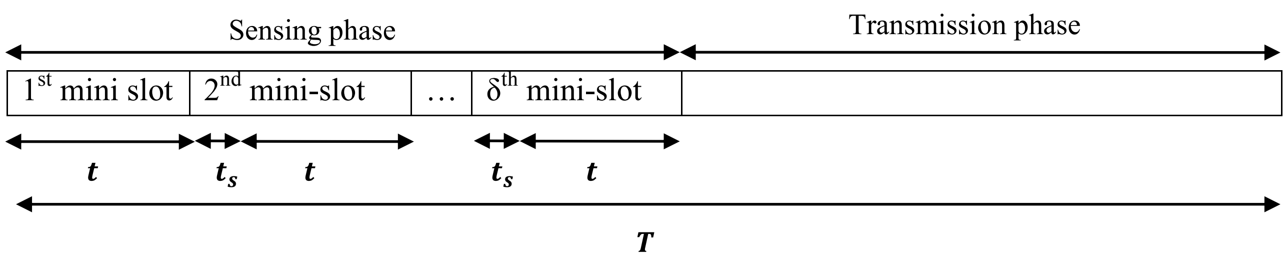


Figure 2.1: The structure of a channel time slot.

2.3 The proposed algorithms

In the following sections four new algorithms are introduced for the multichannel CRN, aimed first to serve the requests of the PUs, and then trying with various techniques to increase the aggregate throughput of the SUs, by exploiting the unused licensed spectrums when an opportunity is found. The first approach is referred to as *"PPRA with adaptive probabilities"*, the second as *"Algorithm with awareness"*, the third as *"Algorithm with limited awareness"* and the fourth as *"Algorithm with default channels"*. The first approach is based on the assumption that the SUs independently select their transmission channels, without coordination by a centralized entity. The second, the third and fourth algorithms examine the case where a centralized entity exists and assigns the network channels to the SUs. In the following four subsections, the proposed algorithms are presented.

2.3.1 PPRA with adaptive probabilities

We modified the *"PPRA"* (p-persistent random access) algorithm, introduced and described in [1], so that the traffic loads of the SUs are distributed more evenly to all the available channels. If a channel is estimated to be idle with large probability, we want the larger percentage of SUs to choose that channel as a transmission channel. Once we know that a PU tends to occupy its channel for a long period of time, that channel would be less likely to be available for a SU, as a result, sensing on such a channel would likely be a waste of time and energy.

Each SU chooses a channel based on the estimated primary user arrival rates, i.e., according to the following probabilities:

$$P_j = \frac{(1-\lambda_{est,j})}{\sum_{i=1}^{N_p}(1-\lambda_{est,i})} \quad ,j = 1, \dots, N_p \tag{2.3}$$

For this to be possible, each SU needs to have the estimates of the packet arrival rates of all the PUs. We assume that each PU k collects the feedback from its dedicated channel for the last x slots and it then estimates its packet arrival rate by:

$$\lambda_{\text{est,k}} = \frac{Number\ of\ ACKs}{x},\ k = 1, \dots, N_p \tag{2.4}$$

In the initial $x - 1$ slots, the estimated primary user arrival rates are unknown, so the SUs will choose the channels with equal probability during that time period. We further assume that the PUs broadcast the channel selection probabilities $\mathrm{P_j}$ $, j = 1, \dots, N_{p,}$ to the SUs on a separate broadcast channel.

In each mini-slot of the slot, the SU senses the channel with probability p and chooses to skip the sensing procedure with probability $1 - p$. If the last possibility happens or the examined channel was sensed as busy, then the SU will be routed to the next mini-slot. We assume that at the beginning of each slot, the SUs have all channels as candidates for transmission and the channels are selected based on the probabilities in (2.3). In case a SU senses the selected transmission channel as busy, at a mini-slot of the specific time slot, then the specific channel is removed from its list of candidate channels at the subsequent mini-slot of the specific time slot. The above procedure is repeated until the maximum number of admissible handovers is reached or a transmission opportunity is found.

The modifications in our proposed algorithm are two, compared to the *"PPRA"* algorithm introduced in [1]. Firstly, in our proposed algorithm the SUs select their transmission channels based on the estimated PU arrival rates and not with equal probability, as happens in [1]. Secondly, whenever a SU fails to transmit on a channel (because that channel was sensed as busy) then the specific channel is removed from the list of candidate transmission channels at the subsequent mini-slot of the slot.

2.3.2 Algorithm with awareness

The goal of the proposed algorithm is to prevent the SUs from selecting a busy channel to sense. In the proposed algorithm we take into account the presence of a *Coordinator*, which assigns the network channels to the SUs in a circular manner. At the beginning of each time slot the channels are sorted in ascending order base on the estimated PU arrival rates, as calculated by (2.3). Subsequently, the channel with the lowest PU arrival rate is assigned to a SU; the channel with the next lower PU arrival rate is assigned to another SU etc., until all the SUs have a pre-selected transmission channel. Note that, if the number of SUs is less than the total number of the available channels, no channel contention issue arises between the SUs, because all the SUs will have a different pre-selected transmission channel. If the number of SUs is larger than the number of channels then the channels with low PU arrival rates are selected by more SUs. An example of circular channel assignment at the first mini-slot of the slot is shown in Figure 2.2. Assume that channel 1 has the lowest PU packet arrival rate followed by channels 2, 3, 4, 5, 6 and we show the case we have: (a) 3 SUs and 6 channels and (b) 8 SUs and 6 channels.

What happens in the first mini-slot of a time slot is important, because if the SU finds an idle channel, the transmission duration is longer compared with that at the other mini-slots of the specific time slot. Note that if the number of channels is larger than the number of SUs we want all the SUs to sense the pre-selected transmission channel. Otherwise, we want the SUs to choose the sensing procedure with probability p and to skip the sensing procedure with probability $1 - p$, in order to reduce the collision probability between the SUs, in case they choose the same idle channel for transmission. The *Coordinator* is aware of the total number of available channels and the total number of SUs; therefore at the beginning of the system operation can sent a binary signal to SUs. If the signal is set to 1 means that we have fewer channels than SUs; so the SUs choose to sense the channels with probability p and skip the sensing procedure with probability $1 - p$, for all the mini-slots of each time slot. Otherwise, the SUs sense the pre-selected transmission channel at the first mini-slot of each time slot with probability $p = 1$ and in the following mini-slots of the time slot, they decide to sense (or not) the channels with probability p, $(1 - p)$, respectively.

If the SU senses the pre-selected transmission channel, a transmission opportunity is found and the SU correctly senses the channel as idle, then the secondary transmission takes place in the remaining time of the slot. Otherwise, if an idle channel is sensed incorrectly as busy or a busy channel is correctly detected as busy, then the SU will be routed to the next mini-slot. Finally, if we have a collision between the SUs or between a SU and a PU, no primary or secondary transmission takes place. In that case the PU's packet is retransmitted at the beginning of the next slot and the SU is dropped from the specific time slot.

The *Coordinator* in each mini-slot of a specific slot selects the SUs, which do not already transmit in the specific slot with equal probability; therefore the SUs have equally likely transmission opportunities. At each mini-slot of the slot, the channel with the lower PU packet arrival rate which has never been assigned to a SU will be the next candidate channel for assignment. If all the channels are assigned to a SU, then the candidate channel will be the channel with the lowest PU packet arrival rate that is not busy by SU. The above procedure is repeated until the maximum number of admissible handovers is reached or transmissions opportunities are found.

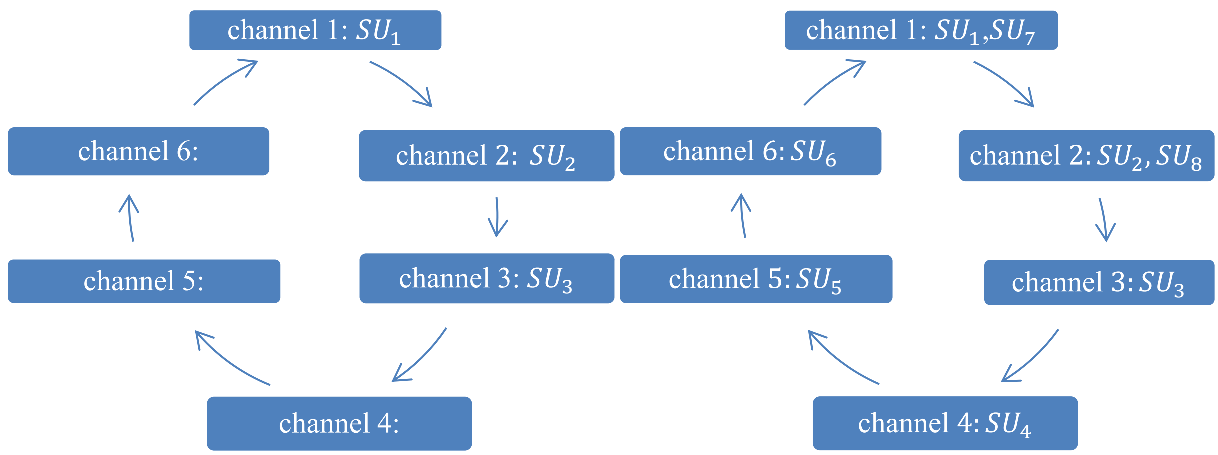


Figure 2.2: Circular channel assignment at the first mini-slot of the slot: (a) 6 sorted channels and $SU_i, i = 1, .., 3$ (b) 6 sorted channels and $SU_j, j = 1, \dots, 8$.

From the overall description of the algorithm, it is clear that the *Coordinator* should monitor the PUs's feedbacks, in order to compute the estimated PU packet arrival rates. Note that in a real scenario if the arrival rates of the PUs do not vary abruptly with time, the *Coordinator* does not have to monitor the channel feedbacks for the entire duration of the system operation. In such case, the *Coordinator* have to monitor the channel feedbacks every y slots, and for a duration of x slots, where $y >> x$ and x and y are design parameters. Also the *Coordinator* has to monitor the SUs's feedback at the end of each mini-slot of the slot, to know the channels that are occupied by the SUs, in order to remove them from the list of candidate channels for assignment.

The specific algorithm requires large amounts of energy and is not expected to be applicable in many practical systems. We present this algorithm in order to provide a performance upper bound (not the optimal). Given that complexity has been significantly increased, we focus on developing efficient algorithms in the next sections, which reduce the complexity while maintaining good performance.

2.3.3 Algorithm with limited awareness

In the proposed algorithm we try to reduce the energy requirements imposed by algorithms which require strong cooperation among the *Coordinator* and the SUs, as the *"Algorithm with awareness"*. The proposed algorithm is a combination of the two previous algorithms, "*PPRA with adaptive probabilities"* and *"Algorithm with awareness",* described in sections 2.3.1 and 2.3.2, respectively. As already mentioned in the "*PPRA with adaptive probabilities"* algorithm, the SUs select their transmission channels without coordination by a centralized entity, and in the *"Algorithm with awareness",* a centralized entity exists and assigns the network channels to the SUs. In the proposed algorithm we assume that the *Coordinator* intervenes only at the first mini-slot of the first slot. In the remaining mini-slots of a slot, the algorithm functions as the algorithm "*PPRA with adaptive probabilities"*, so that algorithm allows coordination by a centralized entity only once during the system operation.

At the first mini-slot of the first time slot, we assume that the *Coordinator* assigns channels to the SUs in a circular manner, as happens in the *"Algorithm with awareness"*. When the channel assignment is completed each SU will has a pre-selected transmission channel and will chooses that channel at the first mini-slot of each slot. The channels are sorted from the channel with the lowest PU packet arrival rate to the busiest channel using (2.3). Subsequently, the channel with the lowest PU packet arrival rate is assigned to a SU; the channel with the next lower PU packet arrival rate is assigned to another SU etc., until all the SUs have a pre-selected transmission channel. In case the number of SUs is less than the number of channels, the secondary transmission will fail at the first mini-slot only if false alarm occurs, or the channel is occupied by the PU.

We further assume that the *Coordinator* intervenes at the beginning of the simulation in order to determines if the sensing probability will be applied at all the mini-slots of the time slot or at all the mini-slots of the time slot beyond the first, as happens in the previous algorithm. If the binary signal is set to 0 means that the number of available channels is larger than the number of SUs and all the SUs sense the pre-selected transmission channel at the first mini-slot of each time slot. In the remaining mini-slots of each time slot the sensing probability is applied. Otherwise, the sensing probability is applied for all the mini-slots of each time slot.

Note that in a real scenario if the arrival rates of the PUs do not vary abruptly with time, the *Coordinator* does not have to monitor the channel feedbacks for the entire duration of the system operation. In such case, the *Coordinator* intervenes after y slots, when the PU arrival rates change, and then assigns new transmission channels to the SUs. The new transmission channels will be selected at the first mini-slot of each slot.

For the remaining mini-slots of the time slot, the same procedure as the *"PPRA with adaptive probabilities"* algorithm is implemented. Recall the key points of that algorithm, the SUs choose channels based on the estimated PU arrival rates using (2.3). At each mini-slot of the time slot the SUs decide to sense the channel with probability p and to skip the sensing procedure with probability $1-p$. If the SU's transmission failed in the previous mini-slot, because the channel was occupied (by PU or by SU) then the specific channel is removed from the SU's list of candidate channels for transmission. This procedure is continued until the maximum number of admissible handovers is reached or a transmission opportunity is found.

2.3.4 Algorithm with default channels

In this algorithm default channels are pre-assigned to each SU by the *Coordinator* in order to balance the overall traffic load of the SUs on all the channels.

At the beginning of the system operation (i.e., only once during the network operation) the *Coordinator* assigns channels to the SUs in a circular manner, so that all the channels will be assigned. The channels are sorted from the channel with the lowest PU packet arrival rate to the busiest channel using (2.3). When the channel assignment is completed, each SU has its own SS. If the primary user arrival rates change, an event which we assume it happens on average every y slots, the *Coordinator* has to estimate the new primary user arrival rates using (2.3) and then to assign new channels to the SUs. Figure 2.3, presents an example of system initialization in the case we have (a) 6 channels and 3 SUs and (b) 2 channels and 3 SUs. Assume that the ch_1 has the lowest PU packet arrival rate, followed by channels 2, 3, 4, 5, 6.

If the number of SUs is less than the total number of available channels, then some of the SUs will have more than one candidate transmission channels. At each mini-slot of a slot the SU may sense one channel from its SS with probability p.

If the number of channels is larger than the number of SUs (i.e. $N_p > N_s$), then all the SUs sense the preselected channel at the first mini-slot of the slot with probability one. In the remaining mini-slots of the slot, the sensing is done according to the value of the sensing probability. Otherwise, the sensing probability is applied for all the mini-slots of each time slot.

If the SU senses the channel within the duration of a time slot, then it is trying to transmit first on the channel with the lowest PU packet arrival rate from its SS. If that channel is sensed idle then the SU transmits in the remaining time of the slot. However, if that channel is sensed busy or the SU chooses to skip the sensing procedure (because its sensing probability is less than one), then it will be routed to the next mini-slot. In case the secondary transmission does not take place (because the previously examined channel was sensed busy), then the next channel with the lowest PU packet arrival rate from the specific SU's SS, is chosen. The SU will sense that channel with probability p in the next mini-slot. The process is repeated cyclically from the lightest to the heaviest channel at each subsequent mini-slot, until the maximum number of handovers is reached or a transmission opportunity is found.

If the number of channels is larger than the number of SUs, the algorithm guarantees that no contention among the SUs takes place. Note that even if a SU found all the channels in its SS busy, it continues to sense the channels with probability p. This is because false alarm may have occurred in the previous mini-slots of the specific slot.

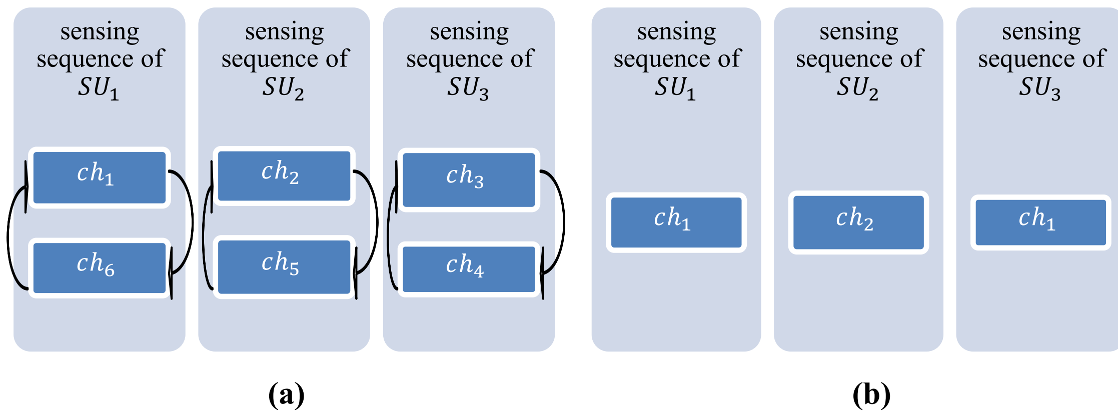


Figure 2.3: Example of system initialization: (a) 6 channels and 3 SUs and (b) 2 channels and 3 SUs

2.4 Performance Evaluation

2.4.1 Performance metrics

The main goal of the CRN is to efficiently manage the utilization of the available spectrum by allowing the SUs to identify the temporally vacant portions of licensed spectrum. Each PU has its own channel; therefore the PUs throughput is 100%. However, the PUs transmissions are suffering small delays due to the collisions caused by the SUs. As already mentioned, if a busy channel is sensed incorrectly as idle by a SU, then a collision occurs and no primary or secondary transmission takes place. In that case the PU's packet is retransmitted at the beginning of the next slot and the SU is dropped from the current slot (is not allowed to the SU to have any action in the specific time slot). Note that in case of collision the SU will perceive that its transmission failed at the end of the specific time slot. SU infers that a collision has occurred whenever it fails to receive an acknowledgment (ACK) for a transmitted data packet. In our simulation we study three performance metrics: (1) the normalized average throughput of the SUs, (2) the SU dropping probability and (3) the total number of successful transmissions occurring during the various mini-slots of the slots. Below the performance metrics are explained.

Secondary User's average throughput is the primary performance metric that provides the maximum packet transmission rate per time slot.

The SU dropping probability corresponds to the fraction of dropped SUs from the system in relation to the total number of channel senses. As already mentioned if more than one SUs choose the same idle channel and more than one SUs correctly detect it as idle then we will have a collision between the SUs and no secondary transmission takes place. Also if a SU mistakenly senses a busy channel as idle then a collision occurs and the SU is dropped from the current time slot. The dropping probability is an indication of the effectiveness of the algorithms in preventing collisions among the SUs and among SUs and PUs.

Finally, the total number of successful transmissions shows the number of successful transmissions occurring during the $1^{st}, 2^{nd}, \dots, \delta^{th}$, mini-slot from all the time slots of the system operation. The specific metric shows the effectiveness of the algorithms, in quickly finding an idle channel.

2.4.2 Simulation model

We conducted event-driven simulations to evaluate the performance of the proposed algorithms. In order to compare the results of our algorithms with those of the algorithm in [1], we modified the proposed therein "*PPRA*" algorithm as follows. As in our scheme, described in Fig. 2.1, when we have collision in a time slot then the packets transmitted by both the PU and the SU are both considered lost.

We consider a time slotted (synchronous) CRN with N_s SUs, which attempt to opportunistically transmit on the N_p channels, each dedicated to one of the N_p PUs. We further assume that the network is comprised of homogeneous channels, i.e. that all the channels have the same transmission rates. We assume that the PUs's packets arrive according to a Poisson process. Let λ_i (arrivals/slots) be the arrival rate of the PU packets at channel $i, i = 1,.., N_p$. The minimum allowable value of detection probability, P_d^{min}, the maximum admissible false alarm probability, P_{fa}^{max} and the time slot duration, T, are chosen according to *IEEE 802.22* standard [2]. In our simulation the worst case scenario is considered and we set $P_{d,i} = P_d^{min}, i = 1, \dots, N_p$ and $P_{fa,i} = P_{fa}^{max}, i = 1, \dots, N_s$. The normalized average throughput, the dropping probability and the total number of successful transmissions of the SUs are computed by simulating each scenario for a time period of 1000 time slots. The default values for all the parameters are shown in Table 2.1. The average achievable normalized throughput is computed as:

$$T_h = \frac{1}{N_s} \sum_{j=1}^{N_s} \frac{1}{N} \left(\sum_{i=1}^{N} \frac{R_i^{(j)}}{T} \right) \quad (2.5)$$

, where $R_i^{(j)}$ denotes the SU's j remained time at the slot i (see equation (2.1)) and N is the number of time slots per simulation run.

To compare the performance of the proposed algorithms we examined a scenario in which the contention between the SUs is not strong and a scenario in which the contention between the SUs is strong. In the first scenario the number of channels is larger than the number of SUs and we set $N_p = 10$ and $N_s = 5$. In the second scenario the number of channels is less than the number of SUs and we set $N_p = 5$ and $N_s = 8$. In the sequel, each scenario was divided into three sub-scenarios and we examined the impact of lightly loaded channels, of heavily loaded channels, of asymmetrically loaded channels and finally the impact of varying the PU arrival rates. The results are presented and discussed in the following sections.

TABLE 2.1*: SYSTEM PARAMETERS AND DEFAULT VALUES*

Notation	Definition (default value)
$P_{d,i}$	Probability of correct detection (0.9), $i = 1, \dots, N_p$
$P_{fa,i}$	Probability of false alarm (0.1), $i = 1, \dots, N_p$
T	Time slot duration (10ms)
t_s	Required time for handover (0.01ms)
t	Required sensing time (2.4ms)
N	Number of time slots per simulation run (1000)
x	Required slots for estimating the PU arrival rates (100)
y	Required slots for detecting changes in the PU arrival rates (2000)
δ	Total number of mini-slots (4)

2.4.3 Simulation results

An event-driven simulator is implemented to evaluate the effectiveness of the proposed algorithms: *"PPRA with adaptive probabilities"*, *"Algorithm with awareness"*, *"Algorithm with limited awareness"* and *"Algorithm with default channels"*, which employs the best algorithm from [1], namely *"PPRA"*. Sections 2.4.3.1 and 2.4.3.2 present the results of the first and second scenarios, respectively.

2.4.3.1 Simulation results in case the channels are more than the SUs

In the next sections we present the result of the first scenario in which the number of PUs is 10 and the number of SUs is 5. The results show that the *"Algorithm with default channels"*, performs equally well with the *"Algorithm with awareness"*. The *"Algorithm with default channels"* outperforms the *"Algorithm with limited awareness",* the *"PPRA with adaptive probabilities"* algorithm and the *"PPRA"* algorithm, in all examined scenarios. As seen by the following results, the *"PPRA with adaptive probabilities"* behaves more efficiently compared to the *"PPRA"* algorithm, in case the channels have different PU arrival rates. However, this improvement is small compared to the algorithms that require coordination by a centralized entity. As already mentioned in section 2.3, the *"Algorithm with awareness"*, the *"Algorithm with limited awareness"* and the *"Algorithm with default channels"*, allow coordination by a centralized entity while in the algorithms *"PPRA with adaptive probabilities"* and *"PPRA"*, the SUs select their transmission channels without coordination by a centralized entity.

In the following three subsections we present the results for the system performance metrics by examining the impact of "light" primary traffic load, of "heavy" primary traffic load, and of asymmetric primary traffic load, as a function of the sensing probability. Subsequently, we set the sensing probability equal to 1 and we examine the impact of varying the PU arrival rates.

2.4.3.1.1 The case of lightly loaded channels

In this section we examine the impact of sensing probability on the performance metrics: normalized average throughput and dropping probability, in the case of lightly loaded channels. Also the performance metric total number of successful transmissions is evaluated, in the case when the sensing probability is set equal to 1. We assume that the PU arrival rates are equal for all the channels are set equal to 0.2 packets per slot.

The results in Fig 2.4 show the impact of the sensing probability on the normalized average throughput of SUs. In the algorithmic description of the algorithms, *"Algorithm with awareness"*, *"Algorithm with limited awareness"* and *"Algorithm with default channels",* it was explained that in the specific scenario where $N_p > N_s$, the illustrated sensing probability is applied only at the mini-slots of a slot beyond the first. Note that in the *"Algorithm with awareness"* and the *"Algorithm with default channels"* the SUs are distributed to N_p different channels (in the specific scenario $N_p \geq N_s$), so there is no reason to skip the sensing procedure, unless we have energy restrictions.

In the examined scenario, the *"PPRA with adaptive probabilities"* performs close to the *"PPRA"* algorithm, because the estimated PU arrival rates are the same for all the channels. Equation (2.3) gives the channel selection probability approximately equal to $\frac{1}{N_p}$, as happens in [1].

From the results in Fig 2.4-Fig.2.6 it is evident that the algorithms which require coordination by a centralized entity outperform those without coordination by a centralized entity, in all system performance metrics. This is due to the fact that the SUs were distributed more efficiently to the available channels. Under the assumption that the number of SUs is less than the number of channels, the *"Algorithm with limited awareness",* guarantees that the SUs will not compete for an idle channel at the first mini-slot of each slot and the *"Algorithm with awareness"* and the *"Algorithm with default channels"* guarantee that no contention between the SUs takes place during the entire system operation. Compared with the results in [1] when the sensing probability is equal to 1, the *"Algorithm with awareness"*, the *"Algorithm with limited awareness"* and the *"Algorithm with default channels"*, achieve on average a **49.01%**, a **37.27%** and a **44.67%** improvement on the normalized average throughput of SUs, respectively.

As seen by the results shown in Fig 2.5 the *"Algorithm with awareness"*, the "*Algorithm with limited awareness"*, and the *"Algorithm with default channels"* achieve the smaller dropping probability. In the *"Algorithm with awareness"* and "*Algorithm with default channels*" we can only have collisions between a SU and a PU (due to a missed detection). Note that only 20% of the slots are occupied by the PUs and the probability of missed detection is equal to 0.1, so the expected dropping probability is approximately 0.02, for the algorithms that guarantee that the contention among the SUs does not exist (i.e. for the algorithms *"Algorithm with awareness"* and "*Algorithm with default channels*"). In the algorithms *"PPRA with adaptive probabilities"* and *"PPRA"* the dropping probability is approximately 0.2 when the sensing probability is equal to 1, because the SUs decide based on their own knowledge, i.e., there is no *Coordinator*, and thus is more likely to compete for an idle channel, compared to the other three algorithms. The *"Algorithm with limited awareness"* performs well in terms of dropping probability because in the first mini-slot of the slot most of the SUs successfully transmit (see Fig 2.6), therefore they will not compete at the remaining mini-slots of the slot.

Fig 2.6 shows the effectiveness of the algorithms *"Algorithm with awareness", "Algorithm with limited awareness",* and *"Algorithm with default channels"* in quickly finding a transmission opportunity, when the sensing probability is set equal to 1. This was expected because in the aforementioned algorithms the *Coordinator* assigns different channels to the SUs.

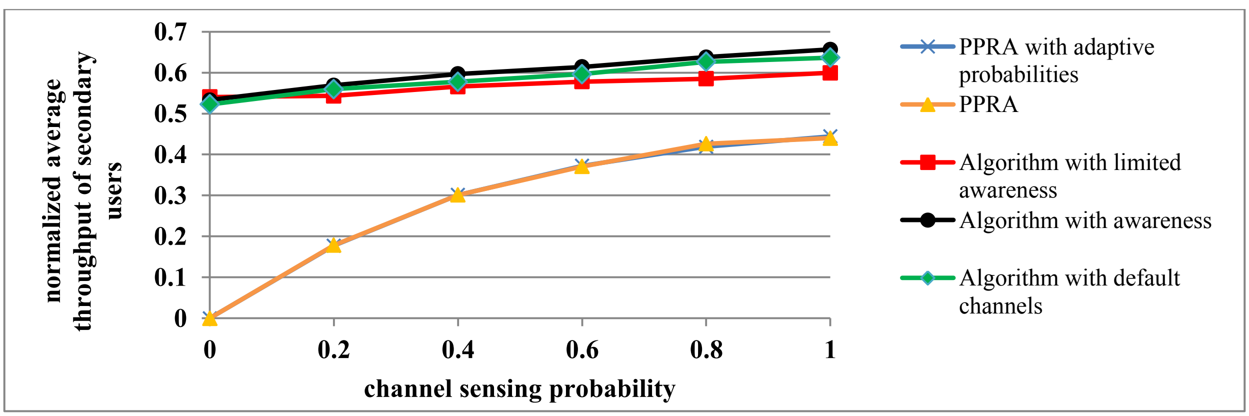


Figure 2.4: The impact of sensing probability on the normalized average throughput of SUs in the case of lightly loaded channels.

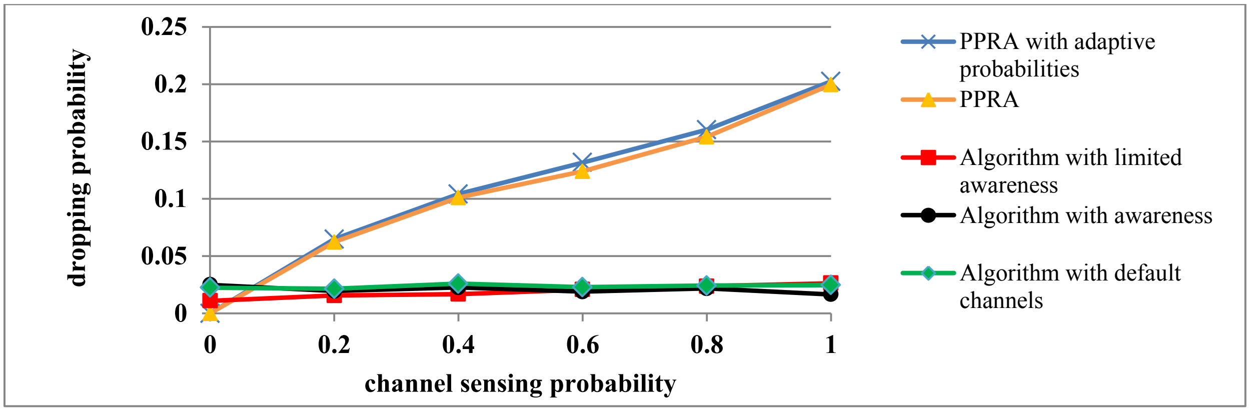


Figure 2.5: The impact of sensing probability on the dropping probability in the case of lightly loaded channels.

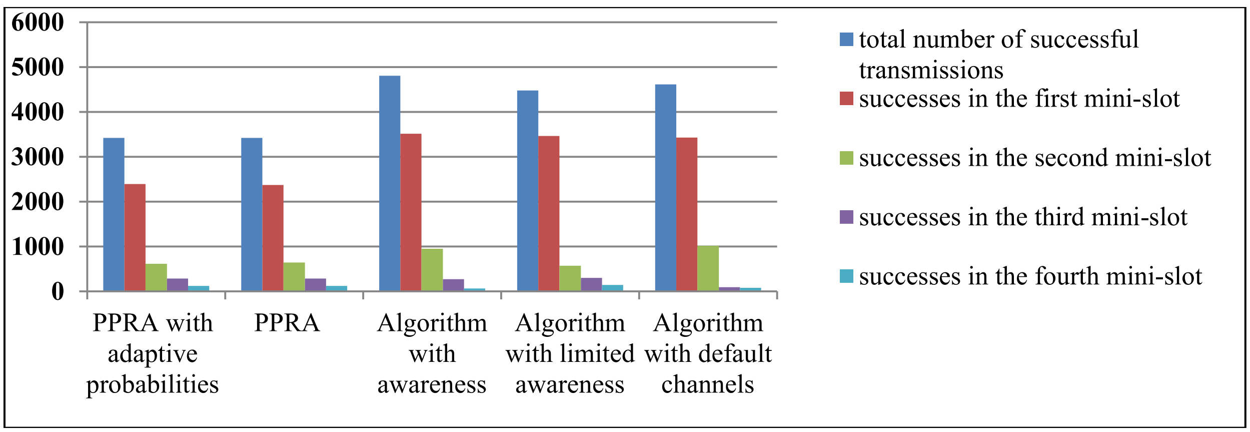


Figure 2.6: Total number of successful transmissions in the case of lightly loaded channels.

2.4.3.1.2 The case of heavily loaded channels

In this section we examine the impact of sensing probability on the performance metrics: normalized average throughput and dropping probability, in the case of heavily loaded channels. Also the performance metric total number of successful transmissions is evaluated, when the sensing probability is set equal to 1. We assume that the PU arrival rates are equal for all the channels and are set equal to 0.8 packets per slot. As expected the achieved normalized average throughput of the SUs is reduced compared with the previous scenario, because about 80% of the slots are occupied by the PUs and the secondary transmission opportunities have dramatically decreased. Note that from the remaining slots, the SUs lose 24% of them in order to sense the channels. Ideally the maximum normalized average throughput of SUs is expected to be equal to 0.152 [0.76*(1-0.8)].

The results in Fig 2.7 show the impact of the sensing probability on the normalized average throughput of SUs. As seen from the results presented in Figures 2.4 and 2.7, the throughput for the algorithms *"PPRA with adaptive probabilities"* and *"PPRA"*, is highly affected by the amount of the sensing probability. In the examined scenario the number of channels is larger than the number of SUs; therefore the secondary throughput is increased by using all the transmission opportunities at all the mini-slots of each slot. The throughput for the *"Algorithm with awareness"*, the *"Algorithm with limited awareness"*, and the *"Algorithm with default channels"*, is not significantly affected by the value of the sensing probability, because in these algorithms most of the SUs successfully transmit from the first mini-slot of the slot (see Fig.2.9). Note that in the aforementioned algorithms if the sensing probability is 0, only the first mini-slot of each slot is used, the remaining mini-slots are not exploited by the SUs. Compared with *"PPRA"* algorithm in [1], when the sensing probability is equal to 1, the *"Algorithm with awareness"*, the *"Algorithm with limited awareness"* and the *"Algorithm with default channels"*, achieve on average a **53.06%**, a **14.22%** and a **41.35%** improvement on the normalized average throughput of SUs, respectively.

As seen by the results shown in Fig 2.8, when the channels are occupied by the PUs transmissions with high probability the dropping probabilities for the algorithms are close to each other. In the specific scenario it is most likely to have collisions between PUs and SUs (due to a missed detection), because most of the slots are occupied by the PUs transmissions. Rarely the channels will be idle, to provoke contention among the SUs, (for algorithms in which contention among the SUs are possible).

Fig 2.9 shows the total number of successful transmissions at the various mini-slots of the slots. Compared with the corresponding results in Fig 2.6, the total number of successful transmissions is decreased because the PU packet arrival rates have been increased. In the examined scenario here we have PU arrival rates equal to 0.8, so the secondary transmission opportunities are dramatically decreased. The algorithms *"Algorithm with awareness"* and *"Algorithm with default channels"* outperform the algorithms *"PPRA"* and *"PPRA with adaptive probabilities"*, because in that algorithms the SUs are not compete for an idle channel during the entire system operation.

