



**ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ  
ΤΜΗΜΑ ΗΛΕΚΤΡΟΝΙΚΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ  
ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ**

**ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ**

**Αλγόριθμοι αίσθησης καναλιών για γνωστικά ραδιο-δίκτυα**

**ΨΑΡΟΜΑΝΩΛΑΚΗΣ ΝΙΚΟΛΑΟΣ**

**A.M: 2010030110**

**Εξεταστική επιτροπή: Καθ. Μ. Πατεράκης (Επιβλέπων)  
Καθ. Α. Λιάβας  
Αναπλ. Καθ. Π. Κουτσάκης**

**Χανιά, 2015**

## *Ευχαριστίες*

Αρχικά θα ήθελα να ευχαριστήσω την οικογένεια μου, η οποία υπήρξε ένα ανεκτίμητο στήριγμα για μένα και της οφείλω όλη τη διαδρομή των σπουδών μου, μέχρι σήμερα. Εν συνέχεια, ευχαριστώ θερμά τον επιβλέποντα της διπλωματικής εργασίας Καθηγητή κ. Μιχάλη Πατεράκη. Με τις συμβουλές, το ενδιαφέρον, τις γνώσεις και την πολύτιμη καθοδήγησή του κατάφερα να υλοποιήσω επιτυχώς τη διπλωματική εργασία. Επίσης, θέλω να ευχαριστήσω τα μέλη της εξεταστικής επιτροπής κ. Λιάβα και κ. Κουτσάκη. Τέλος, θα ήθελα να ευχαριστήσω την κα. Μαρία Θεοδώρου για την πολύτιμη βοήθεια της, στη διάρκεια εκπόνησης της διπλωματικής εργασίας.

## Περίληψη

Τα τελευταία χρόνια έχει παρατηρηθεί δραματική αύξηση της ζήτησης φάσματος από τις ανερχόμενες ασύρματες εφαρμογές. Αυτό οφείλεται στο όλο και αυξανόμενο ενδιαφέρον των καταναλωτών για χρησιμοποίηση ασύρματων υπηρεσιών, το οποίο με τη σειρά του οδηγεί στην εξέλιξη των ασύρματων δικτύων σε δίκτυα υψηλής ταχύτητας. Μελέτες έχουν δείξει τη μη-πλήρη αξιοποίηση του φάσματος από τα παραδοσιακά συστήματα ασύρματης επικοινωνίας αλλά και τη συνεχή μείωση των διαθέσιμων συχνοτήτων για νέες εφαρμογές. Τα γνωστικά ασύρματα ραδιο-δίκτυα (Cognitive radio networks) θεωρούνται μία πολλά υποσχόμενη τεχνολογία, με σκοπό την αντιμετώπιση του παραπάνω προβλήματος. Τα γνωστικά ασύρματα ραδιο-δίκτυα επιτρέπουν σε μη – εξουσιοδοτημένους χρήστες (δευτερεύοντες χρήστες-SUs) να συνυπάρχουν, χωρίς να προκαλούν παρεμβολές, με εξουσιοδοτημένους χρήστες (πρωτεύοντες χρήστες-PUs) στο ίδιο φάσμα.

Το αντικείμενο της παρούσας διπλωματικής εργασίας είναι η μοντελοποίηση και η ανάπτυξη αλγορίθμων αίσθησης της κατάστασης των καναλιών από τους μη-εξουσιοδοτημένους χρήστες (SUs) με σκοπό την καλύτερη αξιοποίηση του φάσματος, όταν αυτό δεν χρησιμοποιείται από τους εξουσιοδοτημένους χρήστες (PUs). Στο πρώτο μέρος της εργασίας, αναπτύσσονται αλγόριθμοι αίσθησης καναλιών σε CRN με ομογενή κανάλια. Όλοι οι δευτερεύοντες χρήστες λαμβάνουν με την ίδια ισχύ τα σήματα που στέλνουν όλοι οι πρωτεύοντες χρήστες με το κέρδος του καναλιού να είναι σταθερό για όλες τις δυνατές συνδέσεις πρωτευόντων-δευτερευόντων χρηστών. Οι αλγόριθμοι αυτοί στη συνέχεια μελετώνται και συγκρίνονται, σε διάφορα σενάρια που εξετάζονται, με αποδοτικούς αλγορίθμους που έχουν προταθεί στην βιβλιογραφία και σχολιάζονται τα αποτελέσματά τους.

Στο δεύτερο μέρος της εργασίας, αναπτύσσονται αλγόριθμοι αίσθησης καναλιών σε CRN με ετερογενή κανάλια. Στην περίπτωση αυτή, οι χρήστες είναι διασκορπισμένοι στο χώρο, με αποτέλεσμα κάθε δευτερεύων χρήστης να λαμβάνει σήματα με διαφορετική ισχύ από κάθε πρωτεύοντα χρήστη ανάλογα με την απόσταση που έχει από αυτόν. Έτσι, είναι λογικό να μπορεί να ανιχνεύσει καλύτερα την ύπαρξη χρηστών στο φάσμα που είναι σε κοντινότερη απόσταση από αυτόν, με δεδομένο ότι όλοι οι πρωτεύοντες χρήστες υποτίθεται ότι έχουν την ίδια εκπεμπόμενη ισχύ. Επίσης η χωρητικότητα κάθε καναλιού για κάθε δευτερεύοντα χρήστη είναι διαφορετική, αφού βάσει του θεωρήματος Shannon-Fano αυτή εξαρτάται από την απόσταση των δύο χρηστών (πομπού – δέκτη). Έπειτα, οι προτεινόμενοι αλγόριθμοι μελετώνται και συγκρίνονται σε διάφορα σενάρια που εξετάζονται και σχολιάζονται τα αποτελέσματά τους.

Το τρίτο μέρος της εργασίας, περιέχει τον επίλογο. Αρχικά, αναλύονται τα κύρια τελικά συμπεράσματα και συζητείται η συνεισφορά της εργασίας. Τέλος, αναφέρονται ορισμένες ιδέες για μελλοντικές επεκτάσεις.

## Περιεχόμενα

ΚΕΦΑΛΑΙΟ 1 <sup>ο</sup> : Εισαγωγή .....	7
1.1 Εισαγωγή στα γνωστικά ασύρματα ραδιο-δίκτυα .....	7
1.1.1 Φασματικά κενά .....	8
1.2 Λειτουργίες των γνωστικών ασύρματων ραδιο-δικτύων (CRNs) .....	9
1.2.1 Αίσθηση του φάσματος (spectrum sensing) στα γνωστικά ασύρματα ραδιο-δίκτυα .....	10
1.2.2 Απόφαση για το φάσμα (spectrum decision) στα γνωστικά ασύρματα ραδιο-δίκτυα .....	12
1.2.3 Μοίρασμα του φάσματος (spectrum sharing) στα γνωστικά ασύρματα ραδιο-δίκτυα .....	13
1.2.4 Κινητικότητα του φάσματος (spectrum mobility) στα γνωστικά ασύρματα ραδιο-δίκτυα .....	13
1.3 Τρόποι λειτουργίας των γνωστικών ασύρματων ραδιο-δικτύων ( CRN ).....	15
1.4 Περιγραφή διαγράμματος διπλωματικής εργασίας .....	16
ΚΕΦΑΛΑΙΟ 2 <sup>ο</sup> : Σχεδιασμός και αξιολόγηση αποδόσεων των προτεινόμενων αλγορίθμων για αίσθηση καναλιών σε CRNs με ομογενή κανάλια .....	18
2.1 Εισαγωγή .....	18
2.2 Περιγραφή συστήματος.....	19
2.2.1 Απλός αλγόριθμος τυχαίας επιλογής καναλιού (PPRA) .....	21
2.2.2 Κατανεμημένος αλγόριθμος (Distributed Algorithm).....	22
2.2.3 Αλγόριθμος κατασκευής SS (Build SS) .....	23
2.2.3.1 Ανάλυση λογικής κατασκευής της λίστας καναλιών ( SS ).....	24
2.2.3.2 Ανάλυση λογικής ανά-κατασκευής SS - διαγραφή καναλιών .....	26
2.2.4 Αλγόριθμος κατασκευής SS με διαφορετική στρατηγική αποθήκευσης και διαγραφής (Build SS_2) .....	28
2.2.5 Αλγόριθμος κατασκευής SS με συνεργασία (Build SS coop).....	29
2.3 Αξιολόγηση απόδοσης .....	30
2.3.1 Μετρική απόδοσης .....	30
2.3.2 Μοντέλο προσομοίωσης.....	30
2.3.3 Αποτελέσματα προσομοιώσεων .....	31
2.3.3.1 Αποτελέσματα προσομοιώσεων σχετικά με τον κατάλληλο αριθμό συνεχόμενων συγκρούσεων που πρέπει να υποστεί ένας SU προτού διαγράψει το κανάλι στον αλγόριθμο Build SS .....	32

2.3.3.2 Αποτελέσματα προσομοιώσεων όταν μεταβάλουμε την πιθανότητα P_delete_channel στον αλγόριθμο Build SS .....	35
2.3.3.3 Σύγκριση αλγορίθμων Build SS και Build SS_2 .....	40
2.3.3.3.1 Αποτελέσματα προσομοιώσεων για $N_p > N_s$ .....	40
2.3.3.3.2 Αποτελέσματα προσομοιώσεων για $N_p < N_s$ .....	41
2.3.3.3.3 Επίδραση μεταβολής του ρυθμού άφιξης πακέτων λ των PUs .....	42
2.3.3.3.4 Σχολιασμός αποτελεσμάτων σύγκρισης αλγορίθμων Build SS και Build SS_2 .....	43
2.3.3.4 Αποτελέσματα προσομοιώσεων για $N_p > N_s$ .....	44
2.3.3.4.1 Περίπτωση όπου τα κανάλια είναι ελαφρώς φορτωμένα .....	44
2.3.3.4.2 Περίπτωση όπου τα κανάλια είναι υψηλά φορτωμένα .....	46
2.3.3.4.3 Περίπτωση όπου τα κανάλια είναι άνισα φορτωμένα .....	47
2.3.3.4.4 Επίδραση μεταβολής του ρυθμού άφιξης πακέτων λ των PUs .....	48
2.3.3.4.5 Σχολιασμός αποτελεσμάτων για την περίπτωση όπου $N_p > N_s$ .....	48
2.3.3.5 Αποτελέσματα προσομοιώσεων για $N_p < N_s$ .....	50
2.3.3.5.1 Περίπτωση όπου τα κανάλια είναι ελαφρώς φορτωμένα .....	50
2.3.3.5.2 Περίπτωση όπου τα κανάλια είναι υψηλά φορτωμένα .....	52
2.3.3.5.3 Περίπτωση όπου τα κανάλια είναι άνισα φορτωμένα .....	53
2.3.3.5.4 Επίδραση μεταβολής του ρυθμού άφιξης πακέτων λ των PUs .....	54
2.3.3.5.5 Σχολιασμός αποτελεσμάτων για την περίπτωση όπου $N_p < N_s$ .....	55
ΚΕΦΑΛΑΙΟ 3 <sup>ο</sup> : Σχεδιασμός και αξιολόγηση αποδόσεων των προτεινόμενων κατανεμημένων αλγορίθμων για αίσθηση καναλιών σε CRNs με ετερογενή κανάλια .....	58
3.1 Εισαγωγή .....	58
3.2 Περιγραφή συστήματος .....	59
3.3 Ανάλυση προβλήματος .....	60
3.3.1 Αλγόριθμος κατασκευής ακολουθίας SS χρησιμοποιώντας την εκτίμηση της χωρητικότητας του καναλιού (Build SS using capacity) .....	62
3.3.2 Αλγόριθμος αίσθησης ιδανικότερων ως προς την εκτίμηση καναλιών (ICESA- ideals channels for the estimation sensing algorithm) .....	63
3.3.3 Αλγόριθμος αίσθησης ιδανικότερων ως προς την εκτίμηση καναλιών χρησιμοποιώντας την πιθανότητα λανθασμένης ανίχνευσης (ICESA using Pmd - ideals channels for the estimation sensing algorithm using Pmd) .....	65
3.4 Αξιολόγηση απόδοσης .....	66
3.4.1 Μετρική απόδοσης .....	66
3.4.2 Μοντέλο προσομοίωσης .....	67
3.4.3 Αποτελέσματα προσομοιώσεων .....	68

3.4.3.1 Αποτελέσματα προσομοιώσεων βάσει του αριθμού των συνεχόμενων συγκρούσεων που πρέπει να υποστεί ένας SU προτού αποφασίσει να προσπαθήσει να αλλάξει θέση ένα κανάλι στη λίστα του στον αλγόριθμο ICESA.....	69
3.4.3.2 Αποτελέσματα προσομοιώσεων βάσει της τιμής της πιθανότητας $p\_change\_position$ στον αλγόριθμο ICESA.....	72
3.4.3.3 Σύγκριση αλγορίθμων ICESA και ICESA using Pmd.....	74
3.4.3.4 Αποτελέσματα προσομοιώσεων για $N_p > N_s$ .....	76
3.4.3.4.1 Περίπτωση όπου τα κανάλια είναι χαμηλά φορτωμένα.....	76
3.4.3.4.2 Περίπτωση όπου τα κανάλια είναι υψηλά φορτωμένα.....	77
3.4.3.4.3 Περίπτωση όπου τα κανάλια είναι άνισα φορτωμένα.....	78
3.4.3.4.4 Σχολιασμός αποτελεσμάτων για την περίπτωση όπου $N_p > N_s$ .....	80
3.4.3.5 Αποτελέσματα προσομοιώσεων για $N_p < N_s$ .....	81
3.4.3.5.1 Περίπτωση όπου τα κανάλια είναι χαμηλά φορτωμένα.....	81
3.4.3.5.2 Περίπτωση όπου τα κανάλια είναι υψηλά φορτωμένα.....	81
3.4.3.5.3 Περίπτωση όπου τα κανάλια είναι άνισα φορτωμένα.....	82
3.4.3.5.4 Σχολιασμός αποτελεσμάτων για την περίπτωση όπου $N_p < N_s$ .....	84
ΚΕΦΑΛΑΙΟ 4 <sup>ο</sup> : Επίλογος .....	86
4.1. Συνεισφορά της διπλωματικής εργασίας.....	86
4.2. Ιδέες μελλοντικών επεκτάσεων .....	88
Βιβλιογραφία.....	89

# ΚΕΦΑΛΑΙΟ 1<sup>ο</sup>: Εισαγωγή

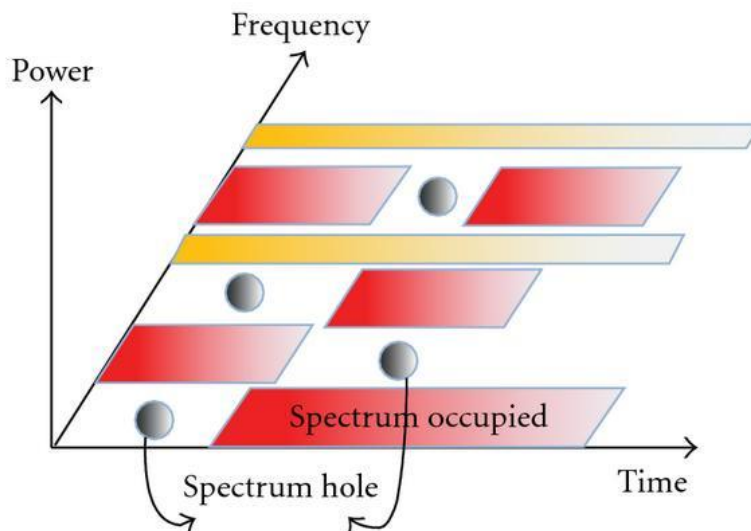
## 1.1 Εισαγωγή στα γνωστικά ασύρματα ραδιο-δίκτυα

Γνωστικά ασύρματα ραδιο-δίκτυα (CRNs) ονομάζονται τα δίκτυα που χρησιμοποιούν τεχνολογία γνωστικής ραδιοεπικοινωνίας, η οποία επιτρέπει τη δυναμική πρόσβαση στο διαθέσιμο ηλεκτρομαγνητικό φάσμα. Ο όρος «cognitive radio» χρησιμοποιήθηκε πρώτη φορά από τον Joseph Mitola [7] ο οποίος περιέγραψε πώς το CR μπορεί να ενισχύσει την ευελιξία των προσωπικών ασύρματων υπηρεσιών μέσω μίας νέας γλώσσας που ονομάζεται γλώσσα αναπαράστασης της ραδιο-γνώσης (Radio Knowledge Representation Language - RKRL). Η τεχνική της δυναμικής πρόσβασης στο φάσμα αποτελεί μία υποσχόμενη λύση στο πρόβλημα της χαμηλής αξιοποίησης του φάσματος αλλά και της ανεπάρκειας του, που παρατηρείται σε πολλές χώρες. Η FCC (Federal Communication Commission) εκτίμησε ότι η χρήση του φάσματος κυμαίνεται από 15%-85% [4], ενώ σύμφωνα με την DARPA (Defense Advance Research Projects Agency) μόνο το 2% του φάσματος είναι σε χρήση οποιαδήποτε χρονική στιγμή στις Η.Π.Α. Είναι σαφές ότι η λύση στο παραπάνω πρόβλημα είναι η παρακολούθηση και την αξιοποίηση των φασματικών κενών (spectrum holes).

Τα CRNs παρέχουν αυτήν τη δυνατότητα σε μη εξουσιοδοτημένους χρήστες (δευτερεύοντες). Οι δευτερεύοντες χρήστες μπορούν να έχουν πρόσβαση στο φάσμα, αν αυτό δεν χρησιμοποιείται από τους πρωτεύοντες. Στα δίκτυα αυτά, οι δευτερεύοντες χρήστες προσπαθούν με διάφορες τεχνικές να ανιχνεύσουν την παρουσία ή μη του πρωτεύοντα χρήστη στο φάσμα, δηλαδή την ύπαρξη των φασματικών κενών. Αν αντιληφτούν ότι το φάσμα δεν χρησιμοποιείται από τον εξουσιοδοτημένο χρήστη, το χρησιμοποιούν αυτοί. Έπειτα όταν ανιχνευθεί ξανά η παρουσία του πρωτεύοντα χρήστη σε αυτό, ο δευτερεύων το εγκαταλείπει για να μην προκαλέσει παρεμβολές στον πρωτεύοντα. Τηλεοπτικοί σταθμοί, χρήστες της δημόσιας ασφάλειας, εταιρείες κινητής τηλεφωνίας και point-to-point συνδέσεις μικροκυμάτων είναι μερικά παραδείγματα των εξουσιοδοτημένων χρηστών [5]. Τα πλεονεκτήματα των CRNs αναλύονται στο [6], όπου αναφέρονται τα γνωστικά ραδιο-δίκτυα ως το κλειδί για τη νέα γενιά τηλεπικοινωνιακών δικτύων γνωστή και ως DSA (Dynamain Spectrum Access), και ορίζονται ως τα ραδιο-δίκτυα που μπορούν να αλλάξουν τις παραμέτρους του πομπού βάσει της αλληλεπίδρασης με το περιβάλλον που λειτουργούν.

### 1.1.1 Φασματικά κενά

Όπως αναφέραμε και παραπάνω σε ένα CRN οι δευτερεύοντες χρήστες μπορούν έξυπνα να ανιχνεύουν ποιά κανάλια επικοινωνίας χρησιμοποιούνται σε διάφορες χρονικές στιγμές και ποιά όχι, ώστε να χρησιμοποιήσουν τα φασματικά κενά που δημιουργούνται στο χρόνο και στις συχνότητες (Σχήμα 1.1). Αυτή η μέθοδος μεγιστοποιεί τη χρήση του διαθέσιμου ραδιο-φάσματος ενώ ελαχιστοποιεί τις παρεμβολές σε άλλους χρήστες.



Σχήμα 1.1 Φασματικά κενά(spectrum holes) σε διάφορες συχνότητες

Πιο συγκεκριμένα, στο σχήμα 1.1 βλέπουμε ότι οι πρωτεύοντες χρήστες δεν μεταδίδουν όλες τις χρονικές στιγμές στα κανάλια τους, με αποτέλεσμα πολλές χρονικές στιγμές το κανάλι να μένει ανεκμετάλλευτο. Το κύριο χαρακτηριστικό λοιπόν των γνωστικών ραδιο-δικτύων είναι ότι οι δευτερεύοντες χρήστες μπορούν να αναληφθούν την ύπαρξη του πρωτεύοντα χρήστη στο κανάλι, το οποίο ο πρωτεύων έχει την άδεια να το χρησιμοποιεί, να στείλουν δεδομένα μέσω του καναλιού αυτού αν δεν είναι απασχολημένο προσαρμόζοντας τον τρόπο επικοινωνίας τους για να ελαχιστοποιήσουν τις παρεμβολές που προκαλούν στον πρωτεύοντα χρήστη. Δηλαδή να αφήσουν το κανάλι τη στιγμή που θα αντληθούν ξανά την παρουσία του πρωτεύοντα χρήστη.

Όπως είναι κατανοητό, δεν θα πρέπει να υποτιμάται η προσπάθεια που απαιτείται για τον εντοπισμό των φασματικών τρυπών και την επακόλουθη εκμετάλλευσή τους στη διαχείριση του φάσματος. Από πρακτική άποψη, το έργο της διαχείρισης φάσματος πρέπει να παραμένει ανεπηρέαστο όχι μόνο από τα είδη διαμόρφωσης των πρωτευόντων χρηστών, αλλά και από αρκετούς άλλους παράγοντες, με ένα αρκετά σημαντικό να είναι ο ακόλουθος :



- Προγνωστική ικανότητα για μελλοντική χρήση: ο εντοπισμός μίας φασματικής τρύπας σε μία συγκεκριμένη γεωγραφική θέση και μία συγκεκριμένη χρονική στιγμή θα ισχύει μόνο για αυτήν τη συγκεκριμένη στιγμή και όχι απαραίτητα και για το μέλλον. Έτσι, ο αλγόριθμος διαχείρισης φάσματος στον πομπό πρέπει να περιλαμβάνει δύο προβλέψεις:
  - Συνεχή παρακολούθηση της εν λόγω φασματικής τρύπας.
  - Εναλλακτική φασματική οδό για την αντιμετώπιση του ενδεχομένου να χρειαστεί ο πρωτεύων χρήστης το συγκεκριμένο φάσμα για δική του χρήση.

## 1.2 Λειτουργίες των γνωστικών ασύρματων ραδιο-δικτύων (CRNs)

Τα γνωστικά ραδιο-δίκτυα έχουν μία σειρά από νέες λειτουργίες όπως: αίσθηση του φάσματος (spectrum sensing), απόφαση για το φάσμα (spectrum decision), μοίρασμα του φάσματος (spectrum sharing) και τέλος κινητικότητα του φάσματος (spectrum mobility). Η αίσθηση του φάσματος αναφέρεται στη διαδικασία κατά την οποία οι SUs επιτηρούν τις διαθέσιμες ζώνες φάσματος για να ανιχνεύσουν πιθανές τρύπες φάσματος. Όταν η παραπάνω διαδικασία ολοκληρωθεί, ακολουθεί η απόφαση για το φάσμα (spectrum decision). Οι SUs επιλέγουν την κατάλληλη μπάντα βασιζόμενοι στη δραστηριότητα των PUs, προκειμένου να αξιοποιήσουν το φάσμα όσο το δυνατόν περισσότερο. Το μοίρασμα του φάσματος (spectrum sharing) παρέχει τη δυνατότητα να μοιράζονται τους πόρους του φάσματος ευκαιριακά πολλαπλοί SUs με PUs [10]. Έτσι οι μηχανισμοί για το μοίρασμα του φάσματος είναι κρίσιμοι για να επιτευχτεί υψηλή απόδοση. Δεδομένου ότι υπάρχουν πολλοί SUs που θέλουν να εκμεταλλευτούν τις ίδιες μπάντες του φάσματος, το δίκτυο πρέπει να κατανέμει κατάλληλα τους SUs στις διαθέσιμες μπάντες για την αποφυγή συγκρούσεων. Εφόσον οι SUs είναι μη εξουσιοδοτημένοι χρήστες, πρέπει να εγκαταλείψουν το κανάλι-μπάντα μόλις ανιχνεύσουν την ύπαρξη του PU σε αυτό. Η μετάδοση ενός τέτοιου SU μπορεί να συνεχιστεί σε άλλη μπάντα συχνοτήτων αν αυτή διαγνωστεί ως ελεύθερη εκείνη τη χρονική στιγμή. Έτσι η κινητικότητα του φάσματος απαιτεί ένα handoff σύστημα μεταβιβάσεως φάσματος, το οποίο θα ανιχνεύσει τη διακοπή σύνδεσης και θα αλλάζει την τρέχουσα μετάδοση σε μία νέα ζώνη συχνοτήτων, με ελάχιστη υποβάθμιση της ποιότητας [10].

### 1.2.1 Αίσθηση του φάσματος (spectrum sensing) στα γνωστικά ασύρματα ραδιο-δίκτυα

Σε αυτό το εδάφιο παρουσιάζονται οι λειτουργίες της ανίχνευσης φάσματος, όπως η ανίχνευση του PU, και η συνεργασία μεταξύ των SUs. Σε περίπτωση συνεργασίας, η παρατηρούμενη πληροφορία από κάθε SU ανταλλάσσεται με τους γείτονές του, έτσι ώστε να βελτιωθεί η ακρίβεια ανίχνευσης. Στην περίπτωση ανίχνευσης PU, ο SU πρέπει να ανιχνεύσει την παρουσία ή όχι του PU, προκειμένου να προσδιορίσει το διαθέσιμο φάσμα. Οι πιο σημαντικές παράμετροι που επηρεάζουν την απόδοση της ανίχνευσης φάσματος είναι ο διαθέσιμος χρόνος για την αίσθηση των καναλιών και η ισχύς των σημάτων των πρωτευόντων χρηστών [9]. Δύο παράμετροι που χρησιμοποιούνται γενικά για τη μέτρηση της απόδοσης της αίσθησης του φάσματος, είναι η πιθανότητα λανθασμένης ανίχνευσης (η πιθανότητα λανθασμένης ανίχνευσης δραστηριότητας του πρωτεύοντα χρήστη όταν αυτός είναι ανενεργός) και η πιθανότητα λανθασμένου συναγερμού (όταν ανιχνεύεται ότι ο πρωτεύων χρήστης είναι ενεργός, ενώ στην πραγματικότητα δεν είναι).

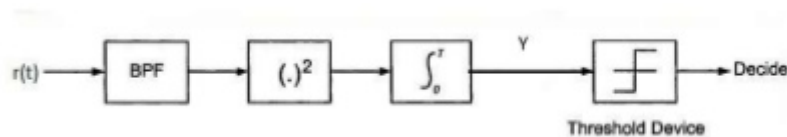
Ο χρόνος αίσθησης των καναλιών επηρεάζει σημαντικά τη σωστή ανίχνευση της χρησιμοποίησης η μη του καναλιού από κάποιο χρήστη. Δηλαδή μπορούμε να πούμε ότι με μεγάλο χρόνο αίσθησης καναλιού μειώνονται οι πιθανότητες λανθασμένου συναγερμού και λανθασμένης ανίχνευσης. Ωστόσο είναι σαφές ότι όσο μεγαλύτερος είναι αυτός ο χρόνος τόσο μικρότερος γίνεται ο χρόνος που απομένει για τη μετάδοση των δεδομένων. Παρακάτω αναλύονται οι τρεις πιο ευρέως γνωστές τεχνικές ανίχνευσης των φασματικών κενών.

- **Χρήση Προσαρμοσμένου φίλτρου (Matched-Filter)**

Το προσαρμοσμένου φίλτρο είναι το γραμμικό βέλτιστο φίλτρο που χρησιμοποιείται για την ανίχνευση σήματος μεγιστοποιώντας το λόγο σήματος-προς-θόρυβο (SNR) παρουσία του προσθετικού στοχαστικού θορύβου. Λαμβάνεται συσχετίζοντας ένα γνωστό αρχικό σήμα του PU  $s(t)$  με ένα λαμβανόμενο σήμα  $r(t)$ . Στη συνέχεια, η έξοδος του προσαρμοσμένου φίλτρου δειγματοληπτείται σε συγχρονισμένο χρόνο. Δεδομένου ότι ο δέκτης έχει a priori γνώση του εκπεμπόμενου σήματος του πρωτεύοντα χρήστη, η σύμφωνη ανίχνευση εξασφαλίζει ότι μόνο  $O(\frac{1}{SNR})$  δείγματα χρειάζονται για αποτελεσματική ανίχνευση. Αν η τιμή του δείγματος είναι μεγαλύτερη από ένα κατώφλι  $\lambda$ , το φάσμα αποφασίζεται ότι είναι κατειλημμένο για τη μετάδοση του PU. Το προσαρμοσμένο φίλτρο απαιτεί όχι μόνο εκ των προτέρων γνώση των χαρακτηριστικών του σήματος του PU αλλά επίσης το συγχρονισμό μεταξύ του πομπού του PU και του SU. Εάν αυτές οι πληροφορίες δεν είναι ακριβείς, τότε το προσαρμοσμένο φίλτρο έχει χαμηλή επίδοση. Επιπλέον, σε περίπτωση όπου υπάρχουν πολλοί πρωτεύοντες χρήστες ο SU θα πρέπει να έχει ένα αφιερωμένο προσαρμοσμένο φίλτρο για κάθε ένα από αυτούς, πράγμα το οποίο αυξάνει το κόστος και την πολυπλοκότητα υλοποίησης

- **Ανίχνευση Ενέργειας (Energy Detection)**

Για να αντιμετωπίσουμε τα παραπάνω προβλήματα, μη σύμφωνη ανίχνευση μπορεί να γίνει μέσω ανιχνευτή ενέργειας. Ο ανιχνευτής ενέργειας είναι βέλτιστος για την ανίχνευση του αγνώστου σήματος, εάν η ισχύς του θορύβου είναι γνωστή.



Σχήμα 1.2 Block diagram ανιχνευτή ενεργείας

Όπως φαίνεται από το σχήμα 1.2 παραπάνω, το μετρούμενο σήμα  $r(t)$  υψώνεται στο τετράγωνο και ολοκληρώνεται στο διάστημα παρατήρησης  $T$ . Η έξοδος του ολοκληρωτή συγκρίνεται με ένα κατώφλι  $\lambda$  για να αποφασιστεί εάν ο PU είναι παρών ή όχι. Ενώ ο ανιχνευτής ενεργείας είναι εύκολος, στην υλοποίηση έχει κάποια μειονεκτήματα. Απαιτεί  $O(\frac{1}{SNR^2})$  δείγματα για μία δεδομένη πιθανότητα ανίχνευσης [10]. Επίσης, αν ένας SU χρειάζεται να ανιχνεύσει αδύναμα σήματα PU ( $SNR$ : -10 dB έως -40 dB), η ανίχνευση ενέργειας πάσχει γιατί χρειάζεται μεγαλύτερο χρονικό διάστημα σε σχέση από της ανίχνευσης με τη χρήση προσαρμοσμένου φίλτρου. Επιπλέον, δεδομένου ότι η ανίχνευση της ενέργειας εξαρτάται μόνο από την  $SNR$  του λαμβανόμενου σήματος, όταν είναι άγνωστη η ισχύς του θορύβου η απόδοσή του δεν είναι η κατάλληλη. Επιπλέον, ενώ ο ανιχνευτής ενέργειας μπορεί να καθορίσει μόνο την παρουσία του σήματος δεν μπορεί να διακρίνει τα είδη σήματος. Έτσι οδηγείται σε λανθασμένες ανιχνεύσεις. Για τους λόγους αυτούς, για να χρησιμοποιείται η ανίχνευση της ενέργειας το γνωστικό ραδιο-δίκτυο (CRN) πρέπει να παρέχει το συγχρονισμό όλων χρηστών, δηλαδή, κάθε SU πρέπει να συγχρονιστεί στα ίδια χρονοσχήματα ανίχνευσης και μετάδοσης. Διαφορετικά, ο SU δεν μπορεί να διακρίνει τα λαμβανόμενα σήματα από τους PU, και ως εκ τούτου οι πράξεις ανίχνευσης του παρεμποδίζεται από τις μεταδόσεις των γειτονικών SU.

- **Ανίχνευση Κυκλοστασιμότητας (Cyclostationary Detection)**

Ψηφιακά διαμορφωμένα σήματα, όπως BPSK, QPSK, AM, OFDM, συνήθως παρουσιάζουν κυκλοστασιμότητα δηλαδή οι στατιστικές τους παράμετροι μεταβάλλονται περιοδικά με το χρόνο, το οποίο προσπαθεί να εκμεταλλευτεί η ανίχνευση κυκλοστασιμότητας. Εφόσον διαφορετικού τύπου σήματα έχουν διαφορετικές μη μηδενικές κυκλικές συχνότητες, μπορούν να αναγνωριστούν από την υπογραφή τους. Ωστόσο, τα σήματα θορύβου (συνήθως λευκός θόρυβος) δεν έχουν κυκλοστάσιμα χαρακτηριστικά. Το κύριο πλεονέκτημα της συγκεκριμένης μεθόδου είναι η απόδοση της όταν η ισχύς του θορύβου είναι αβέβαιη. Αυτή η μέθοδος επιτρέπει στον SU να εκτελεί λειτουργίες ανίχνευσης, ανεξάρτητα από εκείνες των γειτόνων του, δίχως να απαιτείται συγχρονισμός. Παρά το γεγονός ότι η κυκλοστάσιμη ανίχνευση είναι πολύ αποδοτική στα γνωστικά ραδιο-δίκτυα (CRN), είναι υπολογιστικά περιπλοκή και χρειάζεται υψηλό ρυθμό δειγματοληψίας.

### 1.2.2 Απόφαση για το φάσμα (spectrum decision) στα γνωστικά ασύρματα ραδιο-δίκτυα

Η απόφαση για το φάσμα είναι στενά συνδεδεμένη με τα χαρακτηριστικά του καναλιού και τις ενέργειες του PU. Συνήθως αποτελείται από δύο βήματα: πρώτον, κάθε φασματική ζώνη χαρακτηρίζεται με βάση όχι μόνο τις τοπικές παρατηρήσεις ενός SU αλλά και από τα στατιστικά στοιχεία των πρωτευόντων χρηστών. Στη συνέχεια, με βάση αυτό το χαρακτηρισμό, η πιο κατάλληλη ζώνη του φάσματος μπορεί να επιλεγεί. Οι ακόλουθες είναι οι κύριες λειτουργίες που απαιτούνται για την απόφαση του φάσματος:

- **Χαρακτηρισμός του φάσματος:** Με βάση την παρατήρηση, ο SU καθορίζει όχι μόνο τα χαρακτηριστικά του κάθε διαθέσιμου φάσματος, αλλά και το μοντέλο της δραστηριότητας του αντίστοιχου PU. Στενά συνδεδεμένα με το χαρακτηρισμό του φάσματος είναι το περιβάλλον και η δραστηριότητα του πρωτεύοντα χρήστη, τα οποία εξηγούνται παρακάτω.
  1. **Περιβάλλον:** Εφόσον οι διαθέσιμες τρύπες του φάσματος παρουσιάζουν διαφορετικά χαρακτηριστικά, τα οποία μεταβάλλονται με την πάροδο του χρόνου, κάθε τρύπα του ραδιοφάσματος θα πρέπει να χαρακτηρίζεται από την εξέταση τόσο του χρονικά μεταβαλλόμενου περιβάλλοντος όσο και από τις παραμέτρους του φάσματος, όπως η συχνότητα λειτουργίας και το εύρος ζώνης.
  2. **Βασική δραστηριότητα του χρήστη:** Χρειαζόμαστε ένα μέτρο για να μετράμε τη στατιστική συμπεριφορά των πρωτευόντων χρηστών, που ονομάζεται δραστηριότητα του πρωτεύοντα χρήστη. Δεδομένου ότι δεν υπάρχει καμία εγγύηση ότι μία ζώνη ραδιοφάσματος θα είναι διαθέσιμη κατά τη διάρκεια ολόκληρης της μετάδοσης του SU, η εκτίμηση της δραστηριότητας PU είναι ένα πολύ κρίσιμο ζήτημα στην απόφαση του φάσματος. Σε πολλές περιπτώσεις, τίθεται ότι η δραστηριότητα των PUs ακολουθεί χρονικά εκθετική κατανομή.
  3. **Επιλογή φάσματος:** Ο SU βρίσκει την καλύτερη ζώνη συχνοτήτων για την καθορισμένη από άκρο σε άκρο σύνδεση έτσι ώστε να ικανοποιούνται από άκρο σε άκρο οι Quality of Service (QoS) απαιτήσεις. Με βάση τις QoS απαιτήσεις των χρηστών και τα χαρακτηριστικά του φάσματος, ο ρυθμός μετάδοσης των δεδομένων, το αποδεκτό ποσοστό σφάλματος, το όριο καθυστέρησης και το εύρος ζώνης της μετάδοσης μπορούν να προσδιοριστούν. Στη συνέχεια, σύμφωνα με τον κανόνα επιλογής φάσματος, το σύνολο των κατάλληλων ζωνών του φάσματος μπορεί να επιλεγεί.

### 1.2.3 Μοίρασμα του φάσματος (spectrum sharing) στα γνωστικά ασύρματα ραδιο-δίκτυα

Ως μοίρασμα του φάσματος ορίζουμε την κοινή χρήση του φάσματος από πολλούς χρήστες. Επιτρέπεται στους δευτερεύοντες χρήστες να μοιράζονται τις μπάντες του φάσματος των εξουσιοδοτημένων χρηστών (PUs). Ωστόσο, οι SUs πρέπει να περιστελλούν την ισχύ της μετάδοσης τους έτσι ώστε οι παρεμβολές που προκαλούνται στους PUs να είναι κάτω από ένα συγκεκριμένο όριο. Παρακάτω αναλύονται οι κύριες λειτουργίες που απαιτούνται για την κατανομή του φάσματος.

- **Η κατανομή των πόρων:** Βασιζόμενοι στο QoS, οι SUs επιλέγουν τα κατάλληλα κανάλια και ρυθμίζουν την ισχύ εκπομπής τους, έτσι ώστε να επιτύχουν τις QoS απαιτήσεις τους, το δίκαιο μοίρασμα των πόρων αλλά και για να αποφύγουν τη δημιουργία παρεμβολής στον PU.
- **Πρόσβαση στο φάσμα:** Επιτρέπει σε πολλαπλούς SUs να μοιράζονται τους πόρους του ραδιοφάσματος, με προσδιορισμό του ποιος θα έχει πρόσβαση στο κανάλι ή τότε ένας χρήστης μπορεί να έχει πρόσβαση στο κανάλι. Αυτό επιτυγχάνεται συνήθως μέσω μίας τυχαίου μέθοδος πρόσβασης λόγω της δυσκολίας συγχρονισμού των SU.

### 1.2.4 Κινητικότητα του φάσματος (spectrum mobility) στα γνωστικά ασύρματα ραδιο-δίκτυα

Οι δευτερεύοντες χρήστες γενικά θεωρούνται «επισκέπτες» στο φάσμα. Ως εκ τούτου, εάν το συγκεκριμένο τμήμα του φάσματος που χρησιμοποιείται κάποια στιγμή από τον SU απαιτηθεί να χρησιμοποιηθεί ξανά από τον πρωτεύοντα χρήστη, η επικοινωνία του SU πρέπει να συνεχιστεί σε άλλο κενό τμήμα του φάσματος. Αυτή η έννοια ονομάζεται κινητικότητα φάσματος. Η κινητικότητα του φάσματος δημιουργεί τη λεγόμενη μεταπομπή φάσματος (spectrum handoff) στα γνωστικά ασύρματα ραδιο-δίκτυα, κατά την οποία οι δευτερεύοντες χρήστες μεταφέρουν τη σύνδεση τους σε κάποιο άλλο φασματικό κενό. Παρακάτω αναλύονται οι κύριες λειτουργίες του spectrum mobility [11].