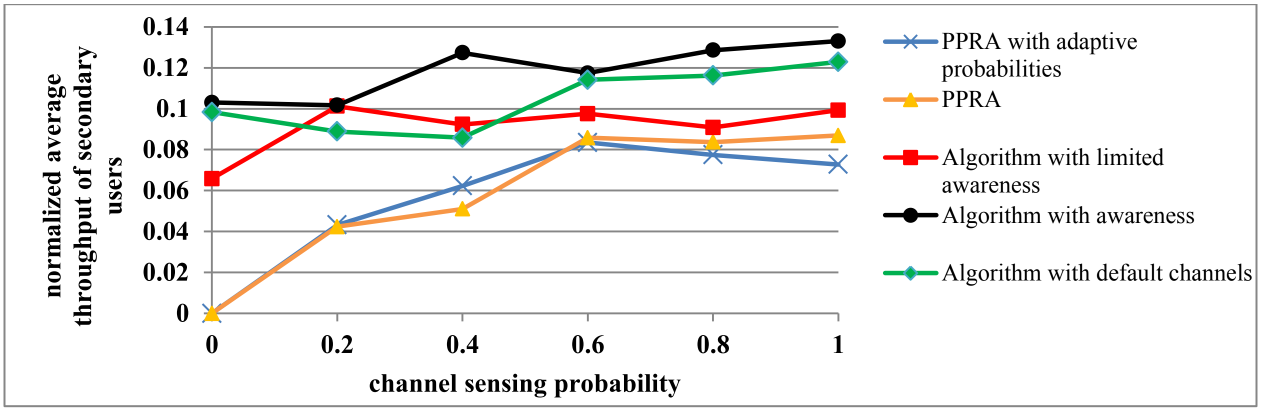


Figure 2.7: The impact of sensing probability on the normalized average throughput of SUs in the case of heavily loaded channels.

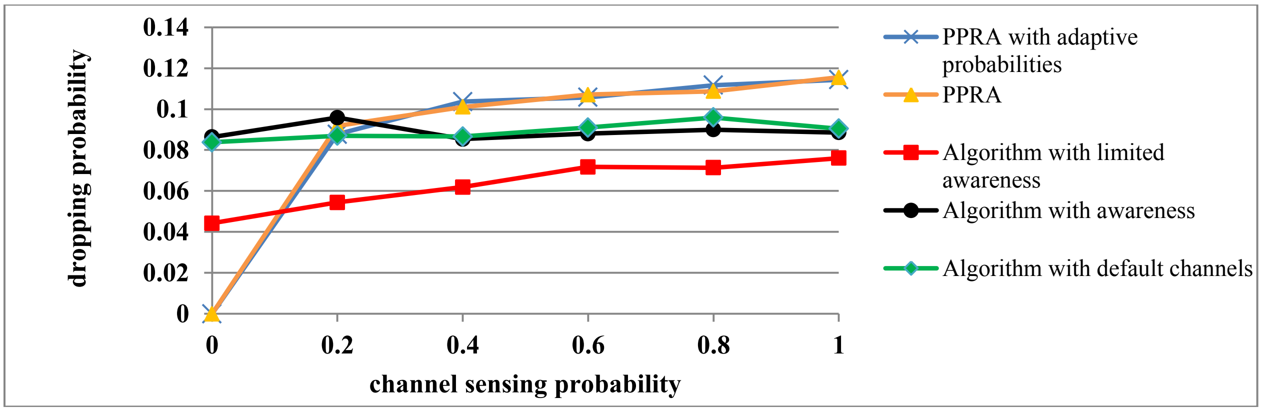


Figure 2.8: The impact of sensing probability on the dropping probability in the case of heavily loaded channels.

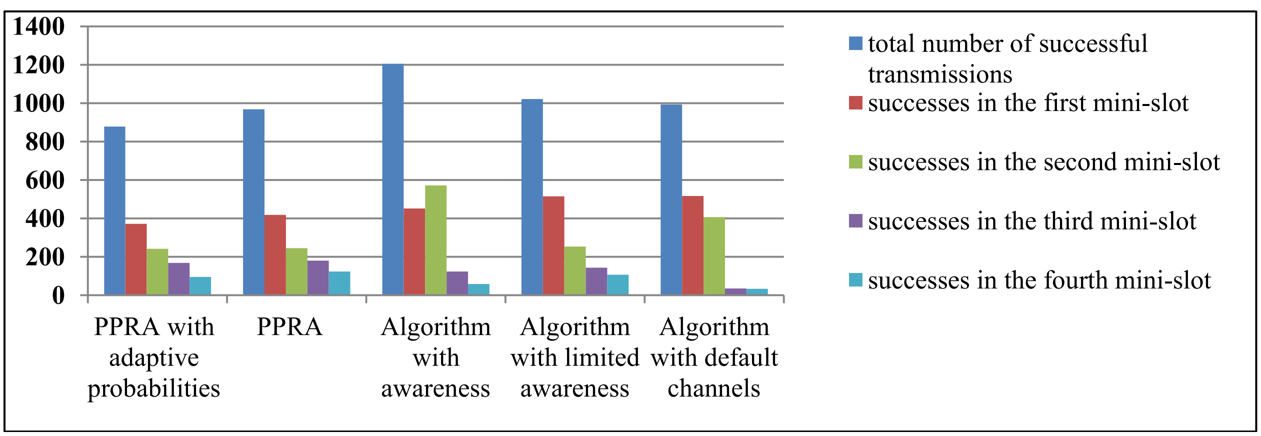


Figure 2.9: **Total number of successful transmissions in the case of heavily loaded channels.**

2.4.3.1.3 The case of unequally loaded channels

Here we examine the impact of sensing probability on the performance metrics: normalized average secondary throughput and dropping probability, in the case of unequally loaded channels. Subsequently, the performance metric total number of successful transmissions is evaluated, when the sensing probability is set equal to 1. We assume that the PU arrival rates are equal to:

$$\lambda_i = \{0.1, 0.1, 0.3, 0.3, 0.5, 0.5, 0.7, 0.7, 0.9, 0.9\}, \mathrm{i} = 1, \dots, \mathrm{N_p} = 10$$

As seen from the results presented in Fig 2.10 the *"PPRA with adaptive probabilities"* algorithm performs more efficiently compared to the *"PPRA"* algorithm, in terms of the normalized average throughput of SUs. The *"PPRA with adaptive probabilities"* algorithm distributes the SUs among the channels based on the estimated PU arrival rates. In the proposed in [1] *"PPRA"* algorithm, the SUs in each mini-slot of a slot are trying to randomly find an idle channel. Thereby, a SU has the same chance to choose the first channel with expected PU arrival rate equal to 0.1 and the tenth channel with expected PU arrival rate equal to 0.9. Compared with "*PPRA"* algorithm, proposed in [1], when the sensing probability is equal to 1, the *"Algorithm with awareness"*, the *"Algorithm with limited awareness"* and the *"Algorithm with default channels"*, achieve on average a **86.51%**, a **65.63%** and a **72.55%** improvement on the average normalized throughput of SUs, respectively.

As seen from the results shown in Fig 2.11 the proposed algorithms *"Algorithm with awareness"*, *"Algorithm with limited awareness"* and the *"Algorithm with default channels"*, perform equally well in terms of dropping probability. This is attributed to the nonexistent or limited contention among the SUs in those algorithms, as already been explained (see section 2.4.3.1.1). In the aforementioned algorithms most of the SUs successfully transmit in the first two mini-slots of the slots, as it can be seen from the results presented in Fig 2.12. The *"PPRA"* algorithm incurs smaller dropping probability compared to the *"PPRA with adaptive probabilities"* algorithm, because it has smaller probability to find an idle channel and as a result has smaller probability to experience contention among the SUs. The *"PPRA"* algorithm chooses with equal probability the heavily loaded channels and the lightly loaded channels, and as a result chooses the channels that are occupied by PU transmissions more frequently compared to the *"PPRA with adaptive probabilities"* algorithm.

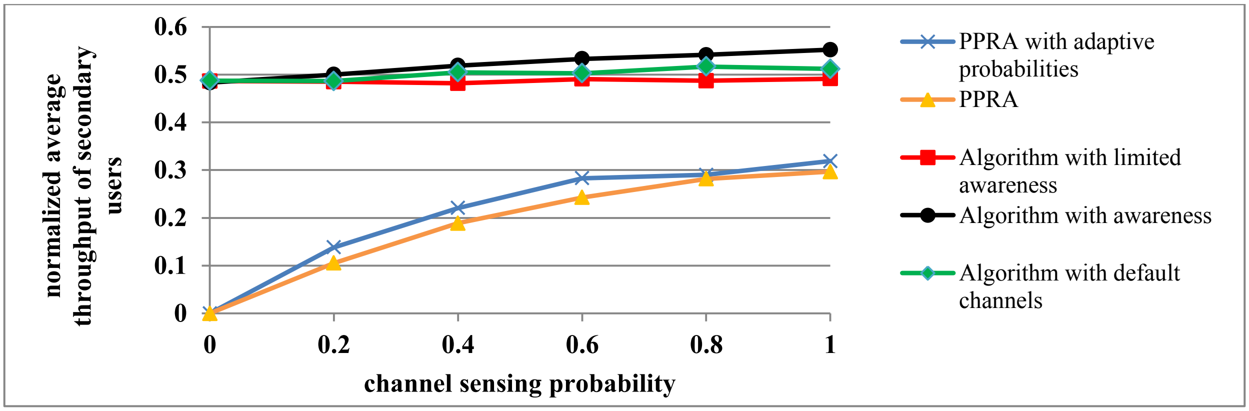


Figure 2.10: The impact of sensing probability on the normalized average throughput of SUs in the case of unequally loaded channels.

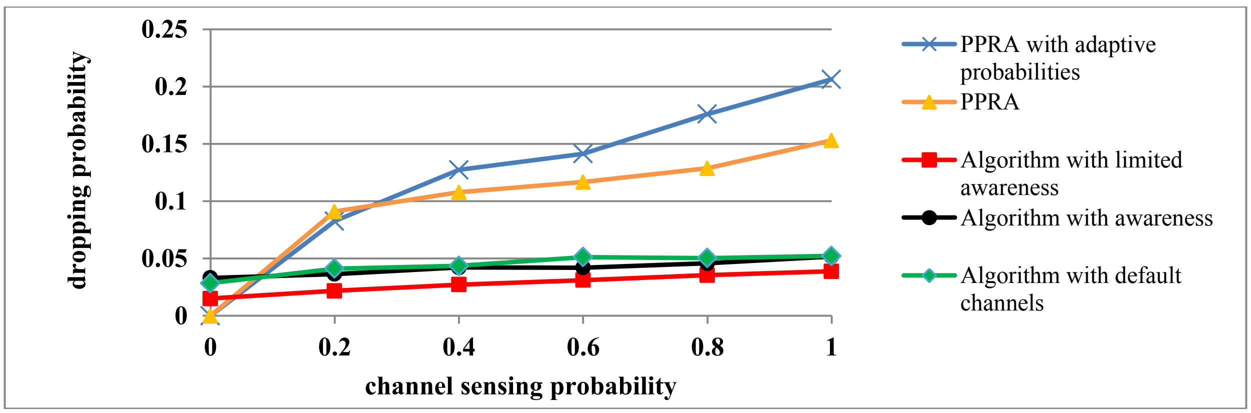


Figure 2.11: The impact of sensing probability on the dropping probability in the case of unequally loaded channels

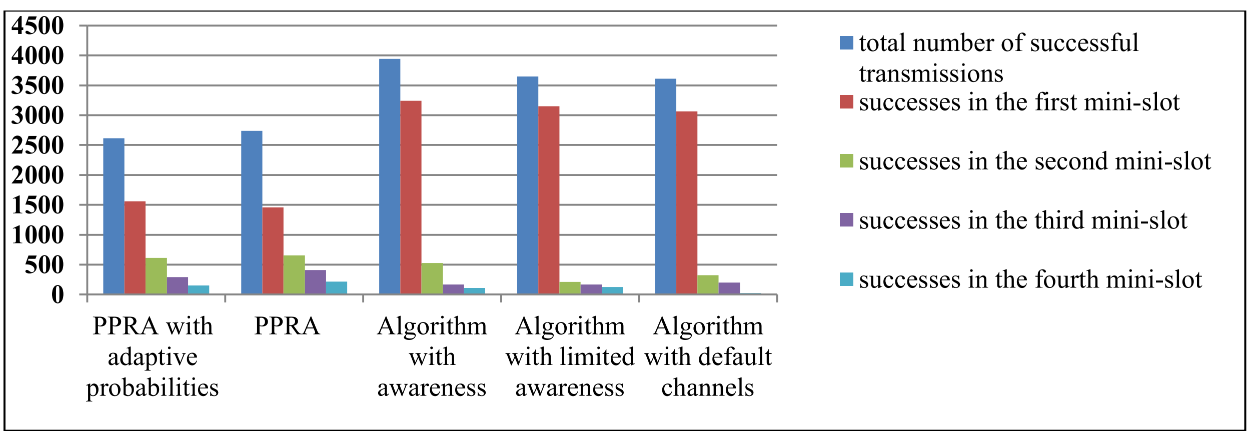


Figure 2.12: Total number of successful transmissions in the case of unequally loaded channels

2.4.3.1.4 The impact of varying the PU arrival rates

Here we examine the impact of various PU arrival rates on the performance metric normalized average secondary throughput. We assume that the PU arrival rates are equal for all the channels and that the sensing probability is set equal to 1.

The results in Fig. 2.13 demonstrate that the *"Algorithm with default channels"*, achieves results close to those of the *"Algorithm with awareness"*. In our simulations we examined the performance of the proposed algorithms, in case the PU arrival rates do not vary abruptly with time during the system operation. Taking this into account, the *"Algorithm with default channels"* requires only one communication between the *Coordinator* and the SUs at the beginning of the system's operation, when the *Coordinator* assigns sensing sequences to the SUs. However, the *"Algorithm with awareness"* requires one communication between the *Coordinator* and the SUs at the beginning of the system's operation in order to assign the pre-selected transmission channels to the SUs for the first mini-slot of each time slot, and in additional requires one communication at the end of each mini-slot of the slot to determine the channels that are occupied by SUs, in order to remove them from the list of candidate channels for assignment. The *"Algorithm with limited awareness"* requires only one communication between the *Coordinator* and the SUs as the *"Algorithm with default channels"*. However, the *"Algorithm with limited awareness"* performs worse because it cannot guarantee that no channel contention arises among the SUs, except from the first mini-slot of each slot. Compared with *"PPRA"* algorithm, the *"Algorithm with awareness"*, the *"Algorithm with limited awareness"* and the *"Algorithm with default channels"*, achieve on average a **37%**, a **27%** and a **35%** improvement on the normalized average throughput of the SUs, respectively.

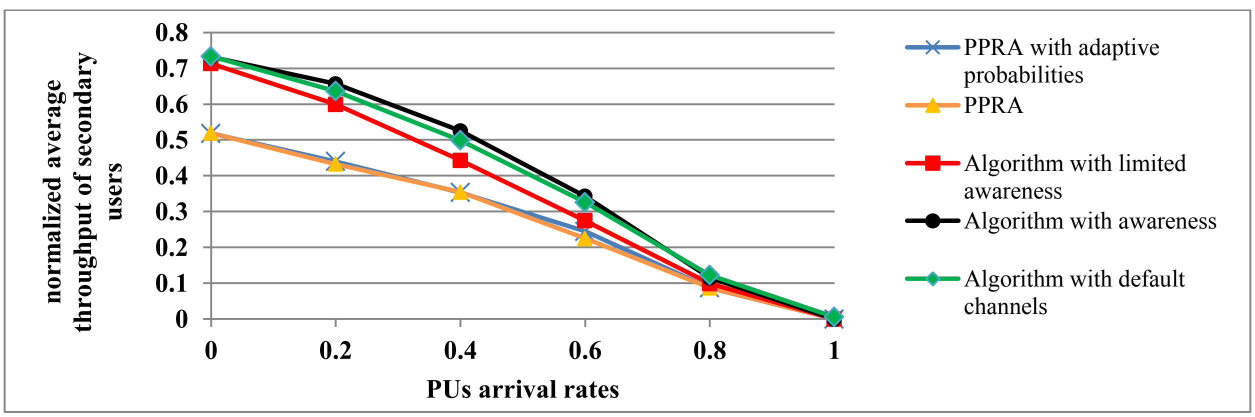


Figure 2.13: The impact of different PU arrival rates on the normalized average throughput of SUs

2.4.3.2 Simulation results when the number of channels is less than the SUs

In the next sections we present the results of the second scenario in which the number of PUs is set equal to 5 and the number of SUs is equal to 8. Note that in this scenario we have fewer channels than SUs and all algorithms experience increased contention among the SUs. The results show that the *"Algorithm with default channels"*, performs well when the sensing probability is relatively small, i.e., between 0.2 and 0.6. As the sensing probability increases above 0.6, the effectiveness of that algorithm decreases because then the contention among the SUs increases. Because of the limited number of channels, all the algorithms perform better when the sensing probability is small.

It is worth recalling that in the algorithms *"Algorithm with awareness"*, *"Algorithm with limited awareness"* and *"Algorithm with default channels",* the illustrated sensing probability in the following figures is applied for all the mini-slots of the slot, because in the examined scenario the number of available channels is less than the number of SUs.

In the following three subsections we present the results for the system performance metrics by examining the impact of "light" primary traffic load, of "heavy" primary traffic load, and of asymmetric primary traffic load among all the channels, as a function of the sensing probability. Subsequently, we set the sensing probability equal to 0.6 and we examine the impact of various PU arrival rates. The sensing probability value of 0.6 was chosen because the algorithms are effective for the specific sensing probability value, in the examined scenario.

2.4.3.2.1 The case of lightly loaded channels

In this section we examine the impact of sensing probability on the performance metrics: normalized average secondary throughput and dropping probability, in the case of lightly loaded channels. Subsequently, the performance metric total number of successes is evaluated, when the sensing probability is set equal to 0.6. We assumed that the PU arrival rates are set equal to 0.2 packets per slot.

In terms of normalized average throughput for the SUs the most efficient algorithm is the *"Algorithm with awareness"*. Subsequently, as it can be seen by the results shown in Fig 2.14, the *"Algorithm with limited awareness"* and the *"Algorithm with default channels"* perform equally well and achieve maximum normalized average throughputs about 0.22 and 0.20, respectively. Note that the *"Algorithm with default channels"* achieves its maximum normalized throughput when the sensing probability is equal to 0.2 and the *"Algorithm with limited awareness"* achieves its maximum normalized throughput when the sensing probability is equal to 0.4. This demonstrates the superiority of the *"Algorithm with default channels"* because smaller sensing probability implies less energy requirements.

As seen from the results presented in Fig 2.15 the *"Algorithm with limited awareness"* achieves the smaller dropping probability. When the sensing probability increases the dropping probability is also increases, because more SUs compete for an idle channel. Note that in case we have fewer channels than SUs all algorithms experience increased contention among the SUs.

As seen from the results presented in Fig 2.16 when the sensing probability is set equal to 0.6 the algorithms *"Algorithm with awareness"*, *"Algorithm with limited awareness"* and *"Algorithm with default channels"* achieve the most of their successful transmissions during the first and the second mini-slot of the slots.

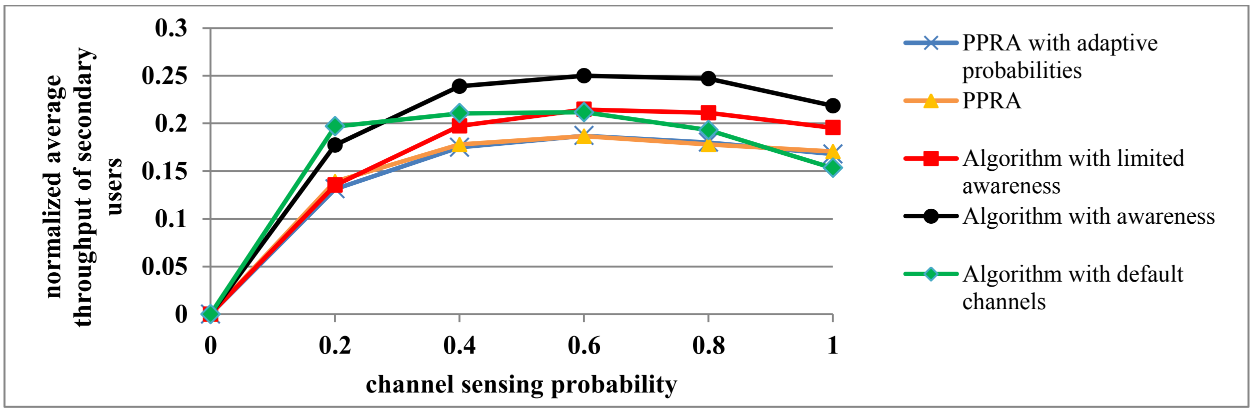


Figure 2.14: The impact of sensing probability on the normalized average throughput of SUs in the case of lightly loaded channels.

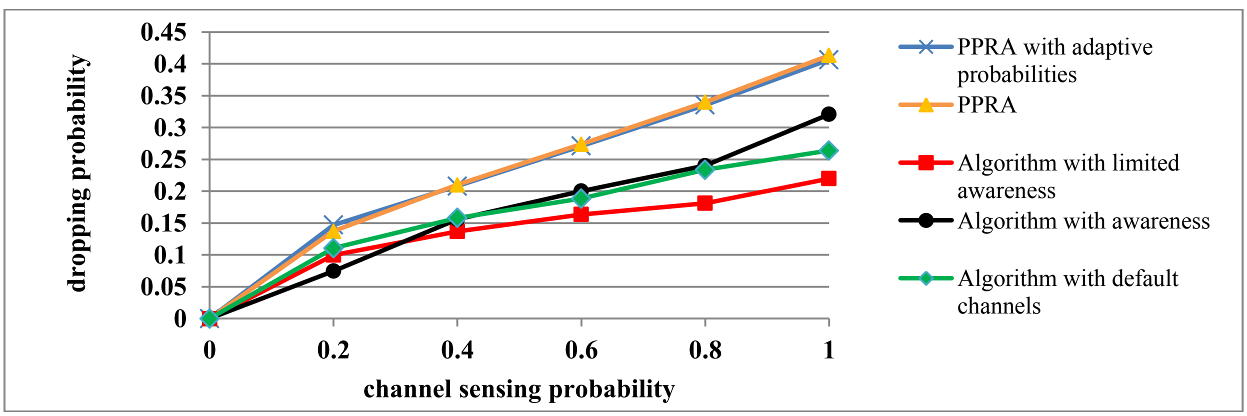


Figure 2.15: The impact of sensing probability on the dropping probability in the case of lightly loaded channels.

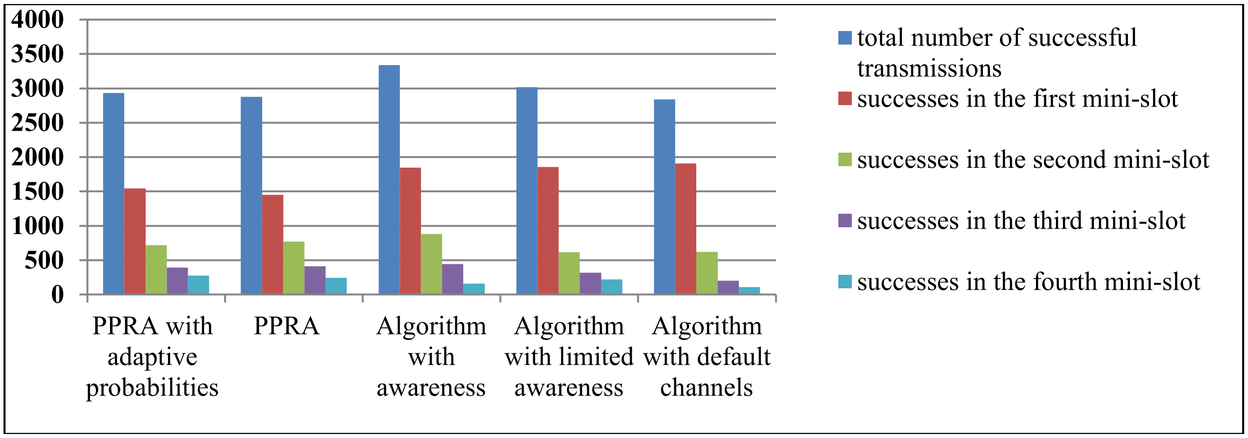


Figure 2.16: Total number of successful transmissions when p=0.6 in the case of lightly loaded channels.

2.4.3.2.2 The case of heavily loaded channels

In this section we examine the impact of sensing probability on the performance metrics: normalized average secondary throughput and dropping probability, in the case of heavily loaded channels. Subsequently, the performance metric total number of successful transmissions is evaluated, when the sensing probability is set equal to 0.6. We assumed that the PU arrival rates are set equal to 0.8 packets per slot.

The algorithms' performance in terms of normalized average throughput and total number of successful transmissions is decreased compared to the case of lightly loaded channels, as was expected (see Fig 2.17 and 2.19). Here about 80% of the slots are occupied by the PUs and in case a channel is idle it is possible that more than one SUs will try to transmit on that channel. As a result the secondary transmission opportunities for SUs become limited because of the intensity of the primary traffic and of the competitions among the SUs.

As seen from the results presented in Fig 2.18 the dropping probability is small, because the channels are occupied with high probability by PUs transmissions and the SUs are not frequently compete for an idle channel.

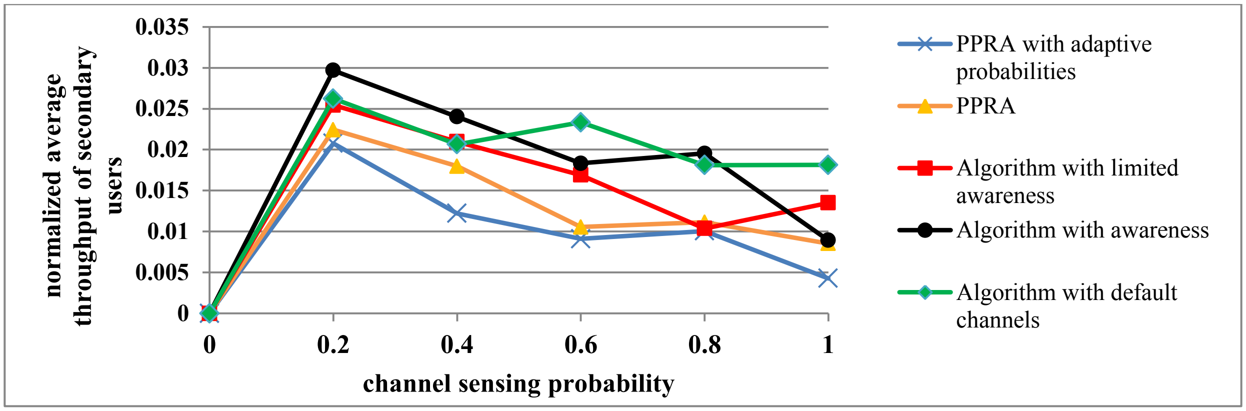


Figure 2.17: The impact of sensing probability on the normalized average throughput of SUs in the case of heavily loaded channels.

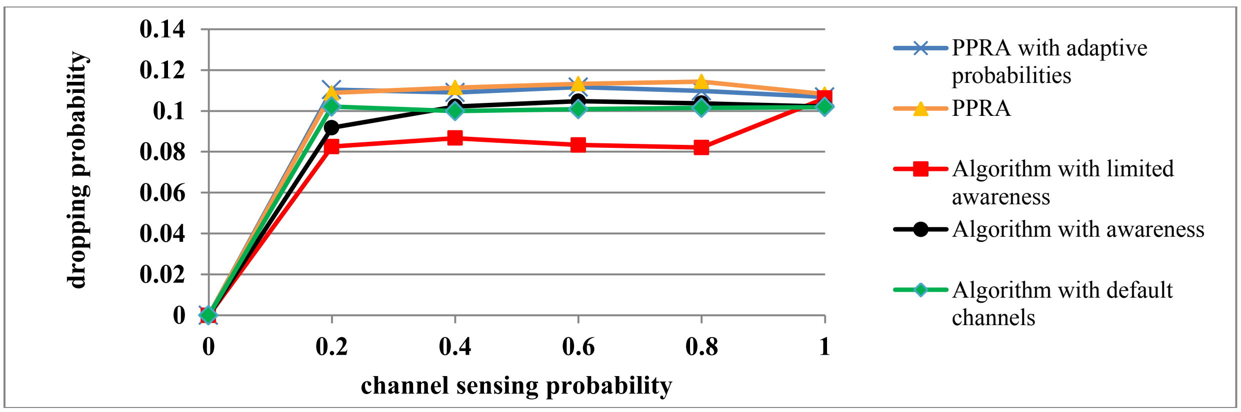


Figure 2.18: The impact of sensing probability on the dropping probability in the case of heavily loaded channels.

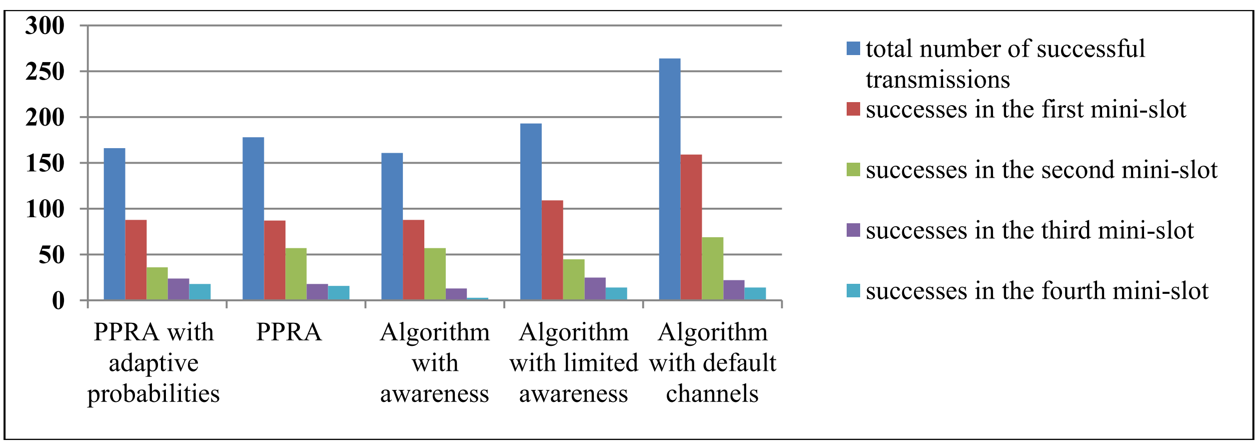


Figure 2.19: Total number of successful transmissions when p=0.6 in the case of heavily loaded channels.

2.4.3.2.3 The case of unequally loaded channels

Here we examine the impact of sensing probability on the performance metrics: normalized average throughput and dropping probability, in the case of unequally loaded channels. The total number of successful transmissions is evaluated when the sensing probability is set equal to 0.6. We assumed that the PU arrival rates are equal to:

$$\lambda_i = \{0.1, 0.2, 0.3, 0.4, 0.5\}, \mathrm{i} = 1, \dots, \mathrm{N_p}=5$$

In terms of normalized average secondary throughput the most efficient algorithm is the *"Algorithm with awareness"*. As seen from the results presented in Fig 2.20 the maximum achievable normalized throughput for the *"Algorithm with limited awareness"* and the *"Algorithm with default channels"* is about 0.17 and 0.16, respectively. Note that the *"Algorithm with default channels"* achieves its maximum normalized throughput when the sensing probability is equal to 0.4 and the *"Algorithm with limited awareness"* achieves its maximum normalized throughput when the sensing probability is equal to 0.6.

As seen by the results in Fig 2.21 the *"Algorithm with limited awareness"* achieves the smaller dropping probability. The *"Algorithm with awareness"* and the *"Algorithm with default channels"* perform equally well. As seen in the case of the algorithms *"PPRA"* and *"PPRA with adaptive probabilities"*, the contention among the SUs for an idle channel is intense. The *"PPRA"* incurs smaller dropping probability compared to the *"PPRA with adaptive probabilities"* algorithm, because the former has smaller probability to find an idle channel and as a result the completion among the SUs is reduced. Recall that in case we have fewer channels than SUs all algorithms experience increased contention among the SUs.

As seen by the results presented in Fig 2.22, when the sensing probability is set equal to 0.6, the algorithms *"PPRA with adaptive probabilities"* and *"PPRA"* achieve more successful transmissions compared to the *"Algorithm with awareness"* and *"Algorithm with default channels"*. However, the last two algorithms achieve the most of their successful transmissions during the first mini-slots of the slots, therefore the slot is exploited for longer duration and that explains the higher achieved throughput.

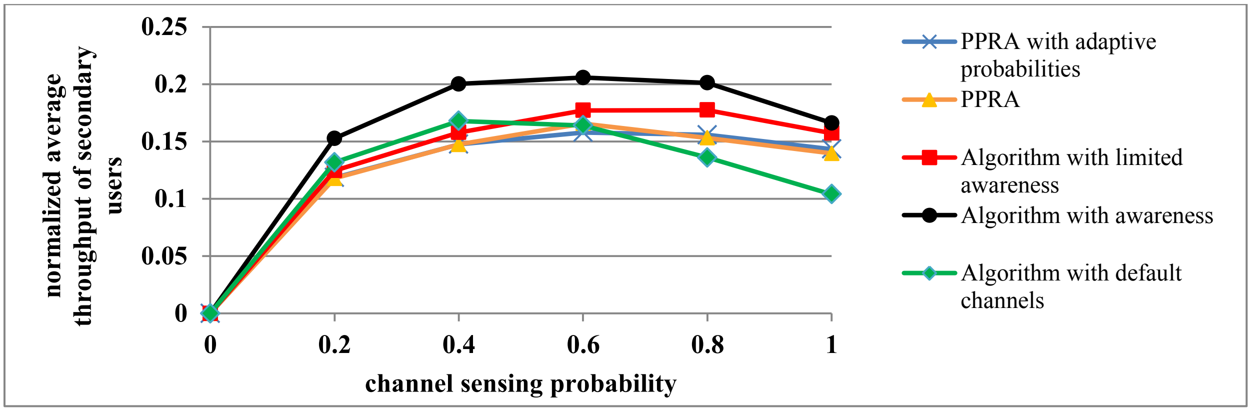


Figure 2.20: The impact of sensing probability on the normalized average throughput of SUs in the case of unequally loaded channels

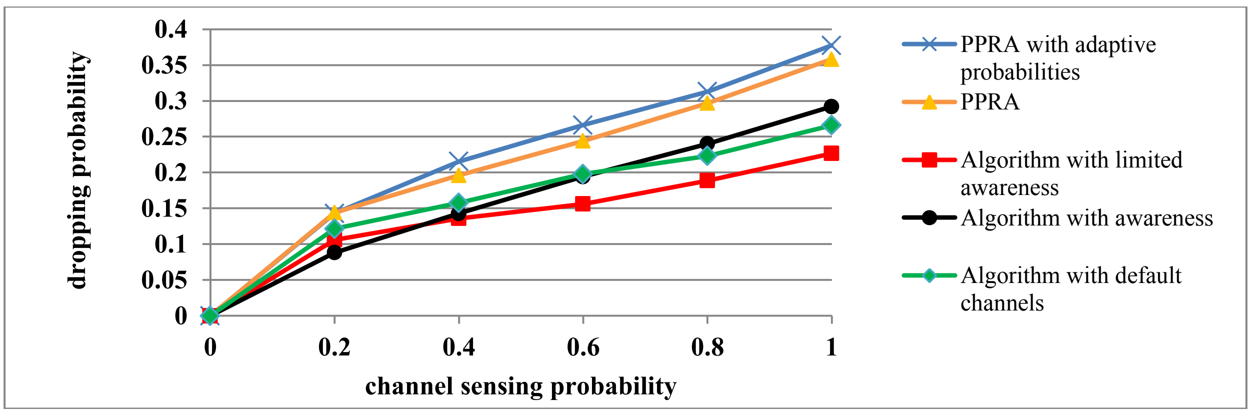


Figure 2.21: The impact of sensing probability on the dropping probability in the case of unequally loaded channels.

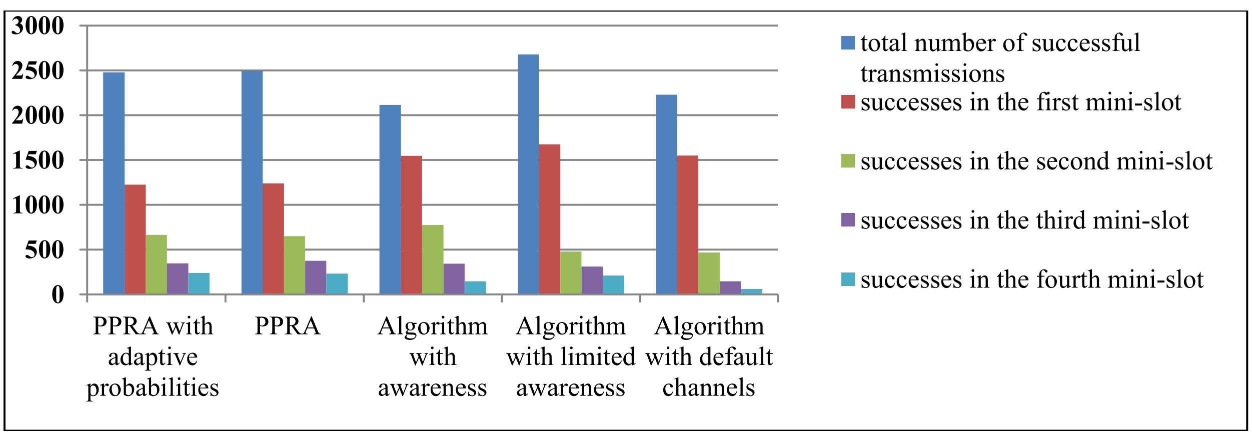


Figure 2.22: Total number of successful transmissions when p=0.6 in the case of unequally loaded channels.

2.4.3.2.4 The impact of varying the PU arrival rates

Here we examine the impact of different PU arrival rates. We assumed that the PU arrival rates are equal for all the channels and that the sensing probability is set equal to 0.6 (since in this case the algorithms perform well). The results in Fig.2.23 show that the *"Algorithm with default channels"*, performs equally well with the *"Algorithm with limited awareness"*. Compared with the results in [1], the *"Algorithm with awareness"*, the *"Algorithm with limited awareness"* and the *"Algorithm with default channels"*, achieve on average a **41.02%**, a **21.88%** and a **33.33%** improvement on the average normalized throughput of the SUs, respectively.

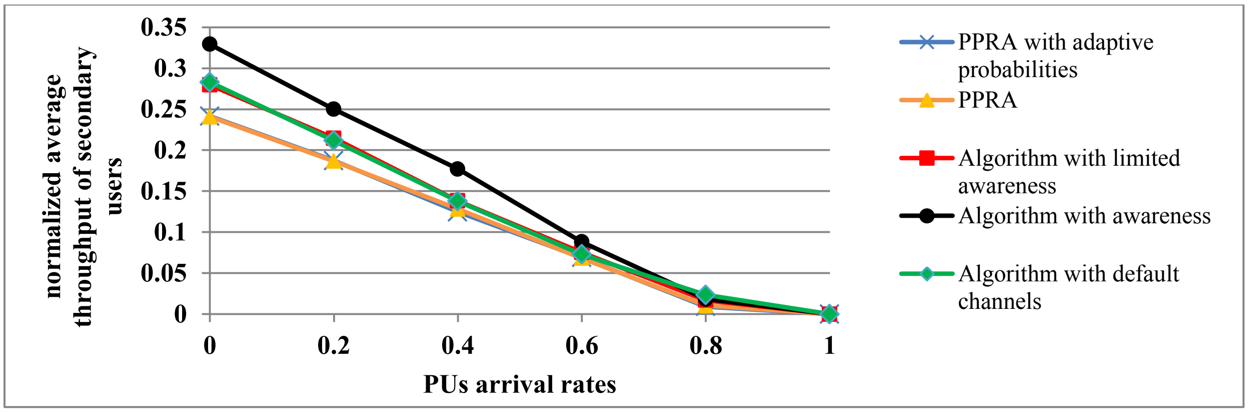


Figure 2.23: The impact of different PU arrival rates on the normalized average throughput of SUs

2.5 Main conclusions

This section presents the main conclusions drawn from our simulation study. From the simulation results on the three performance metrics, the behavior and the effectiveness of each algorithm were examined under the assumption of different conditions.

When the channels are more than the SUs we have seen from the results of the first scenario that the *"Algorithm with default channels"* achieves results close to those of the *"Algorithm with awareness"*, with the difference that the former algorithm requires limited cooperation between the *Coordinator* and the SUs (and not full cooperation) and as a result less amount of energy is expected to be required. Generally, in this scenario we can implement an algorithm, as the *"Algorithm with default channels"*, where the contention among the SUs does not exist. In the *"Algorithm with default channels"*, the evaluated normalized average throughput of the SUs is high because the SUs were distributed among different channels and we can only have collisions between a SU and a PU (due to a missed detection). For the same reason we have small dropping probability and many successful transmissions. The algorithms that require cooperation among the *Coordinator* and the SUs are more effective compared to the algorithms which do not require the SUs to possess centralized, in all examined system performance metrics. We observed that in the latter case the contention among the SUs is strong even when we have enough channels. Finally, we have seen that in case of unequally loaded channels the random channel selection, implemented by *"PPRA"* algorithm, is not effective.

When the channels are less than the SUs, we have seen by the result of the second scenario that the *"Algorithm with default channels"* performs equally well with the *"Algorithm with limited awareness"*, with the difference that the former achieves its good performance when the sensing probability is smaller compared to the sensing probability of the *"Algorithm with limited awareness"*. In this scenario the algorithms are more efficient when they skip the sensing procedure with large probability $(1-p)$, because the contention among the SUs is reduced and transmission opportunities can be found.

Chapter 3: Design and Performance Evaluation of Sequential Channel Sensing Algorithms for Multichannel Heterogeneous CRNs

3.1 Introduction

In this chapter, sequential channel sensing algorithms for multichannel heterogeneous CRNs are studied. In the previous chapter we have examined the case of homogeneous channels, where all the channels have the same transmission rates. In this chapter we study the case of heterogeneous channels, where the channels may have different transmission rates.

More specifically, two new transmission algorithms are introduced and their performance is evaluated via simulations. In the first proposed algorithm we assume that the network's channels are assigned to the SUs by a centralized entity, while in the second algorithm the SUs select their transmission channels without coordination by a centralized entity. When the SUs independently have to search multiple potentially available channels for spectrum opportunities, they face contention from one another in accessing the channels. The end result of this contention is reduced SU throughput due to collisions among SUs or among a SU and the PU that transmits on the same channel.

We are interested in finding a way for distributed SUs to independently reach collision-free sensing orders. Collision-free sensing orders are those in which two or more SUs never simultaneously sense the same channels and therefore never collide with each other. Our results have shown that our proposed algorithms enable the SUs to reduce the likelihood of collisions. More specifically if the probability of missed detection is set equal to 0, the first algorithm that requires coordination by a centralized entity guarantees that no contention issue arises between the SUs during the entire system operation. This is true even if the number of available channels is less than the number of SUs. In the second proposed algorithm, where the SUs cannot cooperate with a centralized entity, if the probability of missed detection is set equal to 0, after a period of time all the SUs will select different sensing sequences (SS) and as a result they will never collide with each other in the remaining time of the system operation.

As in Chapter 2, the PUs are assumed to be oblivious to the presence of the SUs and transmit whenever they have data to send. The structure of a time slot is comprised of several mini-slots corresponding to different stages of spectrum sensing and packet transmission. During the duration of the slot the SUs have a fixed number of opportunities to find an idle channel. We assume that whenever the PUs have a packet for transmission, then the transmission will begin at the beginning of the current slot. Also we assume that the SUs, will always sense the channel before transmission, so that primary transmissions are not interfered by the presence of the SUs. In evaluating the performance of the proposed algorithms, we considered the likelihoods of false alarm and missed detection. Our simulations results show that the two new proposed algorithms improve the normalized average throughput of SUs, compared to the corresponding throughput of the *"γ-persisted strategy"* presented and described in [7].

3.2 Network System Model

The structure of a channel time slot here is the same with the one considered in Chapter 2 (see Fig 2.1). However, here we use different channel selection probabilities. We replace the selection probabilities based on the estimated PU arrival rates:

$$P_j = \frac{1-\lambda_{est,j}}{\sum_{i=1}^{N_p} 1-\lambda_{est,i}}, j = 1, \dots, N_p \tag{3.1}$$

, with selection probabilities based on the channel capacities:

$$P_j = \frac{capacity_j}{\sum_{i=1}^{N_p} capacity_i}, j = 1, \dots, N_p \tag{3.2}$$

The probabilities $P_j, j = 1, \dots, N_p$, denote the selection probabilities for the network channels. This information is considered to be known to the SUs at the beginning of the system operation. There are methods that can be used to estimate the capacity of each network channel, [48]. Recall that the secondary network is assumed saturated, meaning that the SUs always have packets to transmit. Note that if two or more SUs choose the same idle channel, we must consider two different cases. First, only one SU senses the channel as idle and its segmented packet is successfully transmitted during the remaining time of the specific slot. Second, more than one SUs sense the channel as idle; as a result a collision occurs between the SU transmissions and no packet is transmitted successfully.

3.3 The proposed algorithms

In the next sections two new algorithms are introduced for the multichannel heterogeneous CRN, both of which attempt with various techniques to increase the average transmission throughput of the SUs, by exploiting the unused licensed spectrums, owning to the PUs, when an opportunity is found. The first algorithm is referred to as *"Algorithm with default channels and SUs pairs"* and the second as *"Distributed algorithm"*. The first algorithm is based on the assumption that a centralized entity exists which assigns the network's channels to the SUs, while in the second algorithm the SUs independently select the sensing sequences according to which they sense the network channels, without coordination by a centralized entity. In the following two subsections, the proposed algorithms are presented.

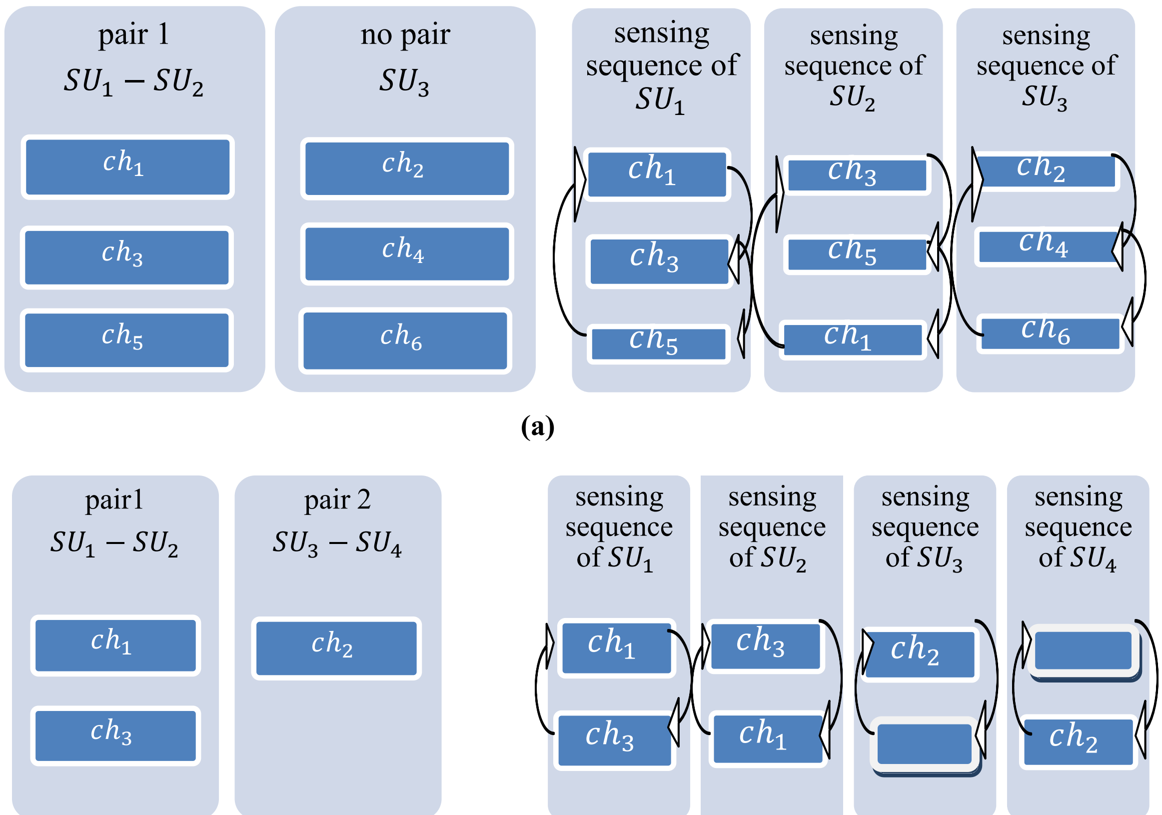
3.3.1 Algorithm with default channels and SUs pairs

We modified the *"Algorithm with default channels"* introduced and described in section 2.3.4 of Chapter 2, so that the SUs are distributed on the channels with higher transmission rates. Also, in the evolved algorithm the collisions between the SUs do not occur even when the number of SUs is larger than the number of available channels.

Firstly, the *Coordinator* creates the SUs pairs. If the number of SUs is even then the *Coordinator* creates $\frac{N_s}{2}$ different SUs pairs. If the number of SUs is odd then the *Coordinator* creates $\frac{N_s-1}{2}$ different SUs pairs and one SU remains without a pair. At the beginning of the system operation (only once during the simulation) the *Coordinator* assigns channels to the SUs pairs in a circular manner so that each channel will be assigned in a unique SU pair. When the channel assignment is completed, the *Coordinator* creates the SS for each SU. At each mini-slot of a slot the SU senses one channel from its SS with probability 1.

The channels are sorted in descending order based on channel capacities using the selection probabilities in (3.2), e.g. $Ch_1, Ch_2, \dots, Ch_{N_p}$ where Ch_1 has the highest capacity. Next, the *Coordinator* assigns cyclically channels to each SUs pair. The channel with the highest capacity is assigned to the first SUs pair; the channel with the next higher capacity is assigned to the second SUs pair, etc. When the channel assignment is completed the *Coordinator* creates different SS for each SU so that each SU will have a different SS compared with the one of its pair. In case a SU pair has been assigned only one channel for transmission, then one SU from that pair will try to transmit in the first mini-slot and in the second mini-slot the same SU will skip the sensing procedure and will continue with the order sense, not sense, sense, not sense, in the remaining mini-slots of each slot. The other SU of the same pair must follow the inverse procedure, e.g. not sense, sense, not sense, sense,… . Figure 3.1, presents an example of system initialization in the case we have: (a) 6 channels and 3 SUs and (b) 3 channels and 4 SUs. The channels are sorted from the channel with the highest capacity to the channel with the lowest capacity. In the first example the SS for SU_1 is $Ch_1, Ch_3, Ch_5, Ch_1, Ch_3, Ch_5 \dots$, the SS for SU_2 is $Ch_3, Ch_5, Ch_1, Ch_3, Ch_5, Ch_1 \dots$, and the SS for SU_3 is $Ch_2, Ch_4, Ch_6, Ch_2, Ch_4, Ch_6 \dots$. In the second example the SS for SU_1 is $Ch_1, Ch_3, Ch_1, Ch_3 \dots$, the SS for SU_2 is $Ch_3, Ch_1, Ch_3, Ch_1 \dots$, the SS for SU_3 is $Ch_2, -, Ch_2, - \cdots$, and the SS for SU_4 is $-, Ch_2, -, Ch_2 \dots$.

As it can be seen from the above algorithmic description, in each mini-slot of a slot each SU has a different transmission channel than the channels of the other SUs. As a result, the SUs never compete for an idle channel. This is true even if the number of available channels is less than the number of the SUs. For reasons of fairness the *Coordinator* must exchange the SUs' SS, after a period of time, so that all the SUs will have equal transmission opportunities on channels with high capacities.



(b)

Figure 3.1: Example of system initialization in case: (a) N_p=6 and N_s=3 (b) N_p=3 and N_s=4

3.3.2 Distributed algorithm

In this section we present an efficient channel sensing order selection strategy for a distributed CRN, where one or more SUs independently select the sensing sequences according to which they sense the available channels, without any coordination by a centralized entity. We propose an approach in which the SS comes from a common predefined Latin Square matrix. A Latin Square is a N_p by N_p matrix of N_p channel indices in which every channel index occurs exactly once in each row and column of the matrix [7].

The numbers in the Latin Square correspond to the channels, e.g. the number 1 corresponds to channel 1. Each SU must select one row from the Latin Square. The selected row corresponds to the SS of the corresponding SU. At the system initialization we assume that the Latin Square has the form shown in Fig.3.2 (a) (in this example we set N_p equal to 6). Note that from the Latin Square Matrix of size N_p by N_p we use only the N_p by δ part because in our study we assume that the number of mini-slots in each slot is equal to δ (as is the length of the SS).

1	2	3	4	5	6
2	3	4	5	6	1
3	4	5	6	1	2
4	5	6	1	2	3
5	6	1	2	3	4
6	1	2	3	4	5

(a)

1	2	3	4
2	3	4	5
3	4	5	6
4	5	6	1
5	6	1	2
6	1	2	3

(b)

Figure 3.2: (a) The Latin Square in the case $N_p = 6$ and $\delta = 4$ (b) the exploited Latin Square

We assume that each SU is equipped with: **(1)** a two position vector in which it stores the index numbers of the slots in which it does not receive a positive acknowledgment (ACK) for its packet transmissions. A SU infers that a collision has occurred whenever it fails to receive an ACK for a transmitted data packet. The abovementioned two position vector is used for the SU to find out when two successive collisions occur with its transmissions during the system operation. As we will see later, the algorithm forces a SU to drop its SS only after it experiences collisions in two consecutive transmissions. Such an event is considered as a strong indication that another SU is using the same SS. Of course there is always a possibility that such an event happens due to a missed detection by the SU. In other words, that the SU failed to detect the presence of a PU transmission or a SU transmission on the channel, thus causing a collision. However, the probability of the latter event happening twice is equal to $(1 - P_d)^2$, which is considered negligible **(2)** a sensing probability $p_i, i = 1, \dots, N_s$ when the contention among the SUs is strong the SUs collide with each other; therefore, the algorithm reduces the probability according to which the SUs decide to sense a channel at the beginning of a slot, and **(3)** a binary variable, referred to as $MySS$, which can take on the values 1 or 0. When $MySS = 1$ for a SU, this indicates that it is highly probable that the particular SU is using a SS, different from the sensing sequences of the other SUs and therefore the particular SU must persist in using the same SS in the next slot. Otherwise, when $MySS = 0$, the SU can drop its SS.

The algorithmic description is presented below:

- For all the SUs, set $MySS = 0$ and the sensing probability $p_i = 1, i = 1, \dots, N_s$.
- Each SU chooses a channel based on the channel capacities, i.e., according to the channel selection probabilities in (3.2).
- The selected by the SU channel, determines the row of the new Latin Square, where the specific channel appears in the first column. In the previous example, if the SU selects channel 2 then from Fig. 3.2 (b) its SS will be the channels 2, 3, 4, and 5.

When multiple independent SUs search multiple potentially available channels for spectrum opportunities, then from an individual's SU perspective one of the following three events can happen in each sensing step, see also the corresponding discussion in [7] : **1)** The SU senses a given channel and it is the only SU to find it free; the SU then has the channel for itself for the remainder of the time slot and it transmits on it; **2)** The SU senses a given channel, finds it occupied by the channel's PU or by another SU, then it continues looking in the next sensing step (mini-slot) of the current slot for another available channel; **3)** The SU senses a given channel, finds it free and transmits on it, but so does at least one other SU; in such case a collision occurs and the SU is not able to transmit until the next time slot, when it again will content for a channel. Another event that may result in a collision is due to the case where the probability of missed detection is not equal to 0 and the SU may mistakenly sense a busy channel as idle and collide with the transmission of the PU, or with the transmission of another SU that already transmits on that channel. Subsequently, we present the way the algorithm addresses the above three events.

As it can be seen by the algorithm's description below (see Table 2.1) in case (a), if the SU successfully transmits in a mini-slot, beyond the first, it cannot set the variable $MySS = 1$ (in other words, it cannot lock the particular SS as its own SS). The importance of the above control is shown by the example where two SUs have chosen the same SS at the beginning of the slot, but one of them did not attempt to transmit in the first mini-slot because of false alarm (i.e., the examined idle channel was mistakenly sensed busy by that SU) and successfully transmits in a subsequent mini-slot of the specific time slot. If the specific control wasn't present the SUs would set $MySS = 1$ and $P_j = 1$ and as a result they would collide in the next time slot. Therefore, the above control prevents the SUs to choose again the same SS in the next time slot. In case the secondary transmission succeeded, the SU increases its sensing probability by a small amount, (equal to 0.1 in our study), since it then assumes that the network is lightly loaded by secondary transmissions. The above increase in the sensing probability is chosen small, because the algorithm must not lead to increased contention among the SUs.

In case (b), where the SU found all the channels in its SS busy, if $MySS = 1$, the SU will persist in using the same SS, because no collision was observed and therefore the SU continues to believe that it has a unique SS, compared to the sequences of the other SUs. However, if $MySS$=0, it is likely that the SU does not have a unique SS compared to the SS of the other SUs, therefore in the next slot it chooses a new SS based on the channel selection probabilities in (3.2).

In the case of collision (case (c)) the SU has to decide whether another SU is using the same SS or the collision is due to a missed detection. The algorithm stipulates that as long as two successive collisions have not been observed, the SU will persist in using the same SS. We have already mentioned that a SU may mistakenly drop its SS due to a missed detection with probability equal to $(1 - P_d)^2$, which is low. Otherwise, the SU will choose another SS based on the channel selection probabilities in (3.2) and it will reduce its sensing probability by a small amount in order to reduce the contention among the SUs. Note that each time the SU collides(with another SU or with a PU) the sensing probability is reduced by a small value 0.1,

TABLE 3.1: THE DISTRIBUTED ALGORITHM

One of three possibilities occurs.

a) The ith SU successfully transmits in the current slot

if (mini-slot == 0) {

$$MySS = 1, P_j = 1, p_i = 1$$

}else{

if ($p_i + 0.1 \leq 1$)

$$p_i = p_i + 0.1$$

}

b) The ith SU found all the channels in its SS busy

if ($MySS = 1$)

$$p_i = 1, P_j = 1,$$

else if ($MySS = 0$) {
Choose one SS from the Latin Square based on the probabilities in (3.2).
}

c) Collision

If the ith SU believes that it has a unique SS ($i.e\ MySS == 1$) then it will drop out its SS only after it experiences collisions in two consecutive transmissions. In that case must updates its variables as:

1. $MySS = 0$
2. in the next slot will choose one SS from the Latin Square based on the probabilities in (3.2).
3. if ($p_i - 0.1 \geq 0.5$)

$$p_i = p_i - 0.1$$

Otherwise, if the ith SU has ($MySS == 1$) and the SU does not observes two successive collision then the SU insists in using the same SS and updates its variables as:

$$p_i = 1, P_j = 1$$

If the ith SU has not a unique SS ($i.e\ MySS == 0$) then the SU:
1. in the next slot will choose one SS from the Latin Square based on the probabilities in (3.2).
2. if (successive collisions ≥ 2){
   if ($p_i - 0.1 \geq 0.5$)
   $p_i = p_i - 0.1$
   }}

because we want the algorithm to gradually reduce the contention among the SUs. In the worst case, where the ith SU experiences many successive collisions, its sensing probability may become as low as $p_i = 0.5$. This is because we do not want the SUs to drop out from the network.

3.4 Performance Evaluation

3.4.1 Performance metrics

In our simulation we study the following performance metric: the average number of Mbits of secondary user traffic transmitted in each time slot. For example if a SU transmits for a period of t sec on a channel with capacity C Mbps, then the specific SU transmits $(t * C)$ Mbits. The average secondary user Mbits transmitted in each slot is calculated as: $\frac{1}{N_1}\sum_{k=1}^{N_1}\frac{1}{N_s * N}\sum_{i=1}^{N_s} bits_i^{tot}$, where $bits_i^{tot}$ is the total Mbits transmitted by the ith SU at the end of the kth simulation run, N is the total number of slots per simulation run and N_1 is the total number of simulation runs.

3.4.2 Simulation model

We use the same simulation model with the one in Chapter 2. For the details we refer to section 2.4.2. We consider a time slotted (synchronous) CRN with N_s SUs, which attempt to opportunistically transmit on the N_p channels, each dedicated to one of the N_p PUs. Each PU is either present for the entire time slot, or absent for the entire time slot. We assume that the PU's packets arrive according to a Poisson process. Let λ_i (arrivals/slots) be the arrival rate of the PU packets at channel $i, i = 1, \dots, N_p$. We further assume that the channels bandwidths are uniformly distributed in the interval [470-862] MHz ([22], [23]). The capacity of channel' i is calculated by:

$$capacity_i = B_i * \log_2(1 + \mathrm{SNR}), \text{ where } \mathrm{B_i}, i = 1, \dots, N_p \text{, is the channel's } i \text{ bandwidth} \qquad (3.3)$$

The average number of Mbits of secondary user traffic transmitted in each time slot is computed by simulating each scenario for a time period of 1000 time slots and each simulation run is independently repeated 1000 times. The default values for all the system parameters are shown in Table 3.2.

To compare the performance of the proposed algorithms we examined a scenario in which the contention between the SUs is not strong ($N_p > N_s$), a scenario in which the contention between the SUs is strong ($N_s > N_p$) and finally a scenario where the number of SUs is equal to the number of PUs. In the first scenario the number of available channels is larger than the number of SUs and we set $N_p = 10$ and $N_s = 5$. In the second scenario the channels are less than the SUs and we set $N_p = 10$ and $N_s = 13$. In the third scenario we set $N_p = N_s = 10$. In the sequel, each scenario is divided into three sub-scenarios and we examined the impact of lightly loaded channels, of heavily loaded channels and finally of asymmetrically loaded channels. The probability of false alarms and the probability of missed detection are defined in the following sub-sections. Following, the results are presented and discussed.

TABLE 3.2*: SYSTEM PARAMETER AND DEFAULT VALUES*

Notation	Definition (default value)
T	Time slot duration (10ms)
t_s	Required time for handover (0.01ms)
t	Required sensing time (2.4ms)
N	Number of time slots per simulation run (1000)
N_1	Number of simulation runs (1000)
SNR	Signal to noise ratio (15db)
B	Bandwidth - Uniformly distributed in [470-862] MHz
δ	Maximum number of mini-slots in each slot (4)

3.4.3 Simulation results

We conducted event-driven simulations to evaluate the performance of the proposed algorithms: *"Algorithm with default channels and SUs pairs"* and *"Distributed algorithm"*, which employs the best algorithm from [7], namely the *"γ-persistent strategy"*. Sections 3.4.3.1, 3.4.3.2 and 3.4.3.3 present the results of the first, the second and the third scenarios, respectively.

3.4.3.1 Simulation results in case the channels are more than the SUs

In the next sections we present the results of the first scenario in which the number of PUs is set equal to 10 and the number of SUs is set equal to 5. The results shown in the first figure of the 1st, 2nd and 3rd sub-sections, the probability of missed detection is set equal to 0 and the probability of false alarm varies. In the second figure of the 1st, 2nd and 3rd sub-sections, the probability of false alarm was set equal to 0.2 and the probability of missed detection varies. Finally, in the fourth sub-section where the PU arrival rates vary, the probability of false alarm was set equal to 0.1 and the probability of correct detection was set equal to 0.9.

The results demonstrate that the *"Algorithm with default channels and SUs pairs"* outperforms the algorithms *"Distributed algorithm"* and *"γ-persistent strategy"*, in all examined scenarios. Recall, that in the former algorithm a centralized entity exists and assigns the network channels to the SUs while in the latter algorithms the SUs independently select their transmission channels without coordinator by a centralized entity. As seen by the following results, the *"Distributed algorithm"* achieves results close to those of the *"γ-persistent strategy"*, when the probability of missed detection is set equal to 0, in the examined scenario. However, the former algorithm behaves more efficiently when the probability of missed detection is positive. Note that in this scenario the SUs are less than the number of available channels and in the *"Distributed algorithm"* the SUs more frequently select the channels with higher capacities, compared to the *"γ-persistent strategy"*.

In the following three subsections we present the results for the system performance metric by examining the impact of "light" primary traffic load, of "heavy" primary traffic load, and of asymmetric primary traffic load, as a function of the probability of false alarm and the probability of missed detection. Subsequently, we examine the impact of various PU arrival rates in case the number of channels is larger than the number of SUs.

3.4.3.1.1 The case of lightly loaded channels

In this section we examine the impact of varying the probabilities of false alarm and missed detection on the performance metric in the case of lightly loaded channels. We assume that the PU arrival rates are equal for all the channels and are set equal to 0.3 packets per slot.

The results in Fig 3.3 show the impact of varying the probability of false alarm on the average secondary transmitted Mbits per slot. In the examined scenario the number of channels is larger than the number of SUs; therefore the average Mbits transmitted by the SUs in a time slot increase by using the channels with higher capacities. In the *"Algorithm with default channels and SUs pairs"* no secondary contention takes place during the entire system operation since all the SUs have different SS and as a result they do not compete with each other for an idle channel. As already mentioned in the algorithmic description of the aforementioned algorithm no primary or secondary collisions occur since the probability of missed detection was set equal to 0; that explains the high achieved average secondary transmitted Mbits per slot. Compared with the results in [7], the *"Algorithm with default channels and SUs pairs"* achieves on average a **9.85%** improvement on the average secondary transmitted Mbits per slot.

The results in Fig 3.4 show the impact of varying the probability of missed detection on the performance metric. The probability of false alarm was set equal to 0.2. As expected the average Mbits transmitted by the SUs in a time slot decrease as the probability of missed detection increases. The algorithms *"Distributed algorithm"* and *"γ-persistent strategy"*, guarantee that the SUs will not compete for an idle channel if all the SUs choose different rows from the Latin Square Matrix. However, if the probability of missed detection is not equal to 0 a busy channel may incorrectly be sensed as idle; therefore the SUs will collide with PUs or with SUs that already transmit on the selected channels. In the *"Distributed algorithm",* if a SU experiences collisions in two consecutive transmissions it must drop the specific SS and choose another one in the next slot using equation (3.2). Although, in the *"Distributed algorithm"* the SUs do not possess centralized information, as happens in the *"γ-persistent strategy"*, the former outperforms the latter because in the *"Distributed algorithm"* the channels with higher transmission capacities are selected by the SUs. In the *"γ-persistent strategy",* the SU has to choose one row from the Latin Square based on probabilities that each SU builds during the system operation. The specific probabilities do not correlate with the channel transmission capacities, but are correlated to channels' collision probabilities. So the SU can choose an inefficient row from the Latin Square and persist in that row, while at the same time other more efficient rows may be available. Compared with the results in [7], the *"Algorithm with default channels and SUs pairs"* and the *"Distributed algorithm"*, achieve on average a **10.5%** and a **2.7%** improvement on the average secondary transmitted Mbits per slot, respectively.

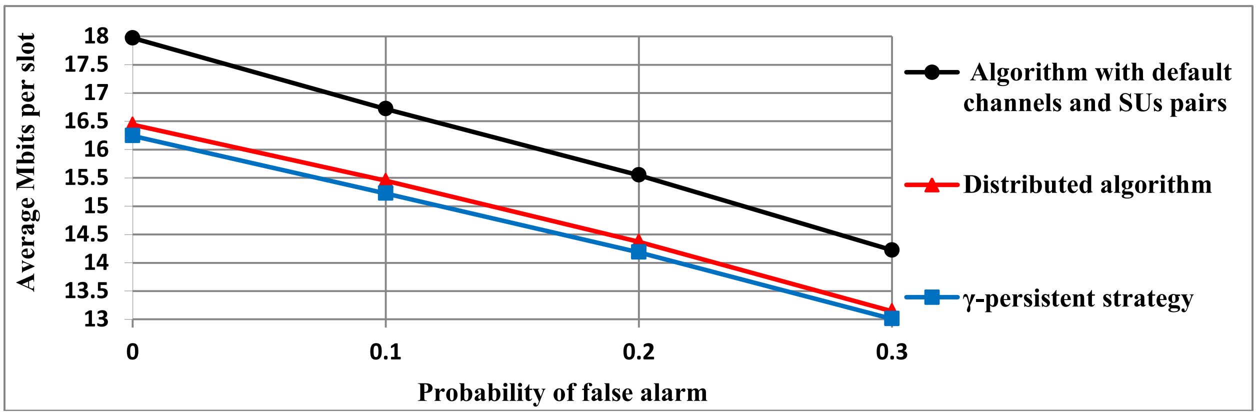


Figure 3.3: The impact of probability of false alarm on the average secondary transmitted Mbits per slot, in the case of lightly loaded channels.

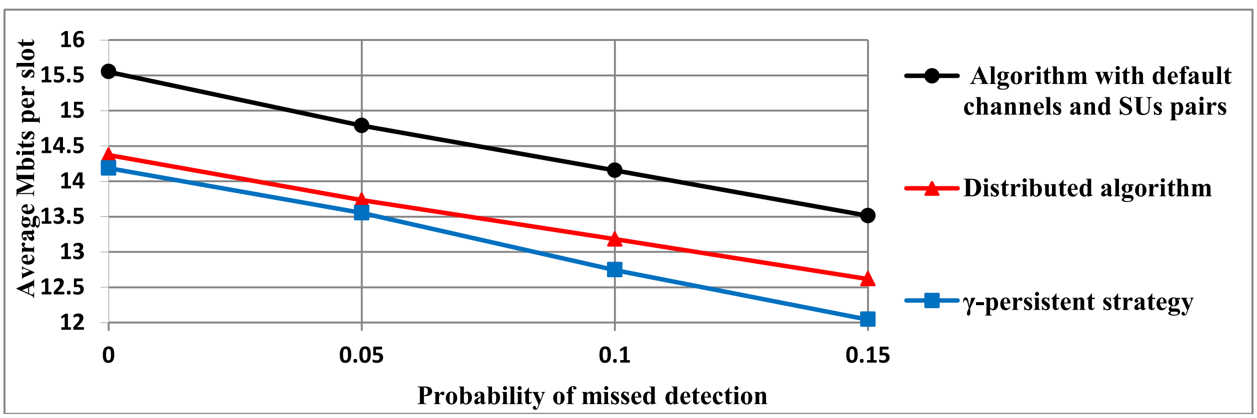


Figure 3.4: The impact of probability of missed detection on the average secondary transmitted Mbits per slot, in the case of lightly loaded channels.