- **Spectrum Handoff:** είναι η διαδικασία μεταφοράς μίας υφιστάμενης σύνδεσης για μετάδοση δεδομένων από το τρέχον κανάλι σε άλλο ελεύθερο κανάλι. Αυτό φυσικά προκαλεί πρόσθετα προβλήματα στην επικοινωνία του SU και τελικά επηρεάζει την απόδοση του συστήματος. Μεταπομπή φάσματος μπορεί να εφαρμοστεί βασιζόμενη σε διαφορετικές στρατηγικές:

1. **Non-Handoff Strategy:** Σε αυτήν την περίπτωση ο SU συνεχίζει να παραμένει στο αρχικό κανάλι και είναι αδρανής μέχρι το κανάλι να καταστεί και πάλι ελεύθερο. Μόλις ο PU αφήσει και πάλι το κανάλι, ο SU συνεχίζει ξανά τη μετάδοση δεδομένων. Αυτή η τεχνική προκαλεί υψηλή καθυστέρηση στους SU επειδή περιμένουν μέχρι να ελευθερωθεί και πάλι το κανάλι από τον PU. Σε συστήματα ευαίσθητα στην καθυστέρηση, αυτή η μέθοδος θα αποτύγχανε να εκπληρώσει τις QoS απαιτήσεις του συστήματος.
  2. **Pure Reactive Handoff Strategy:** Με αυτήν τη μέθοδο ισχύει αντιδραστική ανίχνευση φάσματος και αντιδραστική μεταπομπή φάσματος. Μόλις συμβεί ένα handoff γεγονός, ο SU ξεκινάει ανίχνευση φάσματος ώστε να εντοπίσει νέο ελεύθερο κανάλι. Στη συνέχεια, η επικοινωνία του μεταφέρεται σε αυτό το κανάλι. Η μέθοδος αυτή απαιτεί άμεση αλλαγή του φάσματος χωρίς κανένα χρόνο προετοιμασίας με αποτέλεσμα τη σημαντική υποβάθμιση της ποιότητας της σύνδεσης.
  3. **Pure Proactive Handoff Strategy:** Σε αυτήν τη μέθοδο ο SU εκτελεί ανίχνευση φάσματος για να βρει ένα εφεδρικό κανάλι πριν συμβεί η διακοπή της μετάδοσης του από τον PU. Με λίγα λόγια ο SU προβλέπει τη μελλοντική δραστηριότητα του PU στο κανάλι που εξυπηρετείται και μεταπηδά στο νέο φάσμα λίγο πριν συμβεί η διακοπή, εφόσον πιο πριν έχει εκτελέσει την ανίχνευση του φάσματος. Αυτή η μέθοδος έχει αρκετά πλεονεκτήματα όπως για παράδειγμα τη μικρή καθυστέρηση που θα επιβαρύνει το δευτερεύοντα χρήστη αφού αυτός έχει πραγματοποιήσει την αναζήτηση από πριν. Τα μειονεκτήματα της είναι ότι μπορεί ένα κανάλι να εκτιμήθηκε ως άδαιο αλλά τη στιγμή που θα πρέπει να γίνει η μεταπομπή να είναι κατειλημμένο. Επίσης δεν είναι εύκολο ο SU να έχει ακριβές μοντέλο της κίνησης του κάθε PU.
  4. **Hybrid Handoff Strategy:** Σε αυτήν τη μέθοδο συνδυάζονται οι proactive και reactive μέθοδοι. Η αναζήτηση νέου φάσματος γίνεται από το δευτερεύοντα χρήστη πριν τη διακοπή της υπάρχουσας σύνδεσης, αλλά η μεταπομπή στο νέο κανάλι γίνεται μετά τη διακοπή της σύνδεσης. Μπορεί να επιτευχθεί ταχύτερος χρόνος μεταπομπής, αφού η ανίχνευση έχει γίνει κατά τη διάρκεια μετάδοσης των δεδομένων στο προηγούμενο κανάλι.
- **Διαχείριση της σύνδεσης (Connection management):** Ο SU διατηρεί την ποιότητα των υπηρεσιών ή ελαχιστοποιεί την υποβάθμιση της ποιότητας αυτής κατά τη διάρκεια της αλλαγής φάσματος, με την αλληλεπίδραση με κάθε επίπεδο του πρωτοκόλλου. Όταν η τρέχουσα συχνότητα λειτουργίας καθίσταται απασχολημένη στο μέσο μίας επικοινωνίας κάποιου SU, οι εφαρμογές που εκτελούνται σε αυτόν τον κόμβο πρέπει να μεταφέρουν την επικοινωνία σε άλλη διαθέσιμη ζώνη συχνοτήτων. Ωστόσο, η επιλογή των

νέων συχνοτήτων μπορεί να πάρει χρόνο. Μία σημαντική απαίτηση των πρωτοκόλλων διαχείρισης σύνδεσης είναι οι πληροφορίες καθυστέρησης σχετικά με τη διάρκεια της μεταπομπής φάσματος. Μόλις οι πληροφορίες καθυστέρησης είναι διαθέσιμες, ο SU μπορεί να προβλέψει την επίδραση της προσωρινής αποσύνδεσης σε κάθε στρώμα πρωτοκόλλου, και κατά συνέπεια να διατηρήσει την εν εξελίξει επικοινωνία με την ελάχιστη υποβάθμιση των επιδόσεων.

### 1.3 Τρόποι λειτουργίας των γνωστικών ασύρματων ραδιο-δικτύων ( CRN )

Τα CRNs μπορούν να λειτουργήσουν με διάφορες τεχνικές έτσι ώστε να επιτύχουν το επιθυμητό αποτέλεσμα. Υπάρχουν τριών ειδών τρόποι λειτουργίας που αναλύονται παρακάτω.

- **Δίχως συνεργασία των SUs :** Κάθε δευτερεύων χρήστης δεν έχει καμία επικοινωνία με οποιονδήποτε άλλο δευτερεύοντα χρήστη. Η ανίχνευση του φάσματος και κατά συνέπεια των φασματικών κενών σε αυτό και η επιλογή του καναλιού γίνεται ξεχωριστά από κάθε SU, δίχως ο ένας να επηρεάζει τον άλλο. Αν ο χρήστης εκτιμήσει ότι υπάρχει φασματικό κενό σε κάποιο κανάλι, θα προσπαθήσει να μεταδώσει σε αυτό. Εδώ υπάρχει το πρόβλημα ότι μπορεί παραπάνω από ένας χρήστες να προσπαθήσουν την ίδια χρονική στιγμή να μεταδώσουν μέσω του ίδιου καναλιού, μιας και έχουν ανεξάρτητα εκτιμήσει το κανάλι ως άδειο (δεν επικοινωνούν μεταξύ τους, ώστε να το χρησιμοποιήσει μόνο ένας από αυτούς). Αν κάποιος δευτερεύων χρήστης λάβει με μικρή ισχύ το σήμα του πρωτεύοντα χρήστη, τότε είναι πολύ πιθανό να εκτιμήσει εσφαλμένα το κανάλι ως μη-απασχολημένο και να προσπαθήσει να μεταδώσει σε αυτό.
- **Με συνεργασία μεταξύ των SUs:** Σε αυτήν την περίπτωση οι δευτερεύοντες χρήστες επικοινωνούν μεταξύ τους με στόχο την καλύτερη αξιοποίηση του διαθέσιμου φάσματος. Στις περισσότερες περιπτώσεις, [12]-[15], δημιουργούνται clusters από τους SUs όπου κάθε cluster έχει ένα cluster head (CH) που ορίζεται συνήθως ως ο χρήστης με το μεγαλύτερο αναφερόμενο SNR για το κανάλι. Έπειτα όλοι οι SU παίρνουν τις δικές τους μετρήσεις-εκτιμήσεις για την κατάσταση του καναλιού και στη συνέχεια τις αναφέρουν στο CH. Οι μετρήσεις αυτές έχουν τη μορφή 1 ή 0. Το 1 αντιστοιχεί στην περίπτωση όπου το κανάλι είναι κατειλημμένο και το 0 στην περίπτωση όπου το κανάλι εκτιμηθεί ως μη απασχολημένο. Έπειτα ο CH εκτελώντας συνήθως την εντολή OR με είσοδο όλες την εκτιμήσεις που του σταλήκαν από τους SUs, παίρνει την τελική εκτίμηση για το κανάλι. Στη συνέχεια, γίνεται από τον CH ή από κάποιον κεντρικό συντονιστή η ανάθεση του καναλιού (αν αυτό είναι κενό) σε κάποιον από τους SUs. Με τη συνεργασία μεταξύ των SUs επιτυγχάνεται χαμηλότερη πιθανότητα εσφαλμένης εκτίμησης της



κατάστασης του καναλιού, πράγμα κρίσιμο για τη συνολική απόδοση του συστήματος.

- **Με συνεργασία μεταξύ SUs και PUs:** Με αυτό τον τρόπο ο SU μπορεί να συνεργάζεται με το PU, ώστε η πιθανότητα επιτυχημένης μετάδοσης του PU να αυξάνεται, ενώ ο SU αποκτά περισσότερες ευκαιρίες μετάδοσης. Στην πραγματικότητα, η συνεργασία SU στοχεύει ακριβώς στην αύξηση των επιταχυμένων μεταδόσεων του PU (καθώς έτσι αδειάζει η ουρά των πακέτων του PU με ταχύτερο ρυθμό) μέσω επαναμετάδοσης πακέτων του PU από τον SU (relaying), όταν η αρχική μετάδοση τους από τον PU αποτύχει και το κανάλι μεταξύ του SU και του πρωτεύοντα δέκτη είναι καλύτερο από αυτό μεταξύ PU και του αντιστοίχου δέκτη. Ως εκ τούτου, η συνεργασία αυτή με τον SU έχει τη δυνατότητα, εφόσον ο SU μπορεί να εγγυηθεί τη σταθερότητα της, να αυξήσει τον ρυθμό μετάδοσης της κίνησης του PU. Ο SU που θα βοηθήσει έναν PU να μεταδώσει τα πακέτα του, αποκτά αμέσως πλεονέκτημα στο να χρησιμοποιήσει αυτό το κανάλι όταν ο PU δεν είναι ενεργός. Οι SUs πρέπει να λαμβάνουν ευφυείς αποφάσεις για το αν θα συνεργαστούν ή όχι και με ποιόν PU, προκειμένου να μεγιστοποιούν την απόδοση τους.

## 1.4 Περιγραφή διαγράμματος διπλωματικής εργασίας

Το υπόλοιπο αυτής της εργασίας διαμορφώνεται ως εξής:

Στο κεφάλαιο 2 παρουσιάζεται το πρώτο μέρος της διπλωματικής εργασίας, το οποίο βασίζεται στην υπόθεση ότι υπάρχουν στο σύστημα πολλά ομογενή κανάλια. Δηλαδή υποθέτουμε ότι όλα τα κανάλια έχουν την ίδια χωρητικότητα για όλους τους SU. Στην ενότητα 2.2 γίνεται η περιγραφή του συστήματος, των αλγορίθμων που προτείνουμε αλλά και μερικών αποδοτικών αλγορίθμων που έχουν αναλυθεί στην βιβλιογραφία. Τέλος, στην ενότητα 2.3, παραθέτονται τα σχήματα από τις προσομοιώσεις που έγιναν, συγκρίνονται και αναλύονται τα τελικά συμπεράσματα στα διάφορα σενάρια που μελετήθηκαν.

Στο κεφάλαιο 3 παρουσιάζεται το δεύτερο μέρος της διπλωματικής εργασίας, το οποίο βασίζεται στην υπόθεση ότι οι χρήστες είναι ομοιόμορφα διασκορπισμένοι στο χώρο σε σύστημα με ετερογενή κανάλια. Στην ενότητα 3.2 γίνεται η περιγραφή της τοπολογίας του συστήματος. Στην ενότητα 3.3 αναλύεται το πρόβλημα και οι προτεινόμενοι αλγόριθμοι. Τέλος, στην ενότητα 3.4 παραθέτονται τα σχήματα από τις προσομοιώσεις που έγιναν, συγκρίνονται και αναλύονται τα τελικά συμπεράσματα στα διάφορα σενάρια που μελετήθηκαν.



Το Κεφάλαιο 4 παρουσιάζει τον επίλογο της διπλωματικής εργασίας. Στην ενότητα 4.1 παρουσιάζεται η συνεισφορά της διπλωματικής εργασίας και στην ενότητα 4.2 δίνονται μερικές ιδέες για μελλοντικές επεκτάσεις.

## ΚΕΦΑΛΑΙΟ 2<sup>ο</sup>: Σχεδιασμός και αξιολόγηση αποδόσεων των προτεινόμενων αλγορίθμων για αίσθηση καναλιών σε CRNs με ομογενή κανάλια

### 2.1 Εισαγωγή

Η πιο σημαντική πρόκληση για έναν δευτερεύοντα χρήστη είναι να αποφασίσει ποιό κανάλι (ή κανάλια) θα ανιχνεύσει και θα χρησιμοποιήσει, και πως αυτά θα ανιχνεύονται και θα χρησιμοποιούνται. Σε αυτό το κεφάλαιο μελετάται το πρόβλημα της διαδοχικής ανίχνευσης καναλιών σε Cognitive Radio Network με πολλά ομογενή κανάλια. Πιο συγκεκριμένα, θεωρούμε ένα CRN με πολλαπλά κανάλια, όλα με την ίδια χωρητικότητα, εισάγουμε νέους αλγορίθμους αίσθησης καναλιών και μετάδοσης σε αυτά, και αξιολογούμε την απόδοσή τους μέσω προσομοιώσεων.

Ο PU αγνοεί την παρουσία των SUs στο κανάλι στο οποίο είναι εξουσιοδοτημένος χρήστης και μεταδίδει τα δεδομένα του κάθε φορά που έχει τέτοια προς μετάδοση. Χρησιμοποιούμε την τοπολογία του δικτύου που προτείνεται στο [2]. Η δομή μίας χρονοθυρίδας καναλιού (slot) αποτελείται από πολλές μίνι-χρονοθυρίδες (mini-slots) που αντιστοιχούν σε διαφορετικές φάσεις της ανίχνευσης φάσματος και μετάδοσης πακέτων (σχήμα 2.1). Ως εκ τούτου, κατά τη διάρκεια μίας χρονοθυρίδας οι SU έχουν έναν σταθερό αριθμό ευκαιριών για να βρουν ένα άδειο κανάλι. Ο περιορισμένος αριθμός των πιθανών παρατηρήσεων και η δυναμική φύση των παρατηρούμενων σημάτων μπορεί να οδηγήσει σε ατελή αίσθηση του φάσματος που συνήθως περιγράφεται από τις πιθανότητες λανθασμένης ανίχνευσης και λανθασμένου συναγερμού [1]. Κατά την αξιολόγηση της απόδοσης των προτεινόμενων αλγορίθμων, θεωρούμε την πιθανότητα λανθασμένου συναγερμού, όπου ένα άδειο κανάλι εντοπίζεται λανθασμένα ως απασχολημένο, με αποτέλεσμα η δυνατότητα μεταφοράς δεδομένων σε αυτό να μην αξιοποιείται από τον SU. Θεωρούμε επίσης την πιθανότητα λανθασμένης ανίχνευσης, η οποία προκαλεί συγκρούσεις μεταξύ των μεταδόσεων των πρωτεύοντων και δευτερευόντων χρηστών (ή μεταξύ των δευτερευόντων χρηστών), επειδή το απασχολημένο κανάλι εντοπίστηκε εσφαλμένα ως άδειο.

Εισάγουμε διάφορους αλγορίθμους για την εξερεύνηση του φάσματος του δικτύου, και πιο συγκεκριμένα της κατάστασης των καναλιών από τους δευτερεύοντες χρήστες (secondary users-SUs) σε γνωστικά ασύρματα ραδιο-δίκτυα (cognitive radios networks-CRNs) για την αξιοποίηση των φασματικών κενών που τυχόν παρουσιαστούν. Μελετάμε κυρίως την περίπτωση όπου οι δευτερεύοντες χρήστες δεν επικοινωνούν μεταξύ τους ή επικοινωνούν ελάχιστα. Θεωρούμε ότι το σύστημα που μελετάμε είναι time slotted, με συγχρονισμένους χρήστες. Δηλαδή οι πρωτεύοντες

χρήστες ξεκινούν τη μετάδοση του πακέτου τους (αν έχουν πακέτο προς μετάδοση) στην αρχή κάθε slot. Αντίστοιχα οι δευτερεύοντες χρήστες ξεκινούν την εκτίμηση της κατάστασης του καναλιού στην αρχή κάθε slot. Κάθε δευτερεύων χρήστης προσπαθεί να μεταδώσει σε κάθε slot σε κάποιο κανάλι που είναι μη-απασχολημένο από μετάδοση του πρωτεύοντα χρήστη που το χρησιμοποιεί. Σε κάθε slot θεωρούμε ότι ο PU μεταδίδει ένα πακέτο, αν έχει πακέτο προς μετάδοση. Ο δευτερεύων χρήστης μπορεί να μεταδώσει σε ένα κανάλι όταν είναι άδειο και το ανιχνεύσει επιτυχημένο ως άδειο, δηλαδή όταν δεν το απασχολεί μεταδίδοντας σε αυτό ο εξουσιοδοτημένος χρήστης. Τα αποτελέσματά μας δείχνουν ότι οι νέοι προτεινόμενοι αλγόριθμοι βελτιώνουν την κανονικοποιημένη μέση απόδοση (throughput) των SUs και αυξάνουν τον αριθμό των επιτυχημένων μεταδόσεων που λαμβάνουν χώρα κατά τη διάρκεια της λειτουργίας του συστήματος.

## 2.2 Περιγραφή συστήματος

Όπως στο [2], θεωρούμε ένα time slotted συγχρονισμένο σύστημα με  $N_s$  δευτερεύοντες χρήστες,  $N_p$  πρωτεύοντες χρήστες και  $N_c$  κανάλια, από ένα κανάλι για κάθε ένα πρωτεύοντα χρήστη. Οι δευτερεύοντες χρήστες ψάχνουν ευκαιρία μετάδοσης στα παραπάνω κανάλια. Σε κάθε χρήστη (PU ή SU) θεωρούμε ότι αντιστοιχεί ένας πομπός και ένας δέκτης. Ο πομπός προφανώς, μεταβιβάζει το πακέτο προς το δέκτη. Οι πομποί (transmitters) μεταδίδουν πακέτα στους αντίστοιχους δέκτες (receivers), και με τη σειρά τους οι δέκτες αυτοί, ενημερώνουν με ACK/ NACK τους αντίστοιχους πομπούς, αν τελικά έλαβαν ή όχι σωστά το κάθε πακέτο. Η κάθε χρονοθυρίδα έχει τη δομή που φαίνεται στο *σχήμα 2.1*. Αν ο SU δει ότι το κανάλι είναι άδειο τότε μεταδίδει σε αυτό. Σε κάθε slot γίνονται ακριβώς οι ίδιες ενέργειες. Επομένως κάθε χρήστης επιλέγει με τεχνικές που θα αναλυθούν παρακάτω ποιο κανάλι να αισθανθεί, καταλαβαίνει την κατάσταση του και έπειτα πράττει ανάλογα, μεταδίδοντας ή όχι στον εναπομείναντα χρόνο. Έχουμε θεωρήσει ότι κάθε χρήστης έχει τρεις (3) ευκαιρίες μετάδοσης στη διάρκεια ενός slot, όπου φυσικά κάθε μία διαρκεί μερικά ms μικρότερα από το συνολικό χρόνο που διαρκεί το slot. Αρχικά κάθε ένας από τους SUs επιλέγει ένα κανάλι στο οποίο θα προσπαθήσει να μεταδώσει στη πρώτη ευκαιρία μετάδοσης (1<sup>st</sup> minislot). Αν το εντοπίσει κατειλημμένο τότε ο SU, προχωρά στη δεύτερη ευκαιρία μετάδοσης (2<sup>nd</sup> minislot), όπου επιλέγει ένα διαφορετικό κανάλι, και αν αποτύχει και εκεί, δηλαδή αισθανθεί και αυτό το κανάλι ως απασχολημένο, προχωρά στην τρίτη ευκαιρία μετάδοσης (3<sup>rd</sup> minislot). Αν βρει και αυτό το κανάλι απασχολημένο, τότε δεν μεταδίδει σε αυτό το slot και ψάχνει για ευκαιρίες μετάδοσης από το επόμενο slot. Αν ο SU καταφέρει να μεταδώσει στην πρώτη ευκαιρία μέσα σε ένα slot, τότε μεταδίδει για χρόνο ίσο με  $t = T - t_s$ , όπου  $T$  η διάρκεια του slot και  $t_s$  ο χρόνος που διαρκεί η ανίχνευση της κατάστασης του καναλιού. Αν καταφέρει να μεταδώσει στη δεύτερη ευκαιρία τότε

μεταδίδει για χρόνο ίσο με  $t = T - t_s - t_{ch}$  όπου  $t_{ch}$  είναι ο χρόνος που χρειάζεται για να προετοιμάσει το κύκλωμα αίσθησης του για να μεταβεί στο νέο κανάλι. Επομένως γενικά ο τελικός χρόνος που μεταδίδει ένας χρήστης, ανάλογα με την ευκαιρία μετάδοσης στην οποία επιτυγχάνει είναι  $t = T - t_s - i \cdot (t_s + t_{ch})$ , όπου  $i = 0, 1, 2$ , συμβολίζει την ευκαιρία στην οποία μετάδωσε τελικά ο SU. Επίσης κάθε κανάλι έχει τη δίκια του χωρητικότητα (R) που συνήθως μετριέται σε Mbps, άρα αν κάποιος SU μεταδώσει για χρόνο t, μεταδίδει  $R \cdot t$  Mbits πάνω στο κανάλι.

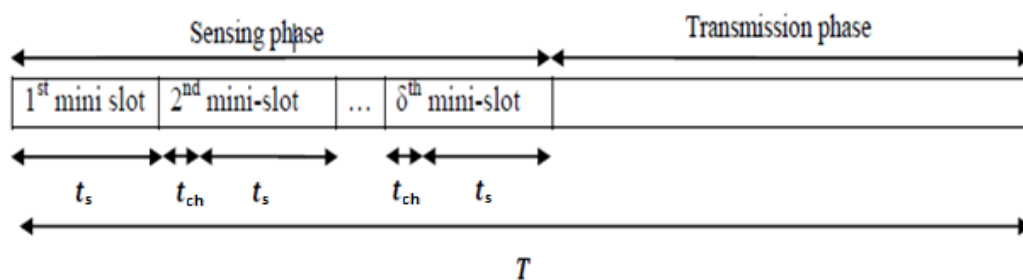
Επιπλέον, πρέπει να σημειωθεί όταν αν ένα κανάλι είναι άδειο, και κάποιος SU μεταδώσει, π.χ στην πρώτη ευκαιρία τότε αυτό αυτόματα καθίστατε κατειλημμένο. Αν λοιπόν κάποιος άλλος SU επιλέξει στη δεύτερη η τρίτη ευκαιρία μετάδοσης να αισθανθεί το συγκεκριμένο κανάλι, τότε θα δει ότι αυτό είναι κατειλημμένο και δεν θα μεταδώσει σε αυτό, εκτός αν γίνει λανθασμένη ανάγνωση της κατάστασης του καναλιού. Υπάρχουν δύο πιθανότητες που παίζουν σημαντικό ρόλο στη λειτουργία των αλγορίθμων που θα παρουσιαστούν και αναλυθούν παρακάτω. Αυτές είναι η πιθανότητα false alarm ( $P_{fa}$ ) και η πιθανότητα miss detection ( $P_{md}$ ). Η  $P_{fa}$  είναι η πιθανότητα να ανιχνευθεί ένα κανάλι από το δευτερεύοντα χρήστη ως κατειλημμένο ενώ δεν είναι. Έτσι ο δευτερεύων χρήστης δεν μεταδίδει σε αυτό το κανάλι, ενώ κανονικά θα μπορούσε. Η πιθανότητα  $P_{md}$  είναι η πιθανότητα να ανιχνευθεί ένα κανάλι ως μη κατειλημμένο ενώ είναι. Τότε ο δευτερεύων χρήστης μεταδίδει αλλά το πακέτο του θα συγκρουστεί σίγουρα με αυτό του χρήστη (PU ή SU) που μετέδιδε ήδη στο κανάλι και επομένως οι εμπλεκόμενοι χρήστες θα λάβουν NACK στο τέλος του slot, με αποτέλεσμα, αν η σύγκρουση είχε γίνει με πακέτο του PU, ο τελευταίος θα πρέπει να επαναμεταδώσει το ίδιο πακέτο στο επόμενο slot. Επιπλέον, αν δύο SUs επιλέξουν το ίδιο κανάλι στην ίδια ευκαιρία μετάδοσης και το ανιχνεύσουν άδειο, τότε μεταδίδουν και οι δύο τα πακέτα τους συγκρούονται και λαμβάνουν NACK στο τέλος του slot. Υποθέτουμε ότι ο PU έχει πακέτα προς μετάδοση, με ρυθμό άφιξης  $\lambda$ . Όσο μεγαλύτερο είναι το  $\lambda$  τόσο περισσότερα πακέτα θα μεταδίδει ο PU στο κανάλι που του αντιστοιχεί, άρα τόσο περισσότερο κατειλημμένο θα είναι αυτό το κανάλι, με αποτέλεσμα να μη μπορεί να μεταδώσει τόσο συχνά σε αυτό ένας SU, όσο σε ένα κανάλι με μικρότερο ρυθμό άφιξης πακέτων ( $\lambda$ ) του πρωτεύοντα χρήστη.

Αν συμβολίσουμε με  $\lambda_i$  τον ρυθμό άφιξης πακέτων του  $PU_i$  στο  $i$ -οστό κανάλι ( με δυνατές τιμές από το 0 έως το 1) και  $P_{fa}$ ,  $P_{md}$  τις πιθανότητες λανθασμένου συναγερμού και λανθασμένης ανίχνευσης αντίστοιχα, τότε το  $i$ -οστό κανάλι ανιχνεύεται ως ελεύθερο όταν : 1) ο PU δεν έχει πακέτο προς μετάδοση (γεγονός που συμβαίνει με πιθανότητα  $(1 - \lambda_i)$ ), ο δευτερεύων χρήστης ανιχνεύσει αυτήν την ευκαιρία (γεγονός που συμβαίνει με πιθανότητα  $(1 - P_{fa})$ ) και συνεπώς η συνολική πιθανότητα του γεγονότος αυτού είναι  $(1 - \lambda_i) \cdot (1 - P_{fa})$  ή 2) το  $i$ -οστό κανάλι είναι κατειλημμένο από τον πρωτεύοντα χρήστη (γεγονός που συμβαίνει με πιθανότητα  $\lambda_i$ ) αλλά ο δευτερεύων χρήστης λανθασμένα εκτιμήσει το κανάλι ως ελεύθερο (γεγονός που συμβαίνει με πιθανότητα  $P_{md}$  και συνεπώς η συνολική πιθανότητα του γεγονότος αυτού είναι  $\lambda_i \cdot P_{md}$ ). Αν δεν συμβεί κανένα από τα παραπάνω γεγονότα, η αίτηση για την ανίχνευση φάσματος μεταφέρεται στην επομένη μίνι-χρονοθυρίδα, όπως φαίνεται

το σχήμα 2.1, γεγονός που μπορεί να συμβεί με πιθανότητα  $P_{fa}^*(1-\lambda_i) + \lambda_i^*(1-P_{md})$ . Εάν έχουμε σύγκρουση μεταξύ PU και SU ή μεταξύ SUs τότε κανένα πακέτο δεν μεταδίδεται επιτυχημένα. Θεωρούμε ότι η μετάδοση ενός πακέτου (είτε από τον PU είτε από τον SU) διαρκεί από τη στιγμή που ξεκινάει τη μετάδοση ο χρήστης μέχρι το τέλος της χρονοθυρίδας. Δηλαδή θεωρούμε ότι τα πακέτα των PUs διαρκούν ακριβώς όσο μία χρονοθυρίδα (αφού η αποστολή των δεδομένων ξεκινά ακριβώς στην αρχή της), και ότι τα πακέτα των SUs μπορούν να υποδιαιρεθούν σε μικρότερα μεγέθη, ανάλογα με τη διάρκεια του χρόνου από την αρχή της μετάδοσης τους μέχρι το τέλος της χρονοθυρίδας.

Ο μέγιστος αριθμός των ανεκτών μεταπομπών (handoffs) καθορίζεται από δύο περιορισμούς. Πρώτον, ο αριθμός των καναλιών που ανιχνεύονται δεν μπορεί να υπερβαίνει τον αριθμό των PUs. Δεύτερον, ο χρόνος που καταναλώνεται για τις διαδικασίες τόσο της ανίχνευσης όσο και μετάδοσης δεν μπορεί να υπερβαίνει το χρονικό διάστημα διάρκειας μίας χρονοθυρίδας. Στην εργασία μας θεωρήσαμε, όπως αναφέρθηκε και παραπάνω, ότι μπορούν να συμβούν το πολύ τρεις μεταπομπές (τρεις ευκαιρίες μετάδοσης) κατά τη διάρκεια μίας χρονοθυρίδας.

Τέλος, αξίζει να σημειωθεί ότι αν ένας ή περισσότεροι SUs επιλέξουν το ίδιο διαθέσιμο κανάλι, πρέπει να λάβουμε υπόψη δύο διαφορετικές περιπτώσεις. Κατ'αρχάς, την περίπτωση όπου μόνο ένας SU ανιχνεύει το κανάλι ως άδειο και το πακέτο του μεταδίδεται επιτυχημένα κατά τη διάρκεια του υπολειπόμενου χρόνου της συγκεκριμένης χρονοθυρίδας. Δεύτερον, την περίπτωση όπου περισσότεροι από ένα SUs αισθανθούν το κανάλι ως άδειο με αποτέλεσμα να προκύψει σύγκρουση μεταξύ των μεταδόσεων τους και κανένα πακέτο SU να μη μεταδοθεί επιτυχημένα.



Σχήμα 2.1 : Η δομή μίας χρονοθυρίδας(slot)

### 2.2.1 Απλός αλγόριθμος τυχαίας επιλογής καναλιού (PPRA)

Ο συγκεκριμένος αλγόριθμος έχει προταθεί στην βιβλιογραφία με μία ευκαιρία μετάδοσης [2]. Για τις ανάγκες αυτής της εργασίας και για να μπορούμε να συγκρίνουμε τα αποτελέσματα του με τους αλγόριθμους που θα προτείνουμε και αναλύσουμε παρακάτω, τον υλοποιήσαμε με τρεις ευκαιρίες μετάδοσης σε κάθε slot. Ας θεωρήσουμε ότι υπάρχουν  $N_p$  κανάλια και συνεπώς  $N_p$  PUs. Όλα τα κανάλια είναι ισοπίθανα για να επιλέγουν από κάποιο δευτερεύοντα χρήστη με πιθανότητα  $1/N_p$ .

Κάθε δευτερεύων χρήστης αν έχει πακέτο προς μετάδοση, μεταδίδει ένα πακέτο ανά slot, όπου το μέγεθος του πακέτου αυτού ορίζεται από τη χωρητικότητα του καναλιού αλλά και από την ευκαιρία ( $1^n$  ή  $2^n$  ή  $3^n$ ) που κατάφερε να μεταδώσει ο δευτερεύων χρήστης. Για παράδειγμα αν η χωρητικότητα του καναλιού είναι  $C=20\text{Mbps}$ , η διάρκεια του slot  $T=10\text{ms}$ ,  $t_s=2.4\text{ms}$  και ο δευτερεύων χρήστης μεταδώσει στην πρώτη ευκαιρία, το πακέτο του θα είναι μεγέθους  $C \cdot (T - t_s) = 20 \cdot 7,6 \cdot 10^{-3} = 0,02 \cdot 7,6 = 0.152 \text{ Mbits}$ . Αν μεταδώσει στη δεύτερη ευκαιρία με  $t_{ch}=0.01 \text{ ms}$  τότε το πακέτο θα έχει μέγεθος ίσο με  $C \cdot (T - t_s - t_s - t_{ch}) = 20 \cdot (10 - 2,4 - 2.4 - 0.01) \cdot 10^{-3} = 0,02 \cdot 5,19 = 0.1038 \text{ Mbits}$ .

Στην αρχή κάθε slot οι SUs επιλέγουν τυχαία ένα κανάλι με πιθανότητα  $1/N_p$ . Όπως είπαμε, κάθε SU αν το βρει κατειλημμένο στην πρώτη ευκαιρία προχωράει στη δεύτερη ευκαιρία επιλέγοντας ξανά κανάλι. Το κανάλι που θα ανιχνευθεί ως άδειο χρησιμοποιείται για τη μετάδοση του πακέτου του. Σε κάθε ευκαιρία η επιλογή του καναλιού είναι τυχαία και κάθε φορά εξαιρείται το κανάλι (κανάλια) που έχει επιλέγει στις προηγούμενες ευκαιρίες, αν υπήρχαν. Στην αρχή κάθε slot, μπορούν να επιλεγούν όλα τα κανάλια με την ίδια πιθανότητα. Και εδώ προφανώς ισχύουν τα όσα αναφέραμε παραπάνω σχετικά με τις πιθανότητες  $P_{fa}$ ,  $P_{md}$ , αλλά και τις περιπτώσεις που μπορεί να προκύψει σύγκρουση του πακέτο με άλλου δευτερεύοντα ή πρωτεύοντα χρήστη.

### 2.2.2 Κατανεμημένος αλγόριθμος (Distributed Algorithm)

Ο αλγόριθμος αυτός έχει προταθεί στην βιβλιογραφία [1]. Κάθε χρήστης έχει τέσσερις ευκαιρίες μετάδοσης σε κάθε slot. Η λογική του αλγορίθμου αυτού είναι ότι κάθε χρήστης δημιουργεί ένα πίνακα (ο οποίος αναφέρεται ως Latin Square) τεσσάρων θέσεων σε κάθε γραμμή όπου η κάθε γραμμή περιέχει κάθε δυνατό συνδυασμό των καναλιών. Η σημαντική ιδιότητα αυτού του πίνακα είναι ότι κάθε κανάλι εμφανίζεται μία φορά σε κάθε γραμμή. Ο πίνακας αυτός είναι ίδιος για όλους τους χρήστες. Για παράδειγμα αν είχαμε 4 κανάλια ο Latin Square πίνακας θα ήταν:

1	2	3	4
2	3	4	1
3	4	1	2
4	1	2	3

Στην αρχή ο αλγόριθμος αυτός εκτιμάει τη χωρητικότητα του κάθε καναλιού. Η εκτίμηση αυτή γίνεται στο αρχικό διάστημα λειτουργίας του συστήματος. Στο δικό μας μοντέλο υποτίθεται ότι όλα τα κανάλια έχουν την ίδια χωρητικότητα. Οι δευτερεύοντες χρήστες δεν γνωρίζουν την κίνηση κάθε καναλιού, αλλά γνωρίζουν τη χωρητικότητα του και το πλήθος των συνολικών καναλιών του συστήματος. Η επιλογή της σειράς γίνεται με βάση την επιλογή του καναλιού

που είναι στην πρώτη θέση της σειράς. Έτσι ο χρήστης επιλέγοντας κάποιο κανάλι από αυτά τα τέσσερα, καταφέρνει να επιλέξει και την αντίστοιχη γραμμή του πίνακα. Αν π.χ επιλέξει το κανάλι 2 στην πρώτη ευκαιρία μετάδοσης θα αισθανθεί αρχικά το κανάλι 2, στη δεύτερη ευκαιρία το κανάλι 3, στην τρίτη ευκαιρία το κανάλι 4 και στην τέταρτη το κανάλι 1. Η πιθανότητα επιλογής καναλιού (άρα και αντίστοιχης λίστας από το πίνακα) είναι  $1/Np$ . Επομένως όλα τα κανάλια έχουν την ίδια πιθανότητα επιλογής.

Παράλληλα κάθε δευτερεύων χρήστης χρησιμοποιεί και μία πιθανότητα  $p\_sense$  η οποία καθορίζει αν στο επόμενο slot αυτός θα επιχειρήσει να μεταδώσει ή όχι. Η πιθανότητα αυτή είναι αρχικά ίση με 1 για όλους τους δευτερεύοντες χρήστες. Όταν ο δευτερεύων χρήστης υποστεί δύο συνεχόμενες συγκρούσεις αυτή η πιθανότητα μειώνεται κατά 0.1 με ελάχιστη δυνατή τιμή που μπορεί να πάρει το 0.5. Αντίθετα, όταν ο δευτερεύων χρήστης μεταδώσει επιτυχώς η πιθανότητα αυτή αυξάνεται κατά 0.1 με μέγιστη δυνατή τιμή το 1. Επίσης, αν ο SU αισθανθεί όλα τα κανάλια απασχολημένα σε κάποιο slot, η πιθανότητα  $p\_sense$  γίνεται ίση με 1.

Τέλος η χρησιμοποιούμενη λίστα (δηλαδή μία σειρά του πίνακα) εγκαταλείπεται από το δευτερεύοντα χρήστη όταν έχουν συμβεί δύο συνεχόμενες συγκρούσεις στα πακέτα που έστειλε χρησιμοποιώντας τη λίστα αυτή. Τότε την εγκαταλείπει γιατί κατά πάσα πιθανότητα κάποιος άλλος δευτερεύων χρήστης έχει επιλέξει την ίδια γραμμή από το πίνακα Latin Square, γεγονός που οδηγεί σε συνεχόμενες συγκρούσεις μεταξύ των δευτερευόντων χρηστών.

Έπειτα από διάφορες μετρήσεις είδαμε ότι η τέταρτη ευκαιρία μετάδοσης χρησιμοποιείται σπάνια, καθώς τότε δύσκολα ο χρήστης θα βρει ελεύθερα κανάλια για να μεταδώσει. Ακόμα και να βρει όμως, ο χρόνος που του μένει για τη μετάδοση των δεδομένων του στη διάρκεια του slot, είναι αρκετά μικρός. Έτσι για την ανάγκη των προσομοιώσεων της εργασίας μας, υλοποιήσαμε το συγκεκριμένο αλγόριθμο με τρεις ευκαιρίες μετάδοσης.

### 2.2.3 Αλγόριθμος κατασκευής SS (Build SS)

Ο συγκεκριμένος αλγόριθμος τον οποίο προτείνουμε και υλοποιήσαμε βασίζεται στον παραπάνω Distributed Algorithm με τη διαφορά ότι προσπαθεί να χτίσει μία ακολουθία (SS) αποτελούμενη από κανάλια, την οποία θα συμβουλευτεί κάθε φορά ως προς το ποιο κανάλι να αισθανθεί στο επόμενο slot με σκοπό να μεταδώσει σε αυτό, αν είναι ελεύθερο. Η ακολουθία αυτή αποτελείται από τρεις θέσεις. Κάθε SU



μπορεί να εκτιμήσει τη χωρητικότητα του κάθε καναλιού στο σύστημα Έτσι θεωρούμε ότι το μόνο που γνωρίζουν οι χρήστες για τα κανάλια είναι το πλήθος τους και η χωρητικότητα τους. Σκοπός του αλγορίθμου αυτού είναι να καταφέρουν οι δευτερεύοντες χρήστες να χτίσουν διαφορετικές ακολουθίες ο ένας από τον άλλο, μιας και υποθέτουμε ότι δεν μπορούν να επικοινωνήσουν μεταξύ τους. Με αυτό τον τρόπο έχουμε λιγότερες συγκρούσεις πακέτων πληροφορίας, αφού σε λιγότερες περιπτώσεις θα προσπαθούν να μεταδώσουν στο ίδιο κανάλι στην ίδια ευκαιρία μετάδοσης (minislot) παραπάνω από ένας δευτερεύοντες χρήστες. Έτσι θα επιτευχθεί αυξημένη απόδοση του συστήματος.

### 2.2.3.1 Ανάλυση λογικής κατασκευής της λίστας καναλιών ( SS )

Στην αρχή της λειτουργίας του συστήματος η ακολουθία κάθε δευτερεύοντα χρήστη είναι άδεια. Έτσι κάθε SU στο πρώτο slot επιλέγει τυχαία ένα κανάλι για την πρώτη ευκαιρία μετάδοσης. Η πιθανότητα να επιλέγει ένα κανάλι παραμένει ίδια και ίση με  $1/N_p$  όπως στον απλό τυχαίο αλγόριθμο. **Όπως είπαμε παραπάνω θεωρούμε ότι οι δευτερεύοντες χρήστες δεν γνωρίζουν τον ρυθμό άφιξης πακέτων από τον πρωτεύοντα χρήστη για κάθε κανάλι, αλλά γνωρίζουν τη χωρητικότητα του και το πλήθος το συνολικών καναλιών του συστήματος.** Αν το κανάλι στην πρώτη ευκαιρία είναι απασχολημένο τότε ο χρήστης αναζητά άλλο κανάλι στη δεύτερη ευκαιρία, και αντίστοιχα άλλο κανάλι στην τρίτη ευκαιρία αν αποτύχει και η δεύτερη, και δεν αποθηκεύει κανένα κανάλι στη λίστα του μέχρι να πετύχει σε κάποιο.