3.4.3.1.2 The case of heavily loaded channels

Here we examine the impact of varying the probabilities of false alarm and missed detection on the performance metric, in the case of heavily loaded channels. We assume that the PU arrival rates are equal for all the channels and are set to 0.7 packets per slot.

The results in Fig 3.5 show the impact of varying the probability of false alarm on the performance metric. As expected the achieved average secondary transmitted Mbits per slot is small, because about 70% of the slots are occupied by the PUs transmissions and the secondary transmission opportunities have dramatically decreased. Note that from the remaining slots, the SUs lose 24% of them in order to sense the channels. The *"Algorithm with default channels and SUs pairs"* achieves the highest average secondary transmitted Mbits per slot, since the SUs do not experience collisions with each other and the transmission opportunities are higher compared to the ones in the other algorithms, where the SUs may compete for the same idle channel if more than one SUs choose the same row of the Latin Square. Note that the probability of missed detection is set equal to 0; therefore the SUs always sense a busy channel as busy. Compared with the results in [7], the *"Algorithm with default channels and SUs pairs"* achieves on average a **7.27%** improvement on the average secondary transmitted Mbits per slot.

The results in Fig 3.6 show the impact of varying the probability of missed detection on the performance metric. The SUs have a limited number of transmission opportunities because of the heavy primary traffic load and as the value of the probability of missed detection increases the SUs do not exploit the limited transmission opportunities and the collisions between the SUs and the PUs increase. Compared to the results of the *"γ-persistent strategy"* in [7], the *"Algorithm with default channels and SUs pairs"* achieves on average a **7.91%** improvement on the average secondary transmitted Mbits per slot.

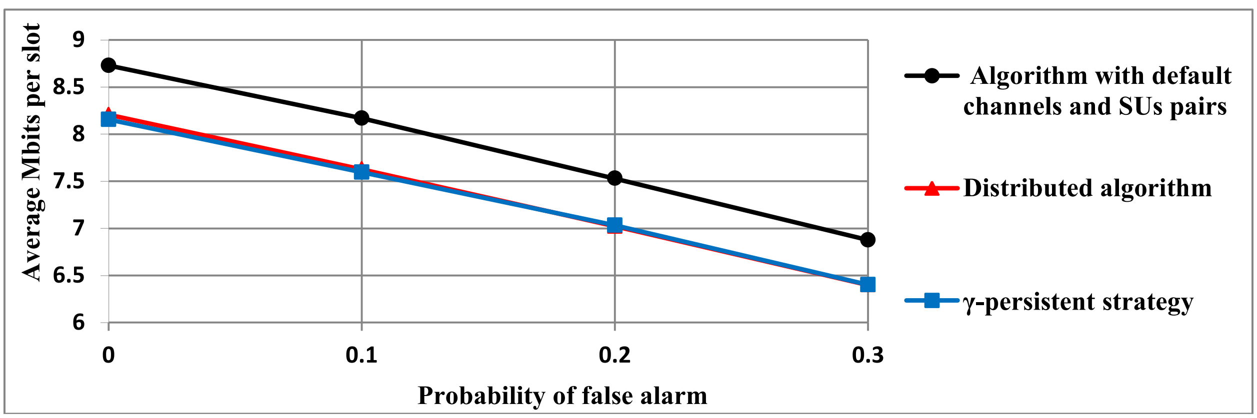


Figure 3.5: The impact of probability of false alarm on the average secondary transmitted Mbits per slot, in the case of heavily loaded channels.

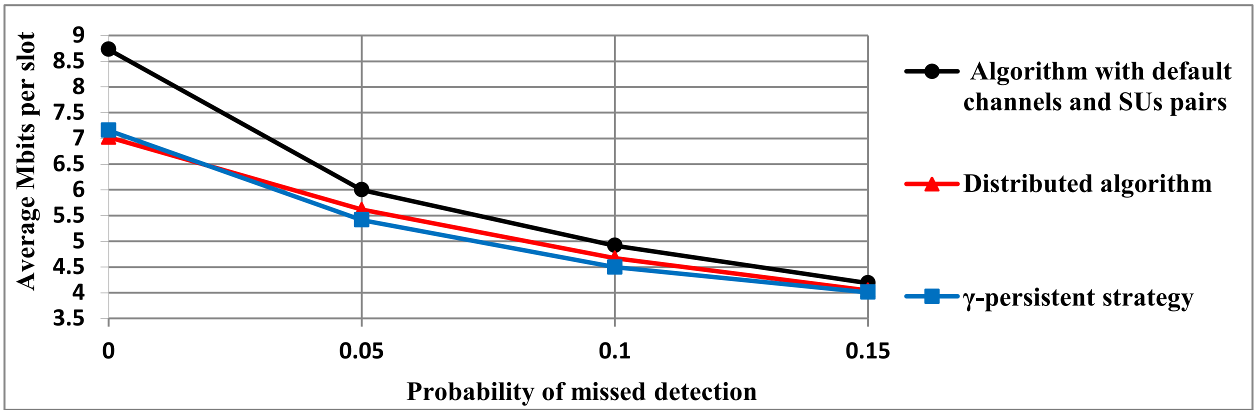


Figure 3.6: The impact of probability of missed detection on the average secondary transmitted Mbits per slot, in the case of heavily loaded channels.

3.4.3.1.3 The case of unequally loaded channels

In this section we examine the impact of varying the probabilities of false alarm and missed detection on the performance metric, in the case of unequally loaded channels. We assume that the PU arrival rates are equal to:

$$\lambda_i = \{0.1, 0.1, 0.2, 0.2, 0.3, 0.3, 0.4, 0.4, 0.5, 0.5\}, \mathrm{i} = 1, \dots, \mathrm{N_p} = 10$$

In Fig 3.7 we plot the average secondary transmitted Mbits per slot as a function of the false alarm probability. As seen by the presented results the *"Distributed algorithm"* outperforms the *"γ-persistent strategy"*. This shows that it is more efficient to select the channels with higher capacities, instead of randomly selecting the channels and persist with them if no collisions are observed, as happens in the *"γ-persistent strategy"*. For example, in the *"γ-persistent strategy"* a SU is possible to select a row from the Latin Square with the first channel be the channel with the lowest capacity and to persist with that row until a collision with another SU takes place. If no collision takes places the SU will persist with the same SS until the end of the system operation. Note that the probability of missed detection is set equal to 0 and secondary-primary collisions do not take place. Generally in each row of the Latin Square the first channel from each row is the first channel that will be used by the SU in the first mini-slot of each slot. If the specific channel has high capacity and low primary traffic load then the SU will frequently find transmission opportunities with high transmission rate. Compared to the results of the *"γ-persistent strategy"* in [7], the *"Algorithm with default channels and SUs pairs"* and the *"Distributed algorithm"*, achieve on average a **10.43%** and a **3.16%** improvement on the average secondary transmitted Mbits per slot, respectively.

The results in Fig 3.8 show the impact of varying the probability of missed detection on the performance metric, in case of unequally loaded channels. In case a SU experiences collision, the *"Distributed algorithm"* persists in keeping the SS of each SU for a longer period of time, compared to the *"γ-persistent strategy"*, because the former algorithm forces a SU to drop its SS only after it experiences collisions in two consecutive transmissions. This control prevents the SU from leaving by mistake its SS, e.g. due to collision occurring due to a miss detection. Compared with the results of the *"γ-persistent strategy"* in [7], the *"Algorithm with default channels and SUs pairs"* and the *"Distributed algorithm"*, achieve on average a **8.32%** and a **3.13%** improvement on the average secondary transmitted Mbits per slot, respectively.

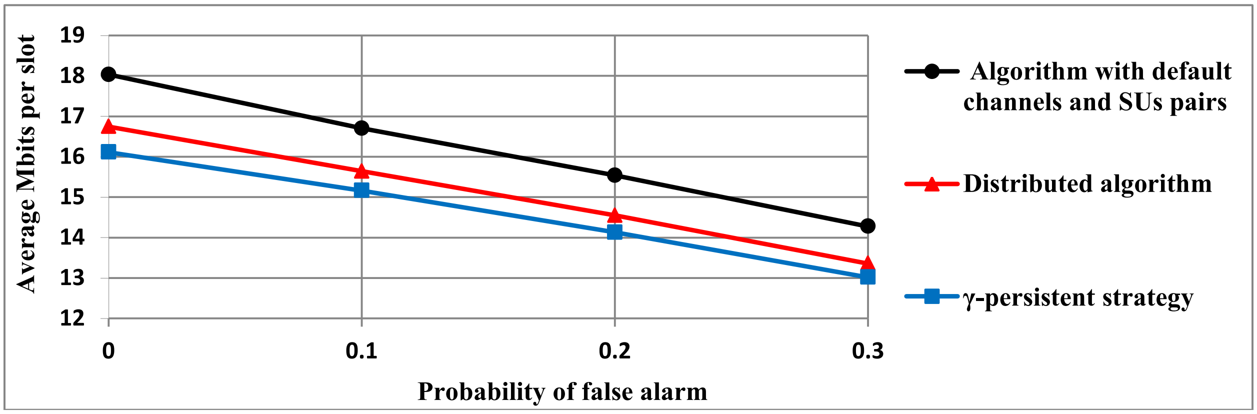


Figure 3.7: The impact of probability of false alarm on the average secondary transmitted Mbits per slot, in the case of unequally loaded channels.

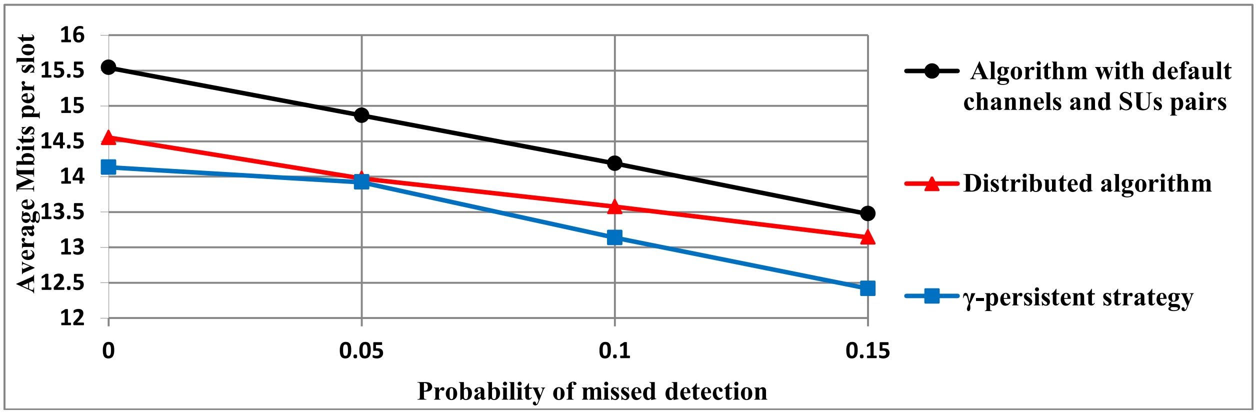


Figure 3.8: The impact of probability of missed detection on the average secondary transmitted Mbits per slot, in the case of unequally loaded channels.

3.4.3.1.4 The impact of varying the PU arrival rates

Here we examine the impact of various PU arrival rates on the performance metric. We assume that the PU arrival rates are equal for all the channels. The probability of false alarm was set equal to 0.1 and the probability of correct detection was set equal to 0.9.

As shown by the illustrated results in Fig 3.9 the *"Algorithm with default channels and SUs pairs"* achieves the highest average secondary transmitted Mbits per slot. When the PU arrival rates increase, the performance of the algorithms decrease because the channels are occupied by the PUs transmissions with higher probability and the secondary transmission opportunities become limited. Compared with the results of the *"γ-persistent strategy"* in [7], the *"Algorithm with default channels and SUs pairs"* achieves on average a **8.2%** improvement on the average secondary transmitted Mbits per slot.

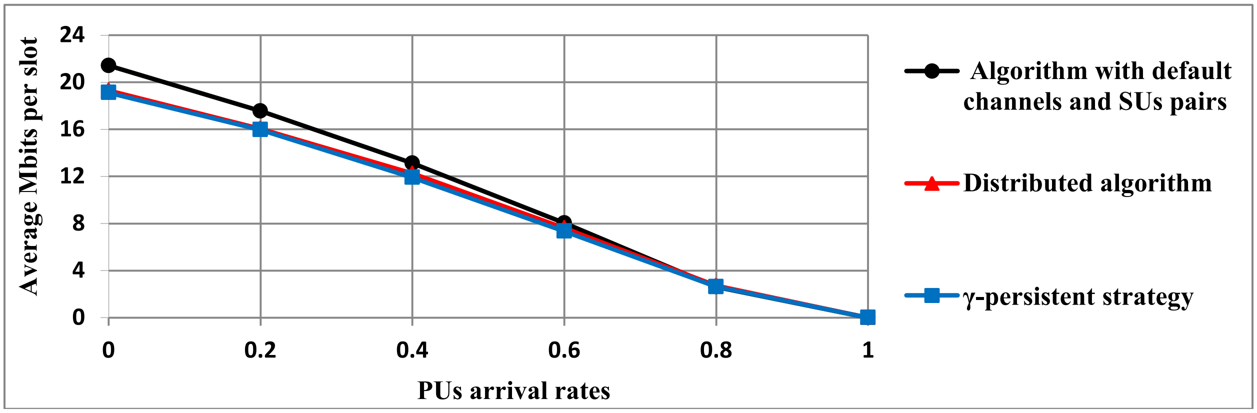


Figure 3.9: The impact of varying the PU arrival rates on the average secondary transmitted Mbits per slot, in the case the number of SUs is less than the number of PUs.

3.4.3.2 Simulation results in case the channels are less than the SUs

In the next sections we present the results of the second scenario. We assume that the number of PUs is set equal to 10 and the number of SUs is set equal to 13. In the first figure of each subsection we set the probability of missed detection equal to 0 while the probability of false alarm varies. In the second figure of each subsection we set the probability of false alarm equal to 0.2 while the probability of missed detection varies.

When $N_p < N_s$ it is not possible for all the SUs to converge to collision-free sensing orders. However, the results show that for $N_p < N_s$, our proposed algorithms enable the SUs to reduce the likelihood of collisions.

The results show that the *"Algorithm with default channels and SUs pairs"* outperforms the algorithms *"Distributed algorithm"* and *"γ-persistent strategy"*, in all the examined scenarios. The following results demonstrate the superiority of the *"Distributed algorithm"* compared to the *"γ-persistent strategy"*. Note that here the number of SUs is larger than the number of available channels and as expected the average secondary transmitted Mbits per slot are lower, compared to the corresponding results in the previous scenario.

In the following three subsections we present the results for the system performance metric by examining the impact of "light" primary traffic load, of "heavy" primary traffic load, and of asymmetric primary traffic load, as a function of the probability of false alarm and the probability of missed detection. Subsequently, we examine the impact of various PU arrival rates.

3.4.3.2.1 The case of lightly loaded channels

Here we examine the impact of varying the probabilities of false alarm and missed detection on the performance metric, in the case of lightly loaded channels. We assume that the PU arrival rates are equal for all the channels and are set equal to 0.3 packets per slot.

The results in Fig 3.10 show the impact of varying the probability of false alarm on the average secondary transmitted Mbits per slot. In the examined scenario the number of channels is less than the number of SUs; therefore the average secondary transmitted Mbits per slot are lower than the corresponding results in Fig 3.3, where the number of SUs was less than the number of channels. Although the channels are not sufficient to provide each SU with a different transmission channel in the *"Algorithm with default channels and SUs pairs"* no secondary contention takes place during the entire system operation because two different events are possible: **(a)** The SUs in the same pair have more than one available channels and the *Coordinator* creates different SS for each SU. The end result of that case will be that the SUs will not compete for an idle channel during the system operation. **(b)** The SUs in the same pair have only one available channel. The first SU must sense that channel alternatively with its pair. Note that in the specific simulation the probability of missed detection was set equal to 0 and as a result in that algorithm there are neither primary nor secondary collisions, and this explains the high achieved average secondary transmitted Mbits per slot.

The *"Distributed algorithm"* outperforms the *"γ-persistent strategy"* because in that algorithm the channels with high transmission rates are more frequently selected for secondary transmissions. In addition, that algorithm adaptively changes the sensing probability of each SU, depending on the successful or not successful transmission attempts occurring during the system operation. As already mentioned in the algorithmic description if the SU experience more than two successive collisions it must reduce its sensing probability by a small amount equal to 0.1 in the next slot, if the SU successfully transmits in a slot then its sensing probability is increased by a small amount equal to 0.1 and if the SU has its own SS then its sensing probability is set equal to 1. That mechanism gradually reduces or increases the potential contention among the SUs, when it is considered necessary. Compared with the results of the *"γ-persistent strategy"*, the *"Algorithm with default channels and SUs pairs"* and the *"Distributed algorithm"*, achieve on average a **35.89%** and a **15.48%** improvement on the average secondary transmitted Mbits per slot, respectively. Although the achieved average secondary transmitted Mbits per slot in Fig 3.10 are less, compared to the corresponding results in Fig 3.5, where the number of SUs was less than the number of channels, the percentage of improvement that is achieved by the *"Distributed algorithm"* is increased here. This is due to the adaptive change of the sensing probability of the SUs which takes place more frequently in the specific scenario, where the contention among the SUs is higher.

The results in Fig 3.11 show the impact of varying the probability of missed detection on the performance metric. The *"Distributed algorithm"* outperforms the *"γ-persistent strategy"*, because the former persists in keeping the SS of a SU for a longer period of time. This prevents the SU from mistakenly dropping its SS, for example because of a collision that occurred due to a missed detection. Compared with the results of the *"γ-persistent strategy"* in [7], the *"Algorithm with default channels and SUs pairs"* and the *"Distributed algorithm"*, achieve on average a **42.29%** and a **16.09%** improvement on the average secondary transmitted Mbits per slot, respectively.

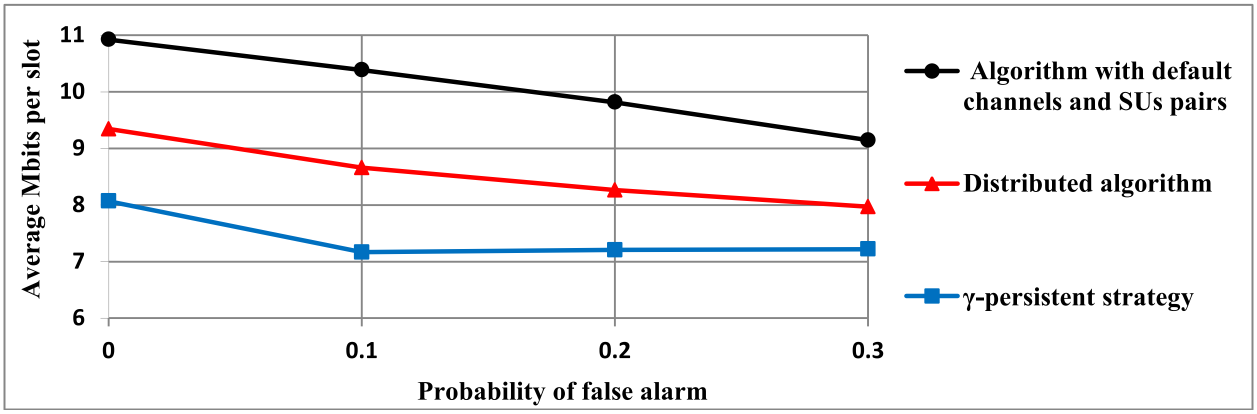


Figure 3.10: The impact of probability of false alarm on the average secondary transmitted Mbits per slot, in the case of lightly loaded channels.

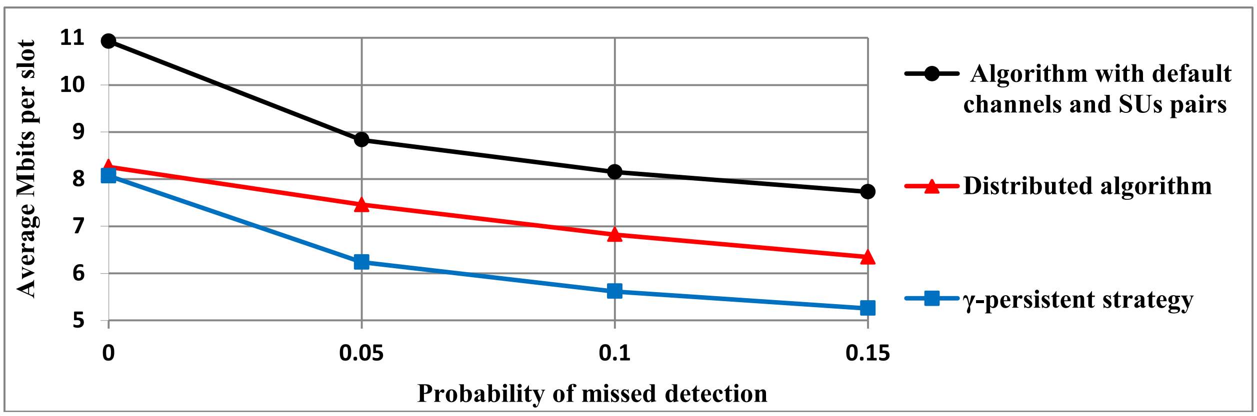


Figure 3.11: The impact of probability of missed detection on the average secondary transmitted Mbits per slot, in the case of lightly loaded channels.

3.4.3.2.2 The case of heavily loaded channels

In this section we examine the impact of varying the probabilities of false alarm and missed detection on the performance metric, in the case of heavily loaded channels. We assume that the PU arrival rates are equal for all the channels and are set equal to 0.7 packets per slot.

In Fig 3.12 we plot the average secondary transmitted Mbits per slot, as a function of the false alarm probability. As seen from the results, the *"Distributed algorithm"* achieves higher average secondary transmitted Mbits per slot, compared to the *"γ-persistent strategy"*. Note that in the examined scenario the available channels are not enough to serve all the secondary transmissions and the primary load is heavy. As the conditions are aggravated, e.g. the probability of false alarm is increased, the secondary transmissions opportunities are limited and that explains the low achieved values for the performance metric. Compared with the results of the *"γ-persistent strategy"*, the *"Algorithm with default channels and SUs pairs"* and the *"Distributed algorithm"*, achieve on average a **24.29%** and a **9.81%** improvement on the average secondary transmitted Mbits per slot, respectively.

Fig 3.13 evaluates the effect of varying the probability of missed detection on the performance metric. Compared with the results of the *"γ-persistent strategy"*, the *"Algorithm with default channels and SUs pairs"* and the *"Distributed algorithm"*, achieve on average a **43.91%** and a **13.81%** improvement on the average secondary transmitted Mbits per slot, respectively.

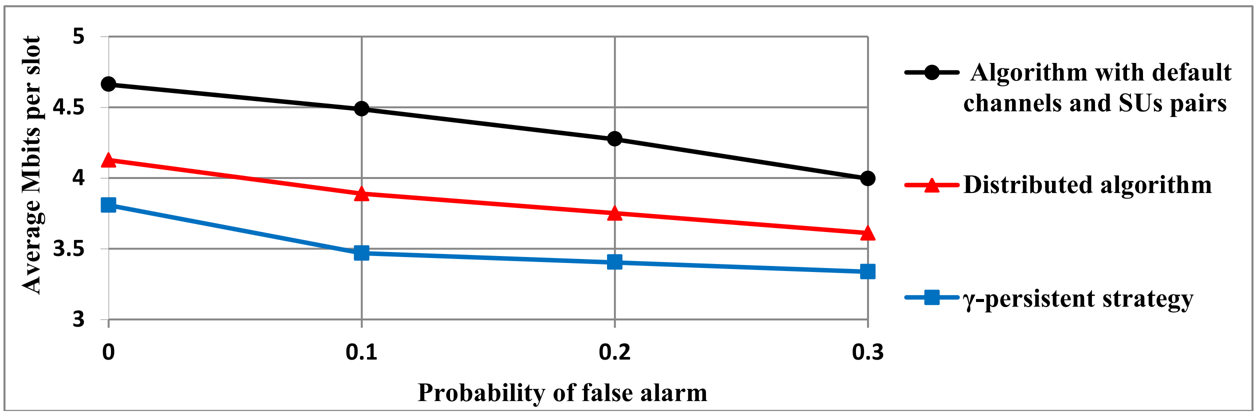


Figure 3.12: The impact of probability of false alarm on the average secondary transmitted Mbits per slot, in the case of heavily loaded channels.

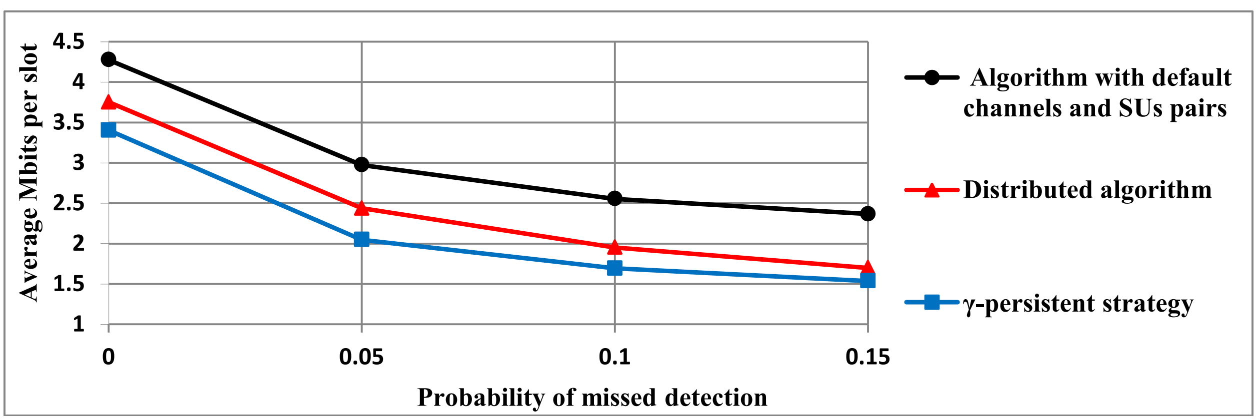


Figure 3.13: The impact of probability of missed detection on the average secondary transmitted Mbits per slot, in the case of heavily loaded channels.

3.4.3.2.3 The case of unequally loaded channels

Here we examine the impact of varying the probabilities of false alarm and missed detection on the performance metric average secondary transmitted Mbits per slot, in the case of unequally loaded channels. We assume that the PU arrival rates are equal to:

$$\lambda_i = \{0.1, 0.1, 0.2, 0.2, 0.3, 0.3, 0.4, 0.4, 0.5, 0.5\}, \mathrm{i} = 1, \dots, \mathrm{N_p} = 10$$

Fig 3.14 evaluates the effect of varying the probability of false alarm on the performance metric. Although the proposed algorithms *"Algorithm with default channels and SUs pairs"* and *"Distributed algorithm"* do not take into account the PU arrival rates when they select a transmission channel, they behave more efficiently compared to *the "γ-persistent strategy"*, because the channels with high capacities are selected with higher probabilities. Recall that in the *"γ-persistent strategy"* at the beginning of the system operation the SS are selected randomly and subsequently, the SS are selected based on collision probabilities that each SU builds during the system operation. Compared with the results of the *"γ-persistent strategy"*, the *"Algorithm with default channels and SUs pairs"* and the *"Distributed algorithm"*, achieve on average a **35.42%** and a **14.98%** improvement on the average secondary transmitted Mbits per slot, respectively.

The results in Fig 3.15 show the impact of varying the probability of missed detection on the average secondary transmitted Mbits per slot, in case of unequally loaded channels. Compared with the results of the *"γ-persistent strategy"* proposed in [7], the *"Algorithm with default channels and SUs pairs"* and the *"Distributed algorithm"*, achieve on average a **42.7%** and a **19.02%** improvement on the average secondary transmitted Mbits per slot, respectively.

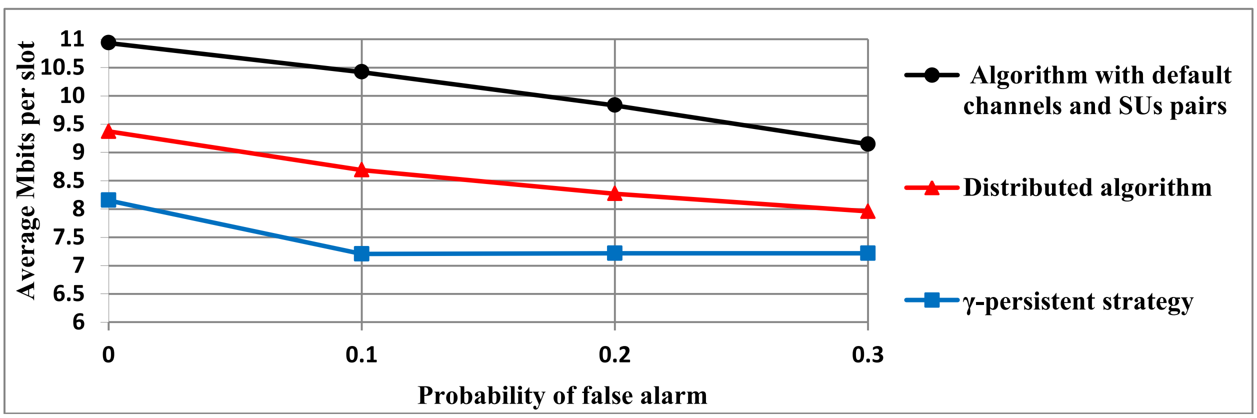


Figure 3.14: The impact of probability of false alarm on the average secondary transmitted Mbits per slot, in the case of unequally loaded channels.

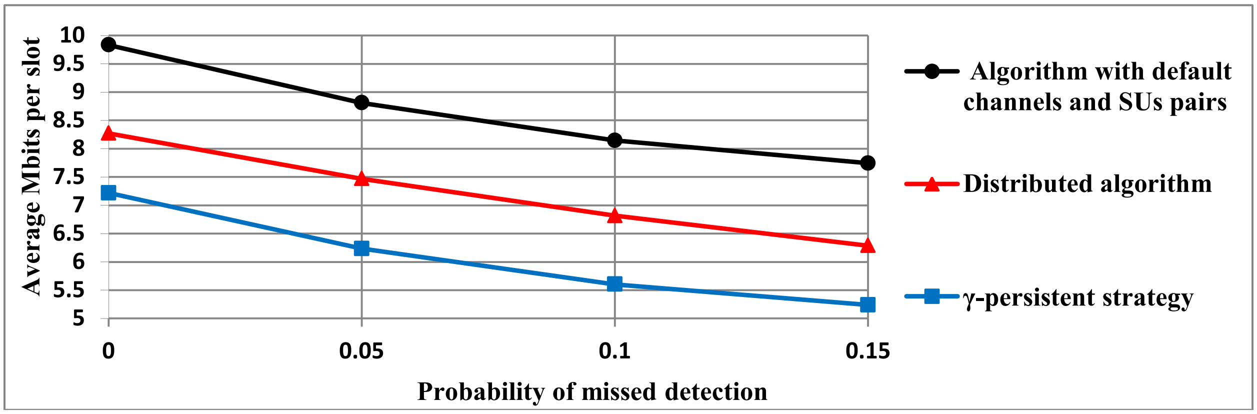


Figure 3.15: The impact of probability of missed detection on the average secondary transmitted Mbits per slot, in the case of unequally loaded channels.

3.4.3.2.4 The impact of varying the PU arrival rates

Here we examine the impact of various PU arrival rates on the performance metric. We assume that the PU arrival rates are equal for all the channels and that the probability of false alarm is set equal to 0.1 and the probability of correct detection is set equal to 0.9.

As seen by the results presented in Fig 3.16 the *"Algorithm with default channels and SUs pairs"* achieves the best results, because contention among the SUs takes place with a negligible probability. In the aforementioned algorithm each SU has a different SS compared with its pair. The SUs in the same pair will experience a collision only if the probability of missed detection is not equal to 0 and one SU loose transmission opportunities (due to a PU transmission or due to a false alarm) and collide with its pair (due to a missed detection) that already transmits. Suppose that the first SU finds the ith channel from its SS idle and transmits on that channel until the end of the specific time slot. If the second SU from the same pair found many of its channels busy it may try to transmit on the ith channel, which exists in its SS in a subsequent mini-slot, and collide with its pair due to a missed detection. Note that the probability of the above event happening is small and is considered negligible. The *"Distributed algorithm"* behaves more efficiently than the *"γ-persistent strategy"*; especially when the primary traffic load is low and the SUs have transmission opportunities in channels with high transmission rates. Compared with the results of the *"γ-persistent strategy"* in [7], the *"Algorithm with default channels and SUs pairs"* and the *"Distributed algorithm"*, achieve on average a **54.6%** and a **22.73%** improvement on the average secondary transmitted Mbits per slot, respectively.

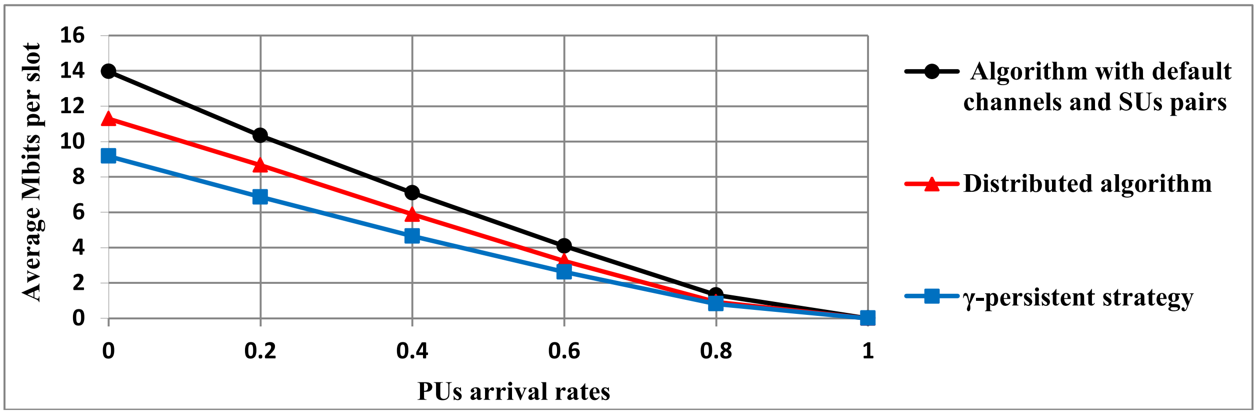


Figure 3.16: The impact of varying the PU arrival rates on the average secondary transmitted Mbits per slot, in the case the number of PUs is less than the number of SUs.

3.4.3.3 Simulation results in case the channels are equal to the SUs

In the next sections we present the results of the third scenario in which the number of PUs is set equal to the number of SUs. We assume that $N_p = N_s = 10$. In the first figure of each subsection we set the probability of missed detection equal to 0 while the probability of false alarm varies. In the second figure of each subsection we set the probability of false alarm equal to 0.2 while the probability of missed detection varies.