Κάθε δευτερεύων χρήστης μεταδίδει σε κάθε slot βάσει μίας πιθανότητας  $p_{sense}$ . Αυτή η πιθανότητα αρχικά είναι 1 για όλους τους χρήστες, και μεταβάλλεται αναλόγως με τις καταστάσεις που αντιμετωπίζει ο χρήστης στη διάρκεια της λειτουργίας του συστήματος. Συγκεκριμένα, όταν ο χρήστης μεταδώσει σε ένα κανάλι και το πακέτο του συγκρουστεί, με αποτέλεσμα να λάβει NACK στο τέλος του slot, η πιθανότητα  $p_{sense}$  μειώνεται κατά 0.1 με την ελάχιστη αποδέκτη τιμή που μπορεί να φτάσει το 0.5. Αντίθετα, αν ο χρήστης μεταδώσει επιτυχώς στο slot η πιθανότητα  $p_{sense}$  αυξάνεται κατά 0.1 με προφανή μέγιστη τιμή το 1. Με αυτό τον τρόπο όταν υπάρχει υψηλός ανταγωνισμός στο δίκτυο (λιγότερα κανάλια από δευτερεύοντες χρήστες ή υψηλή κίνηση στα κανάλια προερχόμενη από τους πρωτεύοντες χρήστες) τότε χρησιμοποιώντας την παραπάνω πιθανότητα είναι πολύ πιθανό να μην προσπαθήσουν να μεταδώσουν όλοι οι δευτερεύοντες χρήστες στο επόμενο slot με αποτέλεσμα να μειώνεται ο ανταγωνισμός στο δίκτυο. Αντίθετα όταν οι δευτερεύοντες χρήστες μεταδίδουν επιτυχώς επανειλημμένα, αυτό σημαίνει ότι το δίκτυο έχει τη δυνατότητα να τους υποστηρίξει όλους χωρίς να συγκρούονται και η πιθανότητα  $p_{sense}$  τείνει να γίνει ίση με 1. Η μείωση και αύξηση της πιθανότητας  $p_{sense}$  κατά 0.1 αλλά και η ελάχιστη δυνατή τιμή της που εδώ υποτίθεται ίση με το 0.5, επιλεχτήκαν πειραματικά έπειτα από διάφορες δοκιμές. Η ελάχιστη τιμή που στην περίπτωση μας είναι 0.5, θα μπορούσε να τεθεί και μικρότερη, αλλά δεν θέλαμε



με αυτόν τον τρόπο να αποτρέψουμε ένα δευτερεύοντα χρήστη από την πρόσβαση στο σύστημα. Αν για παράδειγμα η ελάχιστη τιμή γινόταν 0.1, τότε ο δευτερεύων χρήστης θα προσπαθούσε σπάνια να μεταδώσει.

Όταν ο SU μεταδώσει πρώτη φορά σε κάποιο κανάλι το αποθηκεύει στην ακολουθία του (SS). Κάθε φορά που πετυχαίνει μία μετάδοση σε οποιοδήποτε κανάλι και αυτό δεν υπάρχει στο SS του, το τοποθετεί στην πρώτη θέση που θα βρει κενή από τις τρεις διαθέσιμες της λίστας SS. Για παράδειγμα, με δεδομένο ότι στην αρχή το SS ενός SU είναι άδειο ( $SS=[0\ 0\ 0]$ ), την πρώτη φορά που ο δευτερεύων χρήστης θα επιτύχει σε κάποιο κανάλι, σε οποιαδήποτε ευκαιρία μετάδοσης και αν γίνει αυτό, θα τοποθετήσει το κανάλι αυτό στην πρώτη θέση του SS του. Έπειτα στο επόμενο slot δεν θα πάει να επιλέξει τυχαία κάποιο κανάλι, αλλά θα επιλέξει αυτό που έχει στην πρώτη θέση του SS. Αν το κανάλι είναι απασχολημένο, τότε θα κοιτάξει τη δεύτερη θέση του SS, και αν είναι κενή θα πάει να επιλέξει τυχαία ένα κανάλι με βάση τις πιθανότητες που αναλύθηκαν παραπάνω, με προϋπόθεση ότι το κανάλι αυτό θα είναι διαφορετικό από αυτό που έχει αποθηκεύσει στην πρώτη θέση. Αν το βρει και αυτό απασχολημένο τότε προχωρά στην τρίτη ευκαιρία μετάδοσης όπου κάνει ακριβώς την ίδια λειτουργία, με τη διαφορά ότι το κανάλι που θα επιλέξει πρέπει να είναι διαφορετικό από αυτό στην πρώτη θέση του SS και από αυτό που επέλεξε στη δεύτερη ευκαιρία. Έπειτα αν καταφέρει να μεταδώσει, αποθηκεύει το κανάλι αυτό στη δεύτερη θέση της ακολουθίας του. Αντίστοιχα αν κάποια στιγμή μεταδώσει σε κάποιο κανάλι που δεν υπάρχει στην πρώτη ή στη δεύτερη θέση της λίστας του, θα το αποθηκεύσει και αυτό στην τρίτη θέση.

Βλέπουμε λοιπόν ότι ο κάθε χρήστης χτίζει την ακολουθία του σταδιακά, **με τα κανάλια στα οποία κατάφερε να μεταδώσει όταν χρειάστηκε να επιλέξει τυχαία κάποιο**. Ας αναλύσουμε ένα παράδειγμα για καλύτερη κατανόηση. Έστω ότι έχουμε 4 κανάλια και 3 δευτερεύοντες χρήστες. Θα επικεντρωθούμε σε έναν μόνο SU, έστω στον SU1. Στην αρχή λειτουργίας του συστήματος ο SU1 έχει  $SS=[0\ 0\ 0]$ . Ο SU1 κοιτάει το SS του στην αρχή κάθε slot. Βλέπει ότι δεν έχει αποθηκεύσει κάποιο κανάλι άρα πρέπει να επιλέξει τυχαία ένα από τα 4 διαθέσιμα. Την πρώτη φορά που θα καταφέρει να μεταδώσει, **σε όποια ευκαιρία μετάδοσης και αν γίνει αυτό(1<sup>η</sup> ή 2<sup>η</sup> ή 3<sup>η</sup>)**, θα αποθηκεύσει το κανάλι στην πρώτη θέση. Έστω λοιπόν ότι μεταδίδει στο κανάλι 3, τότε το SS θα γίνει  $[3\ 0\ 0]$ . Στο επόμενο slot, θα κοιτάξει την πρώτη θέση, θα δει το κανάλι 3 και θα προσπαθήσει να μεταδώσει σε αυτό στην πρώτη ευκαιρία μετάδοσης. Αν το βρει απασχολημένο, έχει άλλες δύο ευκαιρίες να βρει κανάλι για να μεταδώσει το πακέτο του στο τρέχον slot. Έστω ότι στη δεύτερη ευκαιρία επιλέγει το κανάλι 1 και αυτό είναι απασχολημένο, τότε προχωράει στην τρίτη ευκαιρία όπου υποθέτουμε ότι επιλέγει το κανάλι 2 στο οποίο τελικά μεταδίδει επιτυχημένα. Τότε θα αποθηκεύσει το κανάλι 2 στο SS του στην πρώτη κενή θέση. Άρα πλέον το SS του SU1 θα είναι  $[3\ 2\ 0]$ . Στην αρχή του επόμενου slot θα κοιτάξει πάλι την πρώτη θέση του SS του, θα προσπαθήσει στο 3 κανάλι και αν αυτό είναι απασχολημένο, στη δεύτερη ευκαιρία θα προσπαθήσει να μεταδώσει στο κανάλι 2 κτλ. Εάν καταφέρει να

μεταδώσει σε κάποιο κανάλι που υπάρχει στη λίστα της ακολουθίας του, τότε δεν κάνει καμία αλλαγή σε αυτή.

Έτσι όπως έχει αναλυθεί ο αλγόριθμος μέχρι στιγμής είναι λογικό να μην είναι αρκετά αποδοτικός, μιας και δεν έχουμε αναφέρει τι γίνεται σε περίπτωση που η ακολουθία που έχει δημιουργήσει ο κάθε χρήστης δεν είναι αποδοτική. Δηλαδή μπορεί η ακολουθία (SS) που έχει χτίσει να τον οδηγεί συνεχώς σε συγκρούσεις με άλλους δευτερεύοντες χρήστες (γιατί μπορεί και αυτοί οι χρήστες να έχουν σχεδόν την ίδια ακολουθία, άρα τα ίδια κανάλια στην ίδια προσπάθεια μετάδοσης) ή μπορεί να μην τον αφήνει να μεταδώσει σχεδόν ποτέ, γιατί τα κανάλια που έχει επιλέξει έχουν υψηλή κίνηση και απασχολούνται σχεδόν μόνιμα από τον πρωτεύοντα χρήστη που μεταδίδει σε αυτό το κανάλι, η από κάποιον άλλο δευτερεύοντα χρήστη που έχει επιλέξει το ίδιο κανάλι αλλά σε προηγούμενη ευκαιρία μετάδοσης. Δηλαδή αν ο SU1 έχει  $SS=[3\ 2\ 4]$  και ο SU2 έχει  $SS=[4\ 3\ 1]$  τότε αν ο SU1 μεταδίδει συνέχεια στο κανάλι 3 όταν αυτό δεν είναι απασχολημένο, και ο SU2 τυχαίνει να μη μεταδίδει στο κανάλι 4 γιατί είναι πάντα κατειλημμένο από τον PU, τότε είναι σχεδόν σίγουρο (εκτός αν γίνει λάθος ανάγνωση – βάσει της πιθανότητας  $P_{md}$ ) ότι ο SU2 θα βρίσκει πάντα απασχολημένο το κανάλι 3 αφού σε αυτό θα προλάβαινε να μεταδώσει ο SU1. Έτσι θα είχε νόημα ο SU2 να αφαιρέσει από τη λίστα του τα κανάλια 4 και 3 αφού τα βρίσκει μόνιμως απασχολημένα. Επιπλέον, όταν παραπάνω από δύο δευτερεύοντες χρήστες έχουν το ίδιο κανάλι στην ίδια θέση της λίστας τους (στις ίδιες ευκαιρίες μετάδοσης) πρέπει να το διαγράψουν (έστω ένας από τους δύο). Δηλαδή το SS για τον πρώτο χρήστη είναι  $[3\ 2\ 1]$  και για το δεύτερο  $[3\ 1\ 4]$ , τότε αν έβρισκαν το κανάλι 3 άδειο θα μετέδιδαν και οι δύο με αποτέλεσμα να συγκρούονταν τα πακέτα τους και θα λάμβαναν NACK στο τέλος του slot που τους ειδοποιεί για την αποτυχία της μετάδοσης τους.

#### 2.2.3.2 Ανάλυση λογικής ανά-κατασκευής SS - διαγραφή καναλιών

Πρέπει λοιπόν να γίνονται κάποιες ενέργειες έτσι ώστε κάθε χρήστης να αλλάζει την ακολουθία του, όποτε κρίνεται αναγκαίο. Για την αντιμετώπιση του προβλήματος που χρήστες έχουν στην ίδια θέση το ίδιο κανάλι, υλοποιήσαμε ένα έλεγχο ο οποίος κοιτάει αν έχουν γίνει τρεις **συνεχόμενες συγκρούσεις**, σε τρία **συνεχόμενα slots** στο ίδιο κανάλι (εξηγείται στα αποτελέσματα των προσομοιώσεων γιατί επιλέχτηκε αυτός ο αριθμός). Κάθε SU έχει 2 μεταβλητές για κάθε θέση της λίστα του, ώστε να μπορεί να μετρήσει αυτά που επιθυμούμε. Τότε οι εμπλεκόμενοι δευτερεύοντες χρήστες διαγράφουν το κανάλι αυτό με μία πιθανότητα  $P_{delete\_channel}$ , και κάνουν μετατόπιση της θέσης των καναλιών, αν αυτή απαιτείται, έτσι ώστε να υπάρχουν κενές θέσεις μόνο στο τέλος της λίστας τους. Αν επιστρέψουμε στο παράδειγμα όπου το SS για τον πρώτο χρήστη ήταν  $[3\ 2\ 1]$  και για το δεύτερο  $[3\ 1\ 4]$ , τότε αν γίνουν τρεις συνεχόμενες συγκρούσεις σε τρία συνεχόμενα slots στο κανάλι 3 οι χρήστες θα διαγράψουν με πιθανότητα  $P_{delete\_channel}$  το κανάλι 3 από τη λίστα τους. Αν το

διαγράψει ο χρήστης 1 η λίστα του θα γίνει [1 4 0] και αν το διαγράψει ο χρήστης 2 η λίστα του θα γίνει [2 1 0]. Αν έχουν υπάρξει δύο συνεχόμενες συγκρούσεις και στο επόμενο slot ο δευτερεύων χρήστης που τις έχει υποστεί, μεταδώσει επιτυχημένα σε οποιοδήποτε κανάλι, ο αριθμός των συγκρούσεων μηδενίζεται για όλα τα κανάλια που ανήκουν στη λίστα του χρήστη αυτού.

Για την περίπτωση που ένα κανάλι είναι κατειλημμένο συνέχεια, υλοποιήσαμε ένα έλεγχο ο οποίος κοιτάει αν το κανάλι είναι κατειλημμένο  $y$  συνεχόμενες φορές όταν χρησιμοποιήθηκε από το δευτερεύοντα χρήστη στην προσπάθεια του να μεταδώσει σε αυτό. Εδώ δεν είναι ανάγκη να είναι σε συνεχόμενα slots κατειλημμένο για να θεωρηθεί ότι δεν είναι «καλό» για το χρήστη. Για παράδειγμα για το  $SS=[1\ 2\ 4]$  αν στο πρώτο slot βρεθεί κατειλημμένο το κανάλι 1 και το κανάλι 2, στο δεύτερο slot ο χρήστης μεταδώσει στο κανάλι 1, και στο τρίτο slot βρει κατειλημμένο και το 1 και το 2 τότε ο αριθμός των συνεχόμενων φόρων που έχει βρεθεί κατειλημμένο το κανάλι 2 είναι δύο αν και αυτό δεν συνέβη σε συνεχόμενα slots. Αν ο SU μεταδώσει επιτυχημένα στο κανάλι 2, ο αριθμός αυτός μηδενίζεται. Τώρα για το αριθμό  $y$  όπως αναφέρθηκε παραπάνω, έγιναν διάφορες μετρήσεις για να δούμε ποιά τιμή είναι καλύτερη για τα σενάρια που έχουμε υποθέσει. Καταλήξαμε ότι ο πλέον κατάλληλος αριθμός για τα κανάλια που ανήκουν στην πρώτη ευκαιρία μετάδοσης είναι 15, για αυτά που ανήκουν στη δεύτερη θέση 10, και αυτά που ανήκουν στην τρίτη 5. Αυτή η μείωση οφείλεται στη λογική ότι όταν ένα κανάλι βρίσκεται στη δεύτερη θέση, σημαίνει ότι μπορεί να είναι κατειλημμένο από τον PU ή από κάποιον SU που το επέλεξε στην πρώτη ευκαιρία μετάδοσης του. Έτσι δεν θέλουμε να κρατήσουμε το κανάλι αρκετή ώρα στην ακολουθία του SU, μιας και λιγότερες μετρήσεις που το βρίσκουν κατειλημμένο είναι αρκετές για να μας δείξουν ότι πρέπει να το διαγράψουμε. Αντίστοιχα, όταν το κανάλι ανήκει στην τρίτη θέση και είναι κατειλημμένο, αυτό μπορεί να οφείλεται είτε στον PU είτε σε SUs που το επέλεξαν στην πρώτη η δεύτερη προσπάθεια μετάδοσης. Άρα είναι ακόμα πιο μεγάλη η πιθανότητα αυτό το κανάλι να είναι κατειλημμένο και μπορούμε να αποφασίσουμε πιο γρήγορα ότι είναι προτιμότερο να διαγραφεί από τη λίστα του συγκεκριμένου SU. Αντίθετα, από τα αποτελέσματα των προσομοιώσεων μας κρίθηκε ότι αν το κανάλι ανήκει στην πρώτη ευκαιρία μετάδοσης, ο αριθμός των συνεχόμενων φόρων που ο συγκεκριμένος SU πρέπει να δει κατειλημμένο το κανάλι για να το διαγράψει είναι μεγαλύτερος. Ο λόγος είναι ότι χρειάζονται περισσότερα slots για να αξιολογηθεί σωστά ένα κανάλι μιας και μπορεί να παραμένει απασχολημένο μόνο από τον PU.

Όσον αφορά την πιθανότητα  $P\_delete\_channel$  που αναφέραμε παραπάνω, δηλαδή την πιθανότητα με την οποία ένας δευτερεύοντας χρήστης διαγράφει ένα κανάλι από τη λίστα του αν έχει υποστεί τρεις συνεχόμενες συγκρούσεις σε αυτό. Αν ένα κανάλι κριθεί από το δευτερεύοντα χρήστη ότι πρέπει να διαγραφεί (λόγω τριών συνεχόμενων NACKS σε αυτό), τότε αυτό διαγράφεται με πιθανότητα  $P\_delete\_channel$ . Υπάρχει δηλαδή και το ενδεχόμενο το κανάλι αυτό να μη διαγραφεί. Αυτό έχει νόημα, γιατί πολλές φορές δεν θέλουμε να διαγράφονται συνέχεια κανάλια από τις λίστες, ώστε να μην υπάρχει υψηλός ανταγωνισμός μεταξύ

των SUs, αλλά και για να καταφέρνουν οι SUs να δημιουργούν πιο σταθερές λίστες. Για παράδειγμα, όπως θα δούμε και από τα αποτελέσματα που παραθέτουμε παρακάτω, έχει νόημα αυτή η πιθανότητα να μην είναι ίση με 1 ή 0 και η κατάλληλη τιμή της έχει να κάνει με τα εκάστοτε χαρακτηριστικά του συστήματος ( $P_{fa}$ ,  $P_{md}$ , πλήθος PUs και SUs, κ.α). Τελικά η τιμή που επιλέχτηκε για αυτήν την παράμετρο είναι το 0.45.

#### 2.2.4 Αλγόριθμος κατασκευής SS με διαφορετική στρατηγική αποθήκευσης και διαγραφής (Build SS\_2)

Ο αλγόριθμος αυτός προσπαθεί να χτίσει τη λίστα με λίγο διαφορετικό τρόπο σε σχέση με τον Build SS. **Πιο συγκεκριμένα δεν υλοποιεί διαγραφές από τη λίστα εάν αυτή δεν έχει γεμίσει και στις 3 θέσεις της.** Όταν αυτή γεμίσει, τότε ανάλογα με το πότε κριθεί από τον αλγόριθμο ότι πρέπει να διαγράψει κάποιο κανάλι από το χρήστη, διαγράφεται και παραμένει κενή η θέση στην οποία αυτό υπήρχε. Αν ο χρήστης μεταδώσει σε ένα κανάλι το οποίο δεν υπήρχε ήδη στη λίστα, τότε αυτό το κανάλι αποθηκεύεται **πάντα στην πρώτη θέση της λίστας**, και τα υπόλοιπα κανάλια που υπάρχουν σε αυτή ολισθαίνουν δεξιά κατά μία θέση. Να υπενθυμίσουμε εδώ ότι στον αλγόριθμο Build SS το νέο κανάλι δεν τοποθετούνταν πάντα στην πρώτη θέση της λίστας, αλλά στην πρώτη κενή θέση που θα συναντούσε.

Για παράδειγμα αν έχουμε πέντε κανάλια, τα 1,2,3,4,5 και το χρήστη 1 με  $SS = [2 \ 3 \ 4]$ , ο οποίος έκρινε ότι πρέπει να διαγραφεί το κανάλι 3 τότε το SS θα γινόταν  $[2 \ 0 \ 4]$ . Αν σε κάποιο από τα επόμενα slots ο χρήστης καταφέρει να μεταδώσει σε διαφορετικό κανάλι, έστω στο 5, τότε η λίστα του χρήστη θα γινόταν  $[5 \ 2 \ 4]$ . Για την αρχική περίοδο των εισαγωγών μέχρι να γεμίσει η λίστα, ο αλγόριθμος ακολουθεί τη διαδικασία που αναλύθηκε παραπάνω.

Ο αλγόριθμος αυτός υλοποιήθηκε για να δούμε πως θα συμπεριφέρεται το σύστημα όταν ο SU τοποθετεί πάντα στην πρώτη θέση της λίστας του το κανάλι που έχει μεταδώσει επιτυχώς, αλλά και πως όταν η λίστα του SU γεμίσει, πράγμα που στον αλγόριθμο Build SS δεν συμβαίνει πάντα, μιας και γίνονται συχνά διαγραφές καναλιών πριν γεμίσουν όλες οι θέσεις της λίστας.

### 2.2.5 Αλγόριθμος κατασκευής SS με συνεργασία (Build SS coop)

Ο αλγόριθμος αυτός, βασίζεται στην ιδέα του αλγόριθμου Build SS με τη διαφορά ότι υπάρχει συνεργασία μεταξύ των SUs στο στην αρχή κάθε slot. Κάθε SU (που έχει κανάλι προς διαγραφή) επικοινωνεί με το συντονιστή, για να του μεταδώσει ότι έλαβε συνεχόμενα NACKs και ετοιμάζεται να διαγράψει το κανάλι. Έπειτα ο συντονιστής επικοινωνεί με τους υπολοίπους SUs. Κάθε δευτερεύων χρήστης που λαμβάνει αυτό το μήνυμα, ελέγχει αν και αυτός μετέδωσε στο ίδιο κανάλι και στέλνει μήνυμα στο συντονιστή. Έπειτα η πληροφορία ελέγχεται από το συντονιστή, ο οποίος τελικά γνωρίζει ποιοί χρήστες συγκρούστηκαν σε ποιο κανάλι, και αποφασίζει τυχαία ποιός από αυτούς (ακριβώς ένας) θα το κρατήσει στη λίστα του, με τους υπόλοιπους να το διαγράφουν. Θεωρούμε ότι υπάρχει ανεξάρτητο κανάλι (από τα  $N_p$ ) για να εξυπηρετεί την επικοινωνία μεταξύ του συντονιστή και των δευτερευόντων χρηστών. Η διάρκεια ολόκληρης της φάσης αυτής, μέχρι να αποφασιστεί ποιοί χρηστές θα το διαγράψουν, διαρκεί αρκετά μικρό χρόνο, μιας και θεωρούμε ότι τα πακέτα που στέλνονται στην φάση αυτή, είναι μικρού μεγέθους.

Για παράδειγμα στον αλγόριθμο Build SS αν οι χρήστες 1,2 και 3 είχαν  $SS1=[1\ 2\ 3]$ ,  $SS2=[1\ 5\ 4]$ ,  $SS3=[1\ 4\ 5]$ , αντίστοιχα και μετέδιδαν όλοι στην πρώτη ευκαιρία, δηλαδή στο κανάλι 1, τότε θα λάμβαναν όλοι NACK. Αν αυτό γινόταν σε τρία συνεχόμενα slots, όπως εξηγήσαμε στον αλγόριθμο Build SS, οι χρήστες θα διέγραφαν το κανάλι 1 από τη λίστα τους, και αυτές πλέον θα γινόντουσαν ίσες με  $SS1=[2\ 3\ 0]$ ,  $SS2=[5\ 4\ 0]$ ,  $SS3=[4\ 5\ 0]$ . Με τον αλγόριθμο Build SS coop, ο χρήστης που έλαβε τρία συνεχόμενα NACKs σε κάποιο κανάλι θα επικοινωνήσει με τους υπολοίπους και το συντονιστή στο τέλος του slot, και ένας από αυτούς, έστω ο χρήστης 2 θα κρατήσει το κανάλι 1 στη λίστα του, ενώ οι υπόλοιποι θα το διαγράψουν. Επομένως οι νέες λίστες των χρηστών 1,2 και 3 θα είναι  $SS1=[2\ 3\ 0]$ ,  $SS2=[1\ 5\ 4]$ ,  $SS3=[4\ 5\ 0]$ . Βλέπουμε λοιπόν, ότι τώρα το κανάλι 2 δεν έχει μείνει τελείως ελεύθερο από SU, εφόσον ο χρήστης 2 θα προσπαθήσει να μεταδώσει σε αυτό, στο αμέσως επόμενο slot. Χωρίς τη συνεργασία που περιγράψαμε όλοι θα το διέγραφαν, και υπήρχε περίπτωση το κανάλι αυτό να μείνει αναξιοποίητο στα επόμενα slots. Τέλος αξίζει να σημειωθεί ότι με τον Build SS coop δεν είναι αναγκαστικό κάποιος χρήστης να έχει τρία συνεχόμενα NACKs για να διαγράψει το κανάλι από τη λίστα του. Μπορεί να συγκρούστηκε μόνο μία φορά σε αυτό το κανάλι, αλλά για κάποιον άλλο δευτερεύοντα χρήστη να ήταν η τρίτη συνεχόμενη σύγκρουση και έτσι να επιλέγεται να το διαγράψει ο χρήστης που είχε μόνο μία σύγκρουση ενώ θα το κρατήσει αυτός που είχε τρεις.

## 2.3 Αξιολόγηση απόδοσης

### 2.3.1 Μετρική απόδοσης

Χρησιμοποιούμε ως μετρική απόδοσης τον κανονικοποιημένο μέσο ρυθμό μετάδοσης **ανά δευτερεύοντα χρήστη**, με μονάδα μέτρησης τα Mbits per sec.

Δηλαδή αν ένας SU μεταδίδει για χρόνο  $t_{tr}$  (ms) σε κανάλι με χωρητικότητα  $C$  Kbps τότε μεταδίδει συνολικά  $C \cdot t_{tr}$  bits στο συγκεκριμένο διάστημα. Αθροίζοντας όποτε μεταδίδει **οποιοσδήποτε** δευτερεύων χρήστης αυτό τον αριθμό με ότι έχουν μεταδώσει μέχρι τότε όλοι οι SUs, στο τέλος της προσομοίωσης έχουμε τα συνολικά Mbits που μετέδωσαν (έστω  $data\_trans\_i$  τα bits που μετέδωσαν όλοι οι δευτερεύοντες χρήστες στη διάρκεια της προσομοίωσης και  $total\_time$  η συνολική διάρκεια της προσομοίωσης σε sec). Έπειτα κάνοντας την πράξη  $\frac{1}{N_s \cdot Total\_time} \sum_{i=1}^{N_p} data\_trans\_i$ , έχουμε την κανονικοποιημένη μέση δυνατότητα μεταφοράς ανά δευτερεύοντα χρήστη, **η πιο απλά το μέσο αριθμό Mbits/sec ανά δευτερεύοντα χρήστη που μεταδόθηκαν καθ' όλη τη διάρκεια προσομοίωσης του συστήματος.**

### 2.3.2 Μοντέλο προσομοίωσης

Για να συγκρίνουμε τα αποτελέσματα του αλγορίθμου Build SS με τον Build SS\_2 αλλά και για τη σύγκριση των αλγορίθμων που προτείναμε τελικά (Build SS και Build SS coop) με τον απλό αλγόριθμο PPRA αλλά και τον Distributed υλοποιήσαμε προσομοιώσεις σε όλους τους αλγορίθμους με τα ίδια δεδομένα.

Θεωρούμε ένα time slotted Cognitive Radio Network με  $N_s$  δευτερεύοντες χρήστες, οι οποίοι προσπαθούν ευκαιριακά να μεταδώσουν σε κάθε slot ένα πακέτο πληροφορίας τους σε κάποιο από τα  $N_p$  κανάλια του δικτύου, καθένα από τα οποία σε ένα PU. Οι αφίξεις πακέτων των PUs υποτίθεται ότι ακολουθούν την κατατομή Poisson με ρυθμό άφιξης  $\lambda_i$ . Επίσης συμβολίζουμε με  $P_{md}$  την πιθανότητα κάποιος SU να αισθανθεί λανθασμένα το κανάλι ως μη απασχολημένο, και να προσπαθήσει να μεταδώσει σε αυτό με αποτέλεσμα να υπάρξει σύγκρουση, και με  $P_{fa}$  την πιθανότητα κάποιος SU να αισθανθεί λανθασμένα το κανάλι σαν απασχολημένο, με αποτέλεσμα να μη μεταδώσει σε αυτό, χάνοντας αυτήν την ευκαιρία μετάδοσης. Τέλος συμβολίζουμε με  $T$  τη διάρκεια του slot.

Τα σενάρια όπου έχουμε προσομοιώσει χωρίζονται κυρίως σε 2 κατηγορίες. Μία όπου τα κανάλια είναι περισσότερα από τους SUs, και πιο συγκεκριμένα  $N_p = 10$  και

$N_s=5$ , αλλά και μία όπου υπάρχει υψηλός ανταγωνισμός μεταξύ των SUs, αφού αυτοί είναι περισσότεροι από τα κανάλια ( $N_p=10$ ,  $N_s=13$ ). Κάθε σενάριο διαιρείται σε αλλά υποσενάρια όπως για παράδειγμα να παραμένει σταθερή η πιθανότητα  $P_{fa}$  και να μεταβάλλεται η  $P_{md}$  ή να παραμένει σταθερή η πιθανότητα  $P_{md}$  και να μεταβάλλεται η  $P_{fa}$ . Επίσης έχουν γίνει μετρήσεις για όλες τις παραπάνω περιπτώσεις με τον Poisson ρυθμό άφιξης πακέτων να είναι  $\lambda_i=0,3$  πακέτα per slot για όλα τα κανάλια (δηλαδή όλα τα κανάλια είναι ελαφρώς φορτωμένα), και  $\lambda_i=0,7$  πακέτα per slot για όλα τα κανάλια (δηλαδή όλα τα κανάλια είναι υψηλά φορτωμένα). Τέλος, έχουν γίνει μετρήσεις και στην περίπτωση που τα κανάλια έχουν διαφορετικούς ρυθμούς άφιξης πακέτων.

Το λαμβανόμενο SNR σε κάθε δευτερεύοντα δέκτη από κάθε πρωτεύοντα και δευτερεύοντα πομπό (transmitter) έχει τεθεί ίσο με 11.13 dB (13 power ratio) για όλες τις δυνατές συνδέσεις. Το εύρος φάσματος κάθε καναλιού είναι 6MHz, ομοιόμορφα καταναμημένο στο διάστημα 470-862 MHz, ομοίως με το [8], στο οποίο ο συγγραφέας αναφέρει ότι αυτές οι συχνότητες χρησιμοποιούνται στις Η.Π.Α για την αναλογική μετάδοση τηλεοπτικών σημάτων. Η χωρητικότητα κάθε καναλιού υπολογίζεται ως  $C_i = B_i \log_2(1 + \text{SNR})$ , όπου  $B_i$ ,  $i = 1, \dots, N_p$ , είναι το εύρος φάσματος του  $i$  καναλιού ( $B_i=6$  MHz,  $i=1, \dots, N_p$ ). Η διάρκεια κάθε slot είναι 10ms. Οι παραπάνω παράμετροι επιλέχτηκαν βάσει του προτύπου IEEE 802.22 [3] και [8].

**Πίνακας 2.1 Παράμετροι του συστήματος με τις καθορισμένες τιμές**

Συμβολισμός	Επεξήγηση (καθορισμένη τιμή)
$T$	Διάρκεια slot (10 ms)
$t_s$	Χρόνος που απαιτείται για να αισθανθεί κάποιο κανάλι ο SU σε μία ευκαιρία μετάδοσης (2.4 ms)
$t_{ch}$	Χρόνος που απαιτείται για προετοιμασία ώστε να αισθανθεί νέο κανάλι ο SU (0.01 ms)
$N$	Αριθμός συνολικών slots μίας προσομοίωσης (10000)
$Sim$	Αριθμός συνολικών προσομοιώσεων για εκτίμηση μέσων τιμών (1000)
$\delta$	Συνολικές ευκαιρίες μετάδοσης ανά slot (3)
SNR	Signal to noise ratio (13 power ratio – 11.13 dB)
$B$	Εύρος φάσματος καναλιού (6MHz) – Ομοιόμορφα καταναμημένο στο διάστημα (470-862 MHz)

### 2.3.3 Αποτελέσματα προσομοιώσεων

Τα αποτελέσματα στην ενότητα αυτή χωρίζονται σε τρεις κατηγορίες:



- Κατάλληλη παραμετροποίηση αλγορίθμου Build SS
- Σύγκριση αλγορίθμων τύπου Build SS δίχως συνεργασία
- Σύγκριση προτεινόμενων αλγορίθμων με άλλους της βιβλιογραφίας

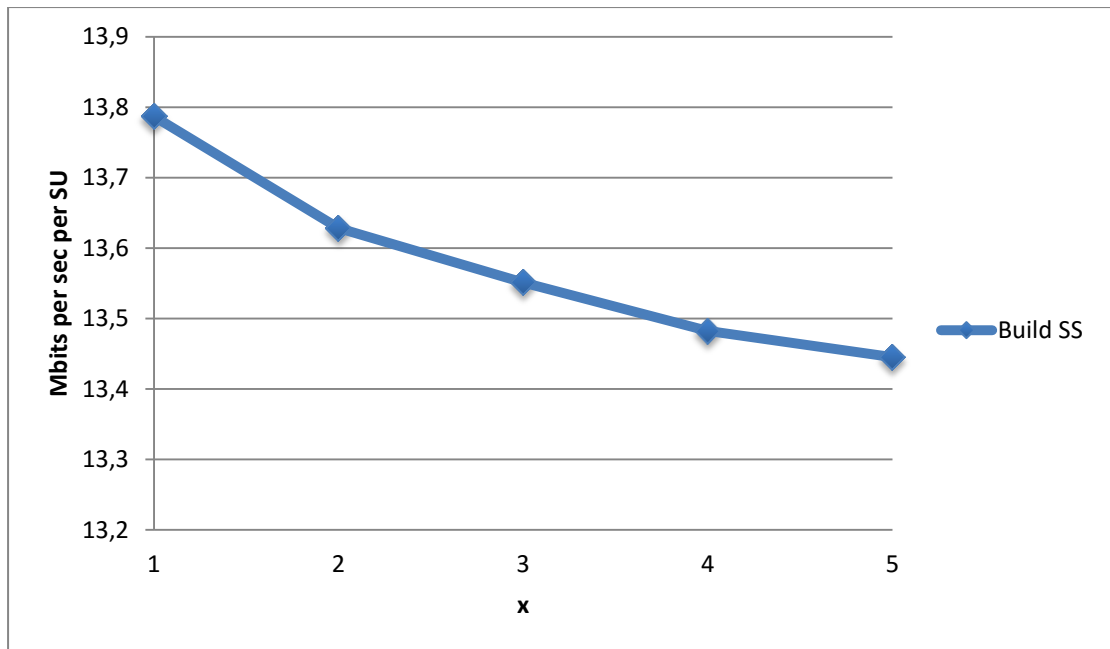
Συγκεκριμένα, η ενότητα 2.3.3.1 περιέχει μετρήσεις σχετικά με τον κατάλληλο αριθμό συνεχόμενων συγκρούσεων που υπόκειται ένας SU προτού εγκαταλείψει το κανάλι στον αλγόριθμο Build SS. Η ενότητα 2.3.3.2 περιέχει μετρήσεις σχετικά με την εξάρτηση της απόδοσης του συστήματος από την πιθανότητα  $P_{\text{delete\_channel}}$  στον αλγόριθμο Build SS. Η επόμενη ενότητα (2.3.3.3), περιέχει τη σύγκριση των αλγορίθμων Build SS και Build SS<sub>2</sub>. Τέλος, οι ενότητες 2.3.3.4 και 2.3.3.5 παρουσιάζουν τα αποτελέσματα των μετρήσεων από τα σενάρια με  $N_p > N_s$  και  $N_s > N_p$  αντίστοιχα, για τους αλγορίθμους Build SS, Build SS coop, Distributed algorithm και PPRA.

Δοθέντος ότι έχουμε θεωρήσει ότι οι δευτερεύοντες χρήστες γνωρίζουν ελάχιστα για το σύστημα, δηλαδή μόνο το πλήθος των καναλιών και τη χωρητικότητα τους, μας αναγκάζει να επιλέξουμε πειραματικά τις τιμές των παραμέτρων του αλγορίθμου Build SS. Σε καμία περίπτωση δεν ισχυριζόμαστε ότι οι τιμές που επιλέξαμε μέσω των προσομοιώσεων είναι οι βέλτιστες τιμές για όλες τις περιπτώσεις και τα σενάρια που μπορούν να υπάρξουν σε ένα πραγματικό σύστημα. Στις ενότητες 2.3.3.1 και 2.3.3.2 παρουσιάζονται ενδεικτικά σχήματα από τις μετρήσεις που έγιναν για την επιλογή των κατάλληλων τιμών των παραμέτρων του αλγορίθμου.

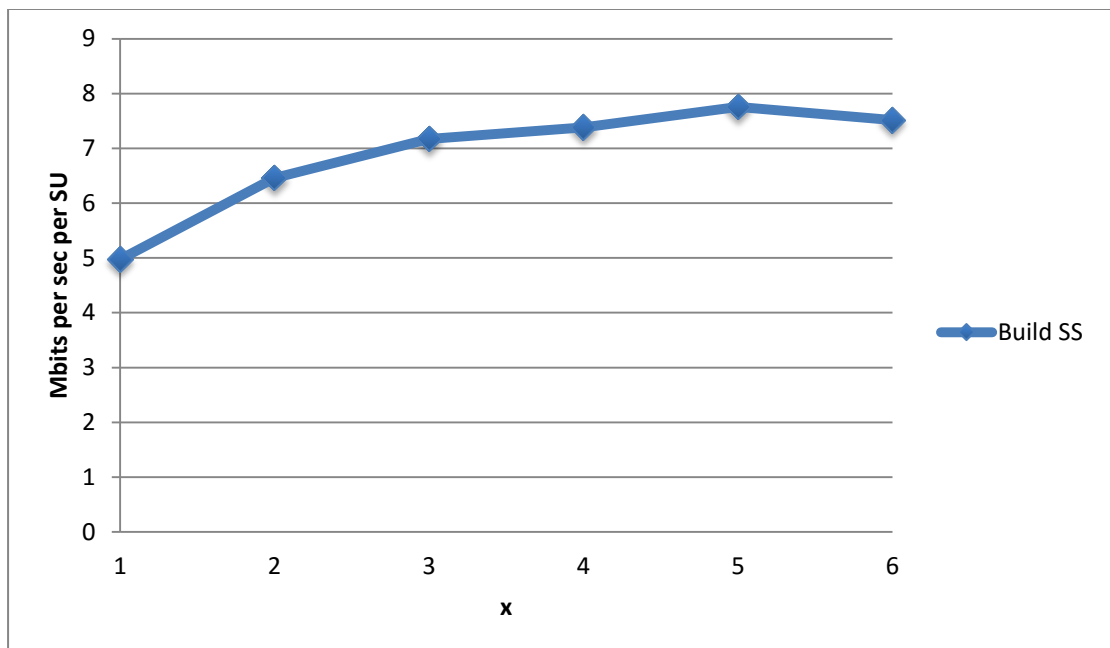
### **2.3.3.1 Αποτελέσματα προσομοιώσεων σχετικά με τον κατάλληλο αριθμό συνεχόμενων συγκρούσεων που πρέπει να υποστεί ένας SU προτού διαγράψει το κανάλι στον αλγόριθμο Build SS**

Αυτός ο αριθμός λοιπόν καθορίζει το πότε ένας χρήστης θα διαγράψει (βάσει τις πιθανότητας  $p_{\text{delete\_channel}}$ ) ένα κανάλι που έχει στη λίστα του όταν έχουν γίνει  $x$  συνεχόμενες συγκρούσεις. Τα παρακάτω σχήματα δείχνουν ποιος αριθμός  $x$  δίνει τα καλύτερα αποτελέσματα. Στο πρώτο σχήμα (2.2) έχουμε χρησιμοποιήσει  $N_p = 10$ ,  $N_s = 5$ ,  $P_{fa} = 0.1$ ,  $P_{md} = 0$ ,  $p_{\text{delete\_channel}} = 1$  και  $\lambda_i = 0.3$  με  $i = 1 \dots N_p$ . Για το δεύτερο σχήμα (2.3) έχουμε τις ίδιες παραμέτρους με τη μόνη διαφορά ότι πλέον  $N_p = 10$  και  $N_s = 13$ .



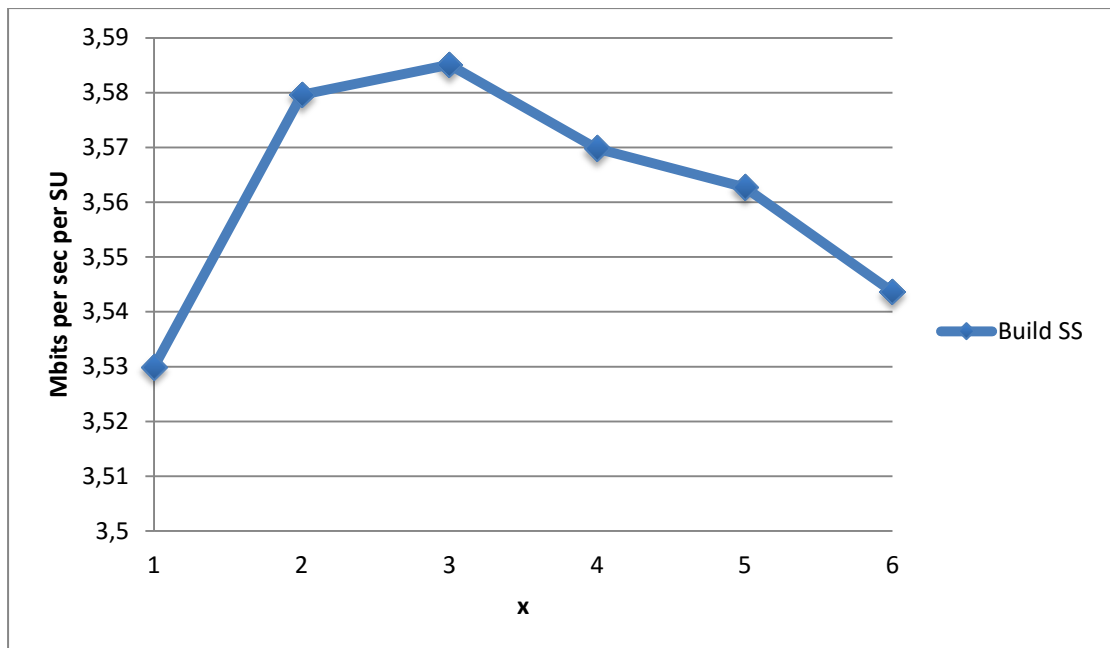


**Figure 2.2** Απόδοση αλγορίθμου *Build SS* σε χαμηλά φορτωμένο δίκτυο ως συνάρτηση του αριθμού των συνεχόμενων συγκρούσεων που πρέπει να υποστεί ο *SU* πριν προχωρήσει στη διαγραφή του καναλιού από τη λίστα του.

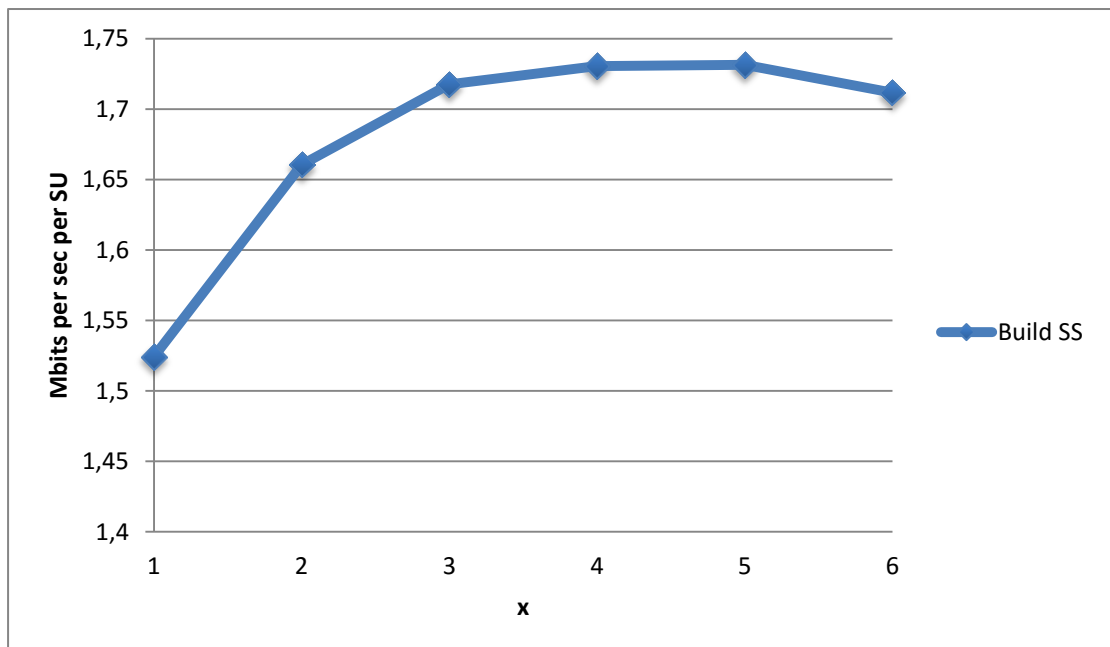


**Figure 2.3** Απόδοση αλγορίθμου *Build SS* σε χαμηλά φορτωμένο δίκτυο ως συνάρτηση του αριθμού των συνεχόμενων συγκρούσεων που πρέπει να υποστεί ο *SU* πριν προχωρήσει στη διαγραφή του καναλιού από τη λίστα του.

Οι παράμετροι που έχουμε χρησιμοποιήσει στα δύο παρακάτω σχήματα είναι: για το σχήμα (2.4)  $N_p = 10$ ,  $N_s = 5$ ,  $P_{fa} = 0.2$ ,  $P_{md} = 0.1$ ,  $p_{delete\_channel} = 1$  και  $\lambda_i = 0.7$ , με  $i = 1 \dots N_p$ , για το σχήμα (2.5)  $N_p = 10$ ,  $N_s = 13$ ,  $P_{fa} = 0.1$ ,  $P_{md} = 0.05$ ,  $p_{delete\_channel} = 1$  και  $\lambda_i = 0.7$ , με  $i = 1 \dots N_p$ .



*Figure 2.4 Απόδοση αλγορίθμου Build SS σε υψηλά φορτωμένο δίκτυο ως συνάρτηση του αριθμού των συνεχόμενων συγκρούσεων που πρέπει να υποστεί ο SU πριν προχωρήσει στη διαγραφή του καναλιού από τη λίστα του.*



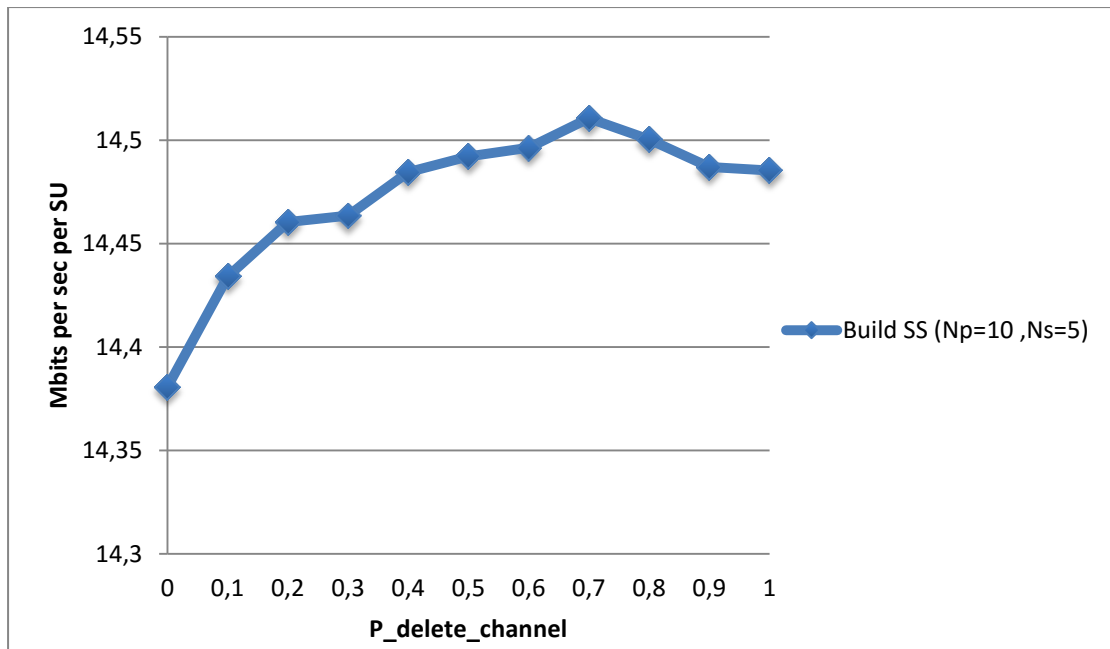
*Figure 2.5 Απόδοση αλγορίθμου Build SS σε υψηλά φορτωμένο δίκτυο ως συνάρτηση του αριθμού των συνεχόμενων συγκρούσεων που πρέπει να υποστεί ο SU πριν προχωρήσει στη διαγραφή του καναλιού από τη λίστα του.*

Από τα παραπάνω σχήματα παρατηρούμε ότι όταν υπάρχει υψηλός ανταγωνισμός μεταξύ SUs (όταν δηλαδή  $N_p < N_s$ ) είναι αποδοτικότερο να διαγράφεται μετά από σχετικά πολλές συγκρούσεις το κανάλι, μιας και αν διαγράφεται έπειτα από λίγες συγκρούσεις, σε κάθε slot πολλοί SUs θα αναζητούν νέο κανάλι με αποτέλεσμα να ανταγωνίζονται μεταξύ τους για λίγα κανάλια. Αντίθετα, όταν  $N_p > N_s$  είναι προτιμότερο οι χρήστες να διαγράφουν σχετικά εύκολα το κάθε κανάλι αφού υπάρχουν περισσότερα κανάλια από τους SUs, επομένως με μεγάλη πιθανότητα θα υπάρχουν κανάλια που δεν έχουν επιλεγεί από κάποιον άλλο χρήστη. Έτσι οι SUs θα έχουν μεγαλύτερη πιθανότητα επιτυχημένης μετάδοσης πακέτου επιλέγοντας νέο κανάλι από το να επιμένουν να μεταδίδουν το πακέτο τους σε κανάλι πάνω στο οποίο έχουν πάρει συνεχόμενα NACKS.

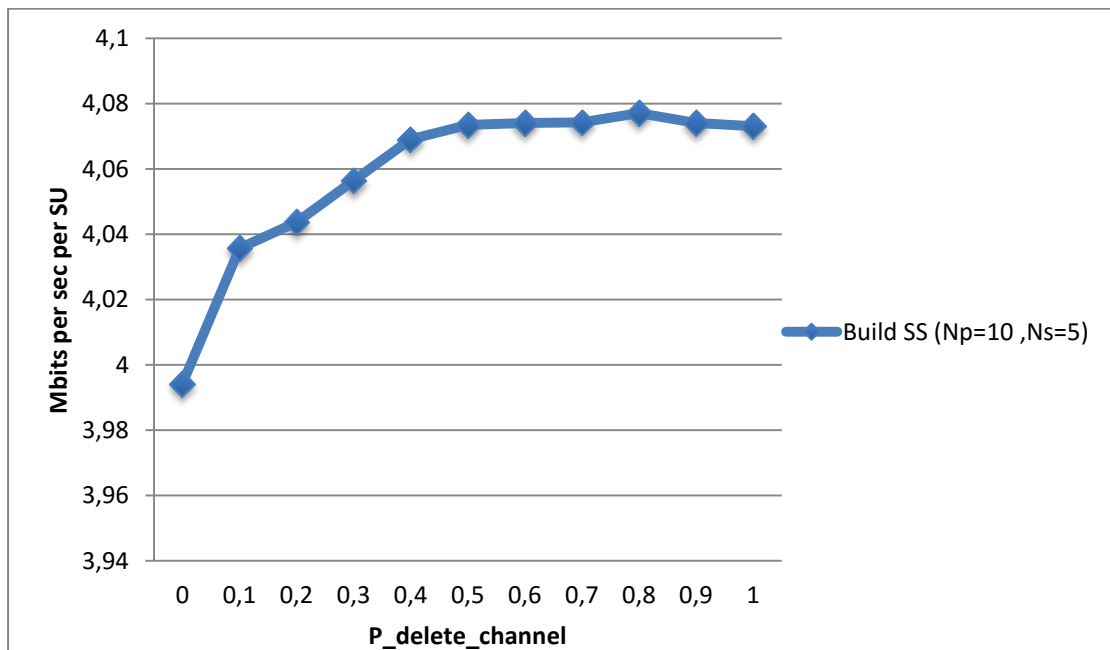
Με δεδομένο ότι δεν μπορούμε να γνωρίζουμε, αν δεν υπάρχει συνεργασία μεταξύ των SUs, πόσοι δευτερεύοντες χρήστες υπάρχουν στο σύστημα και κρίνοντας από τα αποτελέσματα στα παραπάνω σχήματα, αποφασίσαμε να επιλέξουμε σαν κατάλληλη τιμή του  $x$ , **τον αριθμό 3**. Η επιλογή μας αυτή βασίζεται στο ότι με  $x=3$  έχουμε αρκετά καλά αποτελέσματα και για τις δύο περιπτώσεις που εξετάσαμε. Αν αποφασίζαμε κάτι διαφορετικό, για παράδειγμα  $x=5$  ο οποίος είναι ο αποδοτικότερος αριθμός για την περίπτωση όπου  $N_p < N_s$  τότε θα μειώναμε αισθητά την απόδοση του αλγορίθμου στην περίπτωση όπου  $N_p < N_s$ .

### 2.3.3.2 Αποτελέσματα προσομοιώσεων όταν μεταβάλουμε την πιθανότητα $P\_delete\_channel$ στον αλγόριθμο Build SS

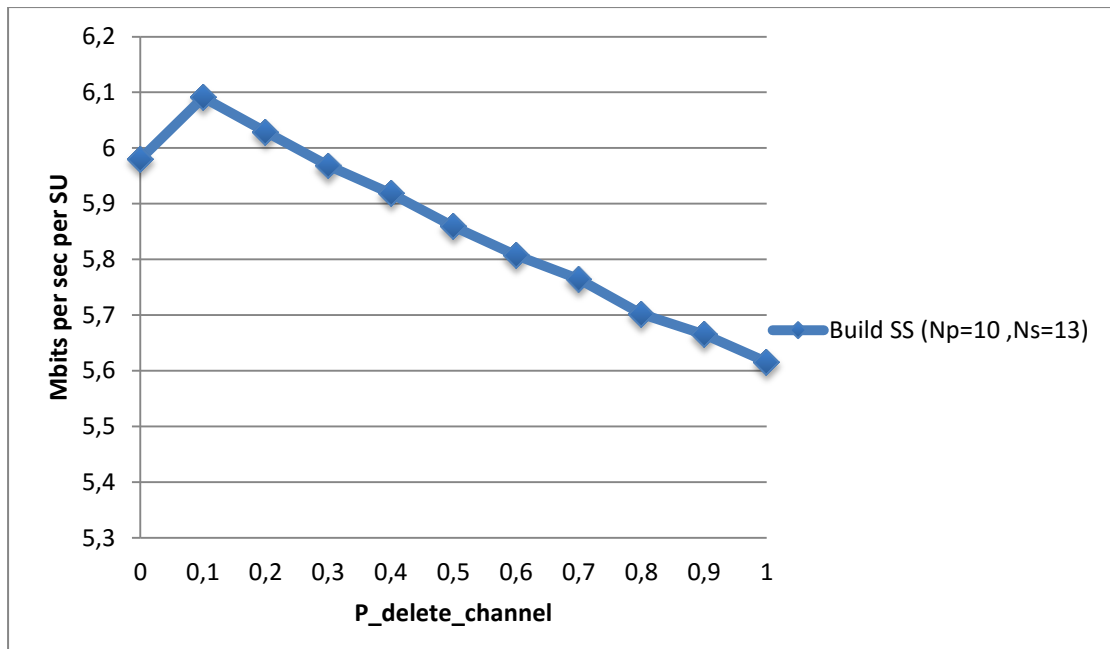
Υπενθυμίζουμε ότι με βάση την πιθανότητα αυτή ένας SU διαγράφει ή όχι τελικά το κανάλι που θεωρεί ότι πρέπει να διαγράψει. Στα σχήματα 2.6 και 2.8 έχουμε τα αποτελέσματα στην περίπτωση όπου  $N_p > N_s$  ( $N_p=10$ ,  $N_s=5$ ) ενώ στα 2.7, 2.9 και 2.10 έχουμε τα αποτελέσματα όταν  $N_s < N_p$  ( $N_p=10$ ,  $N_s=13$ ). Επίσης στα σχήματα 2.6a και 2.10b ισχύει ότι  $P_{fa}=0.0$  και  $P_{md}=0.0$ , στα 2.6b και 2.7 ότι  $P_{fa}=0.1$  και  $P_{md}=0.1$ , στα σχήματα 2.8 και 2.9  $P_{fa}=0.2$ ,  $P_{md}=0$ , και στο σχήμα 2.10a  $P_{fa}=0.1$ ,  $P_{md}=0.05$ . Τέλος, ο ρυθμός άφιξης πακέτων ανά slot είναι  $\lambda_i=0.3$  για τα σενάρια με χαμηλά φορτωμένο δίκτυο και  $\lambda_i=0.7$  για τα σενάρια με υψηλά φορτωμένο δίκτυο,  $i=1, \dots, N_p$ .



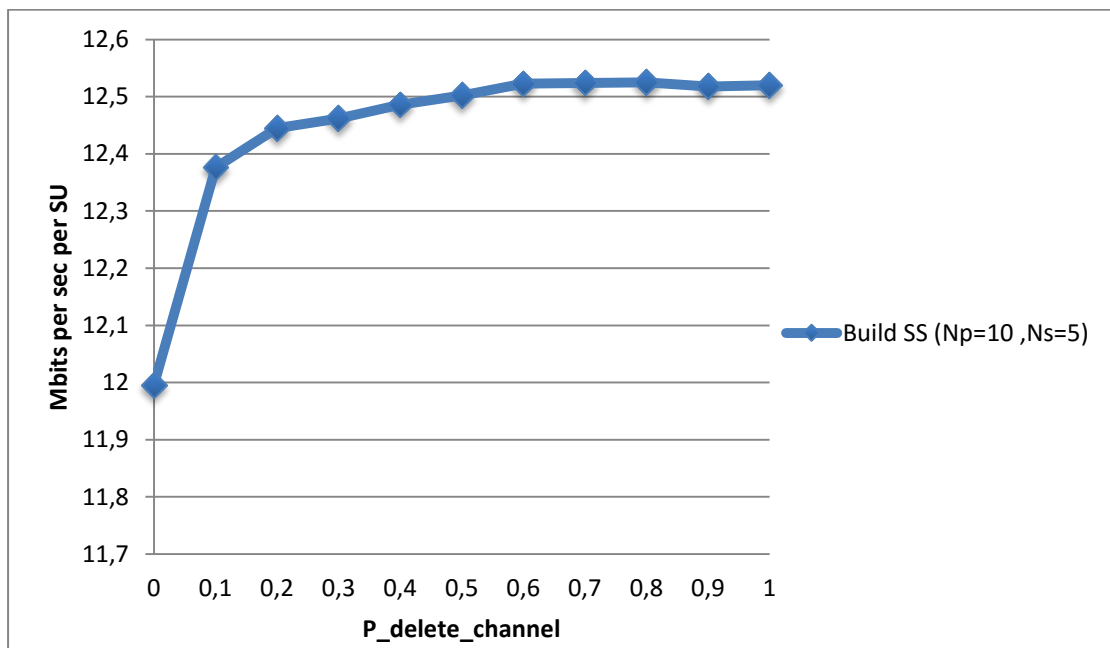
*Figure 2.6a Απόδοση αλγορίθμου Build SS σε χαμηλά φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας διαγραφής καναλιού.*



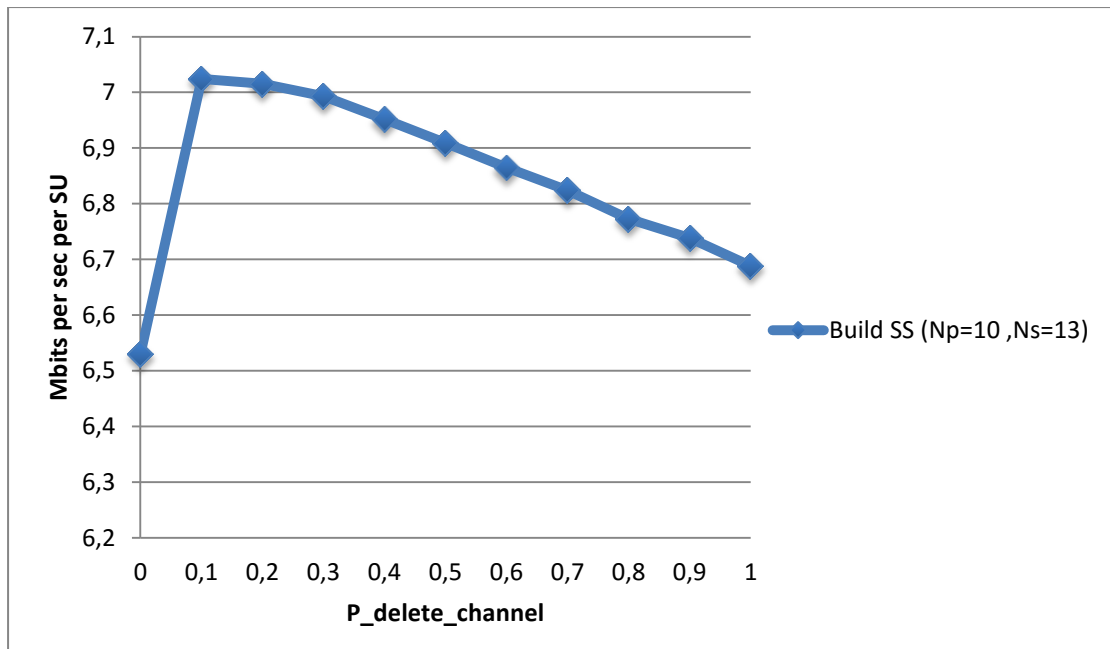
*Figure 2.6b Απόδοση αλγορίθμου Build SS σε υψηλά φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας διαγραφής καναλιού.*



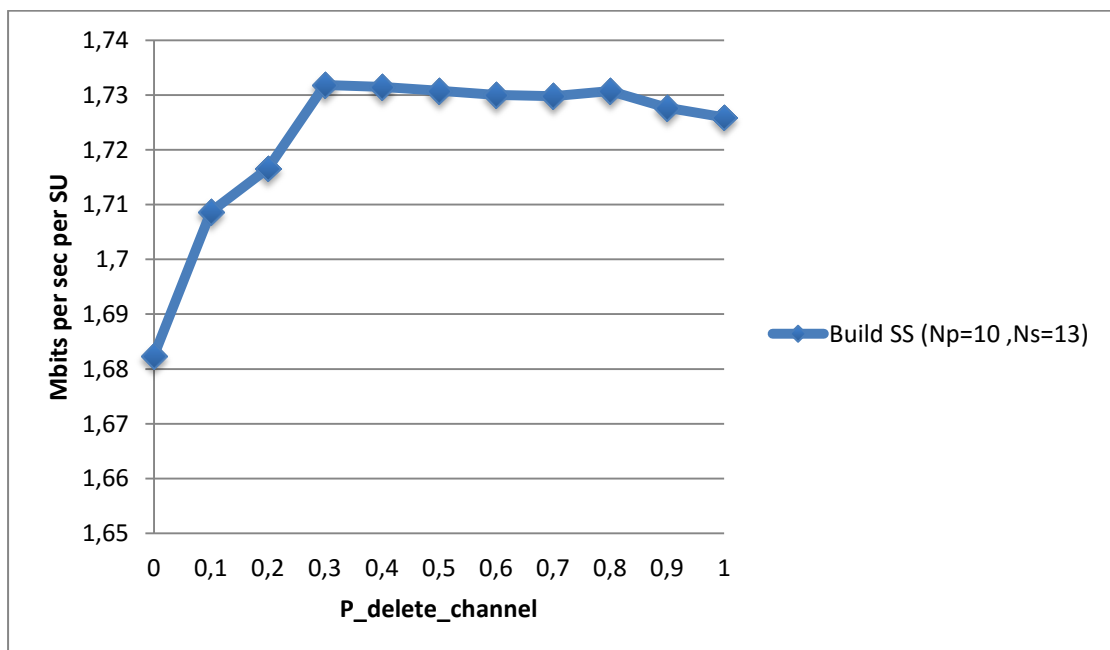
*Figure 2.7 Απόδοση αλγορίθμου Build SS σε χαμηλά φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας διαγραφής καναλιού.*



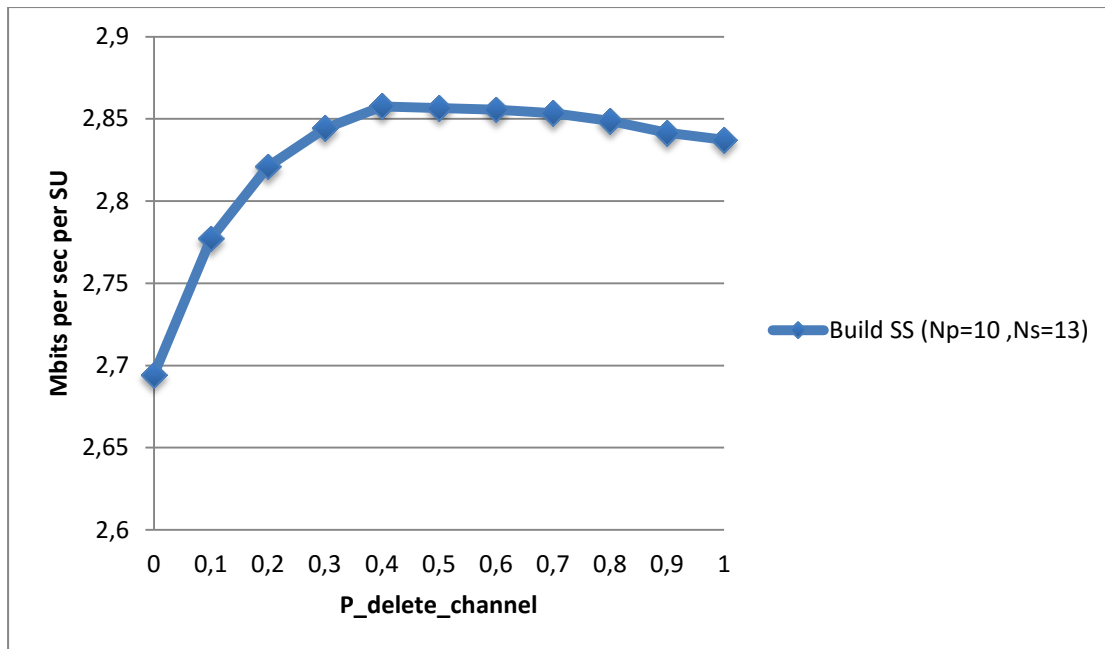
*Figure 2.8 Απόδοση αλγορίθμου Build SS σε χαμηλά φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας διαγραφής καναλιού.*



*Figure 2.9 Απόδοση αλγορίθμου Build SS σε χαμηλά φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας διαγραφής καναλιού.*



*Figure 2.10a Απόδοση αλγορίθμου Build SS σε υψηλά φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας διαγραφής καναλιού.*



**Figure 2.10b Απόδοση αλγορίθμου Build SS σε υψηλά φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας διαγραφής καναλιού.**

Από τα αποτελέσματα στα παραπάνω σχήματα παρατηρούμε ότι όταν τα κανάλια είναι περισσότερα από τους δευτερεύοντες χρήστες και το δίκτυο είναι χαμηλά φορτωμένο η κατάλληλη τιμή για την πιθανότητα διαγραφής καναλιού ( $P_{delete\_channel}$ ) είναι στο διάστημα 0.6-0.8, ενώ στην αντίθετη περίπτωση όπου τα κανάλια είναι λιγότερα από τους δευτερεύοντες χρήστες είναι στο διάστημα 0.1-0.2. Επίσης παρατηρούμε ότι όταν το δίκτυο είναι υψηλά φορτωμένο και τα κανάλια λιγότερα από τους δευτερεύοντες χρήστες η κατάλληλη τιμή για την πιθανότητα διαγραφής καναλιού είναι περίπου 0.3-0.5. Αυτό συμβαίνει γιατί σε αυτήν την περίπτωση τα κανάλια έχουν υψηλή κίνηση, με αποτέλεσμα να μην είναι επιθυμητό να παραμείνει ο χρήστης στο ίδιο κανάλι (αφού σε αυτό θα συγκρούεται με κάποιον άλλο δευτερεύοντα χρήστη), αλλά ούτε και να αλλάζει κανάλι με μεγάλη ευκολία, διότι με μεγάλη πιθανότητα θα βρίσκει κατειλημμένα κανάλια στη νέα αναζήτηση που θα επιχειρήσει, με αποτέλεσμα να μην καταφέρει εύκολα να αποθηκεύσει νέο κανάλι στη λίστα του. Αξίζει να σημειωθεί ότι η τιμή της πιθανότητας  $P_{delete\_channel}$  δεν επηρεάζει σημαντικά την απόδοση του συστήματος στην περίπτωση υψηλής κίνησης όταν οι δευτερεύοντες χρήστες είναι περισσότεροι από τα κανάλια, όπως φαίνεται από τα αποτελέσματα στα σχήματα 2.10.

Τα παραπάνω αποτελέσματα μας δείχνουν ότι όταν δεν υπάρχει υψηλός ανταγωνισμός μεταξύ των SUs καλό είναι τα κανάλια να διαγράφονται πιο συχνά και να αναζητούνται νέα. Αντίθετα, όταν υπάρχει υψηλός ανταγωνισμός, όταν δηλαδή πολλοί δευτερεύοντες χρήστες αναζητούν ευκαιρία μετάδοσης σε λιγότερα κανάλια με υψηλή κίνηση από τους πρωτεύοντες χρήστες, τότε καλό είναι να μην

εγκαταλείπεται εύκολα ένα κανάλι που έχει επιλεγεί, δεδομένου ότι η συχνή αναζήτηση νέων καναλιών από πολλούς SUs θα οδηγήσει σε περισσότερες συγκρούσεις (αφού  $N_p > N_s$ ).

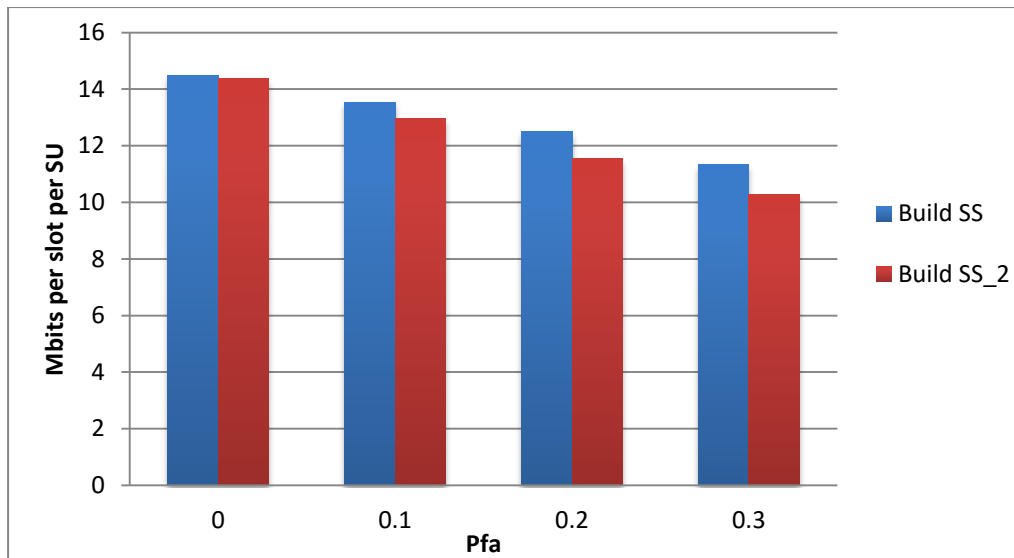
Να υπενθυμίσουμε εδώ ότι όταν ένα κανάλι υπάρχει στη λίστα ενός SU, σημαίνει ότι ο χρήστης έχει μεταδώσει επιτυχημένα σε αυτό, τη στιγμή που το αποθήκευσε. Επομένως, δεδομένου ότι κάθε δευτερεύων χρήστης στο σύστημα δεν μπορεί να γνωρίζει το συνολικό αριθμό των δευτερευόντων χρηστών, από τη στιγμή που αυτοί δεν συνεργάζονται, επιλέξαμε την πιθανότητα  $P_{\text{delete\_channel}}$  να είναι ίση με 0.45. Η επιλογή αυτή έγινε με γνώμονα ότι στις περιπτώσεις που προσομοιώσαμε και αντιπροσωπεύουν σχεδόν όλες τις καταστάσεις που μπορεί να αντιμετωπίσει το σύστημα, έχουμε τη μέγιστη απόδοση αναλόγως το σενάριο για τιμές της  $P_{\text{delete\_channel}}$  ίσες με 0.7, 0.8, 0.1, 0.7, 0.1, 0.3, 0.4 (με τη σειρά που δόθηκαν τα σχήματα). Άρα η μέση τιμή τους είναι περίπου το **0.45**.

### 2.3.3.3 Σύγκριση αλγορίθμων Build SS και Build SS\_2

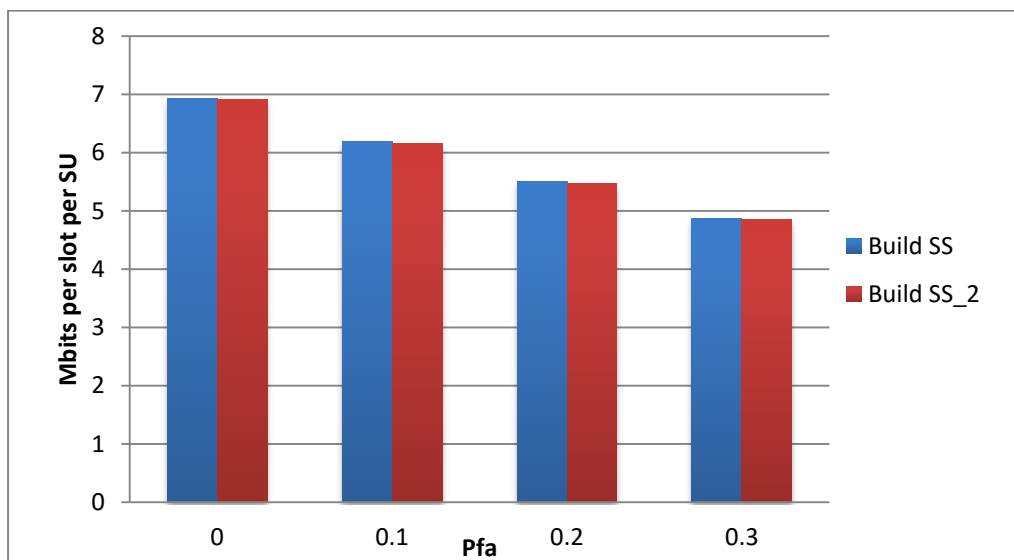
Η πιθανότητα λανθασμένης ανίχνευσης  $P_{\text{md}}$  θεωρείται ίση με 0 στα σχήματα όπου μεταβάλλεται η πιθανότητα λανθασμένου συναγερμού  $P_{\text{fa}}$  από 0 έως 0.3. Αντίστοιχα, στα σχήματα στα οποία μεταβάλλεται η πιθανότητα  $P_{\text{md}}$  από 0 έως 0.15 η πιθανότητα  $P_{\text{fa}}$  θεωρούμε ότι είναι ίση με 0.2. Επίσης ισχύει ότι  $N_p=10$  και  $N_s=5$  στην περίπτωση όπου  $N_p > N_s$  και  $N_p=10$ ,  $N_s=13$  στην περίπτωση όπου  $N_p < N_s$ .

#### 2.3.3.3.1 Αποτελέσματα προσομοιώσεων για $N_p > N_s$



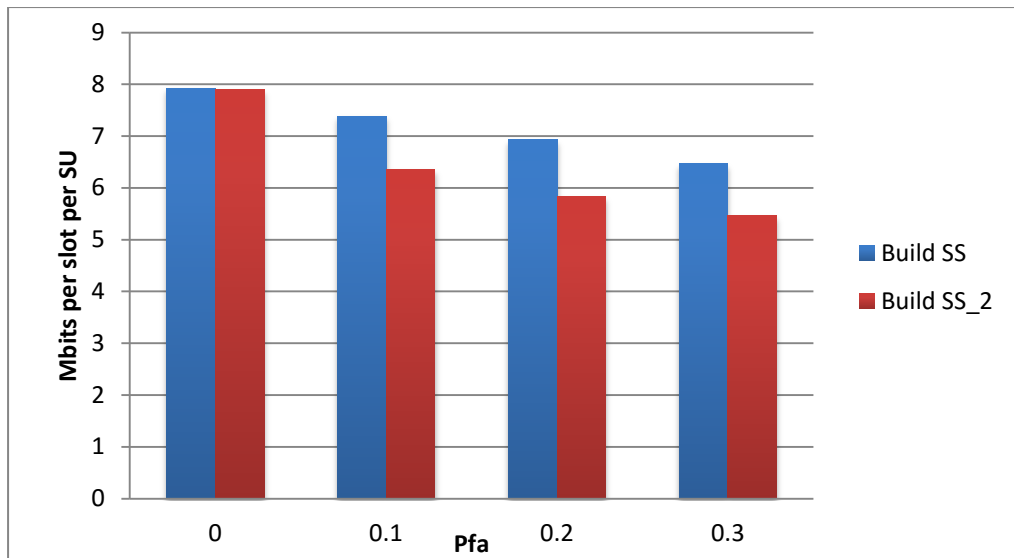


*Figure 2.11 Απόδοση αλγορίθμων σε χαμηλά φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας λανθασμένου συναγερμού  $P_{fa}$ .*

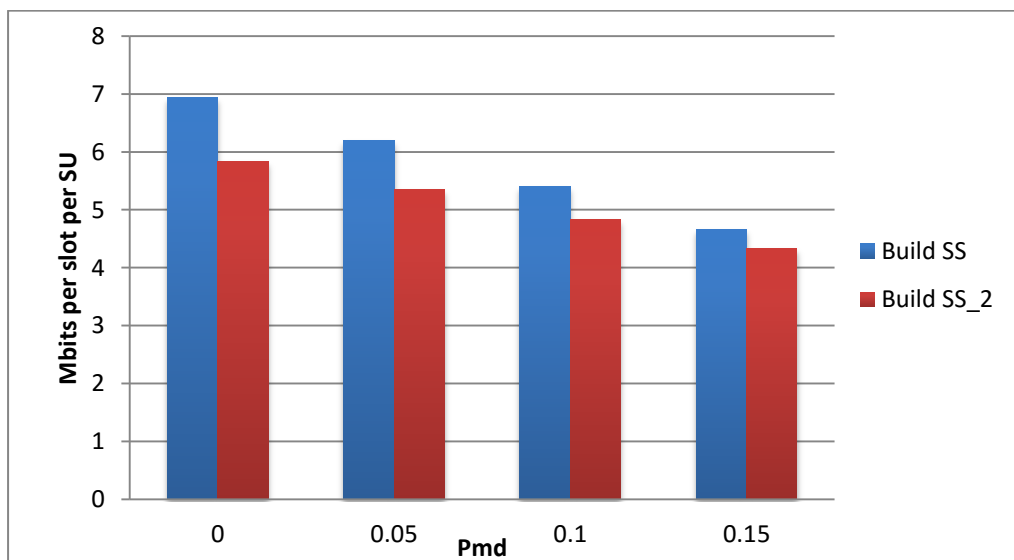


*Figure 2.12 Απόδοση αλγορίθμων σε υψηλά φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας λανθασμένου συναγερμού  $P_{fa}$ .*

#### 2.3.3.3.2 Αποτελέσματα προσομοιώσεων για $N_p < N_s$



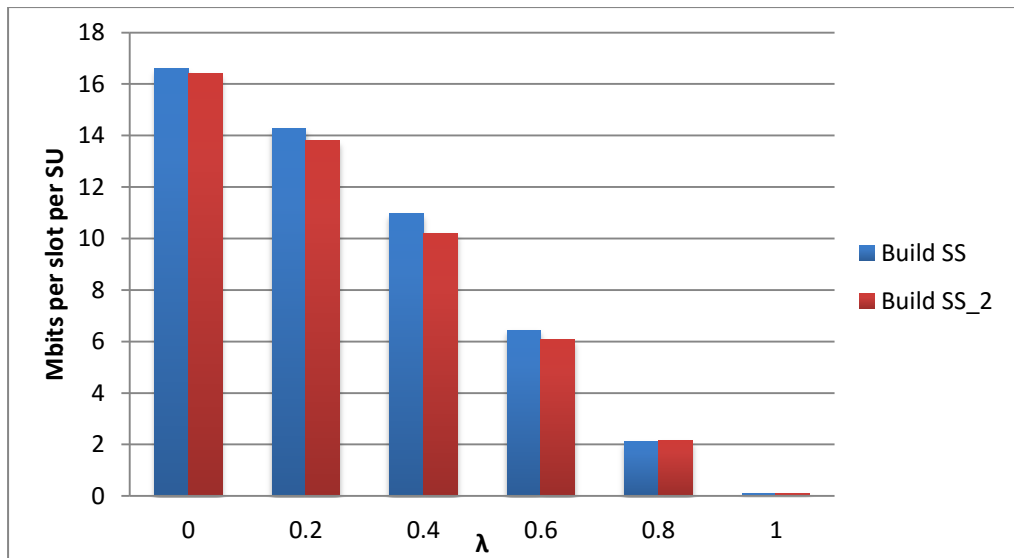
*Figure 2.13 Απόδοση αλγορίθμων σε χαμηλά φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας λανθασμένου συναγερμού  $P_{fa}$ .*



*Figure 2.14 Απόδοση αλγορίθμων σε χαμηλά φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας λανθασμένης ανίχνευσης  $P_{md}$ .*

### 2.3.3.3 Επίδραση μεταβολής του ρυθμού άφιξης πακέτων $\lambda$ των $P_{us}$

Εδώ εξάγουμε τις αποδόσεις των αλγορίθμων για διαφόρους ρυθμούς άφιξης πακέτων, που είναι ίση για όλα τα κανάλια. Έχουμε υποθέσει ότι η πιθανότητα λανθασμένου συναγερμού  $P_{fa}$  και η πιθανότητα λανθασμένης ανίχνευσης  $P_{md}$  είναι ίσες με 0.1. Επίσης ισχύει  $N_p=10$ ,  $N_s=5$ .