The results show that the *"Algorithm with default channels and SUs pairs"* outperforms the algorithms *"Distributed algorithm"* and *"γ-persistent strategy"*, in all examined scenarios. Also in that scenario the *"Distributed algorithm"* behaves the same with the *"γ-persistent strategy"* in case the probability of false alarm varies and the probability of missed detection is set equal to 0. Note that we have equal number of SUs and available channels and at each mini-slot all the channels are expected to be selected by the SUs. As seen by the results, the average secondary transmitted Mbits per slot for the *"γ-persistent strategy"* is highly affected by the value of the missed detection probability, because the SU may mistakenly drop its SS, in case of collision due to a missed detection.

In the following three subsections we present the results for the system performance metric by examining the impact of "light" primary traffic load, of "heavy" primary traffic load, and of asymmetric primary traffic load, as a function of the probability of false alarm and the probability of missed detection. Subsequently, we examine the impact of various PU arrival rates.

3.4.3.3.1 The case of lightly loaded channels

In this section we examine the impact of varying the probabilities of false alarm and missed detection on the performance metric, in the case of lightly loaded channels. We assume that the PU arrival rates are equal for all the channels and are set equal to 0.3 packets per slot.

Fig 3.17 evaluates the effect of varying the probability of false alarm on the performance metric. As seen by the results shown in Fig 3.17 the *"Distributed algorithm"* achieves results close to those of the *"γ-persistent strategy"* when the probability of missed detection is equal to 0. The Latin Square Matrix size in that scenario is $10X4$, because the available channels were set equal to 10 and the number of maximum mini-slots in each slot was set equal to 4. In the aforementioned algorithms, each SU will try to choose a unique row from the Latin Square, and after a period of time all the SUs will choose different SS and persist with them SS until the end of the system operation. Note that collisions between the SUs and the PUs do not occur (since P_d). In case all the SUs choose different SS, then all the available channels are exploited by the SUs ($N_s = N_p$). That explains the comparable average secondary transmitted Mbits per slot achieved by the aforementioned algorithms. The *"Algorithm with default channels and SUs pairs"* outperforms the other two algorithms because from the beginning of the system operation the SUs have different SS and they do not compete for an idle channel during the entire system operation.

The results in Fig 3.18 show the impact of varying the probability of missed detection on the average secondary transmitted Mbits per slot. The *"Distributed algorithm"* outperforms the *"γ-persistent strategy"* when the probability of missed detection varies. In the former algorithm each SU persist with its SS with higher probability compared to the *"γ-persistent strategy"*, because the SU will drop its SS only after it experiences two successive collisions. The probability of having two successive collisions due to missed detection is equal to $(1 - P_d)^2$ which is considered negligible. In the *"γ-persistent strategy"* the SU drops its SS after experience a collision and that leads the strategy to wrong decisions because the SU may mistakenly drop its SS due to a missed detection. Compared with the results of the *"γ-persistent strategy"* proposed in [7], the *"Algorithm with default channels and SUs pairs"* and the *"Distributed algorithm"*, achieve on average a **27.29%** and a **16.42%** improvement on the average secondary transmitted Mbits per slot, respectively.

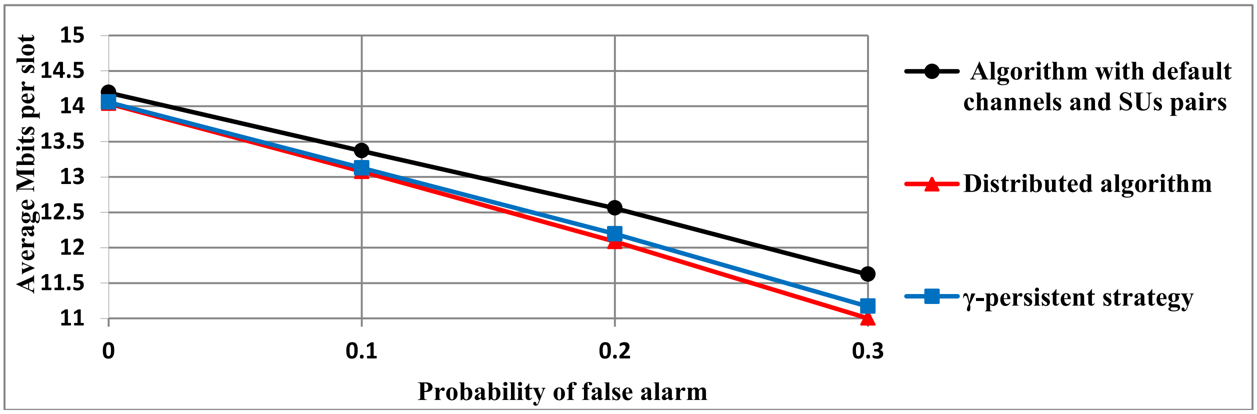


Figure 3.17: The impact of probability of false alarm on the average secondary transmitted Mbits per slot, in the case of lightly loaded channels.

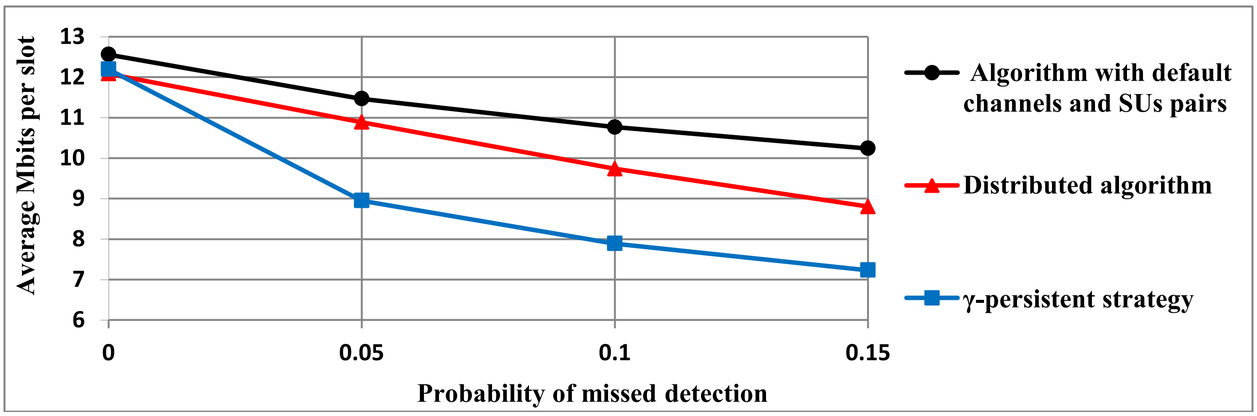


Figure 3.18: The impact of probability of missed detection on the average secondary transmitted Mbits per slot, in the case of lightly loaded channels.

3.4.3.3.2 The case of heavily loaded channels

Here we examine the impact of varying the probabilities of false alarm and missed detection on the performance metric, in the case of heavily loaded channels. We assume that the PU arrival rates are equal for all the channels and are all set equal to 0.7 packets per slot.

The results in Fig 3.19 show the impact of varying the probability of false alarm on the average secondary transmitted Mbits per slot. As seen by the results presented in Fig 3.19, the algorithms *"Distributed algorithm"* and *"γ-persistent strategy"* achieve comparable results because about 70% of the slots are occupied by the PUs transmissions and the secondary transmission opportunities are expected to be equal to 22.8% (0.76*0.3, since the SUs waste 24% of the available to them transmission duration to sense the channels). Therefore, the aforementioned algorithms have similar behavior in the specific scenario, even when the probabilities of false alarm or missed detection vary. The *"Algorithm with default channels and SUs pairs"* outperforms the other two algorithms especially as the probability of missed detection varies (see Fig 3.20) because that algorithm completely eliminates the collisions between the SUs in the first mini-slot of each slot. If a SU transmits on a channel from its SS, then it keeps that channel until the end of the time slot. If the other SU from the same pair skips a mini-slot because of false alarm then it is possible to collide with its pair if it mistakenly senses the specific channel as busy. Note that the probability of such an event happening is very low. Compared with the results of the *"γ-persistent strategy"* proposed in [7], when the probability of missed detection varies the *"Algorithm with default channels and SUs pairs"* achieves on average a **31.99%** improvement on the average secondary transmitted Mbits per slot.

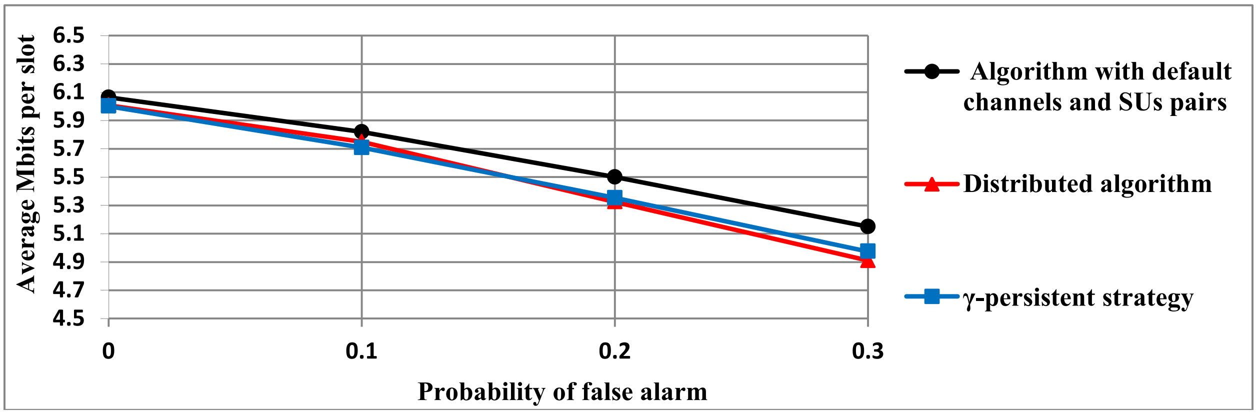


Figure 3.19: The impact of probability of false alarm on the average secondary transmitted Mbits per slot, in the case of heavily loaded channels.

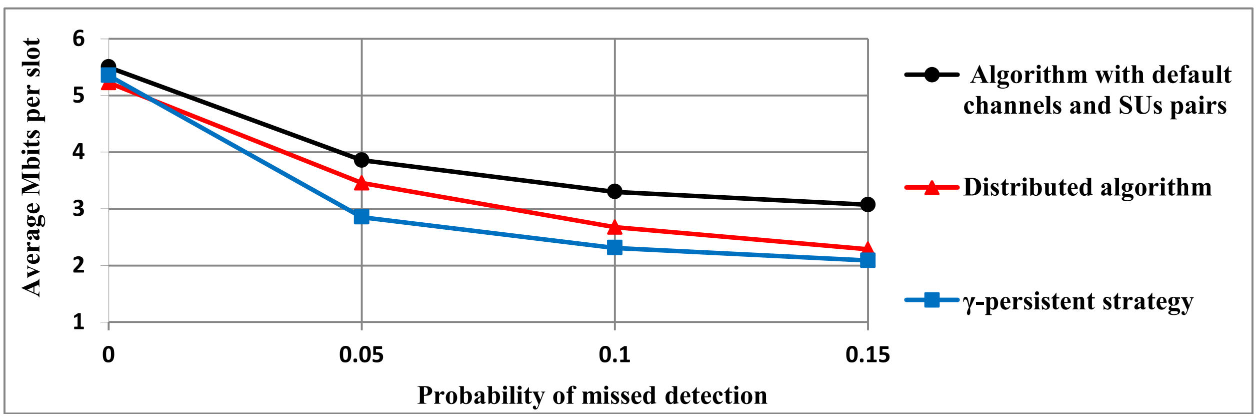


Figure 3.20: The impact of probability of missed detection on the average secondary transmitted Mbits per slot, in the case of heavily loaded channels.

3.4.3.3.3 The case of unequally loaded channels

In this section we examine the impact of varying the probabilities of false alarm and missed detection on the performance metric, in the case of unequally loaded channels. We assume that the PU arrival rates are equal to:

$$\lambda_i = \{0.1, 0.1, 0.2, 0.2, 0.3, 0.3, 0.4, 0.4, 0.5, 0.5\}, \mathrm{i} = 1, \ldots, \mathrm{N_p} = 10$$

As shown by the results in Fig 3.21 when the probability of false alarm varies the algorithms *"Distributed algorithm"* and *"γ-persistent strategy"* achieve comparable results. In the examined scenario we assume that $N_p = N_s$. When the contention among the SUs is completed each SU will have one different row from the Latin Square, therefore in each mini slot all the available channels are exploited by the SUs. The probability of missed detection is set equal to 0, so when the SUs select different SS they will persist with them SS until the end of the system operation, because collisions due to a missed detection cannot occur.

Fig 3.22 shows the improvement introduced by the proposed algorithms: *"Algorithm with default channels and SUs pairs"* and *"Distributed algorithm"*, in case the probability of missed detection varies. The *"Distributed algorithm"* outperforms the *"γ-persistent strategy"* because in the former algorithm each SU must experience two successive collisions before dropping its SS. This persistence prevents the SU from dropping its SS due to a missed detection. Compared with the results of the *"γ-persistent strategy"* proposed in [7], the algorithms *"Algorithm with default channels and SUs pairs"* and *"Distributed algorithm"* achieve on average a **27.01%** and a **16.19%** improvement on the average secondary transmitted Mbits per slot, respectively.

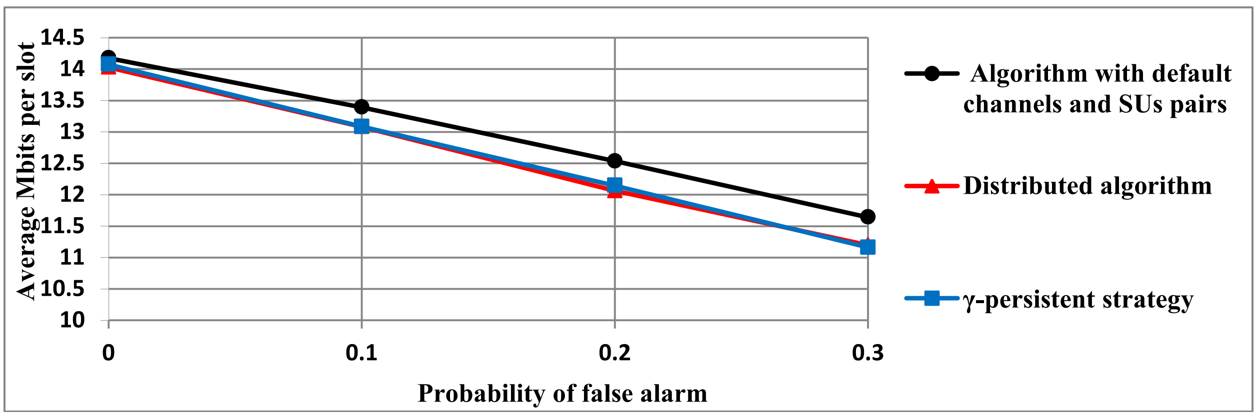


Figure 3.21: The impact of probability of false alarm on the average secondary transmitted Mbits per slot, in the case of unequally loaded channels.

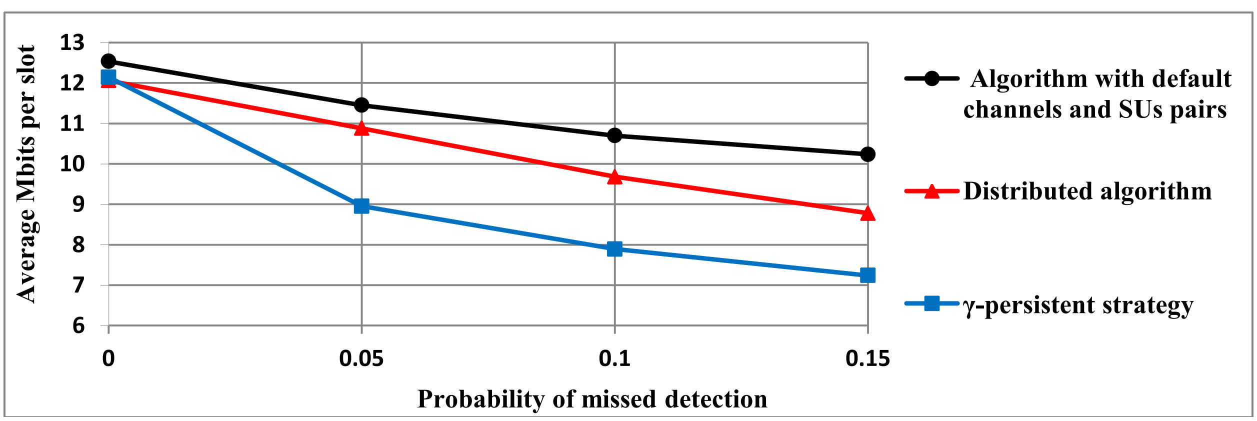


Figure 3.22: The impact of probability of missed detection on the average secondary transmitted Mbits per slot, in the case of unequally loaded channels.

3.4.3.3.4 The impact of varying the PU arrival rates

Here we examine the impact of various PU arrival rates on the performance metric. We assume that the PU arrival rates are equal for all the channels and that the probability of false alarm is set equal to 0.1 and the probability of correct detection is set equal to 0.9.

As seen by the results in Fig 3.23 the *"Algorithm with default channels and SUs pairs"* achieves the highest average secondary transmitted Mbits per slot, because the SUs do not compete for an idle channel in the first mini-slot of each slot (see section 3.4.3.2.4). Note that if the probability of missed detection is equal to 0, that algorithm guarantees that the SUs will never compete for an idle channel during the entire system operation. When the primary traffic load is low the *"Distributed algorithm"* outperforms the *"γ-persistent strategy",* because the SUs exploit the channels with higher capacities, when the transmission opportunities are found. As the PU arrival rates increase the secondary transmission opportunities become rare and all the algorithms achieve comparable results. Compared with the results of the *"γ-persistent strategy"* in [7], the *"Algorithm with default channels and SUs pairs"* and the *"Distributed algorithm"*, achieve on average a **33.07%** and a **17.71%** improvement on the average secondary transmitted Mbits per slot, respectively.

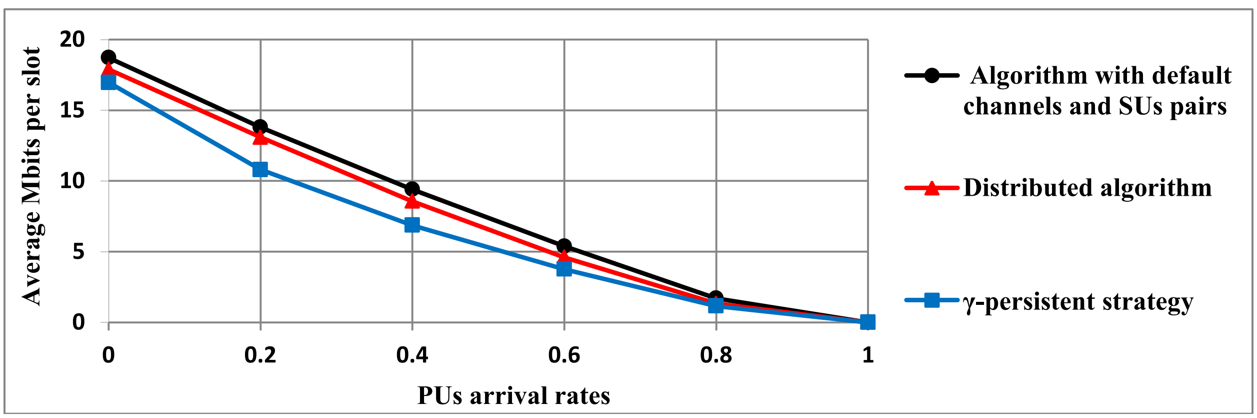


Figure 3.23: The impact of varying the PU arrival rates on the average secondary transmitted Mbits per slot, in the case the number of PUs is equal to the number of SUs.

3.4.3.4 The impact of varying the number of SUs in the case of lightly loaded channels

Here we examine the impact of varying the number of SUs when the number of PUs is set equal to 10, on the performance metric. We assume that the PU arrival rates are equal for all the channels and are set equal to 0.3 packets per slot. The probability of false alarm was set equal to 0.2 and the probability of correct detection was set equal to 0.9.

The illustrated results in Fig 3.24 show the superiority of the *"Algorithm with default channels and SUs pairs"*. As the number of SUs increases the average secondary transmitted Mbits per slot decrease, as expected. Note that as the number of SUs increases the improvement introduced by the *"Distributed algorithm"* increases as well. In the *"Distributed algorithm"* the SUs adaptively change their sensing probability depending on the successful transmission or collisions that experience during the system operation and as a result the competition among the SUs decreases. Compared with the results of the *"γ-persistent strategy"* in [7], the *"Algorithm with default channels and SUs pairs"* and the *"Distributed algorithm"*, achieve on average a **18.6%** and a **10.75%** improvement on the average secondary transmitted Mbits per slot, respectively.

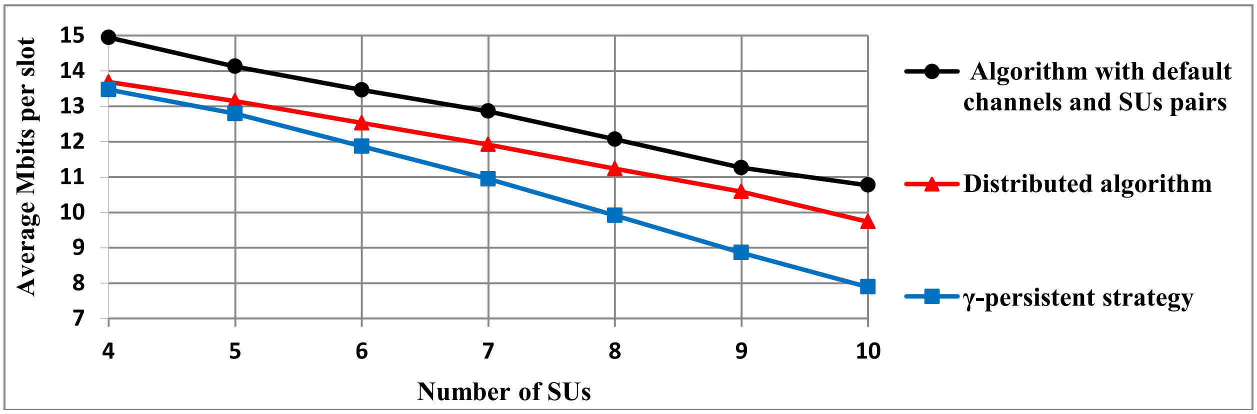


Figure 3.24: The impact of varying the number of SUs on the average secondary transmitted Mbits per slot, in the case of lightly loaded channels.

3.4.3.5 The impact of varying the number of SUs in the case of heavily loaded channels

In this section we examine the impact of varying the number of SUs when the number of PUs is set equal to 10, on the performance metric. We assume that the PU arrival rates are equal for all the channels and are set equal to 0.7 packets per slot. The probability of false alarm was set equal to 0.2 and the probability of correct detection was set equal to 0.9.

Compared to the corresponding results in Fig 3.24, the average secondary transmitted Mbits per slot are reduced because of the heavily loaded channels. Compared with the results of the *"γ-persistent strategy"* in [7], the *"Algorithm with default channels and SUs pairs"* and the *"Distributed algorithm"*, achieve on average a **21.9%** and a **9.15%** improvement on the average secondary transmitted Mbits per slot, respectively.

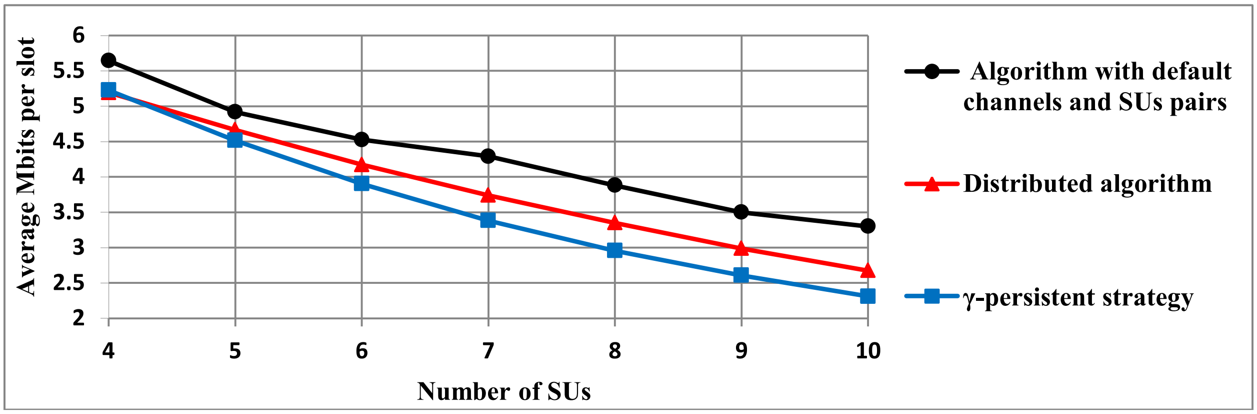


Figure 3.25: The impact of varying the number of SUs on the average secondary transmitted Mbits per slot, in the case of heavily loaded channels.

3.4.3.6 The impact of varying the number of SUs in the case of unequally loaded channels

Here we examine the impact of varying the number of SUs on the performance metric when the number of PUs is set equal to 10. The probability of false alarm was set equal to 0.2 and the probability of correct detection was set equal to 0.9. We assume that the PU arrival rates are equal to:

$$\lambda_i = \{0.1, 0.1, 0.2, 0.2, 0.3, 0.3, 0.4, 0.4, 0.5, 0.5\}, \mathrm{i} = 1, \dots, \mathrm{N_p} = 10$$

As seen by the illustrated results in Fig 3.26, the *"Algorithm with default channels and SUs pairs"* achieves the best results, followed by the *"Distributed algorithm"*. Compared with the results of the *"γ-persistent strategy"* in [7], the *"Algorithm with default channels and SUs pairs"* and the *"Distributed algorithm"*, achieve on average a **16.7%** and a **11.25%** improvement on the average secondary transmitted Mbits per slot, respectively.

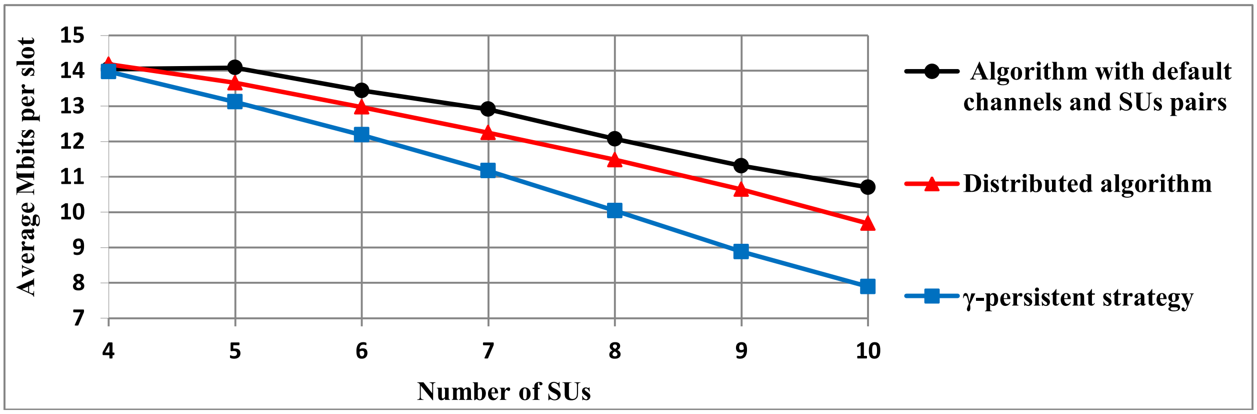


Figure 3.26: The impact of varying the number of SUs on the average secondary transmitted Mbits per slot, in the case of unequally loaded channels.

3.5 Main conclusions

This section presents the main conclusions drawn from our simulation study. As seen by the results in the previous sections, the *"Algorithm with default channels and SUs pairs"* achieves the highest average secondary transmitted Mbits per slot, in all examined scenarios. That algorithm requires cooperation among the *Coordinator* and the SUs. From the algorithmic description it is clear that no contention arises between the SUs in case the probability of missed detection is set equal to 0, because the *Coordinator* assigns different SS to the SUs pairs and in case they have only one transmission channel the SUs will successively sense or not the channels to avoid collisions. The only possibility to have collision among two SUs in the same pair is the event where one SU successfully transmits in a mini-slot and keeps that channel for the remaining time slot; the other SU from the same pair loses a transmission opportunity (due to a false alarm or PU transmission) and in a following mini-slot of the same slot senses the channel on which its pair SU already transmits as idle (due to a missed detection) and collides with the other SU. Note that the above event occurs with a small probability.

We have seen from the results of the first scenario, where the available channels were more than the number of the SUs, that the algorithms *"Algorithm with default channels and SUs pairs"* and *"Distributed algorithm"* outperform the *"γ-persistent strategy"* because in the former algorithms the channels with higher capacities are frequently used. The superiority of the proposed algorithms is more evident in the case the probability of missed detection varies, as we already have demonstrated and explained the weakness of the *"γ-persistent strategy"* in that case.

We have seen from the results of the second scenario, where the channels were less than the number of the SUs, that the *"Distributed algorithm"* outmatches the *"γ-persistent strategy"* because the former algorithm reduces the sensing probability in case of collisions, increases the sensing probability in case of successful transmission and finally, sets the sensing probability equal to 1 in case the SU has its own SS. In the specific scenario the number of channels was limited and the SUs competed for an idle channel more frequently, the above mechanism of the adaptively changing the sensing probability reduced the contention among the SUs, when it was necessary.

Finally, as seen by the results in the third scenario, where the number of available channels was equal to the number of the SUs, the *"γ-persistent strategy"* was highly affected by the value of the probability of missed detection. The *"Distributed algorithm"* persists in keeping the SS of a SU for a longer period of time compared to the *"γ-persistent strategy"*. Recall, that in the *"Distributed algorithm"* the SU will drop its SS only after its experience two successive collisions. In contrasts, in the *"γ-persistent strategy"* when a collision occurs the SU immediately reduces the selection probability for the specific SS in the next slot. As a result a SU may mistakenly drop its SS, due to a missed detection, and then experience competition with the other SUs that are still looking for a unique SS.

Chapter 4: The "*Distributed Algorithm*" in Homogeneous Multichannel CRNs

4.1 Introduction

In the previous Chapter the *"Distributed algorithm"* was proposed for heterogeneous multichannel CRNs. Recall, that in the specific algorithm the SUs select their SS without coordination by a centralized entity. The challenge is to find an algorithm which does not require coordination by a centralized entity and at the same time approaches the effectiveness of the algorithms where a centralized entity exists and assigns the network's channels to the SUs. In this Chapter the idea of the Latin Square Matrix is examined in the case of a homogeneous multichannel CRNs. More specifically, the *"Distributed algorithm"* proposed and introduced in Chapter 3 is modified, and its efficiency is examined in the case of homogeneous multichannel CRNs. As shown by our results, the "*Distributed algorithm*" achieves results close to those of the algorithms that require coordination by a centralized entity.

The same network topology as in Chapters 2 and 3 is used. The SUs have a fixed number of opportunities to find an idle channel during the duration of a time slot. The SU' SS is selected from a predefined Latin Square Matrix. In the first mini-slot of a time slot the SU senses the first channel from its SS with probability p. If the examined channel is sensed busy, then in the next mini-slot the SU will sense the second channel from its SS with probability p. This procedure is continued until the maximum number of admissible handovers δ, is reached or a transmission opportunity is found. When a SU finds an idle channel then it transmits on that channel during the remaining duration of the specific slot. The likelihoods of false alarm and missed detection are taken into account.

4.2 The proposed algorithm

The "*Distributed algorithm*" proposed and introduced in Chapter 3 is modified as follows. The channel selection probabilities based on the channel capacities (see equation (3.2)), is replaced by the channel selection probabilities based on the PU arrival rates. Each SU chooses a channel based on the estimated primary user arrival rates, i.e., according to the following probabilities:

$$P_j = \frac{(1-\lambda_{est,j})}{\sum_{i=1}^{N_p}(1-\lambda_{est,i})} \quad ,j = 1, \dots, N_p \tag{4.1}$$

As in Chapter 2 we assume that each PU k collects the feedback from its dedicated channel for the last x slots and it then estimates its packet arrival rate by:

$$\lambda_{\text{est},k} = \frac{Number\ of\ ACKs}{x}, k = 1, \dots, N_p \tag{4.2}$$

In the initial $x-1$ slots, the estimated PU arrival rates are unknown, so the SUs will choose the channels with equal probability during that time period. We further assume that the PUs broadcast the channel selection probabilities $\mathrm{P_j}$, $j = 1, \dots, N_{p,}$ to the SUs on a separate broadcast channel. Note that in a real scenario if the arrival rates of the PUs do not vary abruptly with time, each PU does not have to monitor its channel feedback for the entire duration of the system operation. In such case, the PUs have to monitor their channel feedback every y slots, and for a duration of x slots, where $y \;>>\; x$.

4.3 Performance Evaluation

4.3.1 Performance metric

The performance metric normalized average secondary throughput is examined. That performance metric corresponds to the average secondary packet transmission rate per time slot.

4.3.2 Simulation model

The same simulation model as in Chapter 2 is used. For the details we refer to section 2.4.2.

4.3.3 Simulation results

An event-driven simulator is implemented to evaluate the effectiveness of the proposed *“Distributed Algorithm”*, with the proposed algorithms in Chapter 2 that require coordination by a centralized entity namely *“Algorithm with awareness”*, *“Algorithm with limited awareness”* and *“Algorithm with default channels”*. Next, the *“Distributed Algorithm”* is compared with the algorithms *“PPRA”* and *“PPRA with improvement”*, which do not require the SUs to possess centralized information. In the following simulation experiments we set the sensing probability equal to 1 for all the examined algorithms. Note that in the *“Distributed algorithm”* the initial value of the sensing probability is set equal to 1. However, as time goes by, each SU will adapt its sensing probability, as dictated by the rules of the algorithm. Recall that, when a the SU collides with another SU or with the PU then the sensing probability is reduced by a small amount equal to 0.1, if the SU successfully transmits during a time slot then the sensing probability is increased by a small amount equal to 0.1 and if the SU has its own SS ($i.e.$ the variable $MySS = 1$) then the sensing probability is set equal to 1.

4.3.3.1 Simulation results in case the channels are more than the SUs

In the next sections we present the result of the first scenario in which the number of PUs is set equal to 10 and the number of SUs is set equal to 5. The results show that the *"Distributed algorithm"*, achieves results close to those of the *"Algorithm with limited awareness"* and to the *"Algorithm with awareness"* in the cases of lightly and unequally loaded channels, respectively.

4.3.3.1.1The case of lightly loaded channels

In this section we examine the case of lightly loaded channels on the performance metric normalized average secondary throughput. We assume that the PU arrival rates are equal for all the channels and are all set equal to 0.2 packets per slot.