**Figure 2.15** Απόδοση αλγορίθμων σε χαμηλά φορτωμένο δίκτυο ως προς το ρυθμό άφιξης πακέτων PU.

#### 2.3.3.3.4 Σχολιασμός αποτελεσμάτων σύγκρισης αλγορίθμων Build SS και Build SS\_2

Από τα παραπάνω σχήματα βλέπουμε ότι η απόδοση του αλγορίθμου Build SS\_2 είναι χειρότερη από του Build SS. Ένα αρνητικό που θεωρούμε ότι έχει ο αλγόριθμος Build SS\_2 είναι ότι μπορεί ένα κανάλι, π.χ το 2 (αν το SS του SU1 είναι [2 3 0]), να είναι αποδοτικό για το δευτερεύοντα χρήστη 1 αλλά μία φορά αυτός να το δει ως απασχολημένο. Τότε αν ο χρήστης μεταδώσει επιτυχώς στο 3, το κανάλι 2 χάνει την πρώτη θέση στη λίστα με αποτέλεσμα ο SU να προσπαθεί πάντα αρχικά να μεταδώσει επιτυχώς στο κανάλι 3 (καθώς το το SS θα γίνει [3 2 0]). Κάποιος άλλος χρήστης ενδέχεται να επιλέξει τυχαία το κανάλι 2 στην πρώτη ευκαιρία μετάδοσης, να μεταδώσει σε αυτό επιτυχώς και ο προηγούμενος SU που πλέον έχει το κανάλι 2 στη δεύτερη θέση του SS του, να το βλέπει πάντα ως απασχολημένο άρα και ως μη αποδοτικό, με σίγουρο αποτέλεσμα τη διαγραφή του καναλιού από το SS του χρήστη αυτού σε κάποιο από τα επόμενα slots. Με λίγα λόγια σε αυτό τον αλγόριθμο, ο SU πολύ εύκολα αλλάζει θέση σε κάθε κανάλι στο SS του, γεγονός που ίσως να μην είναι αποδοτικό.

Επίσης άλλο ένα πρόβλημα του αλγορίθμου Build SS\_2 είναι ότι ο SU αναγκάζεται κατά κάποιο τρόπο να σπαταλάει πιο πολλές ευκαιρίες μετάδοσης σε σχέση με τον Build SS από τη στιγμή που ξεκινάνε οι διαγραφές. Για παράδειγμα αν η λίστα κάποιου δευτερεύοντα χρήστη περιέχει τα κανάλια [3 2 4] και κριθεί ότι πρέπει να διαγράψει το κανάλι 3, τότε με τον αλγόριθμο Build SS και με τον Build SS\_2 η λίστα θα γίνει [2 4 0] και [0 2 4], αντίστοιχα. Έτσι στο επόμενο slot με τον Build SS ο χρήστης θα προσπαθήσει να μεταδώσει στο κανάλι 2, και με τον Build SS\_2 θα ψάξει να βρει νέο κανάλι. Όπως γίνεται κατανοητό στην πρώτη περίπτωση όπου ο SU

επιχειρεί μετάδοση στο κανάλι 2 η πιθανότητα να είναι επιτυχημένη, είναι μεγαλύτερη από την περίπτωση όπου αναζητεί νέο κανάλι, για το λόγο του ότι αφού έχει αποθηκεύσει το κανάλι 2 στη λίστα του σημαίνει ότι **επέτυχε κάποτε μετάδοση** σε αυτό, και εφόσον δεν το έχει διαγράψει ακόμα από τη λίστα, θεωρείται καλό κανάλι για να τον εξυπηρετήσει. Αντίθετα, στην περίπτωση του αλγορίθμου Build SS\_2 στον οποίο ο SU θα αναζητήσει νέο κανάλι, ειδικά αν υπάρχει υψηλός ανταγωνισμός μεταξύ δευτερευόντων χρηστών, η πιθανότητα να βρει νέο κανάλι και να έχει επιτυχημένη μετάδοση είναι μικρή.

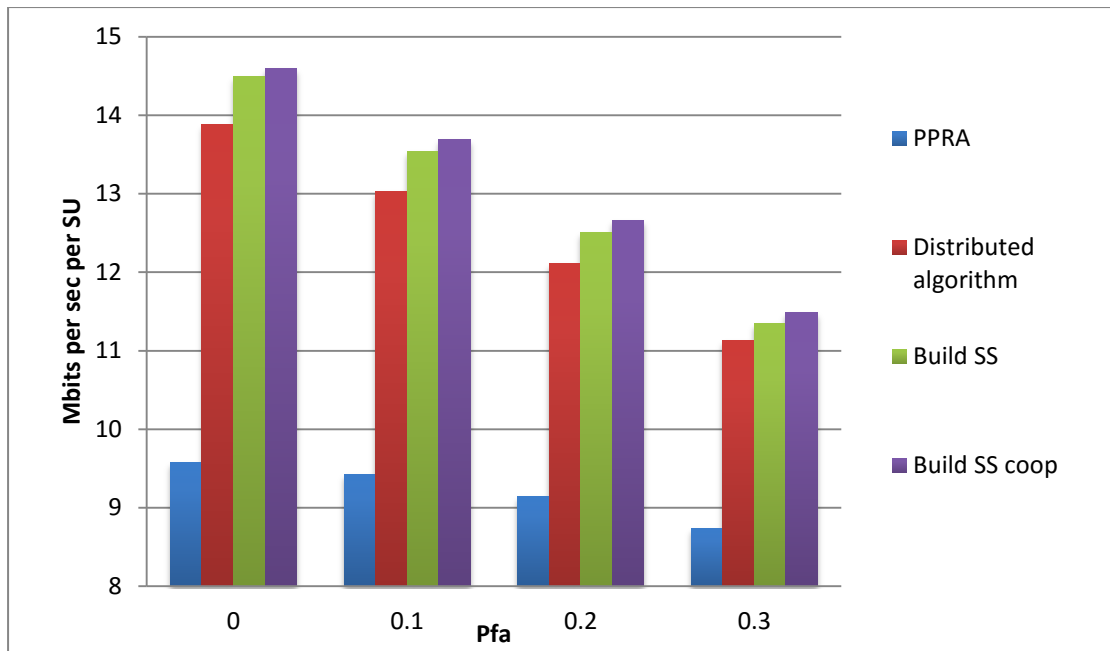
Από τις προσομοιώσεις παρατηρούμε ότι αναφέραμε και παραπάνω. Ο αλγόριθμος Build SS υπερτερεί του αλγορίθμου Build\_SS\_2. Η κυρία αίτια είναι ότι αλλάζει σχετικά εύκολα το κανάλι που βρίσκεται πρώτο στη λίστα ενώ μπορεί να μετέδιδε επιτυχημένα σε αυτό σε πολλά slot. Έτσι ο αλγόριθμος που θα συγκινούμε με άλλους αλγόριθμους της βιβλιογραφίας είναι ο Build SS.

#### 2.3.3.4 Αποτελέσματα προσομοιώσεων για $N_p > N_s$

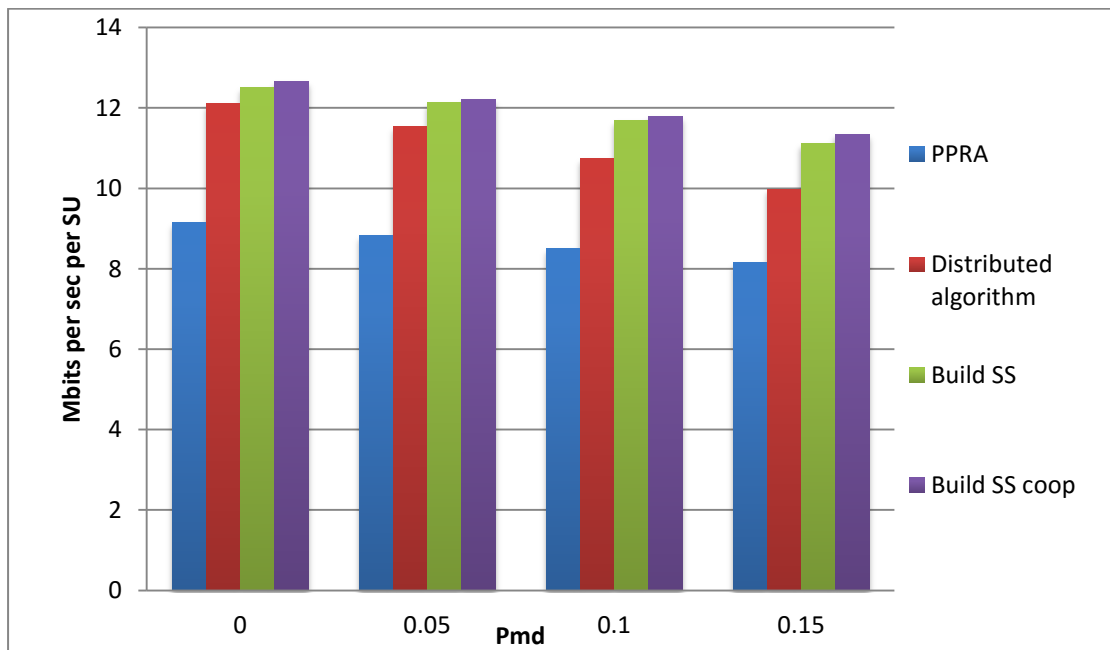
Υπενθυμίζουμε ότι σε αυτήν την περίπτωση έχουμε υποθέσει ότι ο αριθμός των καναλιών είναι 10 και των δευτερευόντων χρηστών είναι 5, δηλαδή  $N_p=10$ ,  $N_s=5$ . Η πιθανότητα λανθασμένης ανίχνευσης  $P_{md}$  θεωρούμε ότι είναι ίση με 0 στα σχήματα όπου μεταβάλλεται η πιθανότητα λανθασμένου συναγερμού  $P_{fa}$  από 0 έως 0.3. Αντίστοιχα, στα σχήματα όπου μεταβάλλεται η  $P_{md}$  από 0 έως 0.15 η  $P_{fa}$  θεωρούμε ότι είναι ίση με 0.2.

##### 2.3.3.4.1 Περίπτωση όπου τα κανάλια είναι ελαφρώς φορτωμένα

Εδώ υποθέτουμε ότι ο Poisson ρυθμός άφιξης πακέτων ανά slot στα κανάλια είναι ίσος με  $\lambda_i=0.3$  packet / slot,  $i=1,...,N_p$ .



*Figure 2.16 Απόδοση αλγορίθμων σε χαμηλά φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας λανθασμένου συναγερμού  $P_{fa}$ .*



*Figure 2.17 Απόδοση αλγορίθμων σε χαμηλά φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας λανθασμένης ανίχνευσης  $P_{md}$ .*

### 2.3.3.4.2 Περίπτωση όπου τα κανάλια είναι υψηλά φορτωμένα

Εδώ υποθέτουμε ότι ο Poisson ρυθμός άφιξης πακέτων ανά slot στα κανάλια είναι ίσος με  $\lambda_i = 0.7$  packet / slot,  $i=1, \dots, N_p$ .

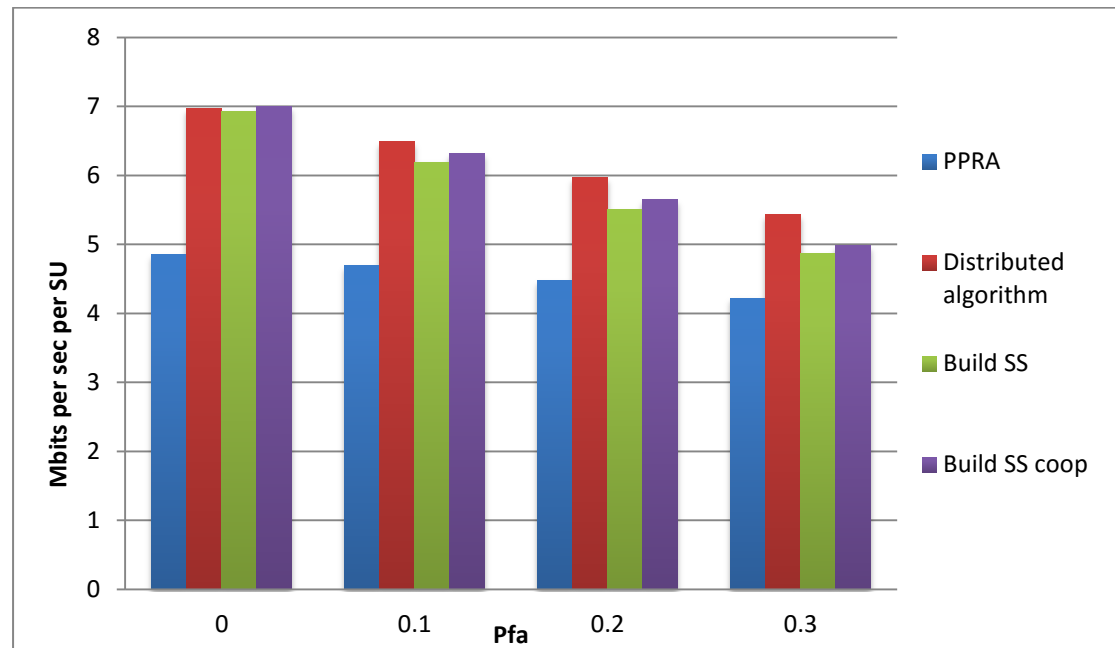


Figure 2.18 Απόδοση αλγορίθμων σε υψηλά φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας λανθασμένου συναγερμού  $P_{fa}$ .

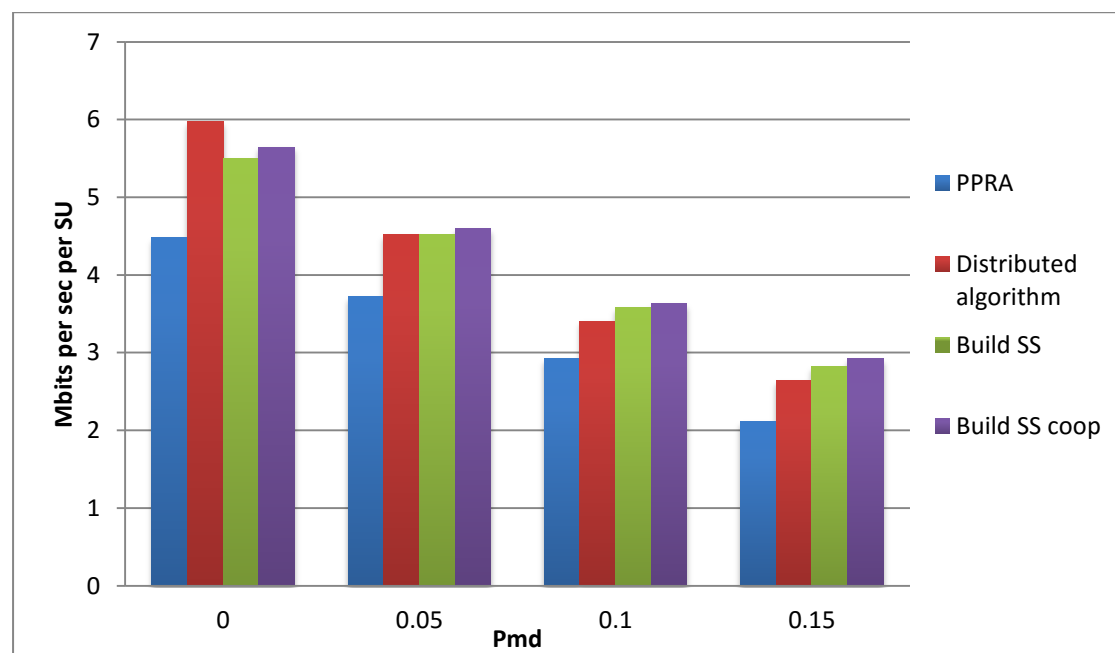


Figure 2.19 Απόδοση αλγορίθμων σε υψηλά δίκτυο σαν συνάρτηση της πιθανότητας λανθασμένης ανίχνευσης  $P_{md}$ .

### 2.3.3.4.3 Περίπτωση όπου τα κανάλια είναι άνισα φορτωμένα

Υποθέτουμε ότι ο Poisson ρυθμός άφιξης πακέτων ανά slot για κάθε κανάλι  $i$ ,  $i=1,\dots,N_p$  παίρνει τις παρακάτω τιμές:

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

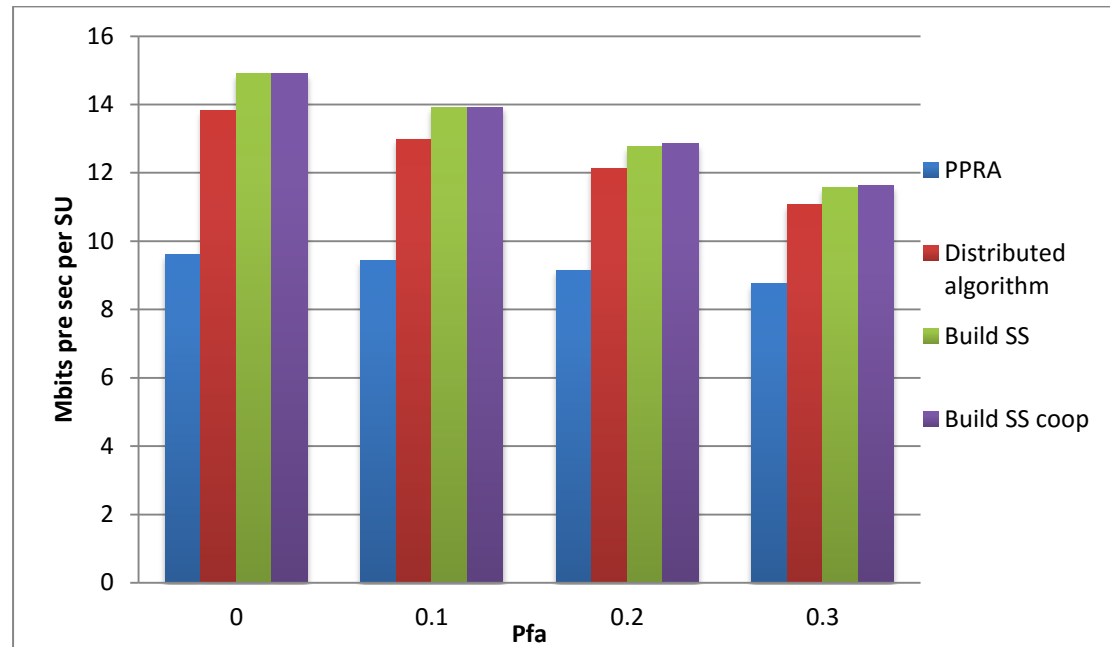


Figure 2.20 Απόδοση αλγορίθμων σε άνισα φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας λανθασμένου συναγερμού  $P_{fa}$ .

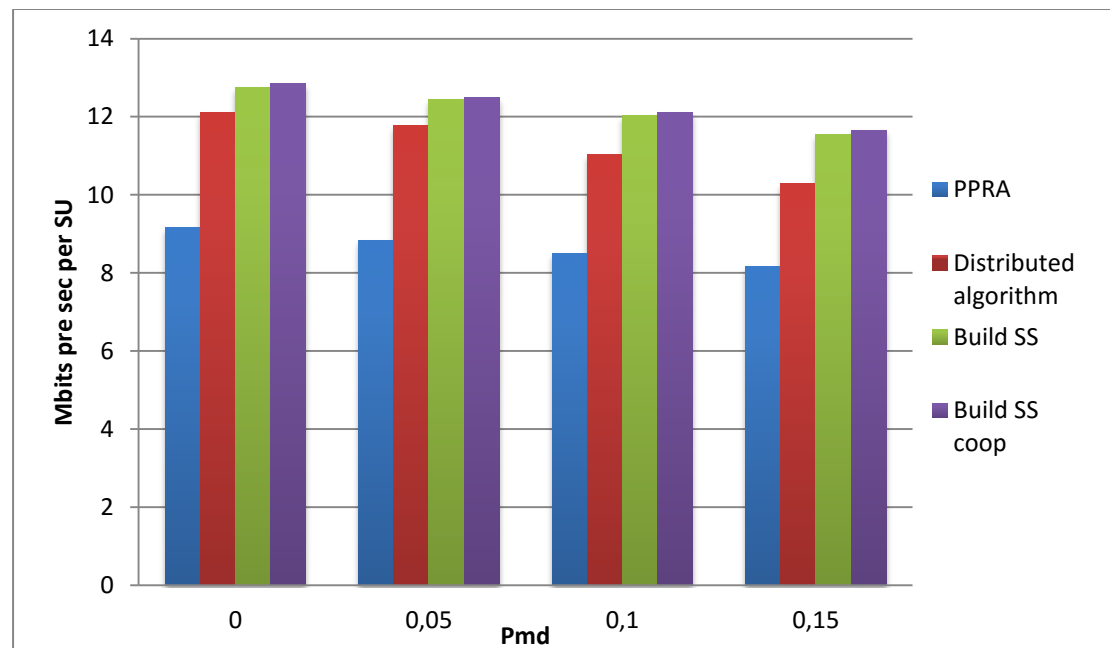


Figure 2.21 Απόδοση αλγορίθμων σε άνισα φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας λανθασμένης ανίχνευσης  $P_{md}$ .

#### 2.3.3.4.4 Επίδραση μεταβολής του ρυθμού άφιξης πακέτων $\lambda$ των PUs

Εδώ παρουσιάζουμε τις αποδόσεις των αλγορίθμων για διαφορετικές τιμές του Poisson ρυθμού άφιξης πακέτων  $\lambda$ , που υποτίθεται ο ίδιος για όλα τα κανάλια. Στο παρακάτω σχήμα έχουμε υποθέσει ότι η πιθανότητα λανθασμένου συναγερμού  $P_{fa}$  και η πιθανότητα λανθασμένης ανίχνευσης  $P_{md}$  είναι και οι δύο ίσες με 0.1.

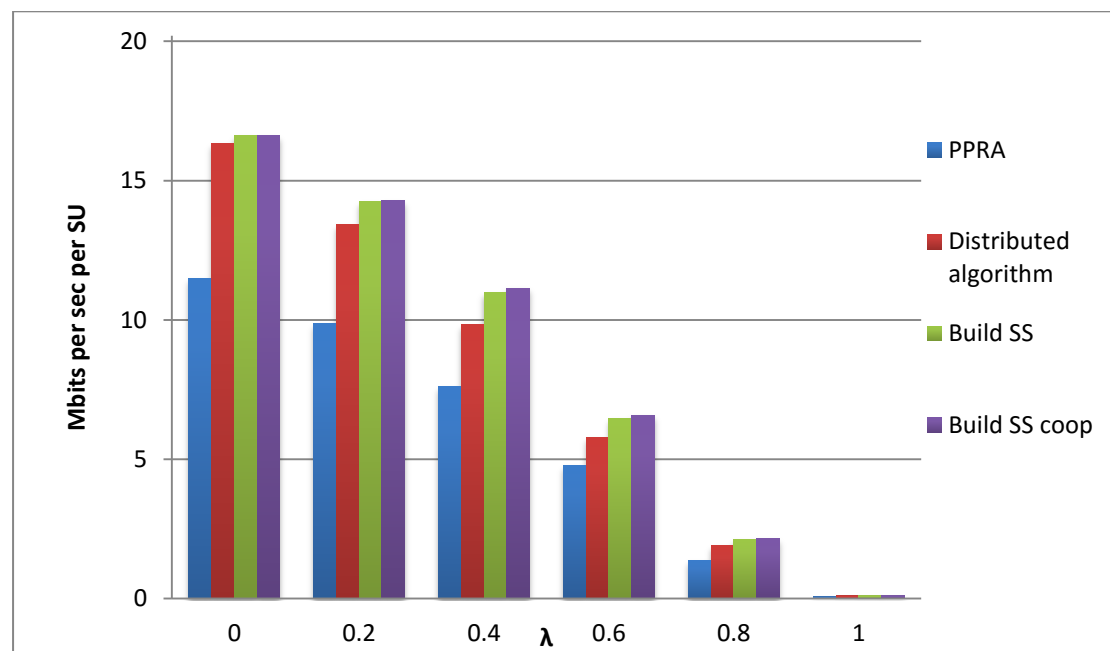


Figure 2.22 Απόδοση αλγορίθμων σαν συνάρτηση του ρυθμού άφιξης πακέτων  $\lambda$  των PUs.

#### 2.3.3.4.5 Σχολιασμός αποτελεσμάτων για την περίπτωση όπου $N_p > N_s$

Από τα αποτελέσματα των παραπάνω προσομοιώσεων για την περίπτωση όπου  $N_p > N_s$  βλέπουμε ότι ο αλγόριθμος Build SS είναι αποδοτικότερος από τον Distributed και τον PPRA στις περισσότερες περιπτώσεις. Όσον αφορά στη σύγκριση του Build SS με τον PPRA, είναι αναμενόμενο να είναι καλύτερος ο Build SS δεδομένου ότι ο PPRA επιλέγει τυχαία κανάλια, και δεν έχει λίστα να τα τοποθετήσει ώστε να τα αξιοποιήσει στο μέλλον. Στη συνέχεια γίνεται η σύγκριση του αλγορίθμου Build SS με τον Distributed.

Στα σχήματα 2.16, 2.17 όπου το δίκτυο είναι χαμηλά φορτωμένο, είναι φανερό ότι υπερσχύει ο αλγόριθμος Build SS. Αυτό οφείλεται στο γεγονός ότι από τη στιγμή που δεν υπάρχει υψηλός ανταγωνισμός στο δίκτυο (έχουμε περισσότερα κανάλια από



δευτερεύοντες χρήστες αλλά και μικρή κίνηση στα κανάλια αυτά προερχόμενη από τους πρωτεύοντες χρήστες), ο Build SS μεταδίδει επιτυχημένα με μεγάλη πιθανότητα σε κάποιο κανάλι και το τοποθετεί στη λίστα του. Αντίθετα ο Distributed παίρνει τη λίστα καναλιών από τον Latin Square πίνακα χωρίς να έχει μεταδώσει επιτυχημένα σε αυτά. Παράλληλα στον Build SS ο χρήστης μπορεί να διαχειρίζεται ξεχωριστά το κάθε κανάλι αφού αυτό μπορεί να τοποθετηθεί και να διαγράφει από τη λίστα ανεξάρτητα από τα άλλα κανάλια. Αντίθετα στον Distributed ο SU επιλέγει τη λίστα των καναλιών από τον Latin Square πίνακα μετά από τυχαία επιλογή. Επίσης στον Distributed αλγόριθμο ο SU διαγράφει μαζικά όλα τα κανάλια της λίστας του όταν έχει δύο συνεχόμενες συγκρούσεις σε οποιοδήποτε από αυτά (όχι απαραίτητα στο ίδιο).

Όσον αφορά τα αποτελέσματα στα σχήματα 2.18 και 2.19, παρατηρούμε ότι ο Distributed αλγόριθμος συμπεριφέρεται καλύτερα όταν το δίκτυο είναι υψηλά φορτωμένο και η πιθανότητα λανθασμένης ανίχνευσης  $P_{md}$  είναι 0. Αυτό οφείλεται στο γεγονός ότι στον Build SS αλγόριθμο ο δευτερεύων χρήστης δεν βρίσκει εύκολα μη απασχολημένο κανάλι για να μεταδώσει (αφού το δίκτυο είναι υψηλά φορτωμένο) και ως αποτέλεσμα δεν τοποθετεί κάποιο κανάλι στη λίστα του. Έτσι αυξάνεται ο ανταγωνισμός των δευτερευόντων χρηστών, καθώς αναζητούν συχνά νέο κανάλι. Αντίθετα ο Distributed αλγόριθμος σε αυτήν την περίπτωση παραμένει περισσότερο σε μία σειρά από κανάλια, αφού την τοποθετεί αυτομάτως στη λίστα του, και έτσι οι χρήστες δεν οδηγούνται σε τόσο συχνές τυχαίες αναζητήσεις νέου καναλιού.

Πιο συγκεκριμένα από το σχήμα 2.18 βλέπουμε ότι όσο μεγαλώνει η πιθανότητα λανθασμένου συναγερμού ( $P_{fa}$ ) τόσο καλύτερος γίνεται ο Distributed αλγόριθμος σε σχέση με τον Build SS αλγόριθμο. Στον αλγόριθμο Build SS, αυξάνοντας την πιθανότητα  $P_{fa}$ , αυξάνονται οι χαμένες ευκαιρίες για μετάδοση και μειώνεται η πιθανότητα ο χρήστης να αποθηκεύσει τελικά κάποιο κανάλι στη λίστα του. Αντίθετα ο Distributed αλγόριθμος δεν αντιμετωπίζει αυτό το πρόβλημα, καθώς και έχει πάντα κανάλια στη λίστα του, δίχως αυτό να εξαρτάται από το αν πέτυχε μετάδοση σε αυτά.

Από τα αποτελέσματα στο σχήμα 2.19 παρατηρούμε ότι όσο μεγαλώνει η πιθανότητα λανθασμένης ανίχνευσης, **οπότε αυξάνονται και οι συγκρούσεις μεταξύ των δευτερευόντων χρηστών**, τόσο μεγαλώνει η απόδοση του αλγόριθμου Build SS σε σχέση με τον Distributed αλγόριθμο, για το λόγο ότι ο Build SS έχει πιο σύνθετους μηχανισμούς για την αντιμετώπιση των συγκρούσεων, αφού μπορεί να διαγράψει κάθε κανάλι ξεχωριστά. Όπως είπαμε και πριν, στον Distributed αλγόριθμο διαγράφεται ολόκληρη η λίστα των καναλιών και έτσι δημιουργείται υψηλός ανταγωνισμός μεταξύ των SUs στην αναζήτηση νέου καναλιού.

Από τα αποτελέσματα στα σχήματα 2.20 και 2.21 όπου η κίνηση στα κανάλια είναι άνιση, με τη μέση κίνηση να είναι και πάλι μικρή, βλέπουμε ότι ο Build SS αλγόριθμος αποδίδει καλύτερα, για τους λόγους που αναφέρθηκαν και παραπάνω στα σενάρια με χαμηλή κίνηση. Επίσης, από τα αποτελέσματα στο σχήμα 2.22 όπου υπάρχει θετική πιθανότητα λανθασμένης ανίχνευσης ο Build SS υπερσχύει και πάλι

του Distributed για όλες τις τιμές των ρυθμών άφιξης των καναλιών. Αξίζει να σημειωθεί ότι ο Build SS έχει μηχανισμούς για την αντιμετώπιση του φαινομένου όπου ένα κανάλι είναι συνεχώς κατειλημμένο, κάτι που ο Distributed δεν έχει. Όταν τα κανάλια έχουν διαφορετικούς ρυθμούς άφιξης το ένα από το άλλο, ο μηχανισμός που αναφέραμε παραπάνω έχει σημαντική επίδραση στην αύξηση της απόδοσης του συστήματος.

Σε σύγκριση τώρα του αλγορίθμου Build SS με τον Build SS coop βλέπουμε ότι ο Build SS coop αλγόριθμος υπερσχύει σχεδόν πάντα. Αυτό οφείλεται στο γεγονός ότι με τον Build SS coop αλγόριθμο μόλις πραγματοποιηθεί μία σύγκρουση, οι χρήστες επικοινωνούν και μέσω του συντονιστή, ακριβώς ένας δευτερεύων χρήστης κρατάει το κανάλι που έγινε η σύγκρουση, στην ίδια θέση της λίστας του. Οι υπόλοιποι δευτερεύοντες χρήστες (αν υπάρχουν) το διαγράφουν. Όταν έχουμε χαμηλή κίνηση είναι αποδοτικό να αναζητούνται σχετικά εύκολα νέα κανάλια, γεγονός που συμβαίνει συχνότερα στον Build SS coop αλγόριθμο, για λόγους που έχουν αναλυθεί στις εξηγήσεις των αλγορίθμων αυτών. Έτσι είναι δεδομένο ότι το κανάλι που οδηγούσε σε συνεχόμενες συγκρούσεις μεταξύ των SUs, δεν θα μείνει ανεκμετάλλευτο στην επομένη ευκαιρία μετάδοσης στο επόμενο slot, αν τότε είναι ελεύθερο από τον πρωτεύοντα χρήστη. Στον Build SS αλγόριθμο υπάρχει η πιθανότητα να κρατήσουν το κανάλι στη λίστα τους πάνω από δύο χρήστες, γεγονός που θα οδηγήσει σε σύγκρουση στο επόμενο slot, αν το αισθανθούν όλοι ελεύθερο (καθώς υπάρχει η πιθανότητα  $P_{fa}$ ).

### 2.3.3.5 Αποτελέσματα προσομοιώσεων για $N_p < N_s$

Εδώ υποθέτουμε ότι ο αριθμός των καναλιών είναι ίσος με 10 και των δευτερευόντων χρηστών είναι ίσος με 13 ,δηλαδή  $N_p=10$ ,  $N_s=13$ . Η πιθανότητα λανθασμένης ανίχνευσης  $P_{md}$  υποτίθεται ίση με 0 στα σχήματα όπου μεταβάλλεται η πιθανότητα λανθασμένου συναγερμού  $P_{fa}$  από 0 έως 0,3. Αντίστοιχα στα σχήματα όπου μεταβάλλεται η  $P_{md}$  από 0 έως 0,15 η πιθανότητα  $P_{fa}$  θεωρείται ίση με 0.2.

#### 2.3.3.5.1 Περίπτωση όπου τα κανάλια είναι ελαφρώς φορτωμένα

Εδώ υποθέτουμε ότι ο Poisson ρυθμός άφιξης πακέτων ανά slot στα κανάλια είναι ίσος με  $\lambda_i=0.3$  packet / slot,  $i=1,...,N_p$ .

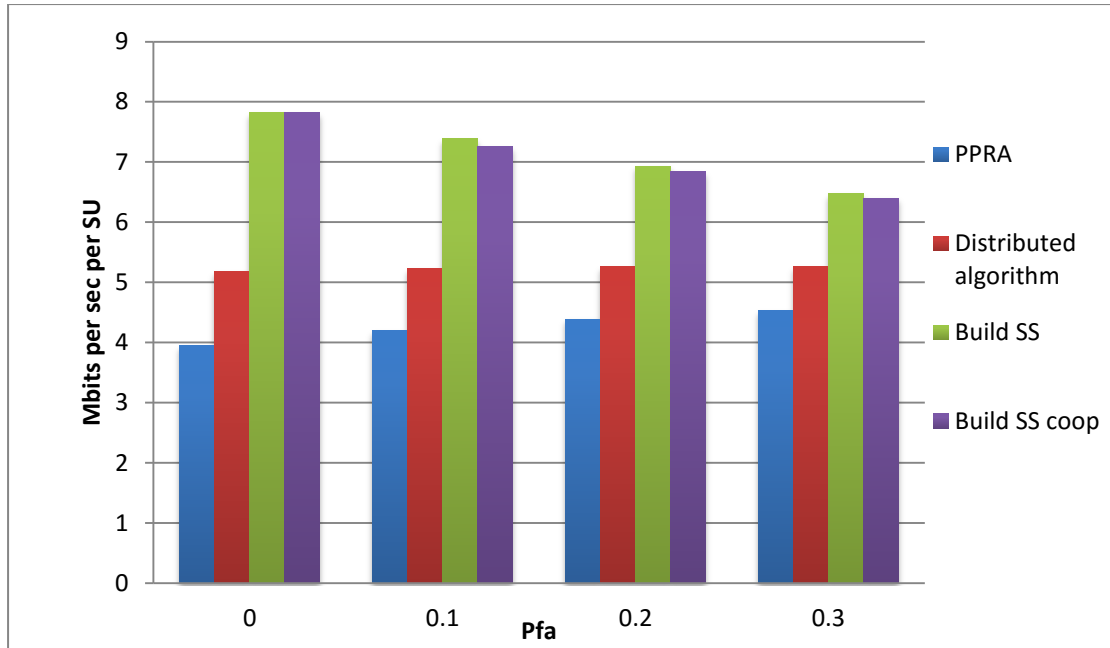


Figure 2.23 Απόδοση αλγορίθμων σε χαμηλά φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας λανθασμένου συναγερμού  $P_{fa}$ .

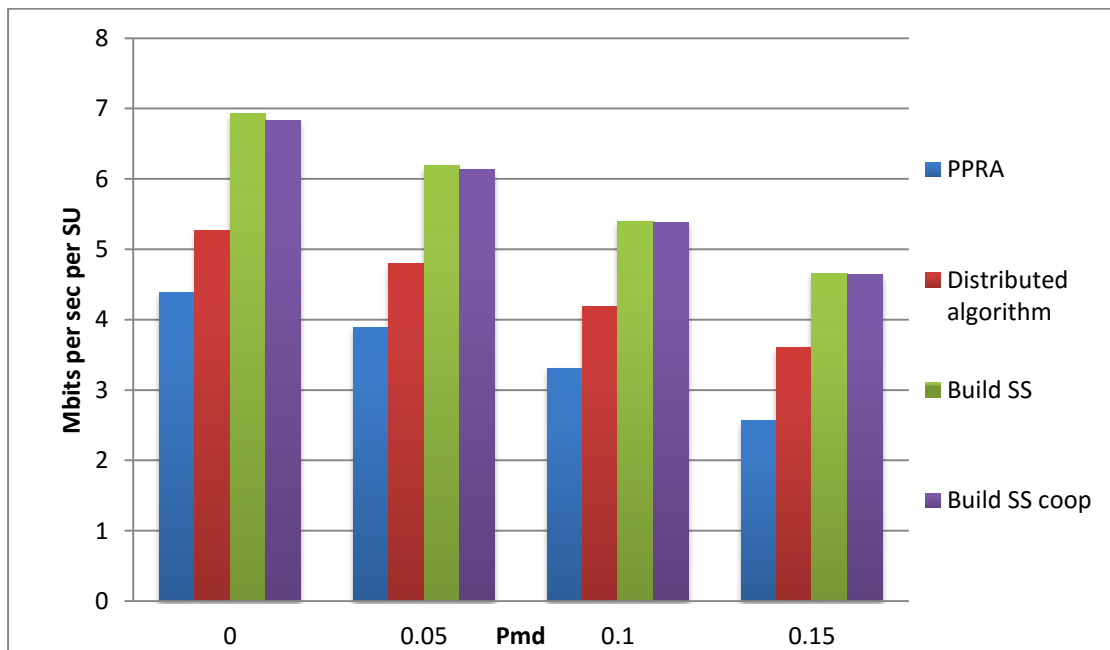


Figure 2.24 Απόδοση αλγορίθμων σε χαμηλά φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας λανθασμένης ανίχνευσης  $P_{md}$ .

### 2.3.3.5.2 Περίπτωση όπου τα κανάλια είναι υψηλά φορτωμένα

Εδώ υποθέτουμε ότι ο Poisson ρυθμός άφιξης πακέτων ανά slot στα κανάλια είναι ίσος με  $\lambda_i = 0.7$  packet / slot,  $i=1, \dots, N_p$ .

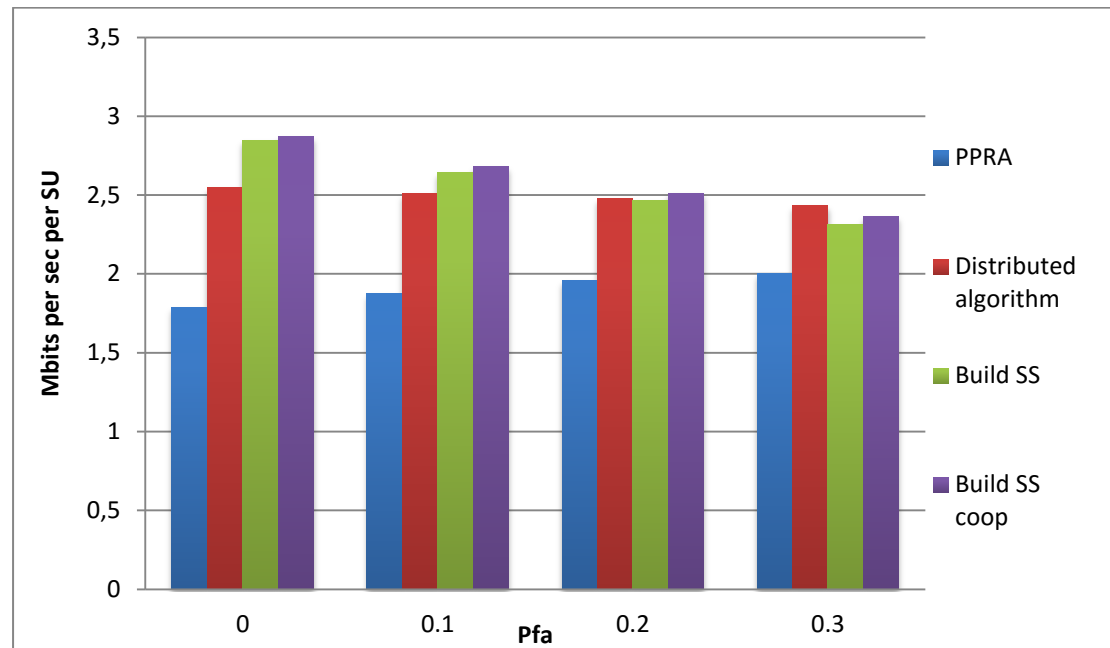


Figure 2.25 Απόδοση αλγορίθμων σε υψηλά φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας λανθασμένου συναγερμού  $P_{fa}$ .

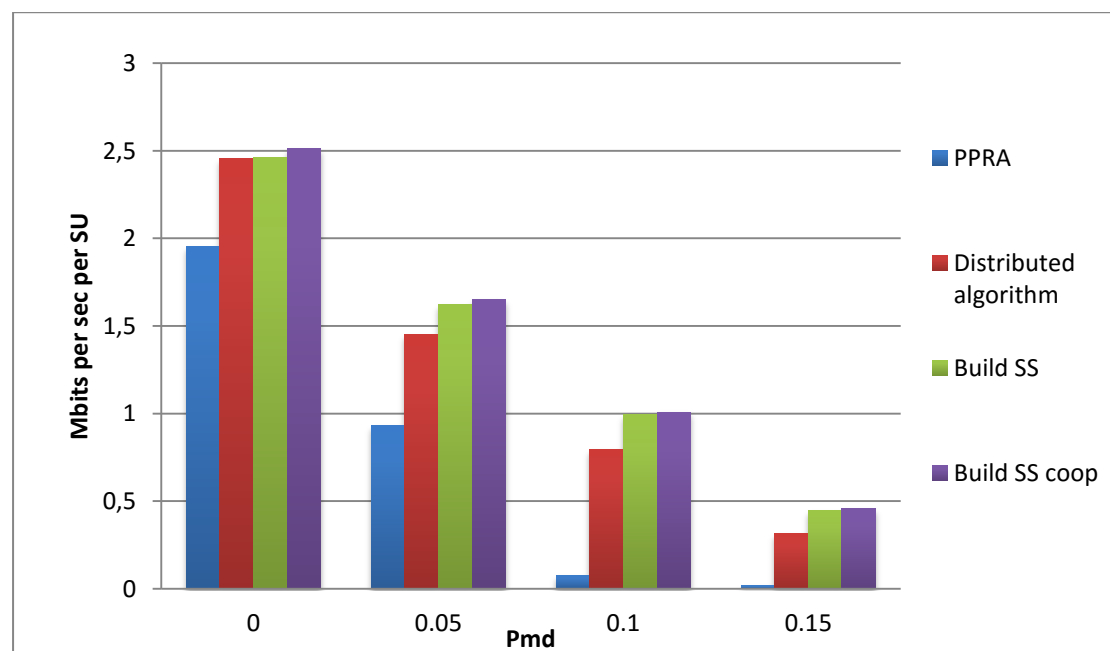
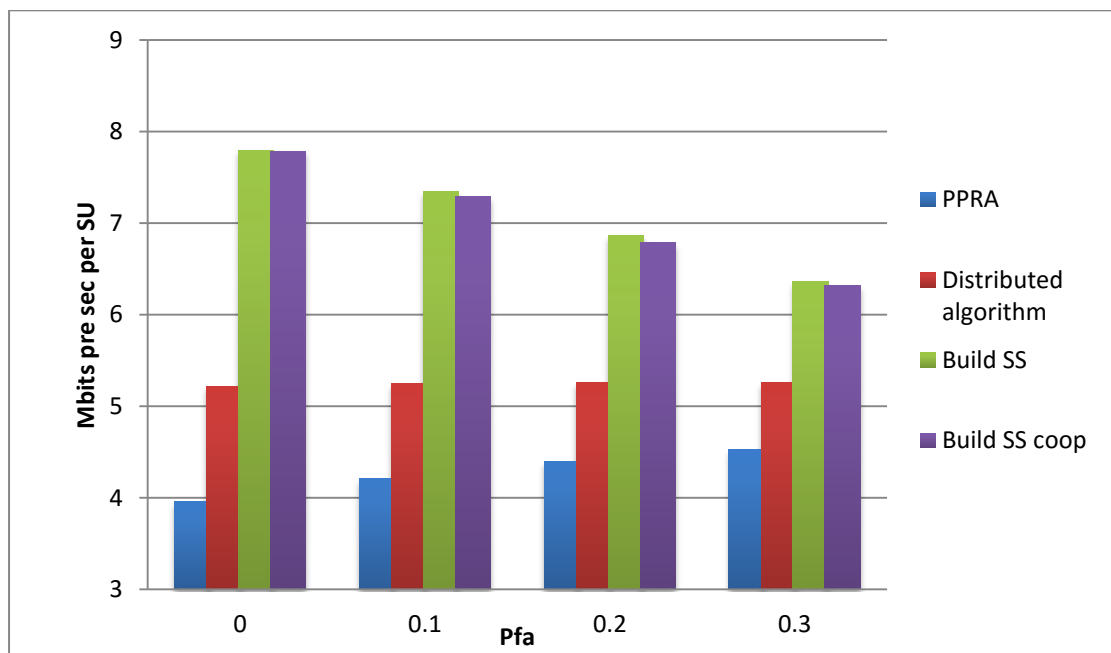


Figure 2.26 Απόδοση αλγορίθμων σε υψηλά φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας λανθασμένης ανίχνευσης  $P_{md}$ .

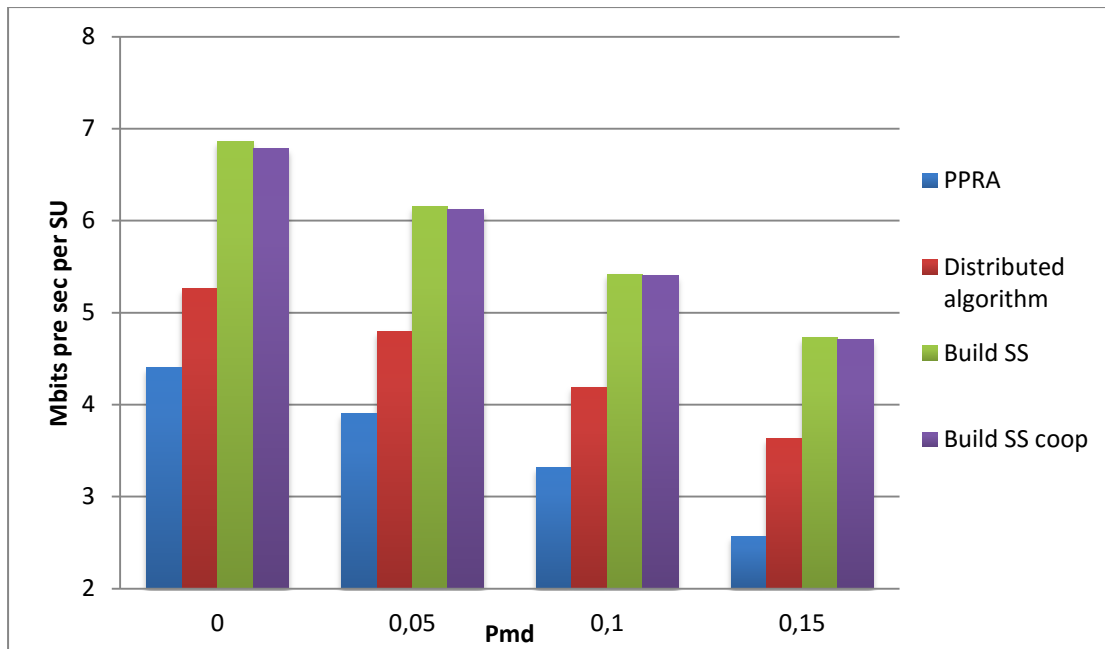
### 2.3.3.5.3 Περίπτωση όπου τα κανάλια είναι άνισα φορτωμένα

Υποθέτουμε ότι ο Poisson ρυθμός άφιξης πακέτων ανά slot για κάθε κανάλι  $i$ ,  $i=1,\dots,N_p$  παίρνει τις παρακάτω τιμές:

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



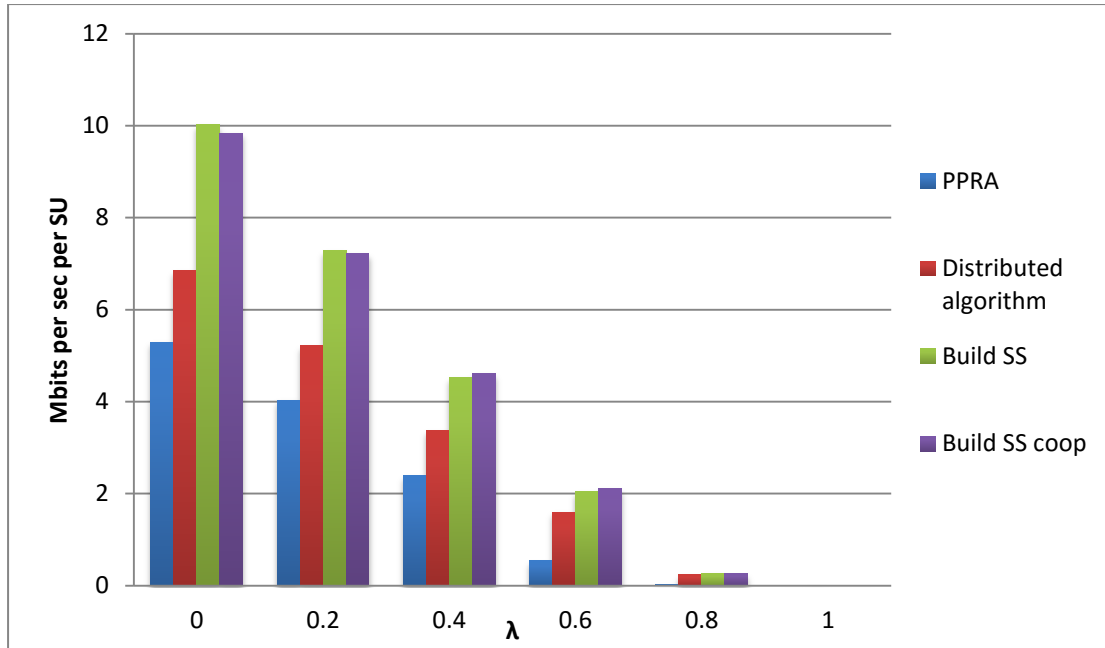
*Figure 2.27 Απόδοση αλγορίθμων σε άνισα φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας λανθασμένου συναγερμού  $P_{fa}$ .*



*Figure 2.28 Απόδοση αλγορίθμων σε άνισα φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας λανθασμένης ανίχνευσης  $P_{md}$ .*

#### 2.3.3.5.4 Επίδραση μεταβολής του ρυθμού άφιξης πακέτων $\lambda$ των $P_{us}$

Εδώ παρουσιάζουμε τις αποδόσεις των αλγορίθμων για διαφορετικές τιμές του Poisson ρυθμού άφιξης πακέτων  $\lambda$ , που υποτίθεται ο ίδιος για όλα τα κανάλια. Στο παρακάτω σχήμα έχουμε υποθέσει ότι η πιθανότητα λανθασμένου συναγερμού  $P_{fa}$  και η πιθανότητα λανθασμένης ανίχνευσης  $P_{md}$  είναι και οι δύο ίσες με 0.1.



**Figure 2.29** Απόδοση αλγορίθμων σαν συνάρτηση του ρυθμού άφιξης πακέτων  $\lambda$  των PUs.

#### 2.3.3.5.5 Σχολιασμός αποτελεσμάτων για την περίπτωση όπου $N_p < N_s$

Από τα αποτελέσματα των προσομοιώσεων παραπάνω για το σενάριο όπου  $N_s > N_p$  βλέπουμε ότι ο Build SS είναι αποδοτικότερος αλγόριθμος των Distributed και PPRA στις περισσότερες περιπτώσεις. Όσον αφορά τη σύγκριση του Build SS με τον PPRA, είναι αναμενόμενο ο πρώτος να είναι καλύτερος, μιας και ο PPRA επιλέγει τυχαία κανάλια και δεν έχει λίστα να τα τοποθετήσει για να αξιοποιήσει στο τέλος. Παρακάτω γίνεται η σύγκριση του αλγορίθμου Build SS με τον Distributed.

Από τα αποτελέσματα στα σχήματα 2.23, 2.24 όπου το δίκτυο είναι χαμηλά φορτωμένο είναι φανερό ότι υπερσχύει ο αλγόριθμος Build SS. Ο Build SS διαγράφει από τη λίστα του SU μεμονωμένα κανάλια, μετά από τρεις συνεχόμενες συγκρούσεις σε τρία συνεχόμενα slots ενώ ο Distributed διαγράφει όλη τη λίστα του έπειτα από δύο συνεχόμενες συγκρούσεις. Όπως έχουμε ήδη αναλύσει, στην περίπτωση όπου  $N_s > N_p$  είναι προτιμότερο να μη διαγράφονται εύκολα κανάλια γιατί οδηγούμαστε σε υψηλό ανταγωνισμό δευτερευόντων χρηστών στο δίκτυο. Έτσι λοιπόν ο μηχανισμός του Build SS για το πότε πρέπει να εγκαταλειφθεί κάποιο κανάλι αποδίδει καλύτερα, σε σχέση με αυτόν του Distributed που διαγράφει ολόκληρη τη λίστα και την εγκαταλείπει σχετικά νωρίς (μετά από δύο συνεχόμενες συγκρούσεις σε όποιο κανάλι και αν έγιναν, ενώ ο Build SS μετά από τρεις συνεχόμενες συγκρούσεις στο ίδιο κανάλι).

Από τα αποτελέσματα στο σχήμα 2.25 παρατηρούμε ότι ο Distributed συμπεριφέρεται καλύτερα από τον Build SS όταν το δίκτυο είναι υψηλά φορτωμένο,

η πιθανότητα λανθασμένης ανίχνευσης  $P_{md}$  είναι 0 και η πιθανότητα λανθασμένου συναγερμού μεγαλύτερη του 0.1. Αυτό οφείλεται στο γεγονός που αναλύθηκε και στην περίπτωση όπου  $N_p > N_s$ , φαινόμενο που αυξάνεται όσο μεγαλώνει η  $P_{fa}$  αφού οι χρήστες χάνουν πιο πολλές ευκαιρίες μετάδοσης ανιχνεύοντας λανθασμένα το κανάλι ως απασχολημένο. Από τα αποτελέσματα στο σχήμα 2.26, παρατηρούμε ότι όσο μεγαλώνει η πιθανότητα λανθασμένης ανίχνευσης, **οπότε αυξάνονται και οι συγκρούσεις μεταξύ των δευτερευόντων χρηστών**, γίνεται καλύτερος ο αλγόριθμος Build SS από τον Distributed, για το λόγο του ότι ο Build SS έχει πιο έξυπνους μηχανισμούς για την αντιμετώπιση των συγκρούσεων όπως τους αναλύσαμε και παραπάνω, σε σχέση με τον Distributed αλγόριθμο.

Από τα αποτελέσματα στα σχήματα 2.27 και 2.28 όπου η κίνηση στα κανάλια είναι άνιση, με τη μέση κίνηση να είναι και πάλι μικρή, βλέπουμε ότι ο Build SS αποδίδει καλύτερα, για τους λογούς που αναφέρθηκαν και παραπάνω στα σενάρια με χαμηλή κίνηση. Επίσης στο σχήμα 2.29 όπου η πιθανότητα λανθασμένου συναγερμού είναι ίση με 0.1, ο Build SS υπερσχύει και πάλι του Distributed για όλες τις τιμές των ρυθμών άφιξης των καναλιών.

Σε σύγκριση τώρα του αλγορίθμου Build SS με τους Build SS coop βλέπουμε ότι ο Build SS coop υπερσχύει σε κάποιες περιπτώσεις του Build SS αλλά όχι πάντα. Αυτό οφείλεται στο γεγονός ότι με τον Build SS coop αλγόριθμο μόλις πραγματοποιηθεί μία σύγκρουση, οι χρήστες επικοινωνούν μεταξύ τους και ακριβώς ένας δευτερεύων χρήστης κρατάει το κανάλι που έγινε η σύγκρουση στην ίδια θέση της λίστας του. Οι υπόλοιποι δευτερεύοντες χρήστες (αν υπάρχουν) το διαγράφουν. Έτσι, αφού υπάρχει υψηλός ανταγωνισμός, οι χρήστες που θα το διαγράψουν θα αναζητήσουν ξανά νέο κανάλι, με μεγάλη πιθανότητα να διαλέξουν κανάλι που κάποιος άλλος χρήστης ήδη το έχει στη λίστα του και θα οδηγηθούν σε νέες συγκρούσεις. Στον Build SS όπου υπάρχει η πιθανότητα να κρατήσουν το κανάλι πάνω από δύο χρήστες, μπορεί να οδηγηθούν λιγότεροι SUs σε αναζήτηση νέου καναλιού, σε σχέση με τον Build SS coop. Βλέπουμε ότι εδώ τα αποτελέσματα μας είναι αντίθετα με αυτά του σεναρίου όπου  $N_s < N_p$ .

Στα σχήματα όπου έχουμε υψηλή κίνηση, οδηγούμαστε σε λιγότερες συγκρούσεις ανάμεσα σε δευτερεύοντες χρήστες, μιας και οι αυτοί δύσκολα θα εντοπίσουν φασματικό κενό για να μεταδώσουν. Πλέον οι δευτερεύοντες χρήστες αλλάζουν συχνά κανάλι βάσει του μηχανισμού διαγραφής καναλιού όταν το κανάλι είναι απασχολημένο. Έτσι είναι δύσκολο να είναι πολλοί δευτερεύοντες χρήστες στο ίδιο κανάλι για να συγκρουστούν. Για αυτό το λόγο ο Build SS coop λειτουργεί αποδοτικότερα, αφού θα είναι λιγότεροι οι SUs που θα συγκρουστούν, ένας από αυτούς θα το κρατήσει και οι υπόλοιποι (που πιθανότατα δεν θα είναι πολλοί) θα αναζητήσουν νέο κανάλι, αν αυτό χρειαστεί.

Τέλος αξίζει να σχολιαστεί το φαινόμενο ότι στα σχήματα στα οποία μεταβάλλεται η  $P_{fa}$  οι αλγόριθμοι PPRA και Distributed, αυξάνουν την απόδοση τους όσο αυξάνεται η πιθανότητα λανθασμένου συναγερμού ( $P_{fa}$ ). Αυτό οφείλεται στο γεγονός ότι έχουμε



υψηλό ανταγωνισμό (περισσότερους δευτερεύοντες χρήστες από κανάλια). Έτσι είναι πολύ πιθανό παραπάνω από ένας χρήστες να επιλέξουν το ίδιο κανάλι στην ίδια ευκαιρία μετάδοσης. Όσο αυξάνεται η πιθανότητα λανθασμένου συναγερμού, τόσο αυξάνεται η πιθανότητα οι χρήστες να ανιχνεύσουν ένα κανάλι λανθασμένα ως απασχολημένο. Όταν ένας μόνο χρήστης το διαγνώσει σωστά ως άδειο, θα μεταδώσει σε αυτήν την ευκαιρία μετάδοσης επιτυχημένα στο συγκεκριμένο κανάλι (εφόσον οι άλλοι το θεωρούν λανθασμένα ως απασχολημένο). Στην περίπτωση που η πιθανότητα λανθασμένου συναγερμού τείνει στο 0, όλοι οι χρήστες θα διαγνώσουν σωστά το κανάλι ως άδειο και θα προσπαθήσουν να μεταδώσουν σε αυτό, οπότε θα οδηγηθούν με σιγουριά σε σύγκρουση. Το φαινόμενο αυτό δεν συμβαίνει στους αλγορίθμους τύπου Build SS, γιατί ο δευτερεύων χρήστης δεν αποθηκεύει το κανάλι στη λίστα του παρά μόνο όταν μεταδώσει επιτυχημένα σε αυτό. Επομένως όσο μεγαλώνει η πιθανότητα  $P_{fa}$  οι SUs οδηγούνται σε περισσότερες χαμένες ευκαιρίες μετάδοσης άρα και σε μικρότερη απόδοση.

## ΚΕΦΑΛΑΙΟ 3<sup>ο</sup>: Σχεδιασμός και αξιολόγηση αποδόσεων των προτεινόμενων κατανεμημένων αλγορίθμων για αίσθηση καναλιών σε CRNs με ετερογενή κανάλια

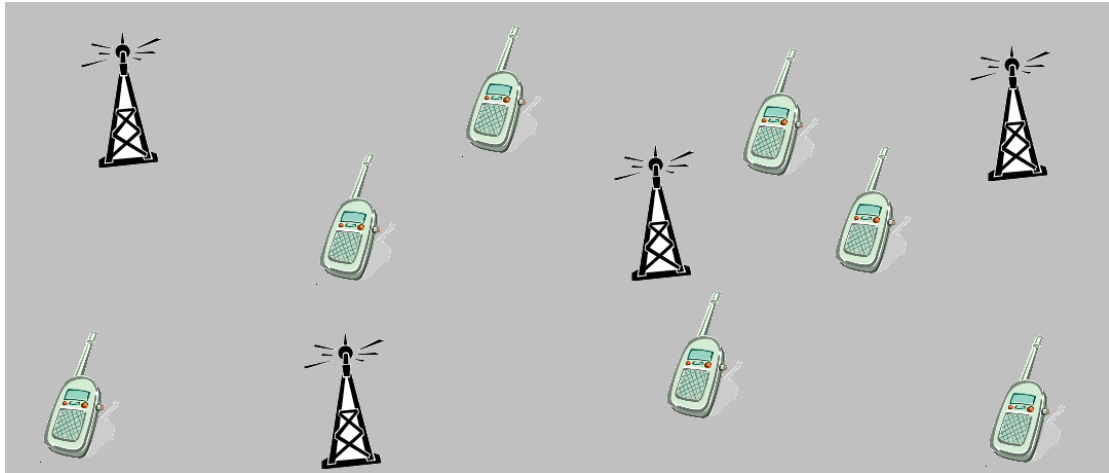
### 3.1 Εισαγωγή

Οι χρήστες σε ένα πραγματικό δίκτυο όπως αυτό στο σχήμα 3.1, είναι διασκορπισμένοι στο χώρο. Αν υποθέσουμε ότι όλοι μπορούν να ‘ακούσουν’ το σήμα του πρωτεύοντα πομπού τότε είναι λογικό αυτοί που βρίσκονται κοντύτερα να το λαμβάνουν με μεγαλύτερη ισχύ από αυτούς που απέχουν περισσότερο. Επίσης βάσει του θεωρήματος Shannon-Fano η χωρητικότητα ενός καναλιού εξαρτάται από το SNR (Signal to Noise Ratio), άρα και από την απόσταση πομπού και δέκτη. Έτσι λοιπόν, παρακάτω θα αναπτύξουμε αλγορίθμους σε σενάρια όπου οι χρήστες (PUs - SUs) έχουν διαφορετικές αποστάσεις μεταξύ τους. Στο Κεφάλαιο 2 είχαμε θεωρήσει ότι για όλα τα κανάλια ισχύει το ίδιο channel gain, δηλαδή ότι οι δευτερεύοντες χρήστες λαμβάνουν την ίδια ισχύ σήματος από όλους τους πρωτεύοντες χρήστες.

Πλέον κάνοντας περισσότερο ρεαλιστικό το σενάριο που εξετάζουμε, δεν απέχουν όλοι οι δευτερεύοντες πομποί την ίδια απόσταση από όλους τους πρωτεύοντες πομπούς και πρωτεύοντες δέκτες. Αυτό οδηγεί τον κάθε δευτερεύοντα χρήστη (πομπό) να ‘βλέπει’ διαφορετική χωρητικότητα για κάθε κανάλι, η οποία εξαρτάται από την απόσταση που έχει από τον αντίστοιχο πρωτεύοντα δέκτη. Εδώ έχουμε υποθέσει ότι οι δευτερεύοντες πομποί μεταδίδουν τα δεδομένα τους στους πρωτεύοντες δέκτες. Θεωρούμε ότι κάθε πρωτεύων δέκτης μπορεί στη συνέχεια να επικοινωνεί και να μεταβιβάζει τα δεδομένα που έλαβε από κάθε δευτερεύοντα πομπό προς τον αντίστοιχο δευτερεύοντα δέκτη.

Πιο συγκεκριμένα, προτείνουμε αλγορίθμους στους οποίους οι αποφάσεις των SUs για το ποιά κανάλια θα τοποθετήσουν στη λίστα τους, δηλαδή σε ποιά κανάλια θα προσπαθούν να βρουν φασματικά κενά για να μεταδώσουν, λαμβάνονται βάσει της χωρητικότητας που οι SUs εκτιμάνε ότι έχει κάθε κανάλι.

Η ατελής αίσθηση του φάσματος και σε αυτό το κεφάλαιο μοντελοποιείται από τις πιθανότητες λανθασμένης ανίχνευσης και λανθασμένου συναγερμού [1]. Αναπαριστούμε μέσω κατάλληλων μοντέλων για την πιθανότητα λανθασμένης ανίχνευσης ( $P_{md}$ ) την περίπτωση όπου, κάθε δευτερεύων χρήστης μπορεί να ανιχνεύσει με μεγαλύτερη σιγουριά την ύπαρξη άλλου χρήστη (πομπού) στο φάσμα όσο πιο κοντά βρίσκεται σε αυτόν, καθώς έχουμε υποθέσει ότι όλοι οι χρήστες (πομποί) έχουν την ίδια εκπεμπόμενη ισχύ. Έτσι όσο μεγαλύτερη είναι η απόσταση μεταξύ των δύο πομπών, τόσο μεγαλύτερη είναι και η πιθανότητα λανθασμένης ανίχνευσης. Τα αποτελέσματά μας δείχνουν ότι οι νέοι προτεινόμενοι αλγόριθμοι βελτιώνουν την κανονικοποιημένη μέση απόδοση του SU.



Σχ. 3.1 Στιγμιότυπο ενός Cognitive Radio Network με τους πομπούς των δευτερευόντων (αναπαριστώνται με τις κινητές συσκευές) και πρωτεύοντων χρηστών (αναπαριστώνται με τους πύργους κεραίες). Υποθέτουμε ότι κάθε δευτερεύων χρήστης μπορεί να λάβει σήμα από κάθε πρωτεύοντα χρήστη και για το λόγο αυτό όλη η περιοχή είναι χρωματισμένη γκρι.

### 3.2 Περιγραφή συστήματος

Χρησιμοποιούμε και σε αυτό το κεφάλαιο την τοπολογία του δικτύου που προτάθηκε στο [2]. Δηλαδή ο PU αγνοεί την παρουσία του SU ή των SUs στο κανάλι στο οποίο αυτός είναι εξουσιοδοτημένος χρήστης και μεταδίδει τα δεδομένα του κάθε φορά που έχει πακέτο για μετάδοση. Η δομή μίας χρονοθυρίδας καναλιού (slot) αποτελείται από περισσότερες από μία μίνι-χρονοθυρίδες (mini-slots), οι οποίες αντιστοιχούν σε διαφορετικές φάσεις της ανίχνευσης φάσματος και μετάδοσης πακέτων. Κατά τη διάρκεια μίας χρονοθυρίδας οι SU έχουν έναν σταθερό αριθμό ευκαιριών για να βρουν ένα άδαιο κανάλι. Οι δευτερεύοντες χρήστες ξεκινούν την εκτίμηση της κατάστασης του καναλιού στην αρχή κάθε slot (για την πρώτη ευκαιρία μετάδοσης). Κάθε χρήστης προσπαθεί να μεταδώσει σε κάθε slot σε κάποιο κανάλι που είναι μη-απασχολημένο. Σε κάθε slot θεωρούμε ότι ο PU μεταδίδει ένα πακέτο, αν έχει πακέτο προς μετάδοση. Ο δευτερεύων χρήστης μπορεί να μεταδώσει στο κανάλι όταν αυτό είναι άδαιο και το ανιχνεύει επιτυχημένα ως τέτοιο.

Όπως στο κεφάλαιο 2, έχουμε ένα time slotted συγχρονισμένο σύστημα με  $N_s$  δευτερεύοντες χρήστες και  $N_p$  πρωτεύοντες χρήστες και κανάλια. Οι δευτερεύοντες χρήστες ψάχνουν ευκαιρίες μετάδοσης σε κάποιο από τα κανάλια. Θεωρούμε ένα CRN όπου οι  $N_p$  πρωτεύοντες πομποί - δέκτες και  $N_s$  δευτερεύοντες πομποί - δέκτες είναι ομοιόμορφα καταναμημένοι στο χώρο. Σε κάθε πομπό, πρωτεύοντα ή δευτερεύοντα αντιστοιχεί και ένας δέκτης. Ο πρωτεύων πομπός μεταδίδει πακέτα στον αντίστοιχο πρωτεύων δέκτη και με τη σειρά του ο πρωτεύων δέκτης ενημερώνει με ACK/NACK τον πρωτεύοντα δέκτη αν τελικά έλαβε ή όχι σωστά το πακέτο. Αντίστοιχα οι δευτερεύοντες πομποί μεταδίδουν πακέτα προς τους πρωτεύοντες

δέκτες στο ίδιο κανάλι με κάποιον πρωτεύοντα πομπό αν αυτός δεν μεταδίδει εκείνη τη χρονική στιγμή. Έπειτα θεωρούμε ότι πρωτεύοντες δέκτες μεταβιβάζουν το πακέτο στον αντίστοιχο δευτερεύοντα δέκτη. Οι αποστάσεις των πρωτευόντων δεκτών – δευτερευόντων πομπών διαφέρουν ανά ζεύγος. Αυτό οδηγεί στο ότι κάθε δευτερεύοντας πομπός “βλέπει” διαφορετική χωρητικότητα σε κάθε κανάλι από αυτή που “βλέπει” κάποιος άλλος δευτερεύοντας πομπός.

Ο τύπος που ισχύει για τη χωρητικότητα κάθε καναλιού, έστω του καναλιού  $i$ , βάσει του θεωρήματος Shannon-Fano είναι :

$$C_i = B_i \log_2(1 + \text{SNR}) , i=1 \dots N_p.$$

Γνωρίζουμε επίσης ότι η σηματο-θορυβική σχέση στο δέκτη είναι της μορφής, [16] :

$$\text{SNR} = \frac{\text{PowerSignal}_{\text{transmitted}} * \text{ChannelGain}^2}{\text{Noise Power}_{\text{atReceiver}}}.$$

και για το channel gain ισχύει :

$h_{pj}$ , channel gain between primary transmitter  $j$  and primary receiver  $j$

$h_{si,pj}$ , channel gain between secondary transmitter  $i$  and primary receiver  $j$

Υποθέτουμε ότι όλοι δευτερεύοντες πομποί γνωρίζουν την ισχύ εκπεμπόμενου σήματος των πρωτευόντων πομπών. Επίσης υποθέτουμε ότι όλοι οι πομποί έχουν την ίδια εκπεμπόμενη ισχύ. Συμβολίζουμε με  $B_i$  το εύρος φάσματος του καναλιού  $i$ ,  $P_i$  την εκπεμπόμενη ισχύ του  $i$  πρωτεύοντα πομπού και  $N_0$  την ισχύ του θορύβου.

Τότε οι χωρητικότητες των διαφορετικών συνδέσμων είναι :

$C_i = B_i \log_2(1 + \text{SNR})$ , όπου στον τύπο του SNR αντικαθιστούμε το channel gain του συνδέσμου που θέλουμε (π.χ  $h_{si,pj}$  αν θέλουμε τον σύνδεσμο μεταξύ του πρωτεύοντα δέκτη  $j$  και δευτερεύοντα πομπού  $i$ ).

### 3.3 Ανάλυση προβλήματος

Οι δευτερεύοντες χρήστες ‘ακούνε’ τις μεταδόσεις άλλων χρηστών και όταν βρουν ευκαιρία μετάδοσης σε κάποιο κανάλι μεταδίδουν σε αυτό στον εναπομείναντα χρόνο του slot. Έχουμε ακριβώς την ίδια λειτουργία του συστήματος όπως αναλύθηκε στην ενότητα 2.2.

Κάθε δευτερεύων χρήστης στα πρώτα slots λειτουργίας του συστήματος θεωρούμε ότι μπορεί να εκτιμήσει τη χωρητικότητα κάθε καναλιού, παρακολουθώντας τα ACK/NACK που στέλνει ο πρωτεύων δέκτης στον πρωτεύοντα πομπό του καναλιού αυτού. Όπως αναφέραμε νωρίτερα, θεωρούμε ότι η πιθανότητα  $P_{md}$  εξαρτάται από την απόσταση που έχει κάθε SU (πομπός) από κάθε άλλο πομπό (SU ή PU). Κάθε

δευτερεύων χρήστης λαμβάνει με μία ισχύ το σήμα που έστειλε ο πρωτεύων ή δευτερεύων χρήστης. Αν η ισχύς του σήματος αυτού είναι μεγαλύτερη από ένα όριο θεωρεί ότι το κανάλι είναι απασχολημένο, αντίθετα αν είναι χαμηλότερη από αυτό το όριο θεωρεί ότι έλαβε μόνο θόρυβο και εκτιμά το κανάλι ως άδειο. Για τις ανάγκες της προσομοίωσης μας, και λαμβάνοντας υπόψη ότι δεν μπορούμε να γνωρίζουμε όλες τις παραμέτρους ενός πραγματικού συστήματος το οποίο είναι αρκετά σύνθετο, μοντελοποιήσαμε την πιθανότητα λανθασμένης ανίχνευσης  $P_{md}$  με τον τρόπο που αναλύεται παρακάτω.

Θεωρούμε ότι έχουμε μία περιοχή με εμβαδόν  $\mathbf{a \times b \ m^2}$  στην οποία κατανέμονται ομοιόμορφα οι χρήστες (primary-secondary transmitters-receivers). Κάθε χρήστης είναι στατικά τοποθετημένος σε μία θέση, έστω ότι οι συντεταγμένες της θέσης αυτής είναι  $(x,y)$ . Η μέγιστη δυνατή απόσταση που μπορούν να έχουν δύο χρήστες βάσει της περιοχής που έχουμε υποθέσει είναι  $\sqrt{(a-0)^2 + (b-0)^2} = \sqrt{a^2 + b^2} = \text{max\_distance}$ . Έτσι χρησιμοποιώντας το πυθαγόρειο θεώρημα, έχοντας δεδομένες τις συντεταγμένες των χρηστών  $i,j$  υπολογίζουμε την απόσταση τους και διαιρώντας τη με τη μέγιστη δυνατή απόσταση ( $\text{max\_distance}$ ) παίρνουμε την κανονικοποιημένη απόσταση των δύο χρηστών, την οποία συμβολίζουμε με  $d_{i,j}$ ,  $d_{i,j} \in [0,1]$ . Είναι λογικό ότι όσο πιο μακριά βρίσκεται ο δευτερεύων χρήστης (πομπός) από ένα δευτερεύοντα ή πρωτεύοντα πομπό (transmitter), τόσο πιο εξασθενημένο λαμβάνει το σήμα του τελευταίου με δεδομένο ότι έχουμε υποθέσει πως όλοι οι πομποί έχουν την ίδια εκπεμπόμενη ισχύ. Άρα όσο πιο μεγάλη είναι η κανονικοποιημένη απόσταση  $d_{i,j}$ , τόσο μικρότερη πιθανότητα έχει ο δευτερεύων χρήστης να προβλέψει σωστά αν το κανάλι που εξετάζει είναι απασχολημένο ή όχι τελικά από άλλο χρήστη.

Συμπερασματικά, όταν ο δευτερεύων χρήστης (πομπός) είναι κοντά στον πρωτεύοντα ή δευτερεύοντα πομπό, λαμβάνει το σήμα του χρήστη αυτού με υψηλή ισχύ και είναι δύσκολο να μην αισθανθεί τη μετάδοση του. Αντίθετα, όταν ο δευτερεύων χρήστης απέχει αρκετά από το χρήστη που μεταδίδει, το σήμα φτάνει εξασθενημένο στο δευτερεύοντα χρήστη και μπορεί αυτός να εκτιμήσει ότι έλαβε μόνο θόρυβο, με αποτέλεσμα να οδηγηθεί σε λάθος απόφαση ότι το κανάλι είναι ελεύθερο και να προσπαθήσει να μεταδώσει σε αυτό δημιουργώντας σύγκρουση.

Για να καταφέρουμε να προσομοιώσουμε στο σύστημα μας όσα αναφέραμε παραπάνω, **αλλάζουμε την πιθανότητα λανθασμένης ανίχνευσης  $P_{md}$  που υπόκειται κάθε δευτερεύων (transmitter) χρήστης για την ανίχνευση του σήματος κάθε άλλου πρωτεύοντα ή δευτερεύοντα (transmitter) χρήστη, βάσει της απόστασης που έχει αυτός από τον τελευταίο.** Έχουμε θεωρήσει ως ελάχιστη δυνατή τιμή της  $P_{md}$  το 0.005, αρκετά κοντά στο 0. Στη συνέχεια υποθέτοντας ότι η  $P_{md}$  ακολουθεί ομοιόμορφη κατανομή με μέση τιμή  $w$ , βρίσκουμε από τον τύπο  $(P_{md\max} - P_{md\min})/2 = w \Rightarrow P_{md\max} = 2w + P_{md\min}$  τη μέγιστη δυνατή τιμή της πιθανότητας. Γνωρίζοντας την κανονικοποιημένη απόσταση  $d_{i,j}$  μπορούμε να εκτιμήσουμε την πιθανότητα λανθασμένης ανίχνευσης του δευτερεύοντα χρήστη  $i$  για σήμα που στέλνει ο χρήστης  $j$  από τον τύπο:

$$P_{md}(i,j) = d_{i,j}(P_{md}max - P_{md}min) + P_{md}min.$$

Έτσι καθορίζεται πλέον η πιθανότητα  $P_{md}$  ανάλογη της απόστασης των χρηστών. Η  $P_{md}(i,j)$  όλου του συστήματος πλέον δεν ακολουθεί ομοιόμορφη κατανομή με μέση τιμή  $w$ , αφού μπορεί οι χρήστες να είναι ομοιόμορφα κατανεμημένοι στις συντεταγμένες  $x$  και  $y$  στο χώρο ( $a \times b \text{ m}^2$ ) με μέση τιμή των  $x,y$  συντεταγμένων το  $a/2$  και  $b/2$  αντίστοιχα, αλλά η κανονικοποιημένη απόσταση κάθε δευτερεύοντα (transmitter) χρήστη από κάθε πομπό (δευτερεύοντα η πρωτεύοντα) **δεν** ακολουθεί ομοιόμορφη κατανομή.