The results in Fig. 4.1 show the superiority of the *"Distributed algorithm"*, compared to the *"PPRA"* and *"PPRA with improvement"* algorithms. In the *"Distributed algorithm"* each SU tries to find a unique SS from a predefined Latin Square Matrix. After a period of time, each SU will have different SS compared with the SS of the other SUs and will persist with that SS if no successive collisions are observed. In the *"PPRA"* algorithm the SUs at each mini-slot of a slot try to randomly find an idle channel, while in the *"PPRA with improvement"* the SUs at each mini-slot of a slot try to find an idle channel based on the estimated PU arrival rates on the channels. As a result the SUs contend for an idle channel more frequently in the latter algorithms.

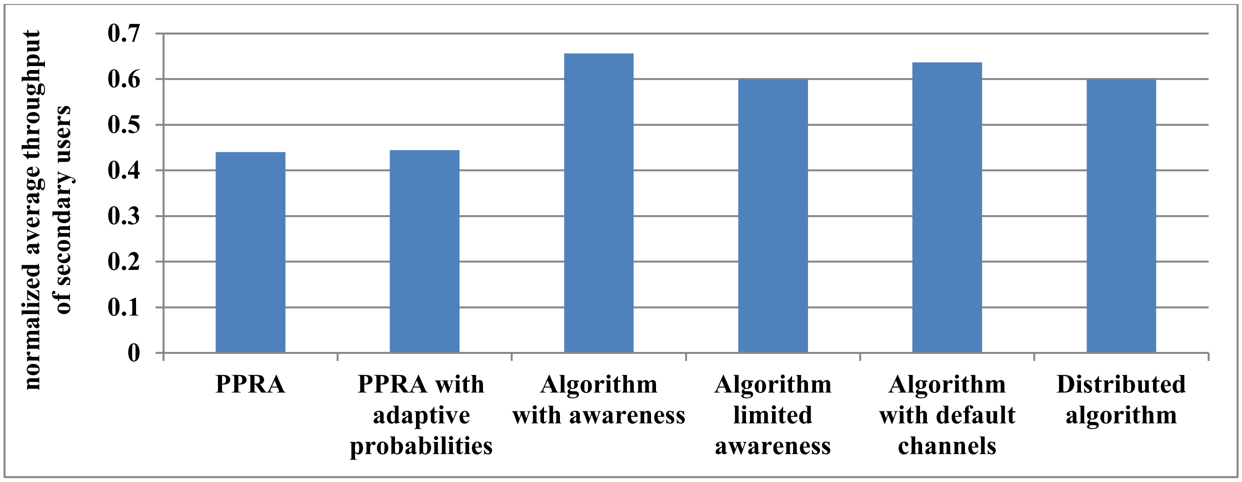


Figure 4.1: Normalized average secondary throughput in case of lightly loaded channels.

4.3.3.1.2 The case of heavily loaded channels

Here we examine the case of heavily loaded channels on the performance metric normalized average secondary throughput. We assume that the PU arrival rates are equal for all the channels and are all set equal to 0.8 packets per slot.

The illustrated results in Fig 4.2 show that the achieved normalized average throughput is very low. Due to the heavy primary traffic the *"Distributed algorithm"* achieves the lower average secondary throughput, this is because the transmission opportunities are scarce and the SUs do not converge into collision-free sensing orders.

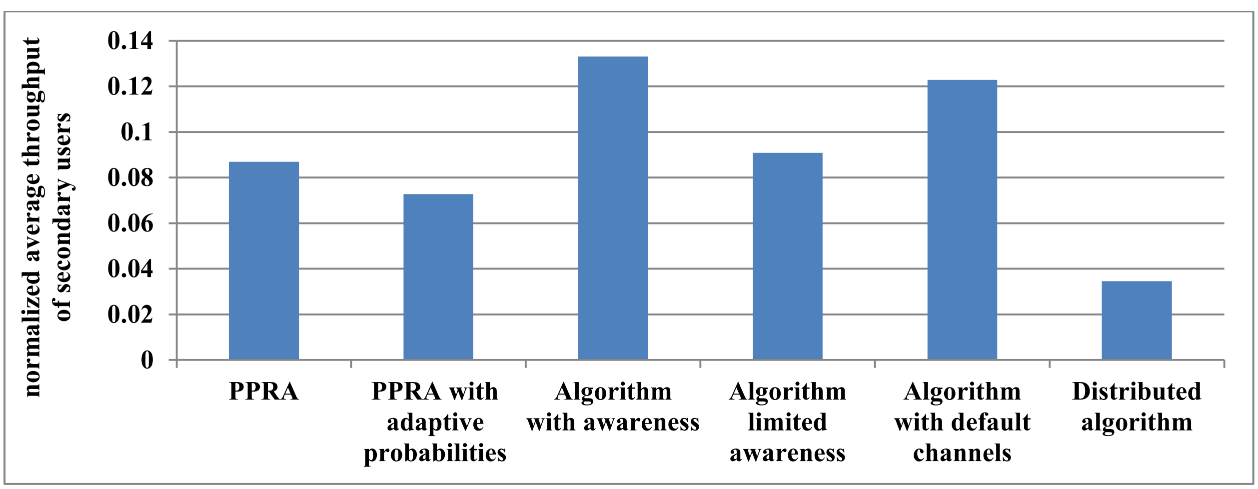


Figure 4.2: Normalized average secondary throughput in case of heavily loaded channels.

4.3.3.1.3 The case of unequally loaded channels

In this section we examine the case of unequally loaded channels on the performance metric normalized average secondary throughput, when the sensing probability is set equal to 1. We assume that the PU arrival rates are equal to:

$$\lambda_i = \{0.1, 0.1, 0.3, 0.3, 0.5, 0.5, 0.7, 0.7, 0.9, 0.9\}, \mathrm{i} = 1, \dots, \mathrm{N_p} = 10$$

The results in Fig. 4.3 show that the *"Distributed algorithm"*, outperforms the *"PPRA"* and the *"PPRA with improvement"* algorithms and achieves normalized average secondary throughput very close to that of the *"Algorithm with limited awareness"*. This shows the superiority of the *"Distributed algorithm"* in the homogeneous channels case; good results can be achieved by an algorithm in which the SUs independently select their transmission channels based on their own statistics, without the need of coordination by a centralized entity. Recall that, in each row and column of the Latin Square Matrix every channel appears only once. In case all the SUs choose different rows from the Latin Square Matrix, then they will not compete during the network operation.

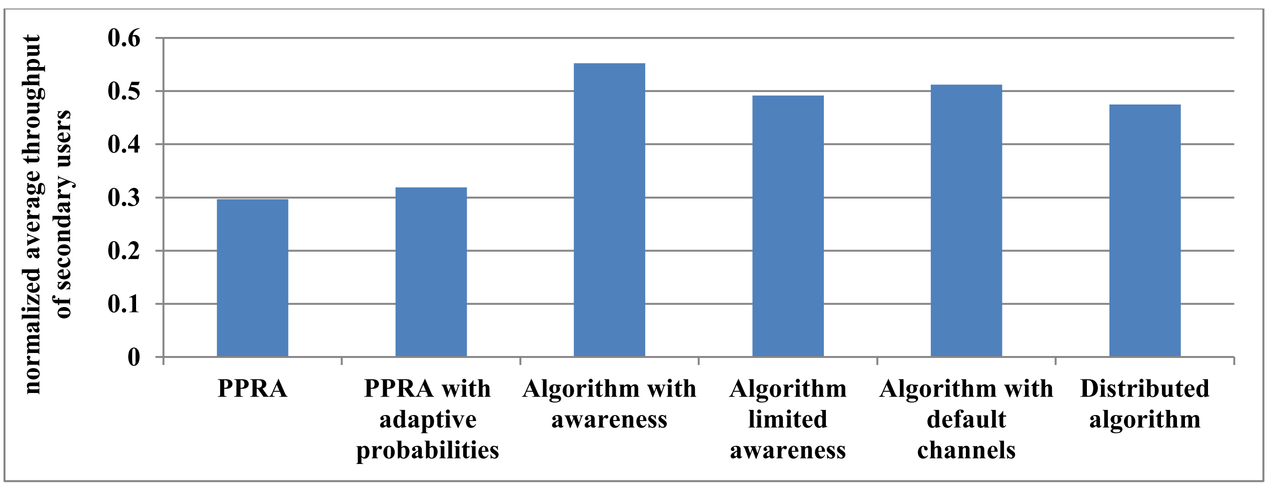


Figure 4.3: Normalized average secondary throughput in case of unequally loaded channels.

4.3.3.1.4 The impact of varying the PU arrival rates

Here we examine the impact of various PU arrival rates on the performance metric normalized average secondary throughput, when the sensing probability is set equal to 1. We assume that the primary user arrival rates are equal for all the channels.

As seen by the illustrated results in Fig 4.4 the *"Distributed algorithm"* achieves a normalized average throughput of the secondary users very close to that of the *"Algorithm with* *limited awareness"*. When the PU arrival rates are high, e.g. in the interval [0.8-1], the *"Distributed algorithm"* achieves the lower average secondary user throughput. This is because due to the heavy primary user load, the SUs have scarce transmission opportunities and are unable to find unique SS's from the Latin Square Matrix.

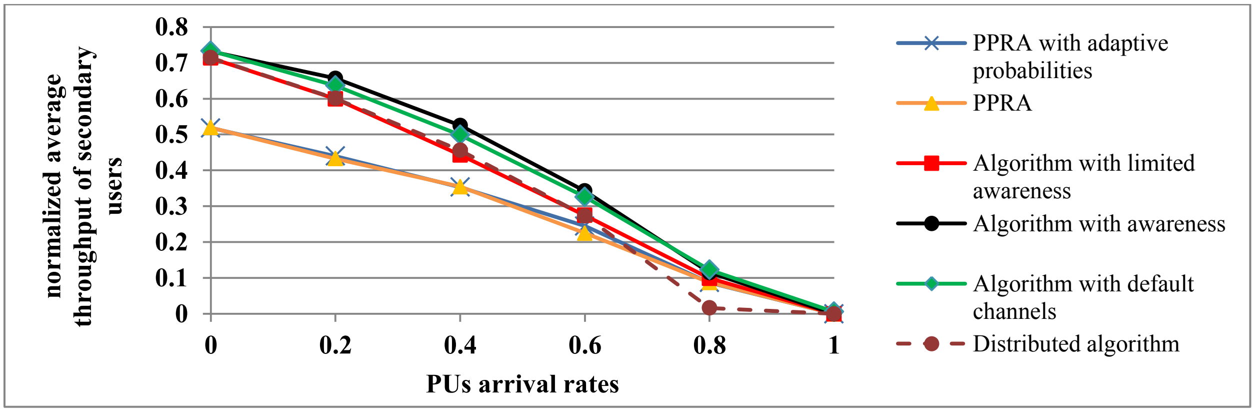


Figure 4.4: The impact of varying the PU arrival rates on the normalized average throughput of secondary users.

4.3.3.2 Simulation results when the number of channels is less than the SUs

In the next sections we present the result of the second scenario in which the number of PUs is set equal to 5 and the number of SUs is set equal to 8. The results show that the *"Distributed algorithm"*, performs equally well with the *"Algorithm with awareness"* and the *"Algorithm with limited awareness"* in the cases of lightly and unequally loaded channels, respectively.

4.3.3.2.1The case of lightly loaded channels

In this section we examine the case of lightly loaded channels on the performance metric normalized average secondary throughput. We assume that the primary user arrival rates are equal for all the channels and are all set equal to 0.2 packets per slot.

Fig 4.5 shows the average secondary throughput in case of lightly loaded channels. The *"Distributed algorithm"*, performs equally well with the *"Algorithm with awareness"*. As already mentioned in the algorithm's description, the *"Algorithm with awareness"* requires large amounts of energy and is not expected to be applicable in many practical systems. However, the *"Distributed algorithm"* is an adaptive algorithm which does not require coordination by a centralized entity, therefore it can be used in practical systems, and achieves high average secondary throughput.

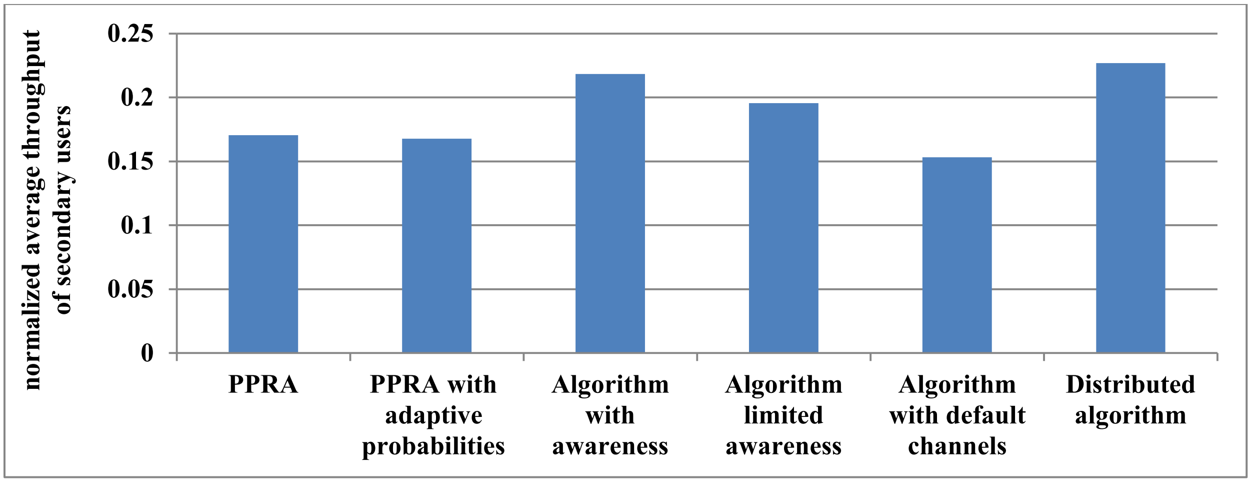


Figure 4.5: Normalized average secondary throughput in case of lightly loaded channels.

4.3.3.2.2 The case of heavily loaded channels

In this section we examine the case of lightly loaded channels on the performance metric normalized average secondary throughput. We assume that the PU arrival rates are equal for all the channels and are all set equal to 0.8 packets per slot.

The illustrated results in Fig 4.6 demonstrate the low achieved average secondary throughput achieved by all the examined algorithms. In this scenario the primary user load is heavy and the number of channels is small, therefore the SUs have very scarce transmission opportunities, which explain the low values of achieved throughputs.

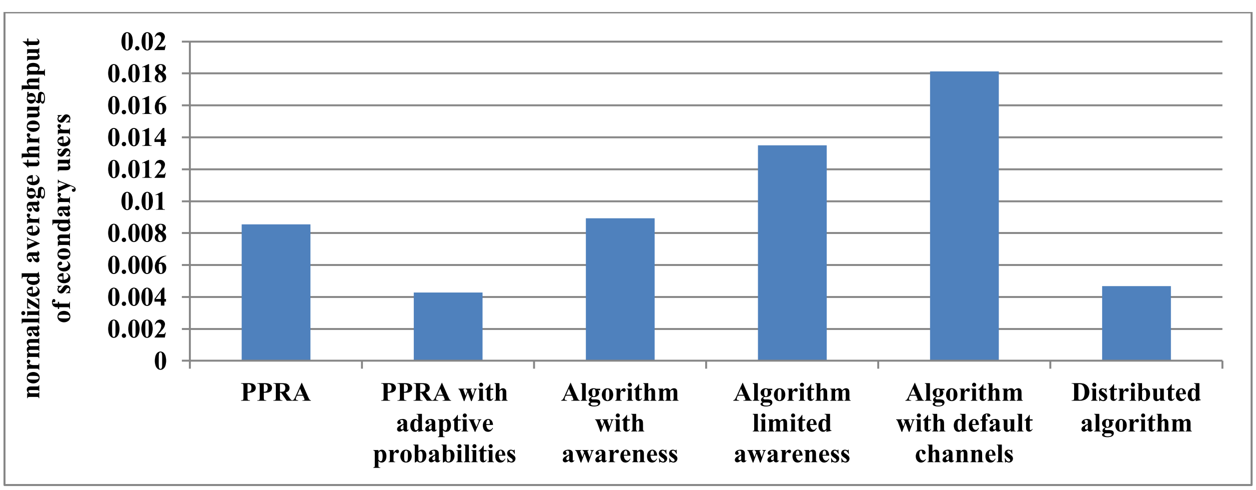


Figure 4.6: Normalized average secondary throughput in case of heavily loaded channels.

4.3.3.2.3 The case of unequally loaded channels

Here we examine the impact of the values of the sensing probability on the performance metric normalized average throughput in the case of unequally loaded channels. We assumed that the PU arrival rates are equal to:

$$\lambda_i = \{0.1, 0.2, 0.3, 0.4, 0.5\}, \mathrm{i} = 1, \ldots, \mathrm{N_p} = 5$$

As seen by the results in Fig 4.7 the *"Distributed algorithm"* achieves a normalized average throughput of the secondary users very close to that of the *"Algorithm with awareness"*. In the second scenario the improvement introduced by the *"Distributed algorithm"* is reduced because the number of available channels is small. In this scenario, $N_p < N_s$ and all the algorithms experience increased contention among the SUs.

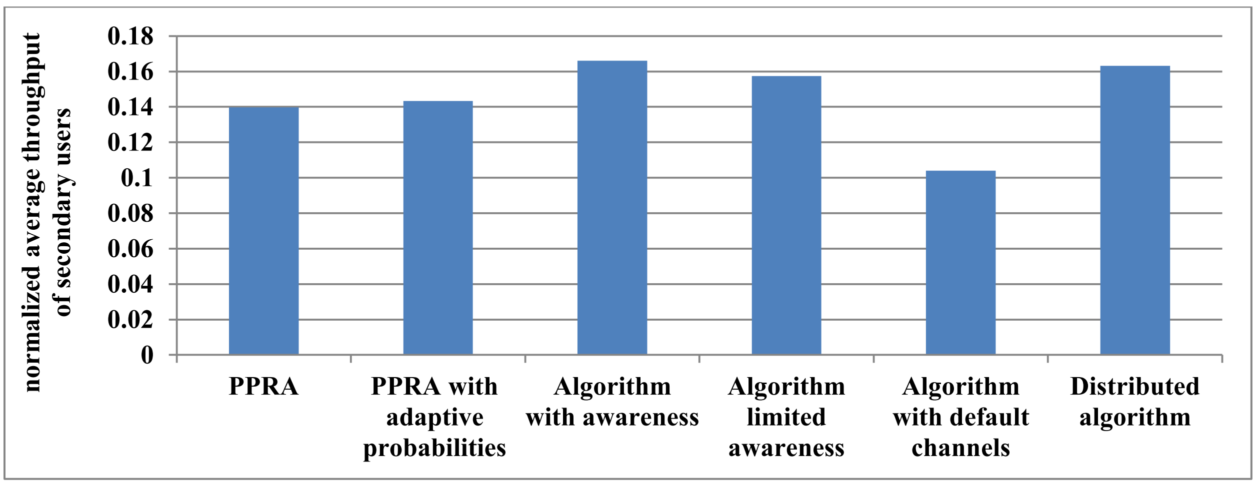


Figure 4.7: Normalized average secondary throughput in case of unequally loaded channels.

4.3.3.2.4 The impact of varying the PU arrival rates

Here we examine the impact of varying the PU arrival rates. We assume that: **(a)** the sensing probability is set equal to 1 and **(b)** the PU arrival rates are equal for all the channels. When the PU arrival rates are light, e.g. in the interval [0-0.3] the *"Distributed algorithm"*, achieves a normalized average throughput of the secondary users very close to that achieved by the algorithms which require coordination by a centralized entity. However, as the PU arrival rates increase the effectiveness of the *"Distributed algorithm"* decreases because of the limited transmission opportunities for the SUs.

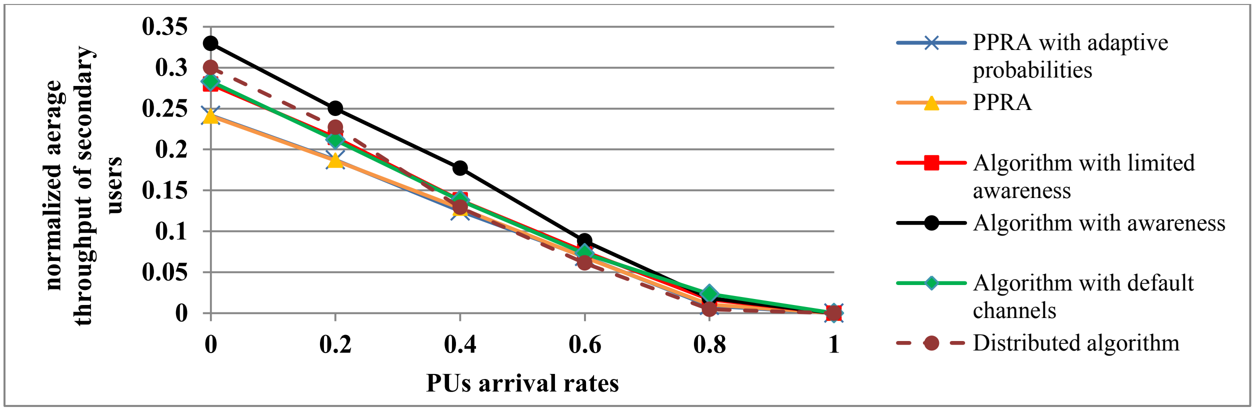


Figure 4.8: The impact of varying the PU arrival rates on the normalized average throughput of secondary users.

4.4 Main conclusions

From the simulations results presented in this Chapter we conclude that the *“Distributed algorithm”* is efficient in homogeneous networks and achieves a normalized average throughput of the secondary users very close to that of the algorithms which use coordination by a centralized entity.

Recall the two main mechanisms of the *“Distributed algorithm”*: **(a)** the persistence of a SU in a SS as long as collisions are not observed in two consecutive transmission attempts and **(b)** the adaptive change of the sensing probability of the SUs. Due to the above two mechanisms, the *“Distributed algorithm”* achieves results close to those of the *“Algorithm with awareness”*, *“Algorithm with limited awareness”* and *“Algorithm with default channels”*. Finally recall that in the *“Distributed algorithm”* each SU has to find a unique SS using the Latin Square and in the latter algorithms the *Coordinator* assigns a unique SS to each SU at the beginning of the system operation.

Chapter 5: Cooperative Relay Selection

5.1 Introduction

The direct transmissions from a primary transmitter to the intended primary receiver may be severely damaged because of the unstable environment in wireless communications due to reasons such as multipath fading and shadowing [32]. In this Chapter, new cognitive cooperative communication protocols are proposed for cognitive radio networks, in which one primary user and N_s SUs cooperate to achieve mutual benefits in their communications. The new cooperation protocols allow active cooperation between PU and SUs so that SUs with better channel conditions assist to relay primary user' signals in exchange for some spectrum released from the primary user. More specifically, we propose a new cooperation protocol with best-relay selection to improve the performance of secondary transmissions while ensuring the quality of service (QoS) of the PU. The proposed cognitive cooperation protocol aims to reduce as much as possible the average primary user packet transmission time, i.e. the average required transmission time for a primary transmitter to send its data packet to the primary receiver potentially using one SU as a relay node. Subsequently, we propose another protocol which employs a stopping criterion that takes into account the time to scan the candidate relays before stopping at a suitable one with a good channel quality. With cooperation, a PU can reduce its primary packet transmission time through the relay service provided by a SU while the SU obtains channel access opportunities as a reward, which can simultaneously improve the performance of both the primary and the secondary network, achieving a “win-win” situation for both. If a secondary relay accepts to cooperate with the PU, then during the duration of the time slot the secondary relay will transmit its own data as a reward and also will relay the primary data to the primary receiver. Note that this is in contrast to our previous work in Chapters 2-4, where the SUs transmit their own signals only if the PU’s channel is found idle, i.e. during the time slot duration if the PU has a packet for transmission then it transmits its data during the entire duration of the specific slot, otherwise if the PU has no packet for transmission and a SU detects the transmission opportunity can transmit its own data during the same idle slot.

The most critical challenge for such a problem of cooperative relay selection is how to select a relay efficiently. Due to the potentially large number of secondary users, it is infeasible for a PU transmitter to first scan all the SUs and then pick the best one. The PU transmitter observes the SUs sequentially. After observing a SU, the PU needs to make a decision on whether to terminate its scan and use the current SU as its relay or to skip it and observe the next SU [32]. We address this problem by using a stopping rule criterion. From our simulation results it is evident that is not efficient to scan all the SUs in order to select for cooperation the relay that minimizes the primary transmission time, especially when the number of SUs is large or the secondary network contains malicious SUs. A malicious SU may delete, modify or replace the bits of the primary data. As a result the primary packet is not transmitted successfully and the PU has to retransmit its packet at the beginning of the next time slot. Without considering that security threat, the PU may choose an untrustworthy SU for cooperation, which may cause the failure of cooperation and degrade the QoS of PU.

The cooperation between the PU and the SUs has attracted a lot of attention in cognitive radio networks research [25]-[47], [49]-[51]. The authors in [25] in order to mitigate interference and reduce delay, proposed a cooperation framework referred to as FTCO by considering the spectrum sharing in both the time and the frequency domains. Then they formulated the multi-hop relay selection problem as a network formation game, in which the multi-hop relay path is computed via performing the primary player's strategies in the form of link operations. The article in [30] studies and analyzes throughput and delay tradeoffs in cooperative multiple access for cognitive radio systems. It focuses on the class of randomized cooperative policies, whereby the SU serves either the queue of its own data or the queue of the PU relayed data with certain service probabilities. The authors in [33] investigated cooperative spectrum access for CRNs. Two types of cooperation schemes were proposed, whereby the PU either cooperates with two SUs or a cluster of SUs, which were referred as relay-jammer (R-J) scheme and cluster-beam forming (C-B) scheme, respectively. The article in [35] studies joint information and energy cooperation between the two systems, i.e., the primary transmitter sends information for relaying and feeds the secondary system with energy as well. This is particularly useful when the secondary transmitter has good channel quality to the primary receiver but is energy constrained.

The best relay selection issue has been investigated in [26] - [29], [31] - [32], [36] - [38], [45] where only the best relay is selected to forward a source node's signal and thus only two channels (i.e., the best relay link and direct link) are required regardless of the number of relays. Many studies [39]-[42] modeled the best relay selection as a Stackelberg game. In the Stackelberg game, the PU acts as the leader and the SU acts as the follower. As the leader, the PU can choose the best strategies, aware of the effect of its decision on the strategies of the follower (the SU); while the SU can just choose its own strategies given the selected parameters of the PU.

The main contributions of the work in this Chapter are described as follows. First, we propose two new protocols for cooperative cognitive radio networks. The first protocol is referred to as the Best Relay Selection Protocol (*BRSP*) and the second as the Stopping Criterion Protocol (*SCP*). From the simulations results it is evident that our proposed protocols reduce the primary packet transmission time, compared with the direct transmission. Second, from the simulations results it can be seen that *BRSP* is inefficient when the number of SUs is large or the SUs behave maliciously. In such a case *BRSP* wastes a significant fraction of the slot duration in order to select the best relay; furthermore the primary transmission will be delayed if the PU cooperates with a malicious SU. Third, from the simulations results it is shown that the *SCP* protocol, which scans the SUs sequentially and stops when the examined SU fulfils certain requirements, outperforms the *BRSP* protocol in many cases, e.g. when the number of SUs is large, when the packet size is small and when the secondary network contains malicious users. Many of the existing works in the literature [26] - [29], [31] - [32], [36] - [38], [45], [46], solve the relay cooperative selection problem as an optimization problem. However, due to the potentially large number of SUs it is infeasible for a PU to scan all the SUs and then pick the best one.

5.2 Network Topology

We consider a cognitive radio system in which primary and secondary networks coexist, as depicted in Fig. 5.1. In the primary network, a primary transmitter (P_t) transmits information to a primary receiver (P_r) with the help of a relay node, which is selected among N_s available relays. If the PU does not benefit from the cooperation, then P_t can send its data directly to the P_r. Meanwhile, in the secondary network, the ith secondary transmitter (S_t^i) transmits its data to the ith secondary receiver (S_r^i) simultaneously with the primary transmissions over the same channel. Notice that the N_s secondary relays are denoted by the set $R = \{S_t^i, \mathrm{i} = 1, \dots, \mathrm{N_s}\}$.

We assume that all the channel gains are perfectly known at the communication nodes. All channel gains for the network can be estimated by assuming channel reciprocity and classical channel estimation approaches [43]. Below the channel gains are defined.

- h_{p,s_i}, channel gain between primary transmitter and ith secondary transmitter
- h_p, channel gain between primary transmitter and primary receiver
- $h_{s_i,p}$, channel gain between the ith secondary transmitter and primary receiver
- h_s^i, channel gain between the ith secondary transmitter and the ith secondary receiver

If the primary transmitter decides to transmit its data directly, then P_t transmits a signal to P_r with a fixed power P_p and data rate R_{dir} (see 5.1). We assume that all the SUs have the same transmission power $P_s^i, i = 1, \dots, N_s$. Let W MHz be the channel bandwidth and N_o be the noise power. Then the transmission rates in the four different links can be computed as follows [45], [47]:

- between primary transmitter and primary receiver

$$R_{dir} = W \log_2(1 + \frac{P_p|h_p|^2}{N_0}) \text{ (Bits/sec)} \tag{5.1}$$

- between primary transmitter and ith secondary transmitter

$$R_{p,s_i} = W \log_2(1 + \frac{P_p|h_{p,s_i}|^2}{N_0}) \text{ (Bits/sec)} \tag{5.2}$$

- between ith secondary transmitter and primary receiver

$$R_{s_i,p} = W \log_2 \left(1 + \frac{P_s^i|h_{s_i,p}|^2}{N_0}\right) \text{ (Bits/sec)} \tag{5.3}$$

- between ith secondary transmitter and ith secondary receiver

$$R_s^i = W \log_2(1 + \frac{P_s^i |h_s^i|^2}{N_0}) \text{ (Bits/sec)} \quad (5.4)$$

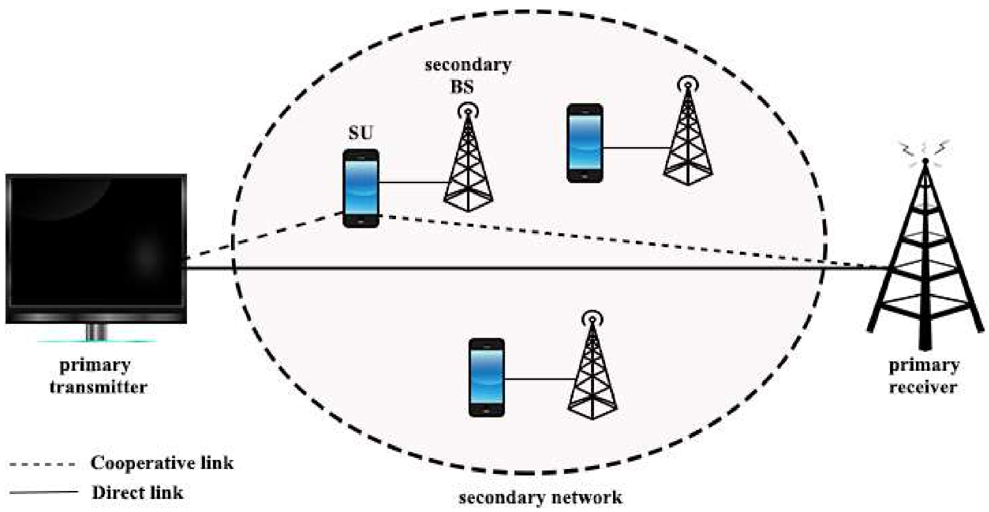


Figure 5.1: Relay model of CR network.

5.3 Problem Formulation

We consider a time slotted network. We assume that the cooperative relay selection is performed at each time slot, and the duration of a time slot is equal to T ms. For simplicity, we further assume that each PU can select at most one SU as a cooperative relay. The time slot duration is divided into four sub-time slots (phases). Fig. 5.2 presents the structure of a channel time slot.

The first phase corresponds to the observation time, in which the PU scans the available relays in order to select one for cooperation. Let t_i be the time needed for observing the ith relay. We set $t_i = \frac{2*L_s}{R_{p,s_i}}$, where L_s is the size of a small packet which is set equal to 16 bits. We assume that when the PU scans the ith SU two signals must be exchanged between them in order the PU to be informed about the quality of services that the ith SU provides. If the PU scans all the available relays then it wastes a total observation time $t_{obs} = \sum_{i=1}^{N_s} t_i$. Also in the 1st phase the PU may lose time t_w, where t_w is the wasted time. At the beginning of each slot the wasted time is equal to zero, i.e. $t_w = 0$. In case the ith SU is chosen for cooperation and it rejects it, then the wasted time is updated according to $t_w = t_w + t_i$. In that case the PU must choose another SU for relaying its data or it can decide to send its data directly. The total duration of the first phase is calculated by:

$$t_{obs} + t_w \tag{5.5}$$

In the second phase, the primary transmitter transmits its packet to the selected secondary transmitter (relay). The total duration of the second phase is calculated by:

$$\alpha_i = \frac{L_p}{R_{p,s_i}} (\text{ms}), i = 1, \dots, N_s \tag{5.6}$$

, where L_p is the size of the PU's packet.

In the third phase the secondary transmitter transmits the primary data packet to the primary receiver. The total duration of the third phase is calculated by:

$$\beta_i = \frac{L_p}{R_{s_i,p}} (\text{ms}), i = 1, \dots, N_s \tag{5.7}$$

The selected relay can transmit its own data to its secondary receiver during the fourth phase, as a reward. The total duration of the fourth phase is calculated by:

$$t_s^i = (T - \alpha_i - \beta_i - t_{obs} - t_w)(\text{ms}), i = 1, \dots, N_s \tag{5.8}$$

To make sure that the packets relayed by the cooperative relay node securely arrive at the destination, the following restriction should be satisfied:

$$0 < a_i R_{p,s_i} < \beta_i R_{s_i,p}, i = 1, \dots, N_s \tag{5.9}$$

We assume that the PU's packets arrive at the primary user according to independent Poisson processes. Let λ (arrivals/slots) be the arrival rate of the PU packets arrival processes. We further assume that the SUs have always a packet for transmission. Finally, we assume that the packets of the SUs can be segmented in smaller sizes depending on the duration of the remaining time in each slot and on the transmission rates R_s^i Mbps/sec. For example if the ith SU transmits data to its secondary receiver for duration equal to t_s^i ms then the transmitted packet size will be equal to $t_s^i * R_s^i$ Kbits. The PU has fixed size packets equal to L_p Kbits and it can transmit one packet during the duration of a time slot. In the remaining time of the slot (4th phase) only the SU that cooperated with the PU within the slot can transmit its data.

In the following sections we present two different cooperation protocols. Section 5.3.1 introduces the Best Relay Selection Protocol (***BRSP***), where the PU scans all the available relays and the best relay is chosen for cooperation. Section 5.3.2 describes the Stopping Criterion

Protocol (***SCP***), which employs a stopping criterion that takes into account the time to scan the candidate relays before stopping at a suitable one with a good channel quality.

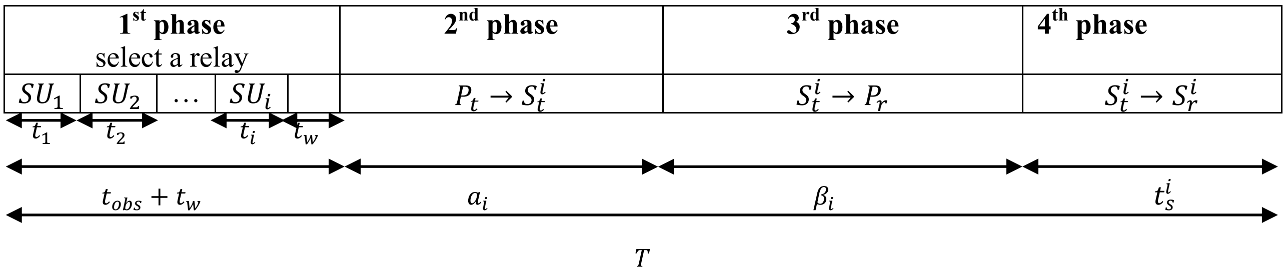


Figure 5.2: The structure of a channel time slot.

5.3.1 Best Relay Selection Protocol

We formulate a minimization problem to select the best relay node. The proposed protocol is referred to as Best Relay Selection Protocol (*BRSP*). In the primary network, a primary transmitter (P_t) transmits information to a primary receiver (P_r), potentially with the help of the best relay node which is selected amongst N_s available relays in such a way that the minimum possible delay is achieved by the destination node via the selected relay path.

For the PU to select the best relay it has to calculate the cooperation time $T_{coop_i}^{BRSP}, i = 1, \dots, N_s$, which is the required transmission time from P_t to P_r using the ith SU as a relay node.

$$T_{coop_i}^{BRSP} = t_{obs} + a_i + \beta_i + t_w \tag{5.10}$$

The SU that minimizes (5.10) over all $i = 1, \dots, N_s$, is selected as the cooperative relay node. Note that the PU will use the ith SU as a relay node only if (5.11) is satisfied:

$$T_{coop_i}^{BRSP} < T_{dir} + t_{obs} + t_w \tag{5.11}$$

, where $T_{dir} = \frac{L_p}{R_{dir}}$ ms, is the required direct transmission time from P_t to P_r.

The ith SU (the best relay) accepts the cooperation with the PU with probability p_i,

$$p_i = \min\left\{\frac{d_{SU_i}^{tol}}{d_{PU}^{tol}}, 1\right\}, i = 1, \dots, N_s \tag{5.12}$$

, where d_{PU}^{tol} denotes the total time duration the ith SU dedicated so far for relaying the primary data and $d_{SU_i}^{tol}$ denotes the total reward time it won so far for its own transmissions from the cooperation with the PU.

At the beginning of the system operation we set $p_i = 1, i = 1, \dots, N_s$. Each time the ith SU accepts the cooperation with the PU, then during the third phase of the corresponding slot it will transmit the primary data to the primary receiver, i.e. it will dedicate total time equal to β_i for the primary transmissions. The time β_i is added to the variable d_{PU}^{tol}. In case of cooperation the ith SU wins a time duration equal to $T - T_{coop_i}^{BRSP}$ to transmit its own data. That time is added to the variable $d_{SU_i}^{tol}$. Furthermore, the PU keeps a list referred to as the *best relays* and in case of cooperation with the ith SU it imports that SU in the tail of the specific list. If the PU has no packet to transmit in a time slot then it examines the list best *relays*. The first SU in the *best relays* list is selected to exploit the idle time slot. In that case the particular SU wins a time duration equal to $(T - t_i)$ for its own transmission. The above reward time is added to the variable $d_{SU_i}^{tol}$. When the secondary transmission is completed the specific SU is removed from the *best relays* list. In case the PU has no packet to transmit and the *best relays* list is empty, then the PU randomly chooses a SU to transmit during the idle slot with equal probability $\frac{1}{N_s}$.

5.3.2 Stopping Criterion Protocol

The Stopping Criterion Protocol (*SCP*) does not require information from all candidate relay nodes as it scans the candidate SU relays sequentially and stops when a suitable relay is identified. Note that *SCP* does not take into account the variable t_{obs}, because the PU does not observe all the available relays during the time slot duration.

The PU scans the available relays $R = \{S_t^i, \mathrm{i} = 1, \dots, \mathrm{N_s}\}$. At the beginning of each time slot the wasted time is set equal to $t_w = 0$. When the PU scans the ith SU then it wastes time t_i, which is added to the variable t_w. The PU stops scanning the available relays at the ith SU if the following condition holds:

$$T_{coop_i}^{SCP} < T_{dir} + t_w \tag{5.13}$$

, where $T_{coop_i}^{SCP} = a_i + \beta_i + t_w$

Therefore, (5.13) simplifies to:

$$a_i + \beta_i < T_{dir} \tag{5.14}$$

The variables a_i and β_i are calculated from equations (5.6) and (5.7), respectively The ith SU accepts the cooperation with probability p_i (see 5.12). The variables d_{PU}^{tol} and $d_{SU_i}^{tol}$ are updated as in the *BRSP* protocol. If the ith SU rejects the cooperation, then the PU examines the next SU from the set R and the procedure is repeated until the PU finds an available relay. In the extreme case where all the SUs reject the cooperation the PU will transmit its data directly.

5.4 Performance Evaluation

5.4.1 Performance metric

In our simulation study we examined the performance metric T_{avg}^{coop}, which is the total average required transmission time for a primary transmitter to send its data packet to the primary receiver using the ith SU as a relay node in the kth slot of a simulation run and is calculated as:

$$T_{avg}^{coop} = \frac{1}{N_1}\sum_{x=1}^{N_1}\frac{1}{N}\sum_{k=1}^{N} T_{coop_i}^{BRSP} \quad \text{, when } BRSP \text{ is used}$$
$$T_{avg}^{coop} = \frac{1}{N_1}\sum_{x=1}^{N_1}\frac{1}{N}\sum_{k=1}^{N} T_{coop_i}^{SCP} \quad \text{, when } SCP \text{ is used} \tag{5.15}$$

, where N is the total number of slots in each simulation run and N_1 is the total number of simulation runs.

We also studied the performance metric S_{th}^{avg}, which is the SUs's average throughput and provides the maximum transmission rate measured in Mbps. Initially we calculated the average total number of bits transferred by all the SUs by:

$$bits_{avg} = \frac{1}{N_1}\sum_{x=1}^{N_1}\sum_{k=1}^{N_s} S_k \tag{5.16}$$

, where S_k is the total achieved throughput by the kth SU at the end of each system simulation run. In our simulation model we used a run duration of 10.000 slots and the time slot duration was set equal to 10ms, i.e. the total duration of the system operation simulated in each run was equal to 100 sec. In order to measure the average throughput we divide the variable $bits_{avg}$ by 100.

5.4.2 Simulation model

We consider a time slotted (synchronous) CRN with N_s SUs and one PU. The PU is either active with a packet for transmission at the beginning of a time slot, or idle at the beginning of a slot. We assume that the PU's packets arrive according to a Poisson process. Let λ (arrivals/slots) be the arrival rate of the PU packet arrival process. The CR network is located in an area of 2000 × 2000 m^2. The PU transmitter and receiver are located at coordinates (0m, 1000m) and (2000m, 1000m), respectively, and are assumed static. The SUs are uniformly distributed in the CRN area. Similar to [39], [41] and [49], by normalizing the distance between

P_t and P_r, the ith SU is approximately placed at a distance $d_{p,s_i} \in (0, 1)$ from the P_t and $d_{s_i,p} \in (0, 1)$ from the P_r. Considering a path loss model, the average power gains between the P_t and the ith secondary transmitter and between the ith secondary transmitter and P_r, are $h_{p,s_i} = \frac{1}{\left(d_{p,s_i}\right)^n}$, and $h_{s_i,p} = \frac{1}{\left(d_{s_i,p}\right)^n}$ respectively, where $n = 2$ is the assumed path loss coefficient. Note that the primary transmitter and receiver are located at a normalized distance of 1 and the channel gain between primary transmitter and primary receiver is $h_p = 1$. We further assume that each SU has its own receiver which is also located uniformly in the CRN area. We calculate the Euclidean distance between the ith secondary transmitter and receiver, and the normalized distance between them is denoted by $d_{s_i} \in (0, 1)$. We define the average power gain between the ith secondary receiver and transmitter as $h_{s_i} = \frac{1}{d_{s_i}^n}$.

The SUs are assumed mobile, and their mobility is modeled by a Random Walk. In the Random Walk, the SUs change their speed and direction at each time interval of constant duration. For every new interval t, each SU randomly and uniformly chooses its new direction $\varphi(t)$ from $[0,2\pi)$. The new speed $v_i(t)$ follows a uniform distribution, $v_i(t) \in [0, v_{max}]$. During the time interval t, the SUs move with the velocity vector $(v_i(t)\cos(\varphi(t)), v_i(t)\sin(\varphi(t)))$. We assume that each SU can change its angle and velocity every x slots, i.e. every $t = x * T$(ms). The time slot duration is set to $T = 10$ms and we assume that during the time slot the channel gains remain constant. At the beginning of each time slot the channel gains h_{p,s_i}, $h_{s_i,p}$ and h_{s_i} of the ith SU, $i = 1, \dots, N_s$, are updated based on the new distances d_{p,s_i}, $d_{s_i,p}$ and d_{s_i}, respectively. The primary transmission power is set to $P_p = 1$mW, the secondary powers to $P_{s_i} = 2$mW and the noise level to $N_0 = 1$mW. The

TABLE 5.1*: SYSTEM PARAMETER AND DEFAULT VALUES*

Notation	Definition (default value)
T	Time slot duration (10ms)
W	Bandwidth (1MHz)
N_0	Noise level (1mW)
P_p	PU power (1mW)
P_{s_i}	SU-i power, $i = 1, \dots, N_s$ (2MW)
n	path loss coefficient (2)
L_s	small packet size (16 bits)
L_p	PU's packet size (5Kbits)
$\varphi(t)$	angle of movement of SUs, uniformly selected in the interval $[0,2\pi)$
$v_i(t)$	SU-i speed velocity, uniformly selected in the interval $[0,v_{max}]$
v_{max}	SU maximum speed velocity (40Km/h)
x	Required slots for changing SUs velocity and angle (1000)
t	time interval for changing SUs velocity and angle (10 sec)
λ	Poisson packet arrival rate at the PU (0.2)
N	Number of time slots in each simulation run(10000)
N_1	Number of simulation runs(100)

bandwidth W is set equal to 1 MHz. The PU has fixed size packets with size equal to $L_p =$ 5Kbits. All the parameters together with their default values are shown in Table 5.1.

5.4.3 Simulation results

An event-driven simulator was implemented to evaluate the performance of the proposed protocols *BRSP* and *SCP*. In the following sections we examine the impact of varying the number of SUs, the PU arrival rates and the PU packet size on the performance metrics average primary total packet transmission time and secondary average transmission throughput.

If all the SUs are well-behaved, both PU and SU can benefit from the cooperation. However, when there are some malicious SUs, the normal operation of Cooperative Cognitive Ratio Network (CCRN) cannot be guaranteed. Specifically, the following security issue arising in CCRN need to been considered [41], [47]. During cooperation, the malicious SUs can alter the packets from the PU or fabricate packets and then forward them to the destination [41]. In our simulation model we examined a scenario in which the SUs are completely honest and a scenario in which the SUs behave maliciously.

5.4.3.1 Fist Scenario: Honest SUs

In the first scenario in which the SUs are assumed honest the *BRSP* and the *SCP* protocols are implemented as described in sections 5.3.1and 5.3.2, respectively. In the following three sub-sections we present the results of this scenario.

5.4.3.1.1 The impact of varying the number of SUs

Here we examine the impact of varying the number of SUs on the performance metrics. We set the PU arrival rate equal to 0.2 packets/slot.

Fig. 5.3 shows the average primary total packet transmission time versus the number of SUs. As seen by the results in Fig. 5.3 *BRSP* outperforms *SCP* when the total number of SUs is less than 50. As the number of SUs increases beyond 50, the *SCP* protocol outperforms *BRSP*. Note that in the *SCP* protocol the average primary total packet transmission time is independent of the number of SUs. On the contrary, the performance of the *BRSP* protocol is highly affected by the number of SUs, because that protocol scans all the available relays in order to select the best relay for cooperation. Furthermore, by the illustrated results it can be seen that the *BRSP* and the *SCP* protocols improve the performance metric T_{coop}^{avg}, which is smaller than the direct transmission time T_{dir}. Generally, cooperation between the PU and the SUs benefits both the primary and secondary networks. The PU can use a relay node with better conditions/channel gain and as a result it can transmit its data with smaller delay and the secondary network can increase its average transmission throughput.

Fig. 5.4 shows the secondary average transmission throughput versus the number of SUs. The *BRSP* protocol achieves higher transmission throughput compared to the *SCP* protocol. The PU in the *BRSP* protocol selects the SU with the best channel conditions as a relay node. The specific SU will take as a reward the remaining time of the slot and in addition one idle slot for its own transmissions. Due to the higher transmission rate that achieves compared to the selected relay by the *SCP* protocol, the *BRSP* protocol achieves higher secondary average transmission throughput.

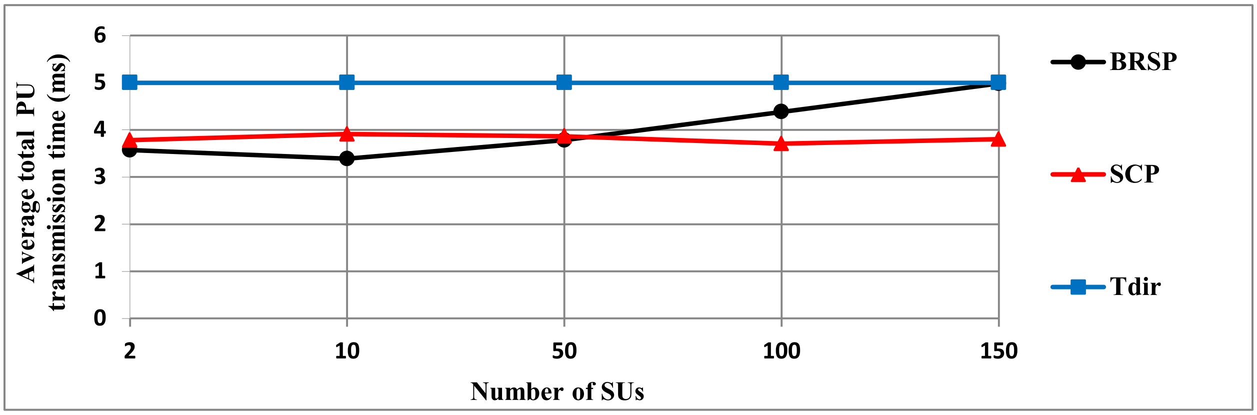


Figure 5.3: **The impact of varying the number of SUs on the average total primary packet transmission time.**

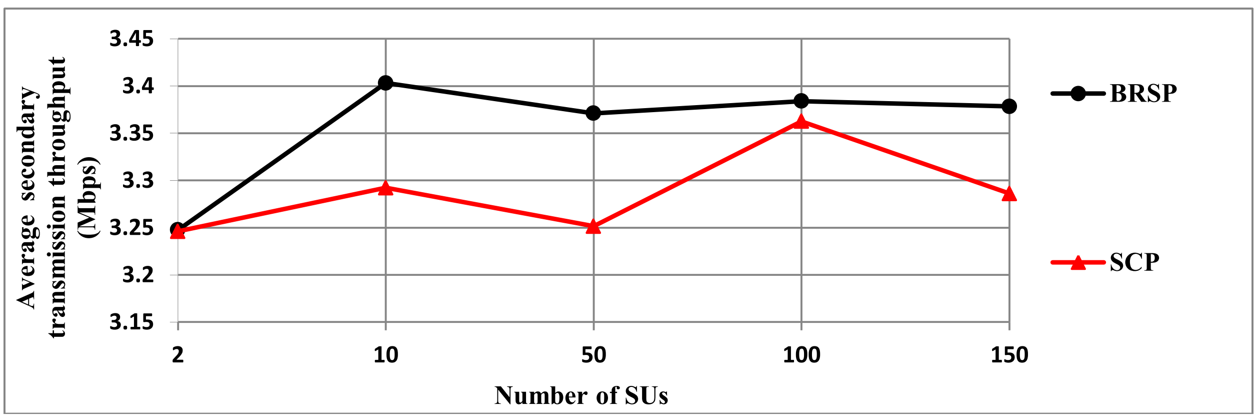


Figure 5.4: The impact of varying the number of SUs on the average secondary transmission throughput.

5.4.3.1.2 The impact of varying the PU arrival rates

Here we examine the impact of varying the PU arrival rate on the performance metrics. We examine the case where the number of SUs is set equal to 50 and the case where the number of SUs is set equal to 100.

As seen by the results in Fig. 5.5 and Fig 5.6, the *BRSP* and the *SCP* protocols achieve results close to each other in case the number of SUs is set equal to 50. In contrast, when the number of SUs is larger, i.e. equal to 100, the performance of the *BRSP* protocol is worst compared with *SCP* on the performance metric average total PU transmission time, since the former protocol is highly affected by the number of SUs (recall that *BRSP* scans all the available relays in order to select the best relay for cooperation). The total primary average transmission time is not affected by the PU arrival rates as expected. The total primary average transmission time depends on the transmission rates between the primary and secondary transmitters and between the secondary transmitter and primary receiver. As the PU arrival rate increases the average secondary transmission throughput decreases (see Fig. 5.6), since in that case the SUs transmit only during the fourth phase of each slot because the idle slots become rare.

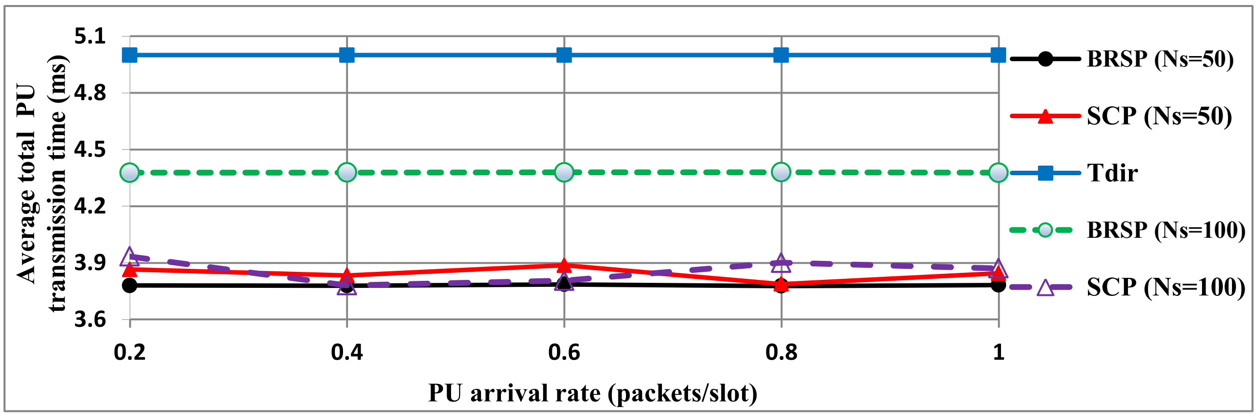


Figure 5.5: The impact of varying the PU arrival rate on the average total primary packet transmission time.

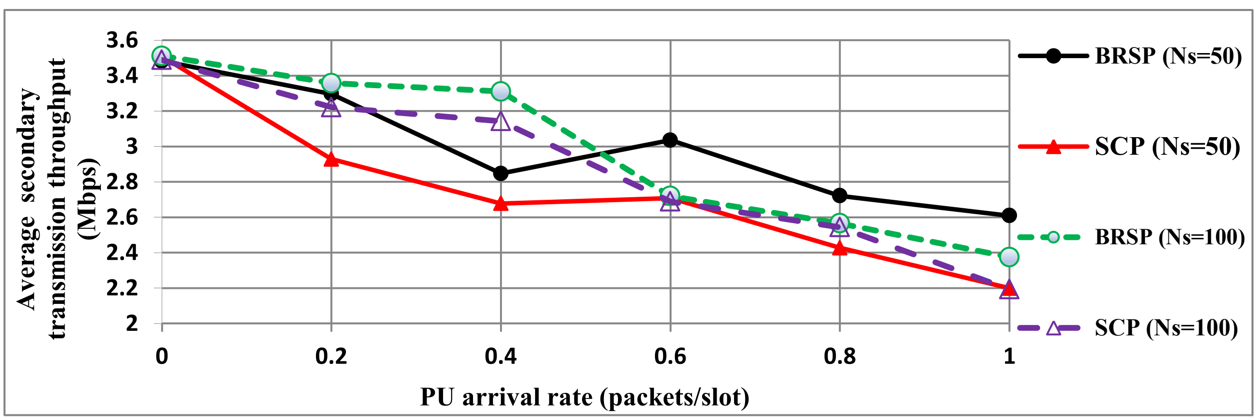


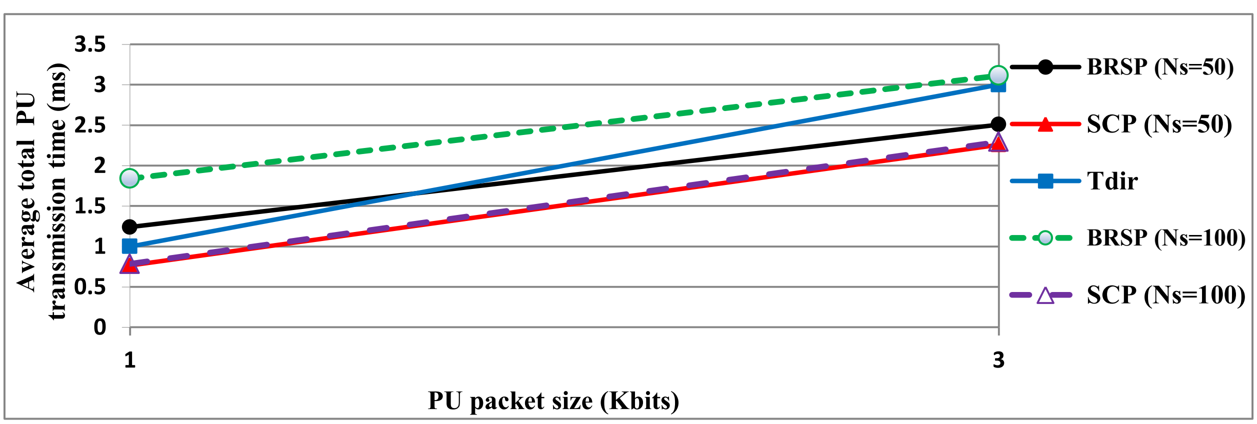
Figure 5.6: The impact of varying the PU arrival rate on the average secondary transmission throughput.

5.4.3.1.3 The impact of varying the PU packet size

Here we examine the impact of varying the PU packet size on the performance metrics. We examine the case where the number of SUs is set equal to 50 and a case where the number of SUs is set equal to 100.

Fig. 5.7 shows the impact of varying the PU's packet size on the average total primary packet transmission time. When the packet size is small, i.e. 1Kbit, the *BRSP* protocol achieves higher average total primary transmission time, than with direct transmission, for both cases $N_s = 50$ and $N_s = 100$. The *BRSP* protocol wastes a period of time equal to t_{obs} to observe all the available relays in order to choose the best relay for cooperation. When the number of SUs is large, i.e. $N_s = 100$, t_{obs} increases and as a result the *BRSP* protocol is inefficient, if the packet size is small. In contrast, the *BRSP* protocol is more efficient, compared with the direct transmission and the *SCP* protocol, in case the packet size is large, i.e. when $L_p \in [5, 9]Kbis$ and the number of SUs is set equal to $N_s = 50$. If the primary packet size is large the PU requires larger duration of the slot to send its packet and the *BRSP* protocol achieves smaller transmission times a_i and β_i, due to its selection of the best secondary channel conditions.

The *SCP* protocol outperforms *BRSP* when the packet size is small, i.e. $L_p \in [1, 3]Kbis$, in both cases $N_s = 50$ and $N_s = 100$. Furthermore, *SCP* outperforms *BRSP* when the packet size is large, i.e. $L_p \in [5,9]Kbis$, and the number of SUs is set equal to $N_s = 100$, since it then achieves lower observation time t_{obs}. As seen by the results presented in Fig. 5.7, it is more efficient to select a cooperative relay that achieves smaller primary transmission time compared with the direct transmission time T_{dir}, without scanning all the available relays, as it happens with the *SCP* protocol. When the packet size is large, e.g. equal to 9Kbits, the *BRSP* and the *SCP* protocols achieve results close to each other. The *SCP* protocol achieves smaller observation time while the *BRSP* protocol achieves smaller transmission times a_i and β_i, due to its selection of the best secondary channel conditions. Due to that the *BRSP* protocol achieves higher average secondary throughput because the SU with the highest transmission rate is selected for cooperation. The particular SU will transmit its data during the 4th phase of the current slot and it will also win a transmission opportunity in a future idle slot.



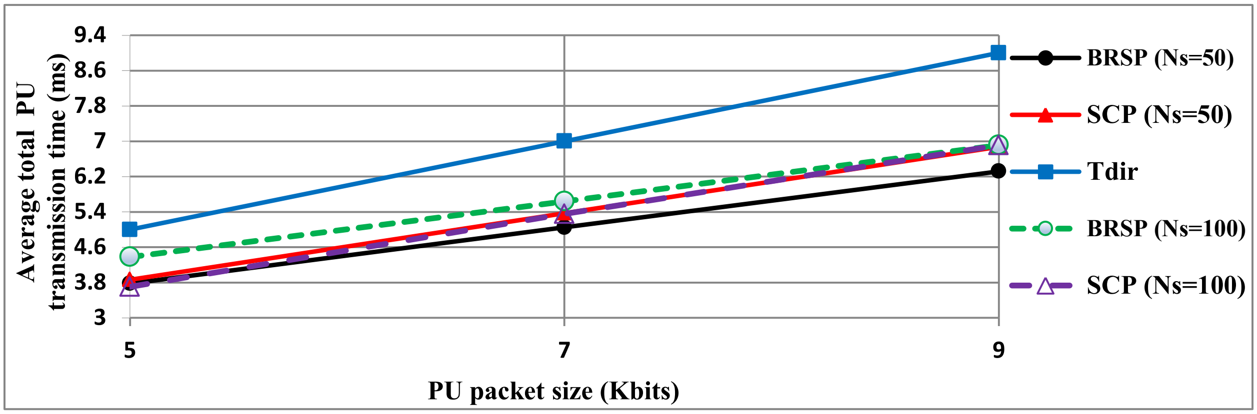


Figure 5.7: The impact of varying the PU packet size on the average total primary packet transmission time.

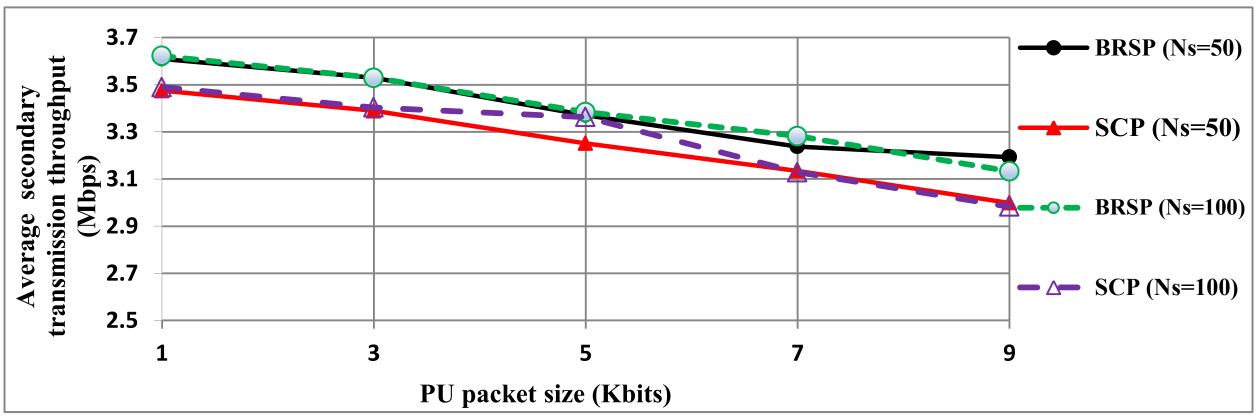


Figure 5.8: The impact of varying the PU packet size on the average secondary transmission throughput.

5.4.3.2 Second Scenario: Malicious SUs

Trust values, $T_r^i, i = 1, \dots, N_s$, are assigned to each SU at the beginning of the system operation and are utilized to characterize the behavior of each of the SUs. Each entity is assumed to behave well with probability T_r^i, and misbehave with probability $(1 - T_r^i)$, independently of the other entities, i.e., the behavior of each entity follows an independent Bernoulli process. In our study, we assume that the trust values $T_r^i, i = 1, \dots, N_s$, are selected uniformly in the interval [0.7, 1].

Note than in a practical system the primary user maintains a table for recording identities and the corresponding estimates for the trust values of the SUs. Each time after cooperation, the behavior of the selected SU is evaluated and its trust value estimate is accordingly updated. The misbehavior of a SU can be detected by the PU based on the success or failure of the transmitted PU packets via an acknowledgment (ACK/NACK) [41].

In the second scenario the SUs can behave maliciously and the *BRSP* and *SCP* protocols are modified as follows. In the *BRSP* protocol the PU selects the best SU that minimizes the cooperative transmission time taking into account the trust value of each SU, i.e. it selects the SU that minimizes the following over $i = 1, \dots, N_s$,

$$P_{coop_i}^{BRSP} = (t_{obs} + t_w + a_i + \beta_i)(1 - T_r^i) \quad (5.18)$$

The SU that minimizes (5.18) is selected as the cooperative relay node. Note that the PU will use the ith SU as a relay node only if the following inequality is satisfied:

$$T_{coop_i}^{BRSP} < (T_{d\,ir} + t_{obs} + t_w) \quad (5.19)$$

, where $T_{coop_i}^{BRSP}$ is calculated from equation (5.10)

In the *SCP* protocol the PU stops scanning the available relays at the ith SU, if the ith SU satisfies the following,

$$(T_{coop_i}^{SCP})T_r^i < (T_{dir} + t_w)T_r^i \quad (5.20)$$

, where $T_{coop_i}^{SCP} = (a_i + \beta_i + t_w)$
The trust values T_r^i can be eliminated from the inequality (5.20), therefore (5.20) is the same with inequality (5.13).

Finally, in case the PU has no packet to transmit and the list of *good relays* is empty, the PU chooses the ith SU to transmit in the idle slot with probability $\frac{T_r^i}{\sum_{j=1}^{N_s} T_r^j}, i = 1, \dots, N_s$.

In the following three sub-sections we present the simulation results of the second scenario. We examine the impact of varying: the number of SUs, the PU arrival rates and the PU packet size on the performance metrics average total primary packet transmission time and average secondary transmission throughput.

5.4.3.2.1 The impact of varying the number of SUs

Fig. 5.9 shows the impact of varying the number of SUs on the average total PU transmission time when the malicious behavior of the SUs is taken into account. We assume that $\lambda = 0.2$ and that the PU packet size is equal to 5Kbits. As seen by the results in Fig. 5.9 the *"SCP with malicious SUs"* outperforms the *"BRSP with malicious SUs",* even when the number of SUs is small. The performance of the *"BRSP with malicious SUs"* is highly affected by the presence of malicious SUs. This is due to the fact that the PU is wasting time in order to discover the best SU for cooperation and it may lose the entire time of the slot duration if the SU chosen is a malicious one. Furthermore, we conclude that the *BRSP* protocol is better than direct transmission when the number of SUs is lower than 150 and around 80 in the cases of honest and malicious SUs, respectively (see Figures 5.3 and Fig 5.9). In contrast, the *SCP* protocol always incurs an average total transmission time lower than T_{dir}. This is due to the fact that the *SCP* protocol can quickly find a SU for cooperation that fulfills the constraint in (5.20) and that the operation of that protocol is not affected by the number of SUs.

Fig. 5.10 shows the secondary average transmission throughput versus the number of SUs. The *"BRSP with malicious SUs"* protocol achieves higher transmission throughput compared with the *"SCP with malicious SUs"* protocol. The former protocol chooses the SU with the best channel conditions and thus is more likely to achieve transmission with higher transmission rate, compared with the one achieved by the *"SCP with malicious SUs"* protocol.

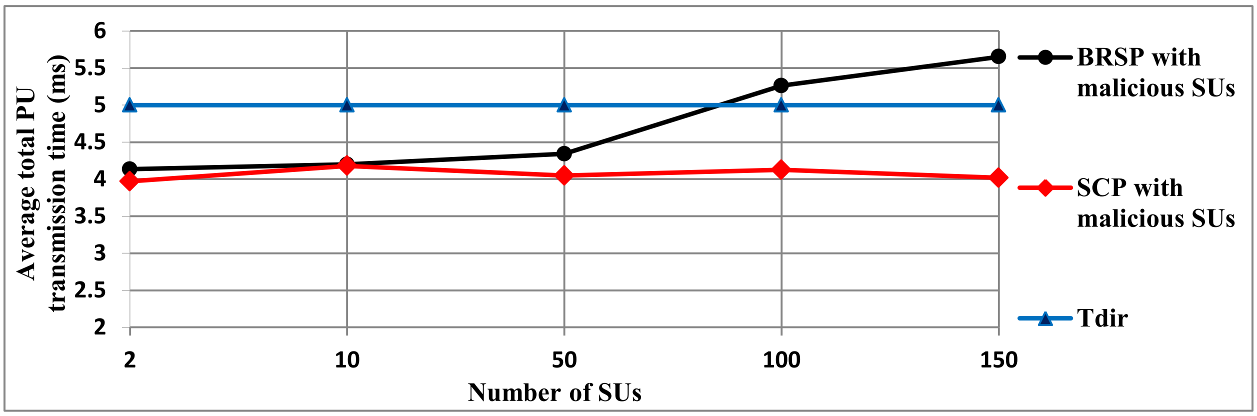


Figure 5.9: The impact of varying the number of SUs on the average total primary packet transmission time.

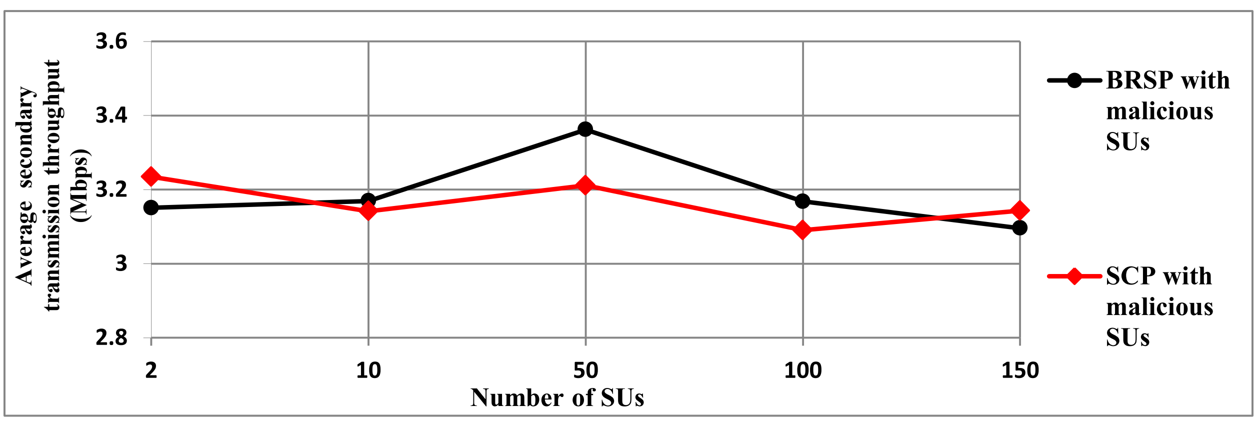


Figure 5.10: The impact of varying the number of SUs on the average secondary transmission throughput.