Επιπροσθέτως όσο πιο κοντά είναι ο δευτερεύων πομπός-χρήστης (secondary transmitter) στον πρωτεύοντα δέκτη-χρήστη (primary receiver), καθώς θεωρούμε ότι σε αυτόν μεταβιβάζει τα δεδομένα του, τόσο μεγαλύτερη μπορεί να εκτιμήσει τη χωρητικότητα του καναλιού, αφού υποθέτουμε ότι όλα τα κανάλια έχουν το ίδιο εύρος φάσματος. Οι δευτερεύοντες χρήστες δεν εκτιμούν την ίδια τιμή χωρητικότητας για τα ίδια κανάλια αφού δεν απέχουν όλοι την ίδια απόσταση από τον κάθε πρωτεύοντα δέκτη. Όπως στα [16] και [17], για το channel gain μεταξύ του δευτερεύοντα χρήστη  $i$  (secondary transmitter  $i$ ) και του πρωτεύοντα δέκτη  $j$  (primary receiver  $j$ ) ισχύει ο τύπος  $h_{si,pj}^2 = 1 / (d_{si,pj})^n$  όπου  $d_{si,pj}$  είναι η κανονικοποιημένη απόσταση  $[0,1]$  των δύο χρηστών και  $n=2$  ο συντελεστής απώλειας διαδρομής (path loss coefficient). Στη συνέχεια η χωρητικότητα υπολογίζεται από τον τύπο που αναφέρθηκε στην ενότητα 3.2.

### 3.3.1 Αλγόριθμος κατασκευής ακολουθίας SS χρησιμοποιώντας την εκτίμηση της χωρητικότητας του καναλιού (Build SS using capacity)

Χρησιμοποιήσαμε τον αλγόριθμο Build SS που προτείναμε στο προηγούμενο κεφάλαιο, κάνοντας μερικές αλλαγές έτσι ώστε κάθε δευτερεύων χρήστης να λαμβάνει υπόψη του την εκτίμηση της χωρητικότητας που έχει κάνει για κάθε κανάλι. Ο δευτερεύων χρήστης μπορεί να εκμεταλλευτεί τις παραπάνω εκτιμήσεις και να επιλέγει τα κανάλια με διαφορετική πιθανότητα, ανάλογη της χωρητικότητας κάθε καναλιού. Επισημαίνουμε ότι δεν εκτιμούν όλοι οι χρήστες την ίδια χωρητικότητα για ένα κανάλι, λόγω των διαφορετικών αποστάσεων που έχουν από τον κάθε δέκτη. Κάθε χρήστης λοιπόν, υπολογίζει την πιθανότητα επιλογής κάθε καναλιού με βάση την εκτίμηση που έχει κάνει για τη χωρητικότητα του. Ο υπολογισμός των πιθανοτήτων του δευτερεύοντα χρήστη  $i$  για να επιλέξει το κανάλι  $j$  γίνεται ως εξής:

$$P_{Si,j} = \frac{C_{i,j}}{\sum_{k=1}^{N_p} C_{i,k}}, i=1,...,N_s, j=1,...,N_p$$

όπου  $C_{i,j}$  η χωρητικότητα του καναλιού  $j$  από τον SU  $i$ .

Με αυτόν τον τρόπο οι δευτερεύοντες χρήστες έχουν μεγαλύτερη πιθανότητα να επιλέξουν για να αισθανθούν κανάλι το οποίο έχουν εκτιμήσει με μεγάλη χωρητικότητα. Έπειτα αν μεταδώσουν επιτυχημένα σε αυτό, το αποθηκεύουν στη πρώτη **κενή θέση** της λίστας τους. Οι ενέργειες που κάνουν οι δευτερεύοντες χρήστες από τη στιγμή που επιλέγεται το κανάλι, όπως δηλαδή η θέση εισαγωγής του στη λίστα ή το πότε κρίνεται ότι πρέπει να διαγραφεί από αυτή, παραμένουν ίδιες με αυτές του αλγορίθμου Build SS όπως αναλύθηκε στην ενότητα 2.2.4 του κεφαλαίου 2.

Αξίζει να τονιστεί ξανά, ότι αυτές οι πιθανότητες επιλογής καναλιού διαφέρουν από χρήστη σε χρήστη με δεδομένο ότι οι χρήστες είναι ομοιόμορφα κατανομημένοι στο χώρο. Έτσι είναι πολύ πιθανό οι χρήστες να κάνουν διαφορετικές επιλογές καναλιών στις ίδιες ευκαιρίες μετάδοσης του slot, με αποτέλεσμα να μην οδηγούνται σε συγκρούσεις αν τελικά μεταδώσουν πακέτο σε αυτό και να καταφέρνουν επιτυχείς μεταδόσεις αποθηκεύοντας το κανάλι στη λίστα τους.

### 3.3.2 Αλγόριθμος αίσθησης ιδανικότερων ως προς την εκτίμηση καναλιών (ICESA-ideals channels for the estimation sensing algorithm)

Με το συγκεκριμένο αλγόριθμο κάθε χρήστης επιλέγει αμέσως να έχει στη λίστα του τα τρία κανάλια που έχει εκτιμήσει ότι είναι καλύτερα για αυτόν. Όπως έχουμε αναφέρει αναλυτικά παραπάνω, στον αλγόριθμο Build SS using capacity κάθε δευτερεύων χρήστης μπορεί να εκτιμήσει στα πρώτα slots της λειτουργίας του συστήματος τη χωρητικότητα κάθε καναλιού, και με βάση αυτές τις εκτιμήσεις δημιουργεί τις πιθανότητες σύμφωνα με τις οποίες θα το επιλέξει. Πλέον με τον ίδιο τρόπο, δημιουργεί το βάρος που έχει κάθε κανάλι για αυτόν. Ο υπολογισμός αυτών των βαρών του δευτερεύοντα χρήστη  $i$  για το κανάλι  $j$  γίνεται ως εξής:

$$W_{i,j} = C_{i,j}, i=1,...,N_s, j=1,...,N_p$$

όπου  $C_{i,j}$  η χωρητικότητα του καναλιού  $j$  από τον SU  $i$ .

Μπορεί εύκολα να γίνει αντιληπτό ότι το κανάλι με το μεγαλύτερο βάρος είναι αυτό που έχει τη μεγαλύτερη χωρητικότητα. Έτσι ο δευτερεύων χρήστης επιλέγει να τοποθετήσει στη λίστα του τα κανάλια με τα τρία μεγαλύτερα βάρη. Στην πρώτη θέση τοποθετεί το κανάλι με το μεγαλύτερο βάρος, στη δεύτερη το κανάλι με το δεύτερο μεγαλύτερο βάρος και στην τρίτη το κανάλι με το τρίτο μεγαλύτερο βάρος.

Μπορούμε να πούμε ότι με αυτόν τον τρόπο δημιουργείται ενός είδους ομαδοποίηση (clustering) ανάμεσα στους δευτερεύοντες χρήστες, για το λόγο ότι δεν απέχουν όλοι το ίδιο από όλους τους πρωτεύοντες δέκτες (receivers) με αποτέλεσμα ο κάθε

χρήστης να κάνει διαφορετική εκτίμηση χωρητικότητας καναλιού και να δημιουργεί διαφορετικά βάρη. Αφού έχουμε θεωρήσει ότι όλοι οι χρήστες κατανέμονται ομοιόμορφα στο χώρο, τα κανάλια θα μοιράζονται στους δευτερεύοντες χρήστες.

Τώρα αν ο χρήστης αντιληφθεί ότι με τη σειρά που έχει τοποθετήσει τα κανάλια στη λίστα του οδηγείτε σε συνεχόμενες συγκρούσεις αλλάζει με μία πιθανότητα  $p\_change\_position$  τη σειρά αυτή. Ειδικότερα, όταν ο χρήστης λάβει επτά συνεχόμενες συγκρούσεις για κάποιο κανάλι σε επτά συνεχόμενα slots τότε τοποθετεί με πιθανότητα  $p\_change\_position$  το κανάλι αυτό στην τελευταία θέση της λίστας και ολισθαίνει τα υπόλοιπα κατά μία θέση αριστερά. Ο λόγος που επιλέχτηκε ο αριθμός επτά εξηγείται στην παρακάτω ενότητα που αναλύονται τα αποτελέσματα των προσομοιώσεων. Για παράδειγμα αν ο χρήστης έχει τα κανάλια 4, 5 και 1 με τα μεγαλύτερα βάρη κατά φθίνουσα σειρά, τα τοποθετεί στη λίστα του ως [4 5 1]. Αν λάβει συνεχόμενες συγκρούσεις από το κανάλι 4 η λίστα θα γίνει με πιθανότητα  $p\_change\_position$  ίση με [5 1 4], ενώ αν λάβει συνεχόμενες συγκρούσεις από το κανάλι 5 η λίστα θα γίνει με πιθανότητα  $p\_change\_position$  ίση με [4 1 5]. Με τον τρόπο αυτό ο χρήστης μπορεί να εντοπίσει τη σειρά των καναλιών που τον εξυπηρετεί περισσότερο.

Τώρα η πιθανότητα  $p\_change\_position$  υπάρχει για το λόγο ότι όταν ένας χρήστης λαμβάνει συνεχόμενα NACKs σε κάποιο κανάλι πιθανότατα συγκρούεται με άλλους δευτερεύοντες χρήστες. Τότε αν δεν υπήρχε αυτή η πιθανότητα θα άλλαζαν όλοι οι χρήστες τη σειρά των καναλιών στη λίστα τους και θα τοποθετούσαν όλοι (το κανάλι που τους οδηγούσε σε συνεχόμενα NACKs) στην τελευταία θέση της λίστας τους. Με την ύπαρξη της πιθανότητας  $p\_change\_position$  δεν θα αλλάζουν όλοι τη σειρά των καναλιών στη λίστα τους με αποτέλεσμα αυτό το κανάλι να παραμείνει σε καλή θέση στη λίστα κάποιων χρηστών ώστε να μπορεί να αξιοποιηθεί πιο σύντομα. Η πιθανότητα  $p\_change\_position$  έπειτα από διάφορες δοκιμές στις προσομοιώσεις επιλέχτηκε να είναι ίση με 0.3.

Επιπλέον, κάθε χρήστης προσπαθεί να μεταδώσει σε κάθε slot βάσει μίας πιθανότητας  $p\_sense$ . Αυτή η πιθανότητα αρχικά είναι 1 για όλους τους χρήστες, και μεταβάλλεται ανάλογα με τις καταστάσεις που αντιμετωπίζει ο χρήστης στη διάρκεια της λειτουργίας του συστήματος. Ειδικότερα, όταν ο χρήστης μεταδώσει σε ένα κανάλι και το πακέτο του συγκρουστεί, με αποτέλεσμα να λάβει NACK στο τέλος του slot, η πιθανότητα  $p\_sense$  μειώνεται κατά 0.1 με την ελάχιστη αποδέκτη τιμή που μπορεί να πάρει το 0.5. Αντίθετα, αν ο χρήστης μεταδώσει επιτυχώς στο slot η πιθανότητα  $p\_sense$  αυξάνεται κατά 0.1 με προφανή μέγιστη τιμή το 1. Με αυτό τον τρόπο όταν υπάρχει υψηλός ανταγωνισμός δευτερευόντων χρηστών στο δίκτυο (λιγότερα κανάλια από δευτερεύοντες χρήστες ή υψηλή κίνηση προερχόμενη από τους πρωτεύοντες χρήστες) τότε χρησιμοποιώντας την παραπάνω πιθανότητα είναι πολύ πιθανό να μην προσπαθήσουν να μεταδώσουν όλοι οι δευτερεύοντες χρήστες στο επόμενο slot με αποτέλεσμα να μειώνεται η συμφόρηση στο δίκτυο. Αντίθετα όταν οι δευτερεύοντες χρήστες μεταδίδουν συνεχώς με επιτυχία, σημαίνει ότι το δίκτυο έχει τη δυνατότητα να τους υποστηρίξει όλους και η πιθανότητα  $p\_sense$



τείνει να γίνει ίση με 1. Η μείωση και αύξηση της πιθανότητας  $p_{\text{sense}}$  κατά 0.1 αλλά και η ελάχιστη δυνατή τιμή της (0.5), επιλέχθηκαν πειραματικά έπειτα από δόκιμες.

### 3.3.3 Αλγόριθμος αίσθησης ιδανικότερων ως προς την εκτίμηση καναλιών χρησιμοποιώντας την πιθανότητα λανθασμένης ανίχνευσης (ICESA using $P_{\text{md}}$ - ideals channels for the estimation sensing algorithm using $P_{\text{md}}$ )

Ο συγκεκριμένος αλγόριθμος λειτουργεί ακριβώς όπως ο ICESA με τη μόνη διαφορά ότι χρησιμοποιεί διαφορετική μετρική για τον υπολογισμό των βαρών που έχει κάθε κανάλι για κάθε δευτερεύοντα χρήστη. Πιο συγκεκριμένα κάθε χρήστης χρησιμοποιεί και την εκτιμώμενη πιθανότητα λανθασμένης ανίχνευσης ( $P_{\text{md}}$ ) που έχει προς όλους του πρωτεύοντες χρήστες (πομπούς), σε συνδυασμό με την εκτιμώμενη χωρητικότητα κάθε καναλιού για να υπολογίσει το βάρος που έχει γι' αυτόν το κάθε κανάλι. Θεωρούμε ότι κάθε δευτερεύων χρήστης εκτιμά με ακρίβεια την πιθανότητα  $P_{\text{md}}$  η οποία είναι ίση με αυτή που υπολογίζεται (ξεχωριστά για κάθε πρωτεύοντα πομπό αναλόγως την απόσταση που έχει από αυτόν) από τους τύπους που αναφέρθηκαν παραπάνω στην ενότητα 3.3. Οι δευτερεύοντες χρήστες δεν μπορούν να εκτιμήσουν την πιθανότητα λανθασμένης ανίχνευσης ως προς τους υπόλοιπους δευτερεύοντες χρήστες, μιας και δεν μπορούν να ξέρουν πόσοι είναι αυτοί αλλά και πάνω σε ποιο κανάλι από τα  $N_p$  διαθέσιμα μεταδίδουν κάθε χρονική στιγμή, πληροφορία που χρειάζεται για να παρακολουθήσουν στη συνέχεια την κίνηση του αντίστοιχου δέκτη.

Επομένως, κάθε δευτερεύων χρήστης στην αρχή λειτουργίας του συστήματος θεωρούμε ότι μπορεί να εκτιμήσει τη χωρητικότητα κάθε καναλιού αλλά και την πιθανότητα λανθασμένης ανίχνευσης για κάθε σήμα πρωτεύοντα πομπού, παρακολουθώντας τα ACK/NACK που στέλνει ο πρωτεύων δέκτης στον πρωτεύον πομπό του καναλιού αυτού. Στην αρχή λειτουργίας του συστήματος οι δευτερεύοντες χρήστες, κάνουν ανιχνεύσεις για την κατάσταση των καναλιών, αν δηλαδή είναι απασχολημένα ή όχι. Αυτή η ανίχνευση γίνεται αναλύοντας το σήμα που λαμβάνουν από κάποιον πρωτεύοντα δέκτη. Αν η ισχύς του σήματος αυτού είναι μεγαλύτερη από ένα όριο, ο δευτερεύων χρήστης θεωρεί ότι το κανάλι είναι απασχολημένο, αντίθετα αν είναι μικρότερη από αυτό το όριο θεωρεί ότι έλαβε μόνο θόρυβο και εκτιμά το κανάλι ως άδειο. Έπειτα παρακολουθεί την κίνηση του αντίστοιχου πρωτεύοντα δέκτη. Αν αυτός στείλει ACK/NACK προς τον πρωτεύοντα πομπό και ο δευτερεύων χρήστης (πομπός) είχε εκτιμήσει το κανάλι ως απασχολημένο, τότε συμπεραίνει ότι έκανε σωστή εκτίμηση της κατάστασης του καναλιού. Αντίθετα αν το είχε εκτιμήσει ως απασχολημένο και ο πρωτεύων δέκτης δεν απάντησε με ACK/NACK στον πρωτεύοντα πομπό, ο δευτερεύων χρήστης θεωρεί ότι έκανε λανθασμένη ανίχνευση του καναλιού.

Ο υπολογισμός των βαρών του δευτερεύοντα χρήστη  $i$  για το κανάλι  $j$  γίνεται πλέον ως εξής:

$$W_{i,j} = C_{i,j} * (1 - P_{md}(i,j)), i=1,...,N_s, j=1,...,N_p$$

όπου  $C_{i,j}$  η χωρητικότητα του καναλιού  $j$  από τον SU  $i$ , και  $1-P_{md}(i,j)$  η πιθανότητα σωστής ανίχνευσης του καναλιού  $j$  από το χρήστη  $i$ .

Εύκολα μπορεί να γίνει αντιληπτό, ότι το κανάλι με το μεγαλύτερο βάρος είναι αυτό για το οποίο το γινόμενο της χωρητικότητας του επί την πιθανότητα σωστής ανίχνευσης δίνει το μεγαλύτερο αποτέλεσμα. Στη συγκεκριμένη περίπτωση ο δευτερεύων χρήστης αν εκτιμήσει ένα κανάλι με μεγάλη χωρητικότητα αλλά με μεγάλη πιθανότητα λανθασμένης ανίχνευσης, πιθανότατα να το θεωρήσει μη αποδοτικό, βάσει της νέας μετρικής, με την έννοια ότι αυτό το κανάλι θα τον οδηγήσει πολύ συχνά σε συγκρούσεις με άλλους χρήστες, αφού δεν θα μπορεί να ανιχνεύσει με μεγάλη σιγουριά την ύπαρξη των μεταδόσεων τους πάνω σε αυτό.

### 3.4 Αξιολόγηση απόδοσης

#### 3.4.1 Μετρική απόδοσης

Και σε αυτό το κεφάλαιο χρησιμοποιούμε ως μετρική απόδοσης τον κανονικοποιημένο μέσο ρυθμό μετάδοσης **ανά δευτερεύοντα χρήστη**, με μονάδα μέτρησης τα Mbits per sec.

Δηλαδή αν ένας SU μεταδίδει για χρόνο  $t_{tr}$  (ms) σε κανάλι με χωρητικότητα  $C$  Kbps τότε μεταδίδει συνολικά  $C*t_{tr}$  bits στο συγκεκριμένο διάστημα. Αθροίζοντας όποτε μεταδίδει **οποιοσδήποτε** δευτερεύων χρήστης αυτό τον αριθμό με ότι έχουν μεταδώσει μέχρι τότε όλοι οι SUs, στο τέλος της προσομοίωσης έχουμε τα συνολικά Mbits που μετέδωσαν (έστω  $data\_trans\_i$  τα bits που μετέδωσαν όλοι οι δευτερεύοντες χρήστες στη διάρκεια της προσομοίωσης και  $total\_time$  η συνολική διάρκεια της προσομοίωσης σε sec). Έπειτα κάνοντας την πράξη  $\frac{1}{N_s * Total\_time} \sum_{i=1}^{N_p} data\_trans\_i$ , έχουμε την κανονικοποιημένη μέση δυνατότητα μεταφοράς ανά δευτερεύοντα χρήστη, **η πιο απλά το μέσο αριθμό Mbits/sec ανά δευτερεύοντα χρήστη που μεταδόθηκαν καθ' όλη τη διάρκεια προσομοίωσης του συστήματος.**

Παράλληλα έχουμε θεωρήσει άλλες δύο μετρικές, οι οποίες είναι η μέση τιμή της πιθανότητας επιτυχημένης μετάδοσης σε ένα slot ανά δευτερεύοντα χρήστη και η μέση τιμή της πιθανότητας μη επιτυχημένης μετάδοσης (σύγκρουσης) σε ένα slot ανά δευτερεύοντα χρήστη. Ειδικότερα, για τον υπολογισμό της πιθανότητα επιτυχημένης μετάδοσης αθροίζουμε τις επιτυχημένες μεταδόσεις που πετυχαίνουν όλοι οι

δευτερεύοντες χρήστες και στο τέλος της προσομοίωσης διαιρούμε τον αριθμό αυτό με το πλήθος των συνολικών slots άλλα και με το πλήθος των δευτερευόντων χρηστών. Αντίστοιχα εκτιμάμε τη μέση τιμή της πιθανότητας σύγκρουσης, αθροίζοντας τις συγκρούσεις που υπόκεινται οι δευτερεύοντες χρήστες και διαιρώντας ομοίως με παραπάνω.

### 3.4.2 Μοντέλο προσομοίωσης

Για να συγκρίνουμε τα αποτελέσματα του αλγορίθμου Build SS που προτάθηκε στο κεφάλαιο 2 με τους αλγορίθμους που εισάγαμε σε αυτό το κεφάλαιο, υλοποιήσαμε σε όλους τις ίδιες προσομοιώσεις, με τα ίδια δεδομένα. Θεωρούμε ένα time slotted Cognitive Radio Network με  $N_s$  δευτερεύοντες χρήστες, οι οποίοι προσπαθούν ευκαιριακά να μεταδώσουν σε κάθε slot ένα πακέτο πληροφορίας σε κάποιο από τα  $N_p$  κανάλια του συστήματος. Η περιοχή στην οποία θεωρήσαμε ότι κατανέμονται ομοιόμορφα οι χρήστες (δευτερεύοντες ή πρωτεύοντες) είναι **2000 x 2000 m<sup>2</sup>**. Στο κανάλι  $i$  ο ρυθμός άφιξης πακέτων από τον πρωτεύοντα χρήστη (transmitter) είναι ίσος με  $\lambda_i$  πακέτα/slot. Υποθέτουμε ότι αυτές οι αφίξεις πακέτων ακολουθούν την κατανομή Poisson. Επίσης, συμβολίζουμε με  $P_{md}$  την πιθανότητα κάποιος SU να αισθανθεί λανθασμένα το κανάλι ως μη απασχολημένο, και να προσπαθήσει να μεταδώσει σε αυτό με αποτέλεσμα να υπάρξει σύγκρουση, και  $P_{fa}$  την πιθανότητα ένας SU να αισθανθεί λανθασμένα το κανάλι σαν απασχολημένο, με αποτέλεσμα να μη μεταδώσει σε αυτό, χάνοντας αυτήν την ευκαιρία μετάδοσης. Συμβολίζουμε με  $T$  το χρόνο διάρκειας του slot.

Τα σενάρια που έχουμε προσομοιώσει χωρίζονται κυρίως σε δύο περιπτώσεις. Μία όπου τα κανάλια (και οι PUs) είναι περισσότερα από τους SUs και πιο συγκεκριμένα  $N_p=10$  και  $N_s=5$ , και μία όπου υπάρχει υψηλός ανταγωνισμός μεταξύ των SUs, αφού είναι αυτοί 13 ( $N_s$ ) και οι PUs (άρα και τα κανάλια) 10 ( $N_p$ ). Κάθε σενάριο διαιρείται σε αλλά υπο-σενάρια στα οποία μεταβάλλονται οι τιμές των πιθανοτήτων  $P_{md}$  και  $P_{fa}$ . Επίσης έχουν γίνει μετρήσεις για τις παραπάνω περιπτώσεις όταν οι ρυθμοί άφιξης πακέτων σε όλα τα κανάλια είναι ίσοι με μέση τιμή  $\lambda_i=0.3$  πακέτα/slot (δηλαδή τα κανάλια είναι ελαφρώς φορτωμένα), όταν  $\lambda_i=0.7$  πακέτα/slot (δηλαδή είναι υψηλά φορτωμένα), και τέλος μετρήσεις όταν τα κανάλια έχουν διαφορετικούς ρυθμούς άφιξης.

Το εύρος φάσματος των καναλιών είναι σταθερό και ίσο με 6MHz για όλα τα κανάλια. Η μπάντα κάθε καναλιού είναι ομοιόμορφα κατανεμημένη στο διάστημα 470-862 MHz και η χωρητικότητα κάθε καναλιού υπολογίζεται από τον τύπο  $C_i = B_i \log_2(1 + \text{SNR})$ , όπου  $B_i, i = 1, \dots, N_p$  είναι το εύρος φάσματος του  $i$  καναλιού. Η διάρκεια κάθε slot υποτίθεται ίση με 10ms. Οι παραπάνω παράμετροι επιλέχτηκαν βάσει του προτύπου IEEE 802.22 [3] και του [8].

Αρχικά θεωρούμε ότι η  $P_{md}$  ακολουθεί ομοιόμορφη κατανομή, έτσι θέτουμε τη μέση τιμή που επιθυμούμε, και βάσει του τύπου της μέσης τιμής στην ομοιόμορφη κατανομή υπολογίζουμε τη μεγίστη τιμή που μπορεί να πάρει. Έτσι βρίσκουμε τη μέση τιμή της πιθανότητας  $P_{md}$  που μπορεί να εκτιμήσει κάποιος χρήστης στους αλγορίθμους του κεφαλαίου αυτού. Όμως όπως αναλύσαμε παραπάνω στην ενότητα 3.1 η κανονικοποιημένη απόσταση  $d_{i,j}$  δεν ακολουθεί ομοιόμορφη κατανομή με αποτέλεσμα η εκτίμηση της  $P_{md}$  να μην ακολουθεί ομοιόμορφη κατανομή. Για το λόγο αυτό οι τιμές που αναφέρονται στα παρακάτω σχήματα έχουν υπολογιστεί από την προσομοίωση, όπου βρίσκουμε τη μέση τιμή της  $P_{md}$  αθροίζοντας τις  $P_{md}$  όλων των χρηστών προς όλους τους χρήστες και διαιρώντας με το πλήθος τους.

**Πίνακας 3.1 Παράμετροι του συστήματος με τις καθορισμένες τιμές**

Συμβολισμός	Επεξήγηση (καθορισμένη τιμή)
$T$	Διάρκεια slot (10 ms)
$t_s$	Χρόνος που απαιτείται για αισθανθεί κάποιο κανάλι ο SU σε μία ευκαιρία μετάδοσης (2.4 ms)
$t_{ch}$	Χρόνος που απαιτείται για προετοιμασία ώστε να αισθανθεί νέο κανάλι ο SU (0.01 ms)
$N$	Αριθμός συνολικών slots ανά προσομοίωση (10000)
$Sim$	Αριθμός συνολικών προσομοιώσεων για εκτίμηση μέσων τιμών (200)
$\delta$	Συνολικές ευκαιρίες μετάδοσης ανά slot (3)
$B$	Εύρος φάσματος καναλιού (6MHz) –Ομοιόμορφα κατανεμημένο στο διάστημα (470-862 MHz)
$K$	Απαιτούμενα slots για την εκτίμηση της χωρητικότητας των καναλιών από κάθε SU στην αρχή της προσομοίωσης (100)
$P_w/N_0$	Ισχύς εκπεμπόμενου σήματος κάθε πρωτεύοντα χρήστη προς την ισχύ θορύβου στο δέκτη (10)
$E$	Απαιτούμενα slots για την εκτίμηση της πιθανότητας λανθασμένης ανίχνευσης από κάθε SU για κάθε κανάλι στην αρχή της προσομοίωσης (100)

### 3.4.3 Αποτελέσματα προσομοιώσεων

Τα αποτελέσματα στην ενότητα αυτή χωρίζονται σε τρεις κατηγορίες:

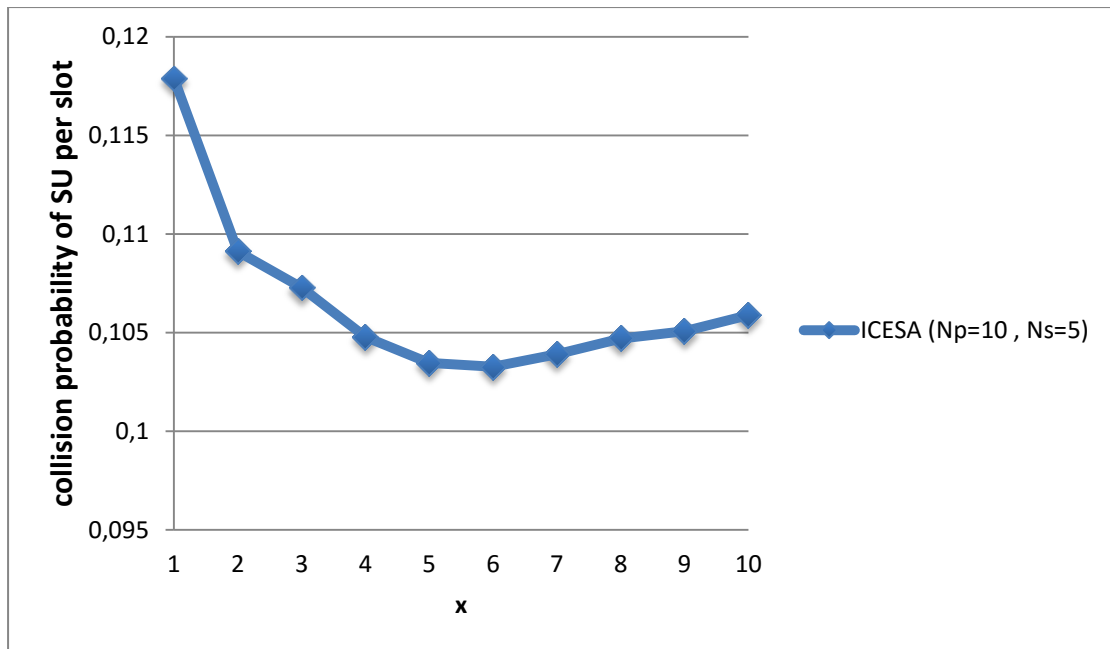
- Κατάλληλη παραμετροποίηση αλγορίθμου ICESA
- Σύγκριση αλγορίθμων τύπου ICESA
- Σύγκριση αλγορίθμων Build SS, Build SS using capacity και ICESA

Συγκεκριμένα, προσομοιώσαμε τις περιπτώσεις-σενάρια που αναλύθηκαν παραπάνω. Οι ενότητες 3.4.3.1 και 3.4.3.2 περιέχουν τις μετρήσεις για διάφορες τιμές των παραμέτρων που χρησιμοποιεί ο αλγόριθμος ICESA και στο πως αυτές επιλεχτήκαν. Στην ενότητα 3.4.3.3 συγκρίνονται οι αλγόριθμοι ICESA και ICESA using Pmd. Τέλος, οι ενότητες 3.4.3.4 και 3.4.3.5 περιέχουν τις συγκρίσεις των αλγορίθμων Build SS, Build SS using capacity και ICESA για τα σενάρια  $N_p > N_s$  και  $N_s > N_p$ , αντίστοιχα.

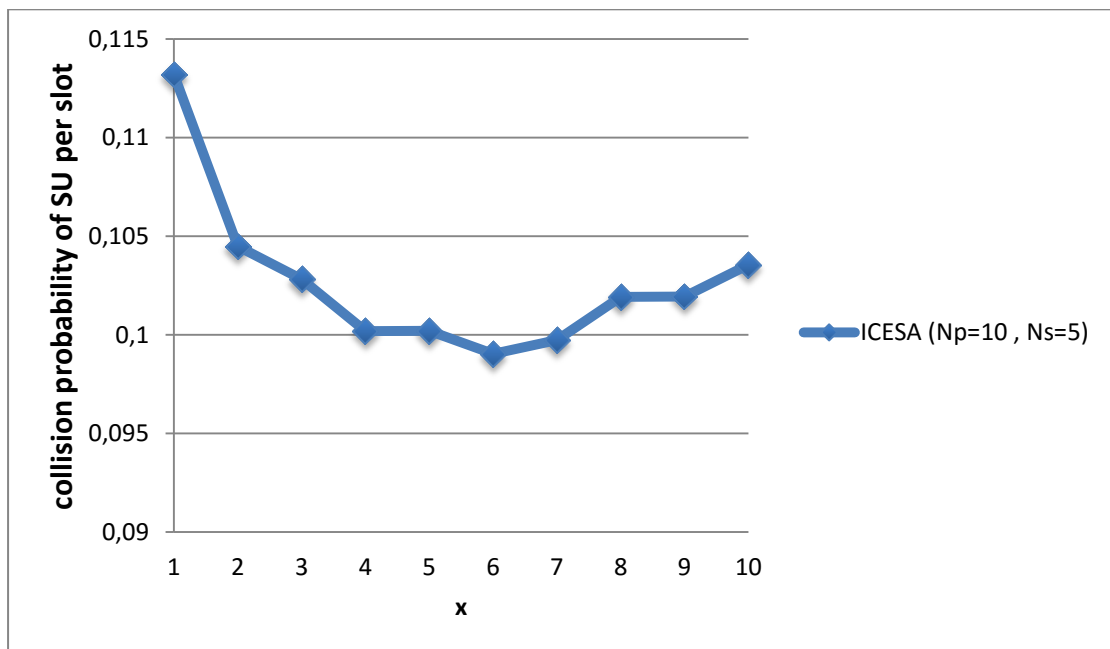
Δοθέντος ότι έχουμε θεωρήσει ότι οι δευτερεύοντες χρήστες γνωρίζουν ελάχιστα για το σύστημα, δηλαδή μόνο το πλήθος των καναλιών και τη χωρητικότητα τους, μας αναγκάζει να επιλέξουμε πειραματικά τις τιμές των παραμέτρων του αλγορίθμου ICESA. Σε καμία περίπτωση δεν ισχυριζόμαστε ότι οι τιμές που επιλέξαμε μέσω των προσομοιώσεων είναι οι βέλτιστες τιμές για όλες τις περιπτώσεις και τα σενάρια που μπορούν να υπάρξουν σε ένα πραγματικό σύστημα. Στις ενότητες 3.4.3.1 και 3.4.3.2 παρουσιάζονται ενδεικτικά σχήματα από τις μετρήσεις που έγιναν για την επιλογή των κατάλληλων τιμών των παραμέτρων του αλγορίθμου.

#### **3.4.3.1 Αποτελέσματα προσομοιώσεων βάσει του αριθμού των συνεχόμενων συγκρούσεων που πρέπει να υποστεί ένας SU προτού αποφασίσει να προσπαθήσει να αλλάξει θέση ένα κανάλι στη λίστα του στον αλγόριθμο ICESA**

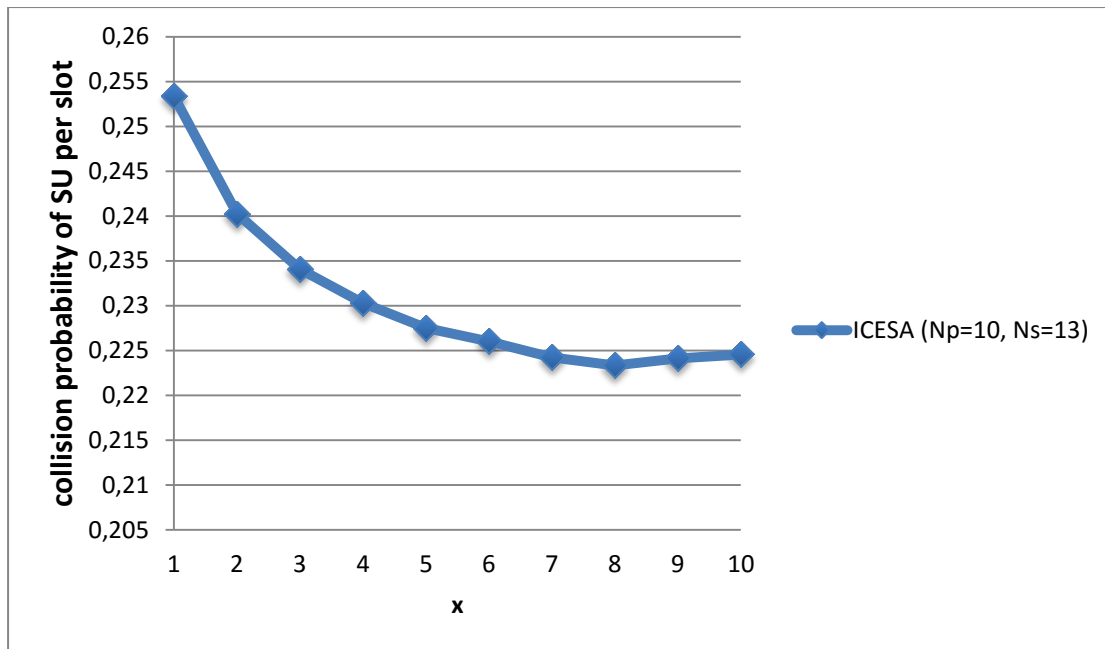
Για τις παρακάτω προσομοιώσεις ισχύει ότι οι μέσες τιμές για την πιθανότητα λανθασμένης ( $P_{md}$ ) ανίχνευσης και την πιθανότητα λανθασμένου συναγερμού ( $P_{fa}$ ) είναι ίσες με 0,075 και 0,1 αντίστοιχα. Έχουν υλοποιηθεί και τα δύο σενάρια, όταν δηλαδή οι πρωτεύοντες χρήστες είναι περισσότεροι από τους δευτερεύοντες ( $N_p=10$ ,  $N_s=5$ ), αλλά και όταν οι πρωτεύοντες είναι λιγότεροι από τους δευτερεύοντες ( $N_p=10$ ,  $N_s=13$ ). Επίσης ο ρυθμός άφιξης πακέτων στον πρωτεύοντα χρήστη κάθε καναλιού είναι ίσος  $\lambda_i=0.3$  πακέτα/slot όταν τα κανάλια είναι χαμηλά φορτωμένα,  $\lambda_i=0.7$  πακέτα/slot όταν τα κανάλια είναι υψηλά φορτωμένα και τέλος  $\lambda_i = \{0.1, 0.1, 0.2, 0.2, 0.3, 0.3, 0.4, 0.4, 0.5, 0.5\}$   $i = 1, \dots, N_p$  όταν είναι άνισα φορτωμένα. Επίσης η πιθανότητα αλλαγής θέσης καναλιού (p\_change\_position) έχει τεθεί ίση με 0.5.



*Figure 3.2 Πιθανότητα σύγκρουσης δευτερεύοντα χρήστη σε ένα slot στον αλγόριθμο ICESA σε χαμηλά φορτωμένο δίκτυο σαν συνάρτηση του αριθμού  $x$  (προσπάθεια αλλαγής θέσης καναλιού στη λίστα του χρήστη μετά από  $x$  συνεχόμενες συγκρούσεις).*



*Figure 3.3 Πιθανότητα σύγκρουσης δευτερεύοντα χρήστη σε ένα slot στον αλγόριθμο ICESA σε άνισα φορτωμένο δίκτυο σαν συνάρτηση του αριθμού  $x$  (προσπάθεια αλλαγής θέσης καναλιού στη λίστα του χρήστη μετά από  $x$  συνεχόμενες συγκρούσεις).*



**Figure 3.4** Πιθανότητα σύγκρουσης δευτερεύοντα χρήστη σε ένα slot στον αλγόριθμο ICESA σε χαμηλά φορτωμένο δίκτυο σαν συνάρτηση του αριθμού  $x$  (προσπάθεια αλλαγής θέσης καναλιού στη λίστα του χρήστη μετά από  $x$  συνεχόμενες συγκρούσεις).

Από τα παραπάνω σχήματα παρατηρούμε ότι είναι αποδοτικότερο ο χρήστης να προσπαθήσει να αλλάξει θέση στο κανάλι μετά από σχετικά μεγάλο αριθμό συγκρούσεων σε αυτό, μιας και αν προσπαθήσει να αλλάξει θέση έπειτα από λίγες συγκρούσεις, σε κάθε slot περισσότεροι SUs θα αλλάζουν θέση στο κανάλι τους, θα τοποθετούν το δεύτερο κανάλι στη λίστα τους στην πρώτη θέση (ή το τρίτο κανάλι της λίστας τους στη δεύτερη θέση) επηρεάζοντας έτσι άλλους χρήστες που μέχρι τώρα έστελναν σε αυτά τα κανάλια.