5.4.3.2.2 The impact of varying the PU arrival rates

Here we examine the impact of varying the PU arrival rates on the performance metrics. In our simulation we examine a scenario where the number of SUs is set equal to 50 and a scenario where the number of SUs is set equal to 100.

Fig. 5.11 shows that the performance metric average primary transmission time is practically not affected by the PU arrival rates in both of the proposed protocols. The aforementioned performance metric is affected by the distance between the primary transmitter and the secondary receiver, and the distance between the secondary transmitter and the primary receiver. As the distance between them increases the channel gains decrease, and as a result the transmission rates decreases as well. In case the SUs behave maliciously, both the *BRSP* and *SCP* protocols required longer time durations for delivering the primary data, compared to the cases with honest SUs. This is because in the case of malicious SUs it is possible for the PU to cooperate with a malicious SU and as a result the PU may lose the entire duration of the time slot, because the ith malicious relay will modify the PU's data with probability $(1 - T_r^i)$. In such case, the PU will retransmit its packet in the next time slot. The *SCP* protocol outperforms *BRSP* in both cases $N_s = 50$ and $N_s = 100$.

Fig. 5.12 presents the impact of varying the PU arrival rates on the performance metric average secondary throughput. As expected, when the PU arrival rates increase the average secondary throughput decreases since the secondary transmissions opportunities in an idle slot become rare.

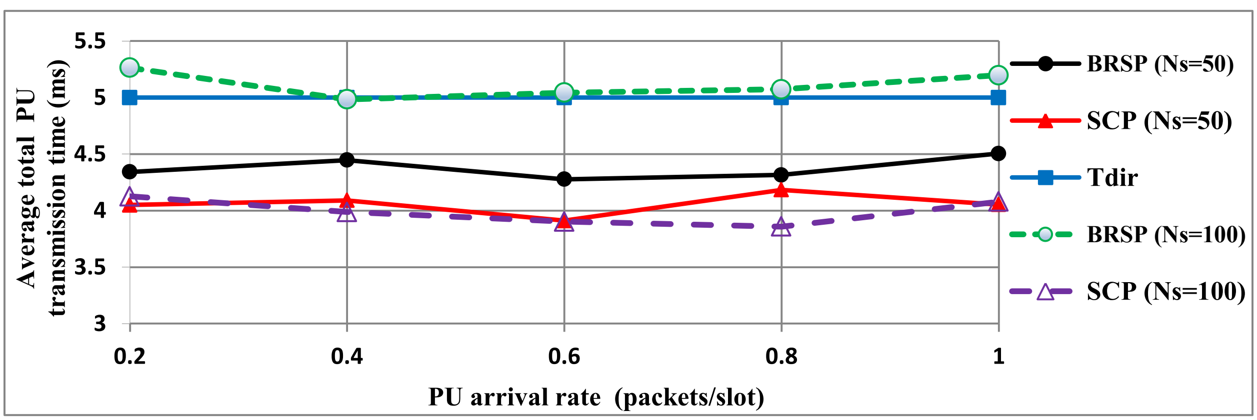


Figure 5.11: The impact of varying the PU arrival rate on the average total primary packet transmission time.

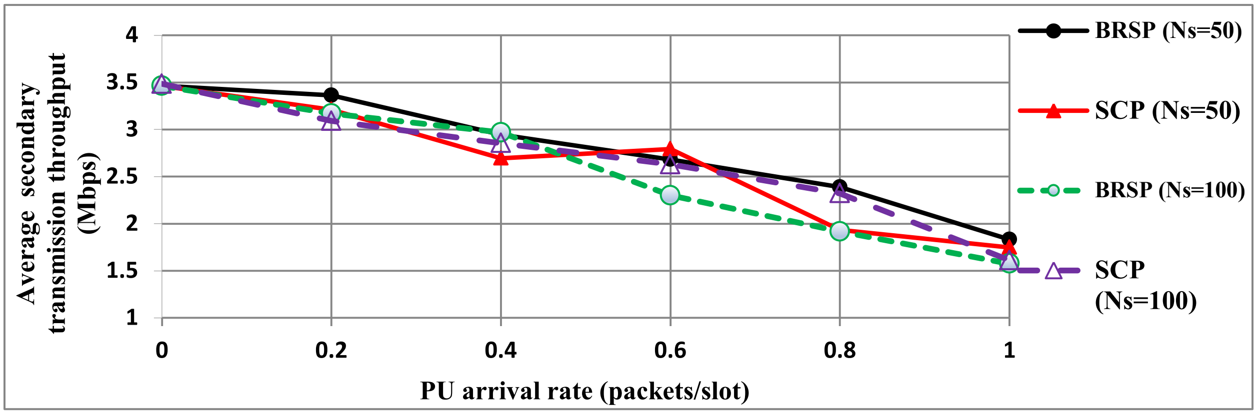


Figure 5.12: The impact of varying the PU arrival rate on the average secondary transmission throughput.

5.4.3.2.3 The impact of varying the PU packet size

Here we examine the impact of varying the PU packet size on the performance metrics. We again examine the cases $N_s = 50$ and $N_s = 100$.

Fig. 5.13 shows the average primary transmission time for the *BRSP* and *SCP* protocols in the case of malicious SUs. The results show that the *"SCP with malicious SUs"* protocol outperforms the *"BRSP with malicious SUs"* protocol, as in the case of honest SUs (see Fig 5.7). The *BRSP* protocol achieves worse performance than direct transmission when the primary packet size is small in both the cases of honest and malicious SUs. Due to the large number of SUs in this case, the *"BRSP with malicious SUs"* protocol wastes a significant fraction of the time of the slot duration to find the best relay for cooperation. In contrast, the *"SCP with malicious SUs"* protocol always outperforms *"BRSP with malicious SUs"* and the direct transmission, because the former protocol scans the SUs sequentially and stops when a SU is found that can send the primary data faster than they can be sent with direct transmission. The latter constraint does not require *SCP* to observe all the available SUs, therefore the *SCP* protocol can quickly find a SU for cooperation. The *BRSP* protocol achieves lower transmission time than the direct transmission only when the packet size is large, for both cases $N_s = 50$ and $N_s = 100$. Due to the larger packet size, the PU requires longer time duration to transmit its packet and a relay with better transmission rate can reduce the required primary transmission time.

Fig. 5.14 shows the average secondary transmission throughput. The *"BRSP with malicious SUs"* achieves the best results. Note however, that the differences between the *"BRSP with malicious SUs"* and the *"SCP with malicious SUs"* protocols are rather small.

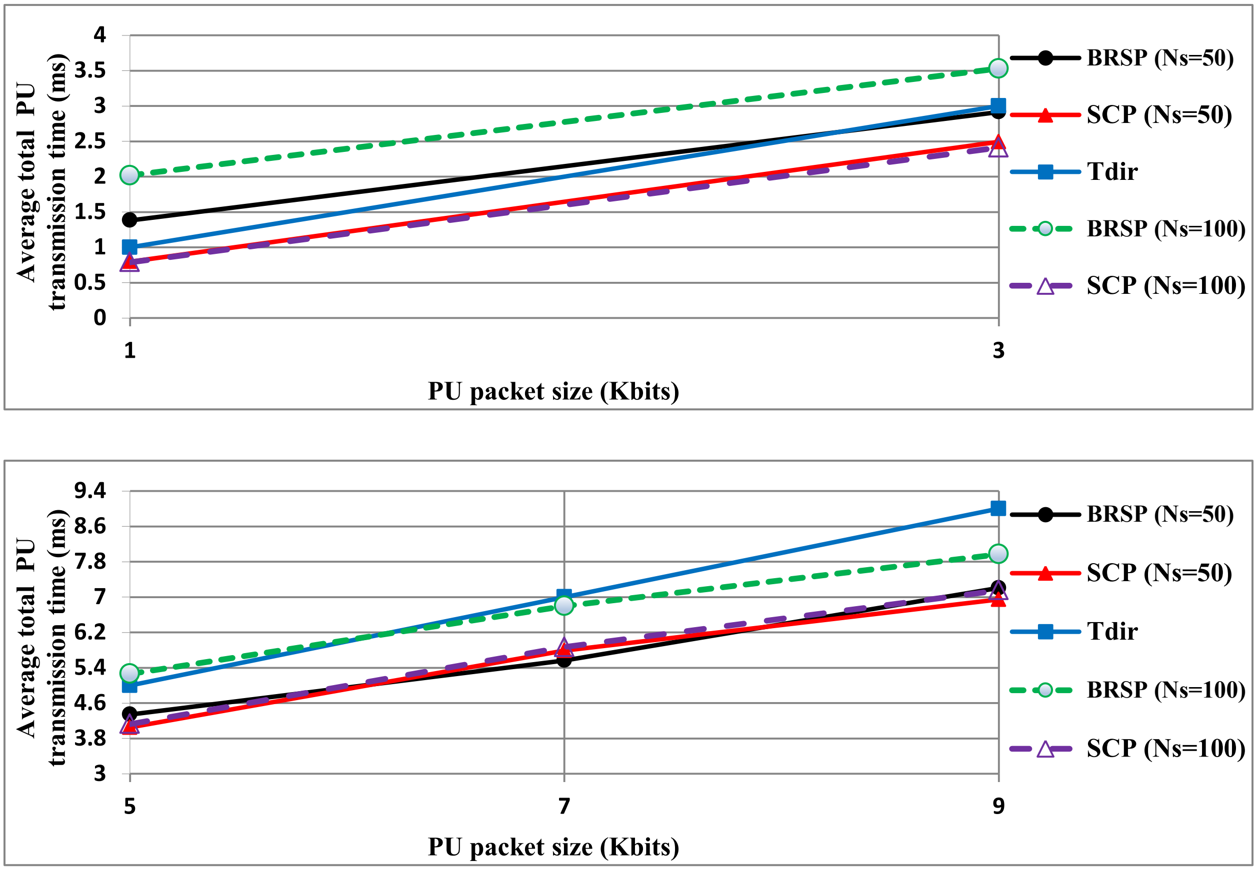


Figure 5.13: The impact of varying the PU packet size on the average total primary packet transmission time.

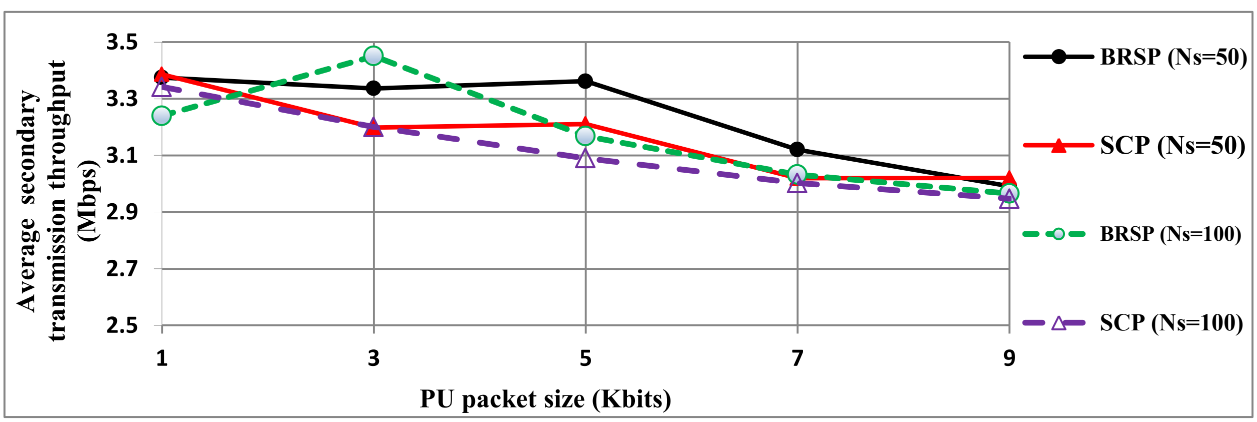


Figure 5.14: The impact of varying the PU packet size on the average secondary transmission throughput.

5.5 Main conclusions

In this Chapter, two new cooperative communication protocols are proposed and evaluated for cognitive radio networks in which one primary user and multiple SUs cooperate for mutual benefit. We proposed a new Best Relay Selection Protocol (*BRSP*) and a new Stopping Criterion Protocol (*SCP*) that achieve cooperation between the PU and the SUs. We examined two scenarios.

The the first scenario we examined assumes that all the SUs are honest, in other words always successfully transmit the PU packets. From the results of the first scenario we concluded that when the number of SUs is small and the size of the primary packet is large the *BRSP* protocol outperforms *SCP*, because the observation the time t_{obss} spent by *BRSP* is small, thus the best relay can be found quickly and that relay transmits the primary data with high transmission rate. When the number of SUs is large, the *SCP* protocol outperforms *BRSP*. Due to the large number of secondary users, *BRSP* wastes a large fraction of the time slot duration to observe all the available relays which results to a large value for the variable t_{obs} and as result the cooperation time $T_{coop_i}^{BRSP}$ increases.

In the second scenario we examined, we assumed that the SUs behave maliciously, in particular with probability $(1 - T_r^i), i = 1, \dots, N_s$ a SU chosen for cooperation will delete, modify or replace the bits in the PU's packet. As a result that PU packet is not successfully transmitted and the PU has to attempt to retransmit it in the next slot. From the results of the second scenario we concluded that *SCP* always outperforms *BRSP*, even when the number of SUs is small and the primary packet size is large. The performance of *SCP* is not affected by the number of SUs, since the protocol it scans the SUs sequentially and it quickly finds a suitable SU

TABLE 5.2: COMPARISION OF T_{coop}^{BRSP} AND T_{coop}^{SCP} IN CASE OF HONEST AND MALICIOUS SUs .

Compare T_{coop}^{BRSP} and T_{coop}^{SCP}	***BRSP* *with honest SUs***	***SCP* with *honest SUs***	***BRSP* *with malicious SUs***	***SCP* *with malicious SUs***
$N_s < 50$ $L_p = 5Kbits$	✓	✗	✗	✓
$N_s > 50$ $L_p = 5Kbits$	✗	✓	✗	✓
$N_s = 50$ $L_p \in [1,3]Kbits$	✗	✓	✗	✓
$N_s = 50$ $L_p \in [5,9]Kbits$	✓	✗	✗	✓
$N_s = 100$ $L_p \in [1,9]Kbits$	✗	✓	✗	✓

for cooperation. In Table 5.2 we summarize the behavior of the two proposed cooperation protocols *BRSP* and *SCP,* in both cases, where the SUs behave honestly and maliciously. As it can be seen from the Table, *SCP* outperforms *BRSP* in most cases and when the SUs behave maliciously in all cases.

Chapter 6: Conclusions

6.1 Overview of Work

In the first part of the Thesis (Chapter 2), we have studied sensing algorithms for homogeneous multichannel CRNs. We have considered the same network topology as in [1]. A time slotted network was considered where each PU has its own channel and the primary packets arrive at each PU according to an independent Poisson process. The SUs are assumed backlogged (i.e., they always have packets for transmission), but their transmission starts only if the PUs are not active in the specific slot. The structure of a channel time slot is comprised of several mini-slots corresponding to different stages of spectrum sensing and packet transmission. When a SU finds an idle channel, then the secondary transmission takes place on that channel until the end of the current slot, otherwise the SU will try to find another idle channel in the next mini-slot of the same slot.

In the second part of the Thesis (Chapter 3), we have studied sensing algorithms for heterogeneous multichannel CRNs. We have considered the same network topology as in [1]. Two new algorithms have been proposed and evaluated. The first algorithm is based on the assumption that a centralized entity exists and assigns the network's channels to the SUs, while in the second algorithm the SUs select their transmission channels without possessing centralized information.

In the third part of the Thesis (Chapter 4), we have studied the algorithm proposed in Chapter 3 in which the SUs select their transmission channels without coordination by a centralized entity, namely the "*Distributed algorithm*", in homogeneous multichannel CRNs.

In the last part of this Thesis (Chapter 5) we have studied cooperative communication protocols for cognitive radio networks. We proposed and evaluate two new protocols, the Best Relay Selection Protocol (*BRSP*) and the Stopping Criterion Protocol (*SCP*) which allow cooperation between a PU and multiple SUs.

6.2 Main Conclusions and Research Contribution

Our simulation results have shown that the sensing algorithms we have introduced for homogeneous multichannel CRNs and the strategies that we used improve the normalized average secondary throughput, significantly reduce the SU dropping probability and increase the total number of successful transmissions occurring during the system operation, compared to the *"PPRA"* algorithm introduced in [1].

From the results presented in Chapter 2, we have seen that the random channel selection used by the *"PPRA"* algorithm in [1], is inefficient compared with the scheme used in the algorithm *"PPRA with improvement"* where the channels are selected based on the estimated PU arrival rates. If a channel is expected to be idle, then the SUs must choose that channel with higher probability. We have also seen that the algorithms which require coordination by a centralized entity outperform the algorithms which do not use such coordination, in all examined scenarios.

From the results in Chapter 3, we have seen that the *"γ-persistent strategy"* proposed in [7], is inefficient compared to our proposed *"Distributed algorithm"* where the channels are selected based on channels capacities. Our simulation results demonstrate the weakness of the *"γ-persistent strategy"* in case the probability of missed detection is positive. In that case, in the *"γ-persistent strategy"* the SU may incorrectly drop its SS due to a missed detection. In contrast, in the *"Distributed algorithm"* each SU persists and keeps its SS for a longer duration, compared to the *"γ-persistent strategy",* and the latter prevents the SUs from wrong decisions, e.g. from dropping a SS due to a missed detection. We have also seen from our results that the adaptive change of the sensing probability of SUs, implemented by the *"Distributed algorithm",* is effective, especially when the number of channels is low and the contention among the SUs is strong.

From the results in Chapter 4 we have seen that in case of homogeneous networks the *"Distributed algorithm"* achieves results close to those of the proposed in Chapter 2 algorithms which require coordination by a centralized entity. The idea of Latin Matrix used by the *"Distributed algorithm"* is efficient, and with the appropriate strategy may lead the SUs to choose different rows from that matrix. As a result, the SUs will not compete during the system operation. In the *"Distributed algorithm"* the SU chooses one row from the Latin Matrix where the first channel of the specific row has low primary traffic. If the first channel of the row is idle then the SU will transmit on that channel for as long as possible. Furthermore, in the *"Distributed algorithm"* the SU persists in using the same SS until it observes two successive collisions and also the algorithm adaptively changes the SU's sensing probability. Due to the above two strategies, the *"Distributed algorithm"* is efficient and achieves high secondary user throughput.

From the results in Chapter 5 we have seen that the cooperation between the PU and the SUs achieves a "win-win" situation. With cooperation, a PU can increase its primary transmission rate via the relay service provided by a SU while the SU obtains channel access opportunities as a reward, which can simultaneously improve the performance of both the primary and the secondary network. Our results show that the *BRSP* protocol introduced in this Thesis is efficient when the number of SUs is small and the PU uses a large packet size. In such case the PU can find the best relay for cooperation with a small observation time t_{obs} and can transmit its data with the highest possible transmission rate. However, if the number of the available relays is large or the secondary network contains malicious SUs it is inefficient to scan all the SUs in order to pick up the best one for cooperation. In such case, the *BRSP* protocol wastes a lot of the time slot duration for observing all the SU relays and the observation time t_{obs} increases. On the contrary, the *SCP* protocol also introduced in this Thesis is not affected by the number of SUs; the PU examines the SUs sequentially and the one that achieves smaller transmission time compared with the direct transmission time by the PU is selected as the cooperative relay. Although the PU may transmit with smaller transmission rate in the *SCP* protocol compared with in the *BRSP* protocol, the primary transmission time is smaller than that of the *BRSP*, in case the number of SUs is large, because the achieved observation time with the *SCP* protocol is smaller.

6.3 Ideas for Future Work

An interesting idea for future work would be to design a non-cooperative algorithm for both the homogeneous and heterogeneous CRN cases. The new algorithm would select the secondary SS without knowledge of the PU arrival rates. Based on discussion in [5] and [21], the PU arrival rates and therefore the availability of each channel are hardly predictable. The new algorithm can use the Latin Square Matrix for selecting the secondary SS and via appropriate controls it may succeed in selecting the best rows of the Latin Matrix.

A different CRN topology may use cooperation between multiple PUs and multiple SUs. In such network the optimal pairing between PUs and SUs must be found. The new topology can take into account the interference between the SUs and the PUs, energy restrictions for the SUs and for the PUs, dishonest and selfishness SUs. A selfish SU may choose a lower transmission power than the required one during cooperation or it may just choose not to forward the PU's packet to save energy. A dishonest SU may provide fake Channel State Information (CSI) to attract cooperation with PUs in order to gain transmission opportunities.

Another idea for future work would be to examine the case of cooperative spectrum sensing, where the SUs exchange their sensing results over a common control channel. In such a system, the SUs cooperate with each other in order to reduce the competition and collisions among them.

Abbreviations and acronyms

ACK	Acknowledgment
BS	Base Station
BRSP	Best Relay Selection Protocol
CCRN	Cooperative Cognitive Ratio Network
CR	Cognitive Ratio
CRN	Cognitive Ratio Network
CSI	Chanel Status Information
PUs	Primary Users
QoS	Quality of Service
SS	Sensing Sequence
SUs	Secondary Users
SCP	Stopping Criterion Protocol

Literature

[1] H.Shokri-Ghadikolaei, F.Sheikholeslami and M.Nasiri-Kenari, "Distributed Multiuser Sequential Channel Sensing Schemes in Multichannel Cognitive Radio Networks", in *IEEE Transactions on Wireless Communications*, vol. 12, no. 5, pp 2055- 2067, May 2013.

[2] C. R. Stevenson, G. Chouinard, W. H. Z. Lei, and S. J. Shellhammer, "IEEE 802.22: the first cognitive radio wireless regional area network standard", in *IEEE Commun. Mag.*, vol. 47, no. 1, pp. 130–138, Jan. 2009.

[3] H.Shokri-Ghadikolaei and M.Nasiri-Kenari, "Optimal and Suboptimal Sensing Sequences in Multiuser Cognitive Radio Networks", in *Proceedings of the Sixth International Symposium on Telecommunications (IST2012)*, Tehran, Iran, pp 243-248, October 2012.

[4] R. Misra and A.P. Kannu, "Optimal Sensing-Order in Cognitive Radio Networks with Cooperative Centralized Sensing", in Proc. *IEEE International Conference on Communications ICC*, pp. 1566–1570, Ottawa, June 2012.

[5] H.T. Cheng and W. Zhuang, "Simple Channel Sensing Order in Cognitive Radio Networks", in *IEEE J. Sel. Areas Communications,*, vol. 29, no. 4, pp.676–688, April 2011.

[6] H. Jiang, L. Lai, R. Fan, and H. V. Poor, "Optimal selection of channel sensing order in cognitive radio", in *IEEE Trans. Wireless Communications*, vol.8, no. 1, pp 297-307, January 2009.

[7] Z. Khan, J. Lehtomaki, L. DaSilva, and M. Latva-aho, "Autonomous Sensing Order Selection Strategies Exploiting Channel Access Information", in *IEEE Transactions on mobile computing* , vol. 12, no. 2, pp. 274 –288, Feb. 2013.

[8] C.F. Laywine and G.L. Mullen, "Discrete Mathematics Using Latin Squares", in *first ed., Series in Discrete Mathematics and Optimization. John Wiley & Sons*, 1998.

[9] M. Gandetto and C. Regazzoni: "Spectrum Sensing: A Distributed Approach for Cognitive Terminals", in *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 3, pp 546- 557, April 2007.

[10] J. Mitola, "Cognitive radio: making software radio more personal", in *IEEE Pers. Comm.*, vol. 6, no. 4, pp. 48–52, August 1999.

[11] FCC, "Spectrum policy task force report," ET Cocket 02-155, in *Federal Communications Commission*, Nov 2002.

[12] F. Akyildiz , W-Y Lee and Kaushik R. Chowdhury, "CRAHNs: Cognitive radio ad hoc networks", doi:10.1016/j.adhoc.2009.01.001, 2009

[13] E. Hossain, L. Le, N. Devroye, and M. Vu: “Chapter 10 Cognitive Radio: From Theory to Practical Network Engineering”, V. Tarokh (ed.), *New Directions in Wireless Communications Research*, pp 251-289, 2009.

[14] E. Matskani, N. Chatzidiamantis, L. Georgiadis, I. Koutsopoulos and L. Tassiulas, “Optimal Primary-Secondary user Cooperation Policies in Cognitive Radio Networks”, arXiv preprint arXiv:1307.5613, July 2013.

[15] Ying-Chang Liang, Y. Zeng, Edward C.Y. Peh, and Anh Tuan Hoang, “Sensing-Throughput Tradeoff for Cognitive Radio Networks”, in *IEEE Transactions on Wireless Communications*, vol.7, no. 4, pp. 1326-1337, April 2008.

[16] R. Fan, H. Jiang, Q. Guo, and Z. Zhang, “Joint Optimal Cooperative Sensing and Resource Allocation in Multichannel Cognitive Radio Networks”, in *IEEE Transactions on Vehicular Technology*, vol. 60, no. 2, pp 722-729, February 2011.

[17] R. Fan and H. Jiang, “Channel Sensing-Order Setting in Cognitive Radio Networks: A Two-User Case”, in *IEEE Transactions on Vehicular Technology*, vol. 58, no. 9, pp 4977- 5008, November 2009.

[18] C.-W. Wang, L.-C. Wang and F. Adachi, “Modeling and Analysis for Proactive-Decision Spectrum Handoff in Cognitive Radio Networks”, *in IEEE Global Telecommunications Conference (GLOBECOM 2010)*, pp. 1-6, December 2010.

[19] H.Shokri-Ghadikolaei, I. Glaropoulos, V. Fodor, C. Fischione, and K. Dimou, “Energy Efficient Spectrum Sensing and Handoff Strategies in Cognitive Radio Networks”, arXiv:1312.0045v1 [cs.NI], pp 1-7, 30 Nov 2013.

[20] A. Ghasemi and E. Sousa, “Optimization of spectrum sensing for opportunistic spectrum access in cognitive radio networks”, in *Proc. IEEE Consumer Communications and Networking Conference (CCNC 2007)*, Las Vegas, Nevada, USA, pp. 1022–1026, January 2007.

[21] Q. Zhao, L. Tong, A. Swami, and Y. Chen, “Decentralized Cognitive MAC for Opportunistic Spectrum Access in Ad Hoc Networks: A POMDP Framework”, in *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 3, pp 589- 600, April 2007.

[22] G. Baldini, O. Holland, V. Stavroulaki, K. Tsagkaris, P. Demestichas, A. Polydoros, S. Karanasios, and D. Allen, “The evolution of cognitive radio technology in Europe: Regulatory and standardization aspects”, Telecommunications Policy, vol. 37, no. 23, pp. 96-107, 2013.

[23] V. P. Kalogirou, E.D Nanou, N.C Capsalis, T-H.N Velivasaki, C.N. Capsalis, “Compatibility of DVBT Services and IMT-2000 Compliant Mobile Telecommunications in the UHF Band of 470 - 862 MHz”, TELSIKS '09, 9th International Conference on *Telecommunication in Modern Satellite, Cable, and Broadcasting Services*, pp 247-254, October 2009.

[24] S. Sayyadi and J.Nourinia, “Performance evaluation of sequential spectrum handoff in cognitive radio networks: a single-user case”, *in International Journal of Communication Systems,* vol. 27, no. 12, pp 3768-3780, December 2014.

[25] W. Li, X. Cheng, T. Jing, and X. Xing, "Cooperative multi-hop relaying via network formation games in cognitive radio networks", *in IEEE INFOCOM* , pp 971 – 979, Turin, April 2013.

[26] Y. Zou, *J.* Zhu, B. Zheng, and Yu-Dong Yao, "An Adaptive Cooperation Diversity Scheme with Best-Relay Selection in Cognitive Radio Networks"*, in IEEE Transactions on Signal Processing*, vol. 58, no.10, pp 5438-5445, September 2010.

[27] Y. Zou, Yu-Dong Yao and B. Zheng, "A Cooperative Sensing Based Cognitive Relay Transmission Scheme without a Dedicated Sensing Relay Channel in Cognitive Radio Networks", *in IEEE Transactions on Signal Processing*, vol.59, no.2, pp 854- 858, January 2011.

[28] M. Seyfi, S. Muhaidat and J. Liang, "Amplify-and-Forward Selection Cooperation over Rayleigh Fading Channels with Imperfect CSI", *in IEEE Transactions on Wireless Communications*, vol. 11, no. 1, pp 199-209, January 2012.

[29] Y. Zou, Yu-Dong Yao and B. Zheng, "Cognitive Transmissions with Multiple Relays in Cognitive Radio Networks", *in IEEE Transactions on Wireless Communications,* vol. 10, no.2, pp 648-659, February 2011.

[30] M. Ashour, A. A. El-Sherif, T. ElBatt, and A. Mohamed, "Cognitive radio networks with probabilistic relaying: stable throughput and delay tradeoffs", http://arxiv.org/abs/1404.6112, April 2014.

[31] F.R.V. Guimaraes, D.B. da Costa, M. Benjillali, T.A. Tsiftsis and G.K. Karagiannidis, "Cooperative spectrum sharing systems with relay selection in the presence of multiple primary receivers", *in IET Communications*, Vol. 8, no. 4, pp 546–553, 2014.

[32] T. Jing, S. Zhu, H. Li, X. Cheng, and Y. Huo, "Cooperative relay selection in cognitive radio networks," *in Proceeding IEEE INFOCOM*, pp 175-179, April 2013.

[33] N. Zhang, N. Lu*,* N. Cheng J. W. Mark*,* X. Shen, "Cooperative Spectrum Access Towards Secure Information Transfer for CRNs", *in IEEE Journal on Selected Areas in Communications*, vol. 31, no. 11, pp 2453- 2464, November 2013.

[34] M. Ashour, A. El-Sherif, T. ElBatt, and A. Mohamed, "Cooperative access in cognitive radio networks: Stable throughput delay tradeoffs," in *Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt),* pp 263-270, May 2014.

[35] G. Zheng, Z. Ho, E. Jorswieck, and B. Ottersten, "Information and energy cooperation in cognitive radio networks," *in IEEE Transactions on Signal Processing*, vol. 62, no.9, pp 2290-2303, April 2014.

[36] X. Zhang, *J.* Xing, Z. Yan, Y. Gao, and W. Wang, "Outage Performance Study of Cognitive Relay Networks with Imperfect Channel Knowledge", *in IEEE Communications Letters*, vol. 17, no. 1, pp 27-30, January 2013.

[37] A. Alsharoa, F. Bader and M-Slim Alouini, "Optimal Relay Selection and Resource Allocation for Two Way DF-AF Cognitive Radio Networks", http://arxiv.org/abs/1303.3489v4, May 2013.

[38] X. Zhang, Z. Yan, Y. Gao, and W. Wang, "On the Study of Outage Performance for Cognitive Relay Networks (CRN) with the *N*th Best-Relay Selection in Rayleigh-fading Channels", in IEEE Wireless *Communications Letters*, vol. 2, no. 1, pp 110-113, February 2013.

[39] J. Zhang and Q. Zhang, "Stackelberg Game for Utility-Based Cooperative CognitiveRadio Networks" *in Proceedings of the tenth ACM international symposium on Mobile ad hoc networking and computing ACM*, pp. 23–32, 2009.

[40] O. Simeone, I. Stanojev, S. Savazzi, Y. Bar-Ness, U. Spagnolini, and R. Pickholtz, "Spectrum Leasing to Cooperating Secondary Ad Hoc Networks", *in IEEE Journal on Selected Areas in Communications*, vol. 26, no. 1, pp 203- 213, January 2008.

[41] N. Zhang, N. Cheng, N. Lu, H. Zhou, J. W. Mark, and X. Shen, "Risk-aware cooperative spectrum access for multi-channel cognitive radio networks," *in IEEE Journal on Selected Areas in Communications*, vol. 32, no. 3, pp. 516–527, 2014.

[42] Y. Wu and K. J. Ray Liu, "An Information Secrecy Game in Cognitive Radio Networks", *in IEEE Transactions on Information Forensics and Security*, vol. 6, no. 3, pp 831- 842, September 2011.

[43] R. Zhang, S. Cui, and Y.-C. Liang, "On ergodic sum capacity of fading cognitive multiple-access channel," *in IEEE Transactions on Information Theory*, vol. 55, no.11, pp. 5161–5178, October 2009.

[44] M. Ge and S. Wang, "Energy-Efficient Power Allocation for Cooperative Relaying Cognitive Radio Networks", *in IEEE Wireless Communications and Networking Conference (WCNC)*, pp 691-696, April 2013.

[45] W. Su, J. D. Matyjas, and S. Batalama, "Active cooperation between primary users and cognitive radio users in cognitive adhoc networks", *in Proc. IEEE International Conference on Acoustics Speech and Signal Processing* (ICASSP), pp. 3174–3177, March 2010.

[46] H. Xu, B. Li, "Resource Allocation with Flexible Channel Cooperation in Cognitive Radio Networks", *in IEEE Transactions on Mobile Computing*, vol. *12*, no. 5, pp 957–970, March 2013.

[47] H. Yao and S. Zhong, "Towards Cheat-Proof Cooperative Relay for Cognitive Radio Networks", *in IEEE Transactions on Parallel and Distributed*, 2013.

[48] M. T. Masonta, M. Mzyece and N. Ntlatlapa, "Spectrum decision in cognitive radio networks: A survey", *in IEEE Communications Surveys & Tutorials*, vol. 15, no. 3, pp 1088–1107, July 2013.

[49] P. Li, *Member,* S. Guo, W. Zhuang, and B. Ye, "On Efficient Resource Allocation for Cognitive and Cooperative Communications", *in IEEE Journal on Selected Areas in Communications*, vol. 32, no. 2, pp 264 – 273, January 2014.

[50] C-X. Wang, *X.* Hong, *H-H.* Chen and J. Thompson, "On Capacity of Cognitive Radio Networks with Average Interference Power Constraints" *in IEEE Transactions on Wireless Communications*, vol. 8, no. 4, pp 1620-1625, April 2009.

[51] X. Chen and *H-H.* Chen, *"*Interference-Aware Resource Control in Multi-Antenna Cognitive Ad Hoc Networks with Heterogeneous Delay Constraints*" in IEEE Communications Letters*, vol. 17, no. 6, pp. 1184-1187, June 2013.

[52] Z. Zhang and H. Jiang, "Cognitive Radio with Imperfect Spectrum Sensing: The optimal Set of Channels to Sense", in *IEEE Wireless Communications Letters*, vol. 1, no. 2, pp 133-136, April 2012.