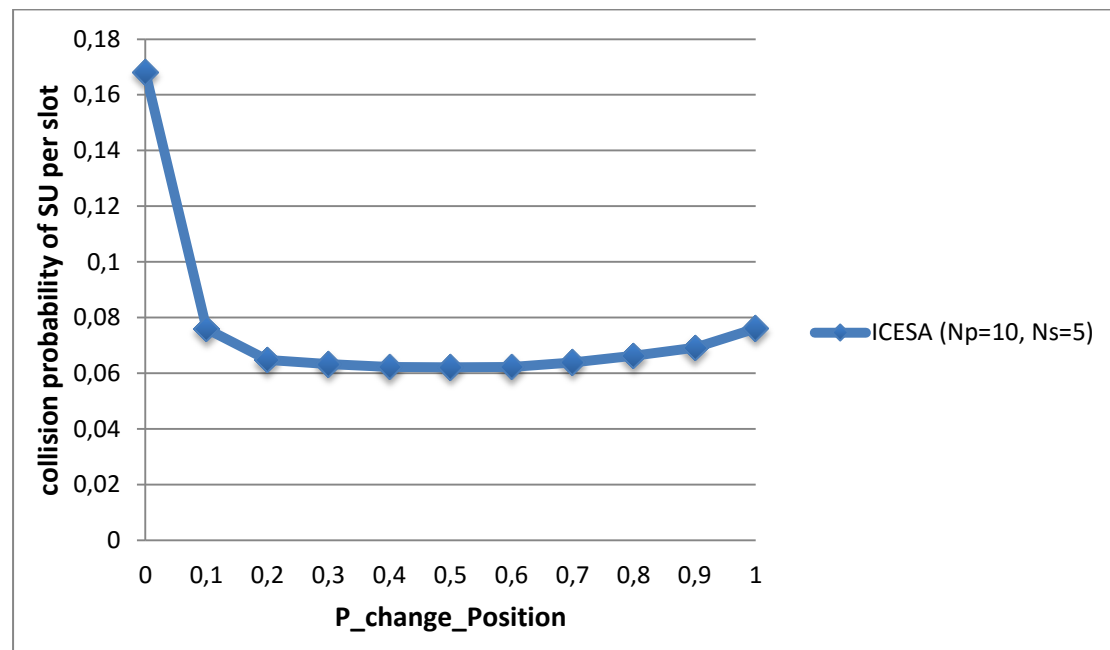
Συμπερασματικά, δεν είναι αποδοτικό να αλλάζουν θέση συνέχεια τα κανάλια στις λίστες των δευτερευόντων χρηστών, αφού με αυτό τον τρόπο δημιουργείται τελικά μία αστάθεια στο σύστημα (όσον αφορά το ποιο κανάλι θα αισθανθεί κάθε χρήστης). Αφού κάθε χρήστης αποφασίζει τα τρία κανάλια που τοποθετεί στη λίστα του και παραμένει σε αυτά καθ' όλη τη διάρκεια της λειτουργίας του συστήματος, δεν έχει νόημα να αλλάζει συχνά θέση στη λίστα σε αυτά τα κανάλια γιατί έτσι θα ανακυκλώνεται συνεχώς η μορφή της λίστας. Επιπλέον, όταν  $N_p > N_s$  βλέπουμε και πάλι ότι ο κατάλληλος αριθμός των συνεχόμενων συγκρούσεων μετά τον οποίο θα γίνεται προσπάθεια να αλλάξει θέση το κανάλι στη λίστα είναι σχετικά μεγάλος, αλλά όχι τόσο μεγάλος όσο στην περίπτωση όπου  $N_p < N_s$ . Αυτό εξηγείται γιατί όταν  $N_p > N_s$ , οι χρήστες δεν συναντούν τόσο υψηλό ανταγωνισμό και μπορούν να βρουν με μεγαλύτερη ευκολία νέο κανάλι που να τους εξυπηρετεί.

Δεδομένου ότι οι δευτερεύοντες χρήστες δεν γνωρίζουν, αν δεν υπάρχει συνεργασία μεταξύ τους, πόσοι δευτερεύοντες χρήστες υπάρχουν στο σύστημα, και κρίνοντας

από τα αποτελέσματα στα παραπάνω σχήματα, αποφασίσαμε να επιλέξουμε ως κατάλληλη τιμή αριθμού συνεχομένων συγκρούσεων μετά τον οποίο ο δευτερεύων χρήστης προσπαθεί να μετακινήσει ένα κανάλι θέση στη λίστα, **τον αριθμό 7**. Η παραπάνω επιλογή βασίζεται στο ότι με τον αριθμό 7 έχουμε αρκετά ικανοποιητική απόδοση για τις περιπτώσεις που εξεταστήκαν.

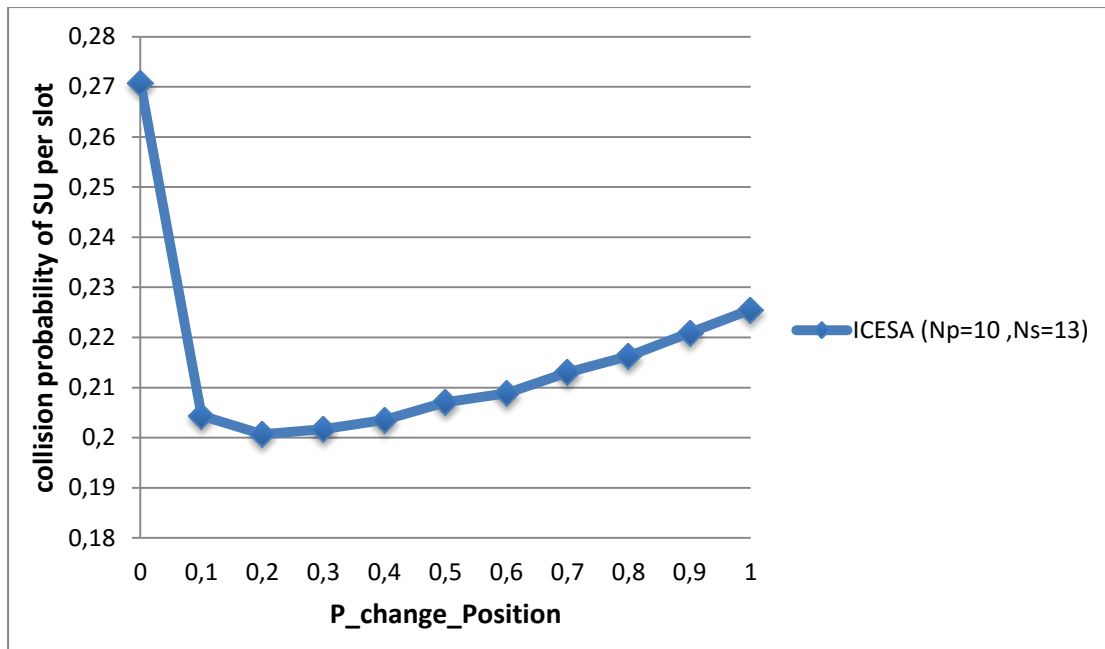
### 3.4.3.2 Αποτελέσματα προσομοιώσεων βάσει της τιμής της πιθανότητας $p\_change\_position$ στον αλγόριθμο ICESA

Για τις παρακάτω προσομοιώσεις ισχύει για τις παραμέτρους ότι ίσχυε στις προσομοιώσεις της ενότητας 3.4.3.1, με τη μόνη διάφορα ότι πλέον ο αριθμός των συνεχομένων συγκρούσεων που πρέπει να υποστεί ένας SU προτού αποφασίσει να προσπαθήσει να αλλάξει θέση ένα κανάλι στη λίστα τίθεται ίσος με 7 και αναζητείται η κατάλληλη τιμή για την πιθανότητα  $p\_change\_position$ .

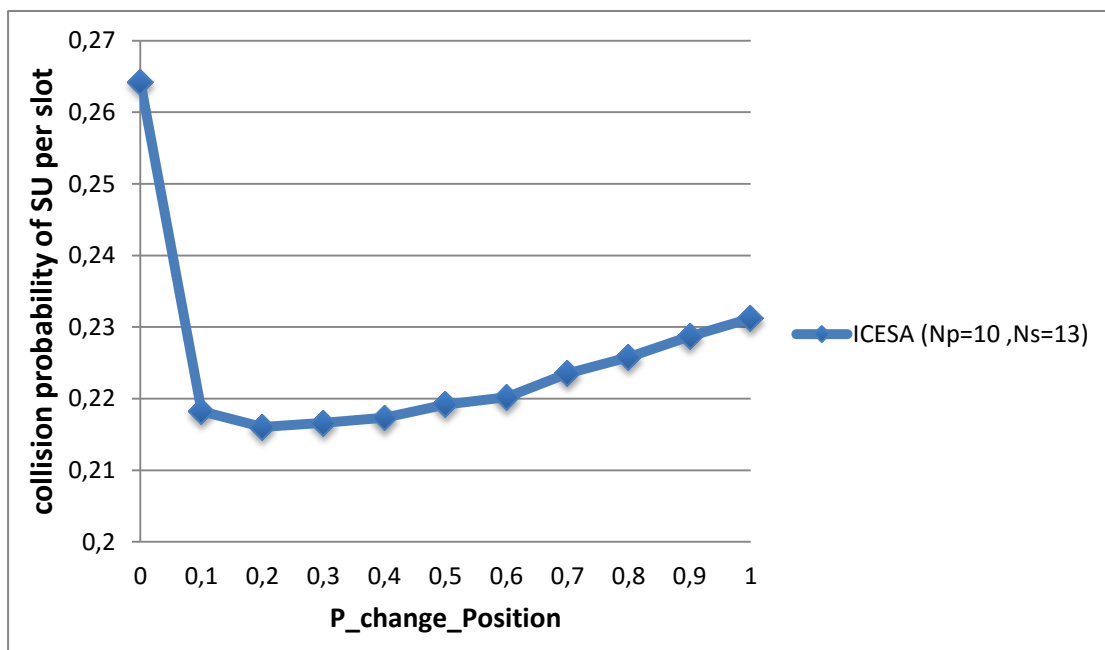


*Figure 3.5 Πιθανότητα σύγκρουσης δευτερεύοντα χρήστη σε ένα slot στον αλγόριθμο ICESA σε χαμηλά φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας αλλαγής θέσης καναλιού.*





**Figure 3.6** Πιθανότητα σύγκρουσης δευτερεύοντα χρήστη σε ένα slot στον αλγόριθμο ICESA σε χαμηλά φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας αλλαγής θέσης καναλιού.

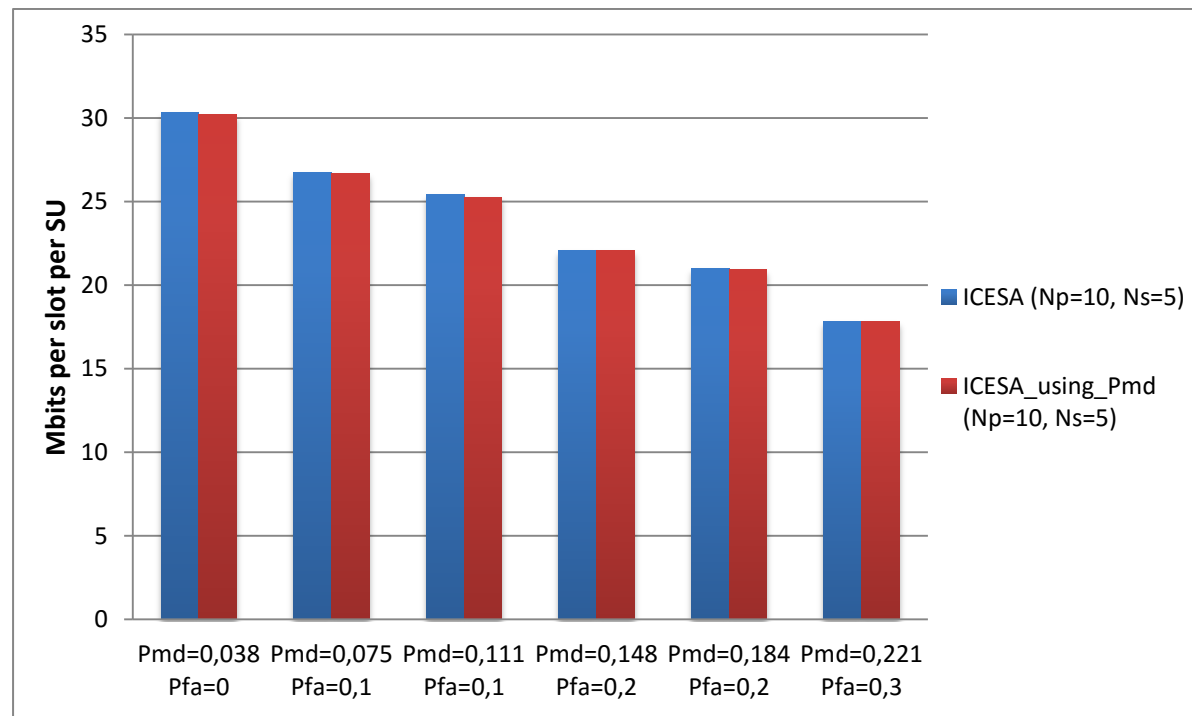


**Figure 3.7** Πιθανότητα σύγκρουσης δευτερεύοντα χρήστη σε ένα slot στον αλγόριθμο ICESA σε άνισα φορτωμένο δίκτυο σαν συνάρτηση της πιθανότητας αλλαγής θέσης καναλιού.

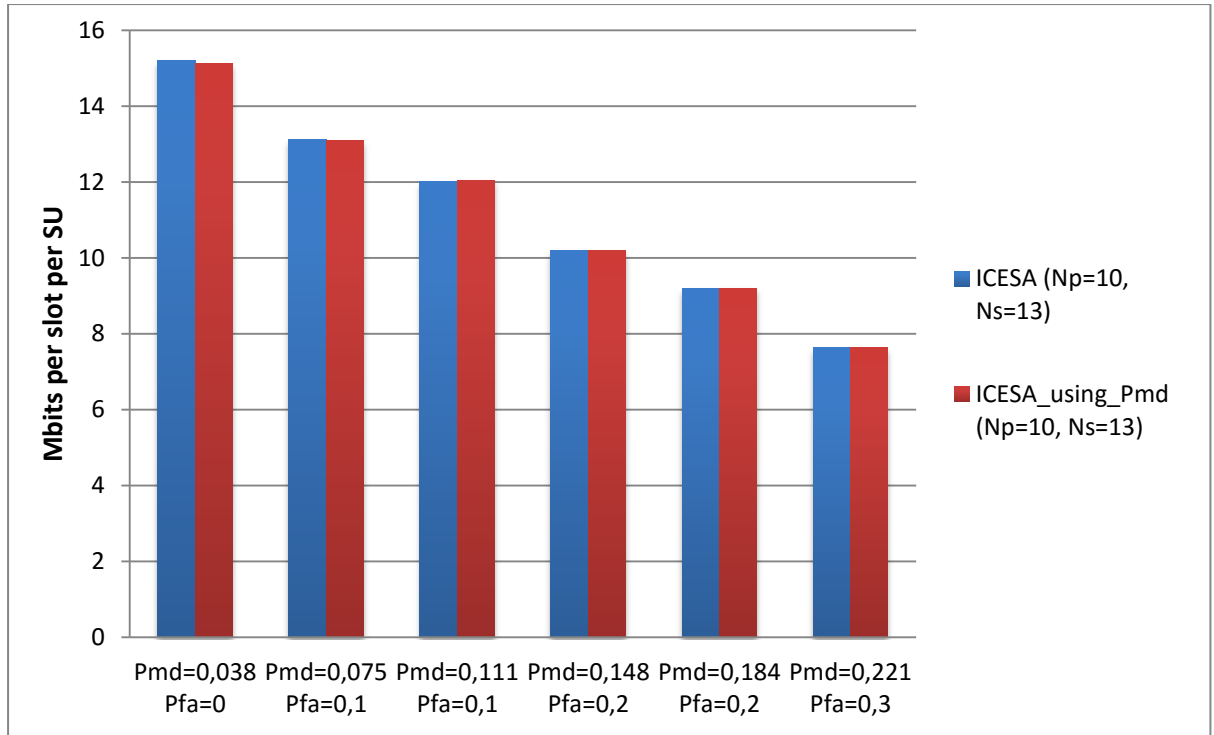
Από τα παραπάνω σχήματα παρατηρούμε ότι όσο πιο μεγάλος είναι ο ανταγωνισμός που υπάρχει στο δίκτυο, τόσο πιο μικρή πρέπει να είναι η πιθανότητα αλλαγής θέσης καναλιού, καθώς τότε έχουμε μικρότερη πιθανότητα σύγκρουσης δευτερεύοντα χρήστη σε ένα slot. Οι μετρήσεις σε αυτήν την ενότητα αλλά και στην προηγούμενη (3.3.3.1) δείχνουν ότι γενικά είναι καλύτερο οι χρήστες να επιμένουν στις αρχικές τους επιλογές και να μην αλλάζουν με μεγάλη ευκολία τη σειρά των καναλιών στη

λίστα τους. Αυτό οφείλεται στο γεγονός ότι (δεδομένου ότι έχουμε υποθέσει ότι όλοι οι χρήστες είναι ομοιόμορφα κατανομημένοι στο χώρο) είναι δύσκολο δευτερεύοντες χρήστες να επιλέξουν ακριβώς την ίδια λίστα τριών καναλιών με άλλους δευτερεύοντες χρήστες. Παρόλο ότι δεν είναι αποδοτικό να γίνονται συχνές αλλαγές θέσεων των καναλιών στη λίστα των χρηστών, είναι όμως αναγκαίο να υπάρχουν αυτοί οι μηχανισμοί, για να μπορεί ο χρήστης να αλλάξει τη λίστα του όταν αυτή δεν είναι αποδοτική. Με βάση τα αποτελέσματα που παραθέτουμε στα σχήματα 3.5-3.7 αλλά και άλλες μετρήσεις που πήραμε και δεν παρουσιάζουμε για οικονομία χώρου, επιλέξαμε ως κατάλληλη τιμή για την πιθανότητα αλλαγής θέσης καναλιού το 0.3.

### 3.4.3.3 Σύγκριση αλγορίθμων ICESA και ICESA using Pmd



**Figure 3.8 Απόδοση αλγορίθμων σε άνισα φορτωμένο σαν συνάρτηση των πιθανοτήτων λανθασμένης ανίχνευσης,  $P_{md}$ , και λανθασμένου συναγερμού,  $P_{fa}$ .**



**Figure 3.9** Απόδοση αλγορίθμων σε άνισα φορτωμένο σαν συνάρτηση των πιθανοτήτων λανθασμένης ανίχνευσης,  $P_{md}$ , και λανθασμένου συναγερμού,  $P_{fa}$ .

Έπειτα από μετρήσεις που έγιναν (παραθέτονται δύο ενδεικτικά σχήματα παραπάνω), παρατηρήσαμε ότι ο αλγόριθμος ICESA (ο οποίος χρησιμοποιεί μόνο τη χωρητικότητα κάθε καναλιού για την εύρεση του βάρους που έχει το κανάλι αυτό), αποδίδει καλύτερα η τουλάχιστον το ίδιο με τον αλγόριθμο ICESA using Pmd. Αυτό οφείλεται στο γεγονός ότι η απόσταση του δευτερεύοντα πομπού από τον πρωτεύοντα πομπό και η απόσταση του δευτερεύοντα πομπού από τον πρωτεύοντα δέκτη διαφέρουν. Η εκτίμηση της πιθανότητας  $P_{md}$  καθορίζεται από την πρώτη απόσταση που αναφέραμε παραπάνω, ενώ η εκτίμηση της χωρητικότητας από τη δεύτερη. Είναι λοιπόν φανερό ότι το κανάλι με τη μεγαλύτερη χωρητικότητα δεν έχει κατ' ανάγκη τη μικρότερη πιθανότητα λανθασμένης ανίχνευσης. Υπάρχει λοιπόν πιθανότητα δευτερεύοντες χρήστες να τοποθετήσουν κανάλια με σχετικά μικρή χωρητικότητα στις τρεις θέσεις της λίστας επειδή σε αυτά έχει εκτιμηθεί μικρή πιθανότητα λανθασμένης ανίχνευσης, και έτσι τα συνολικά bits που θα μεταδώσουν πάνω σε αυτά θα είναι ασφαλώς μικρότερα από ότι με τον αλγόριθμο ICESA. Αξίζει να σχολιαστεί ότι ο αλγόριθμος ICESA using Pmd πετυχαίνει μικρότερη πιθανότητα σύγκρουσης SU σε ένα slot, δίχως όμως αυτό να συμβάλει στην αύξηση της απόδοσης του συστήματος.

Παράλληλα, ο αλγόριθμος ICESA using Pmd έχει αυξημένη πολυπλοκότητα σε σχέση με τον ICESA καθώς χρειάζεται να εκτιμήσει την πιθανότητα  $P_{md}$ . Σημαντικό είναι ότι εκτιμάται μόνο η πιθανότητα  $P_{md}$  προς τον πρωτεύοντα χρήστη ενώ στη διάρκεια της προσομοίωσης κάθε δευτερεύων χρήστης πρέπει να ανιχνεύσει πολλές

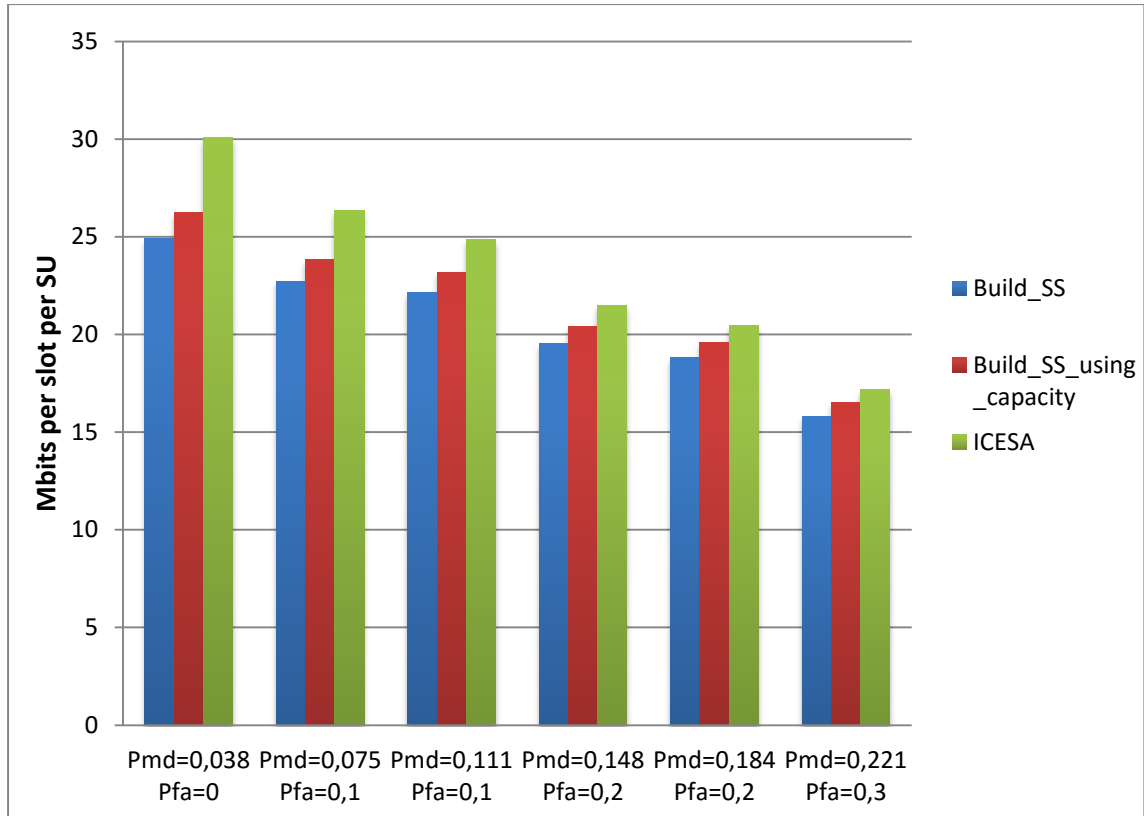
φορές (στη δεύτερη και τρίτη ευκαιρία μετάδοσης) την ύπαρξη άλλου δευτερεύοντα χρήστη στο φάσμα. Τότε ο SU θα υπόκειται διαφορετική πιθανότητα  $P_{md}$  μιας και θα απέχει διαφορετική απόσταση από τον SU ο οποίος στέλνει εκείνη τη στιγμή στο συγκεκριμένο κανάλι, σε σχέση με αυτή που απέχει από τον πρωτεύοντα χρήστη (από την οποία καθορίζεται η εκτιμώμενη  $P_{md}$ ). Συνοψίζοντας όλα τα παραπάνω, ο αλγόριθμος που τελικά προτείνουμε και θα συγκρίνουμε με τον Build SS είναι ο ICESA (που χρησιμοποιεί μόνο τη χωρητικότητα κάθε καναλιού για τον υπολογισμό των βαρών).

Δοκιμαστήκαν και άλλες μετρικές για τα βάρη των καναλιών οι οποίες χρησιμοποιούσαν την πιθανότητα  $P_{md}$  σε συνδυασμό με τη χωρητικότητα, αλλά δεν βρέθηκε καμία με καλύτερα αποτελέσματα από αυτά του αλγορίθμου ICESA.

#### 3.4.3.4 Αποτελέσματα προσομοιώσεων για $N_p > N_s$

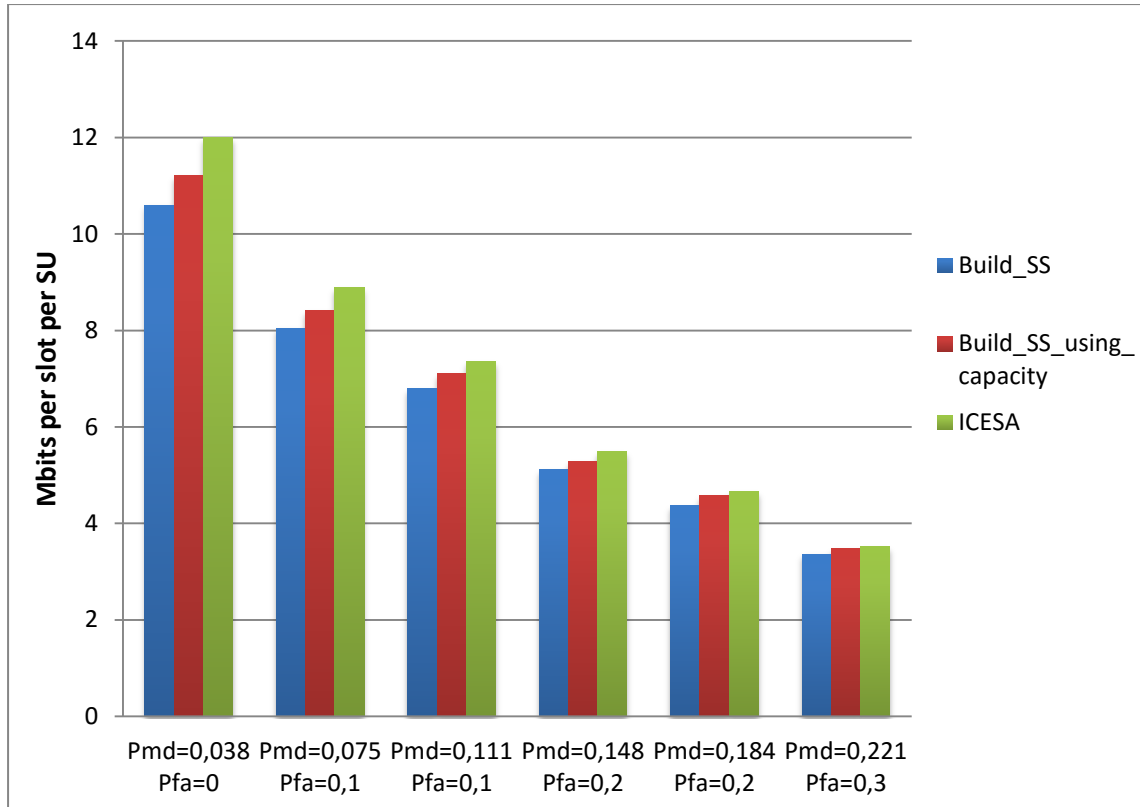
Υπενθυμίζουμε ότι σε αυτήν την περίπτωση έχουμε υποθέσει ότι ο αριθμός των καναλιών είναι  $N_p=10$  και των δευτερευόντων χρηστών είναι  $N_s=5$ . Στην περίπτωση όπου τα κανάλια είναι χαμηλά φορτωμένα έχουμε υποθέσει ότι  $\lambda_i=0.3$  πακέτα/slot,  $i=1 \dots N_p$ , ενώ στην περίπτωση που τα κανάλια είναι υψηλά φορτωμένα  $\lambda_i=0.7$  πακέτα/slot,  $i=1 \dots N_p$ . Τέλος όταν τα κανάλια είναι άνισα φορτωμένα  $\lambda_i = \{0.1, 0.1, 0.2, 0.2, 0.3, 0.3, 0.4, 0.4, 0.5, 0.5\}$ ,  $i = 1, \dots, N_p$ .

##### 3.4.3.4.1 Περίπτωση όπου τα κανάλια είναι χαμηλά φορτωμένα



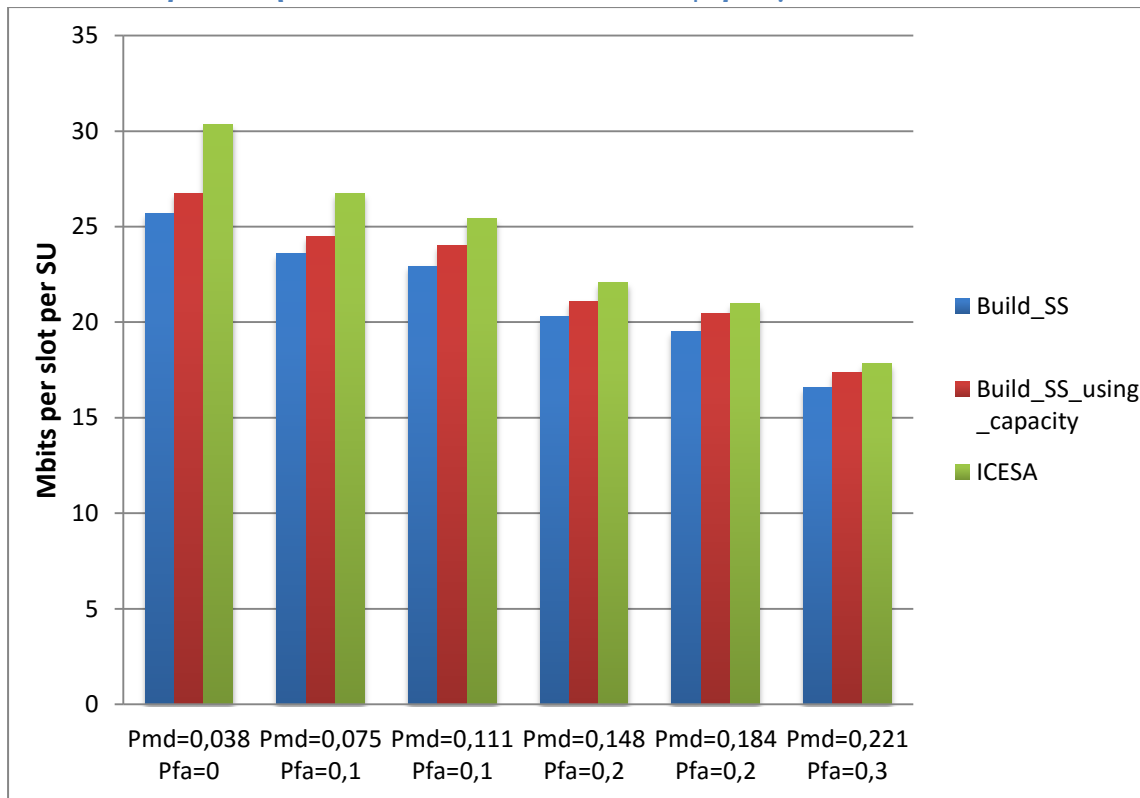
**Figure 3.10** Απόδοση αλγορίθμων σε χαμηλά φορτωμένο δίκτυο σαν συνάρτηση των πιθανοτήτων λανθασμένης ανίχνευσης,  $P_{md}$ , και λανθασμένου συναγερμού,  $P_{fa}$ .

#### 3.4.3.4.2 Περίπτωση όπου τα κανάλια είναι υψηλά φορτωμένα



**Figure 3.11** Απόδοση αλγορίθμων σε υψηλά φορτωμένο δίκτυο σαν συνάρτηση των πιθανοτήτων λανθασμένης ανίχνευσης,  $P_{md}$ , και λανθασμένου συναγερμού,  $P_{fa}$ .

#### 3.4.3.4.3 Περίπτωση όπου τα κανάλια είναι άνισα φορτωμένα



**Figure 3.12** Απόδοση αλγορίθμων σε άνισα φορτωμένο σαν συνάρτηση των πιθανοτήτων λανθασμένης ανίχνευσης,  $P_{md}$ , και λανθασμένου συναγερμού,  $P_{fa}$ .

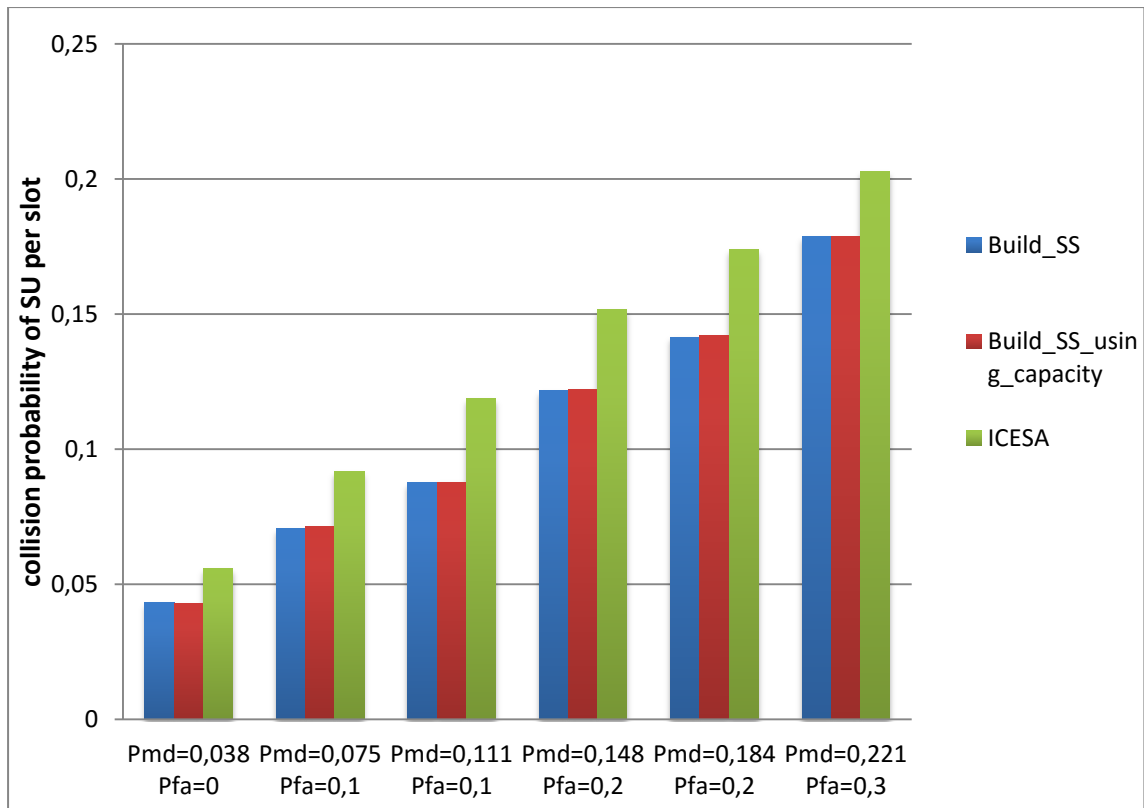


Figure 3.13 Πιθανότητα σύγκρουσης δευτερεύοντα χρήστη σε ένα slot σε υψηλά φορτωμένο δίκτυο σαν συνάρτηση των πιθανοτήτων λανθασμένης ανίχνευσης,  $P_{md}$ , και λανθασμένου συναγερμού,  $P_{fa}$ .

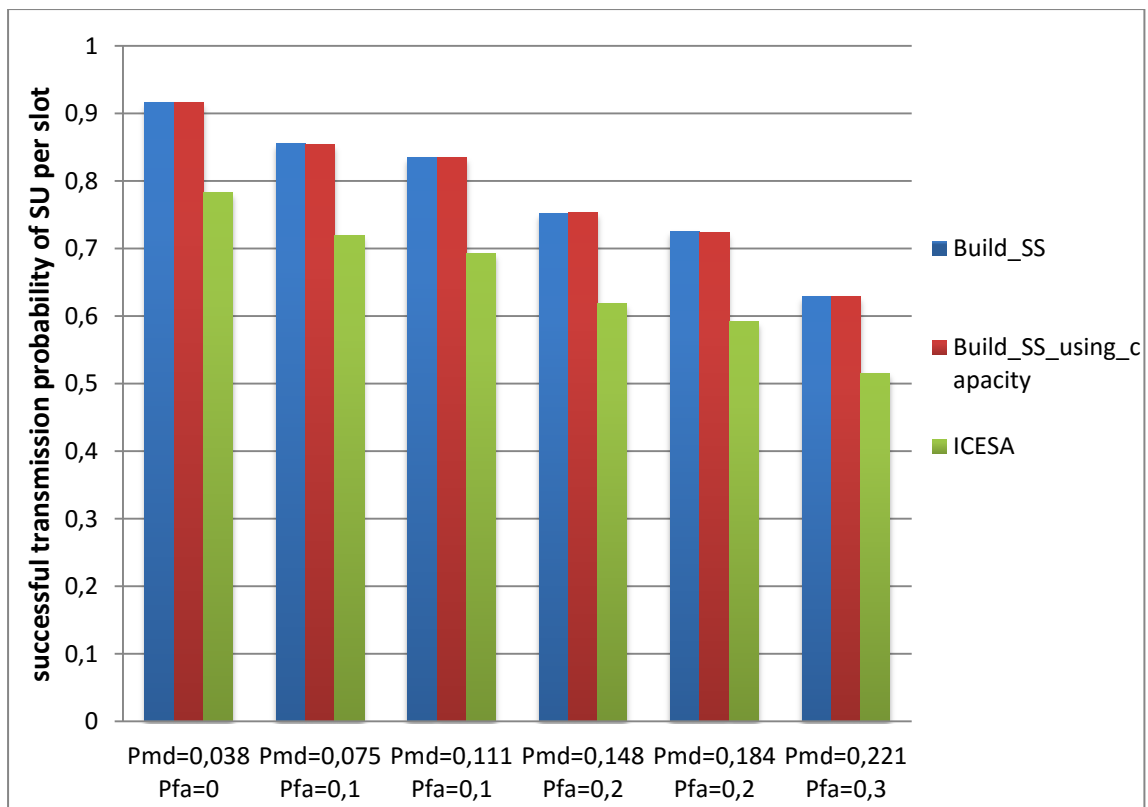


Figure 3.14 Πιθανότητα επιτυχημένης μετάδοσης δευτερεύοντα χρήστη σε ένα

*slot σε υψηλά φορτωμένο δίκτυο σαν συνάρτηση των πιθανοτήτων λανθασμένης ανίχνευσης,  $P_{md}$ , και λανθασμένου συναγερμού,  $P_{fa}$ .*

#### 3.4.3.4.4 Σχολιασμός αποτελεσμάτων για την περίπτωση όπου $N_p > N_s$

Από τα αποτελέσματα των προσομοιώσεων τις παραπάνω προσομοιώσεις για τα σενάρια όπου  $N_p > N_s$  βλέπουμε ότι ο ICESA είναι αποδοτικότερος από τον Build SS και τον Build SS using capacity. Όσον αφορά τη σύγκριση του Build SS με τον Build SS using capacity, όπως είναι αναμενόμενο ο δεύτερος είναι καλύτερος, καθώς ο Build SS επιλέγει ισοπίθانا τα κανάλια σε αντίθεση με τον Build SS using capacity που τα επιλέγει με πιθανότητες εξαρτώμενες από τη χωρητικότητα τους. Έτσι οι μεταδόσεις που πετυχαίνει κάθε SU στον Build SS using capacity με μεγάλη πιθανότητα θα είναι σε κανάλια με μεγάλη χωρητικότητα (σε σχέση με τις χωρητικότητες που έχει εκτιμήσει ο κάθε χρήστης). Αντίθετα στον Build SS ο χρήστης μπορεί να επιλέξει με μεγαλύτερη πιθανότητα κανάλια που έχουν μικρή χωρητικότητα.

Από τα σχήματα 3.10 - 3.12 είναι φανερό ότι υπερिशύει ο αλγόριθμος ICESA έναντι των άλλων σε οποιαδήποτε περίπτωση κίνησης δικτύου. Αυτό οφείλεται στο γεγονός ότι στον ICESA οι SUs τοποθετούν πάντα τα τρία κανάλια με τη μεγαλύτερη χωρητικότητα στις θέσεις της λίστας τους. Έτσι πετυχαίνουν μεταδόσεις όπου καταφέρνουν να στέλνουν πακέτα μεγαλύτερου μεγέθους από αυτά του Build SS using capacity. Παράλληλα στον ICESA δημιουργείται ένα είδους ομαδοποίησης (clustering) δεδομένου ότι οι χρήστες είναι ομοιόμορφα κατανεμημένοι στο χώρο, επομένως αναμένουμε τα κανάλια να μοιράζονται στους χρήστες, αφού κάθε δευτερεύων χρήστης εκτιμάει διαφορετικές χωρητικότητες από κάθε άλλο δευτερεύοντα χρήστη.

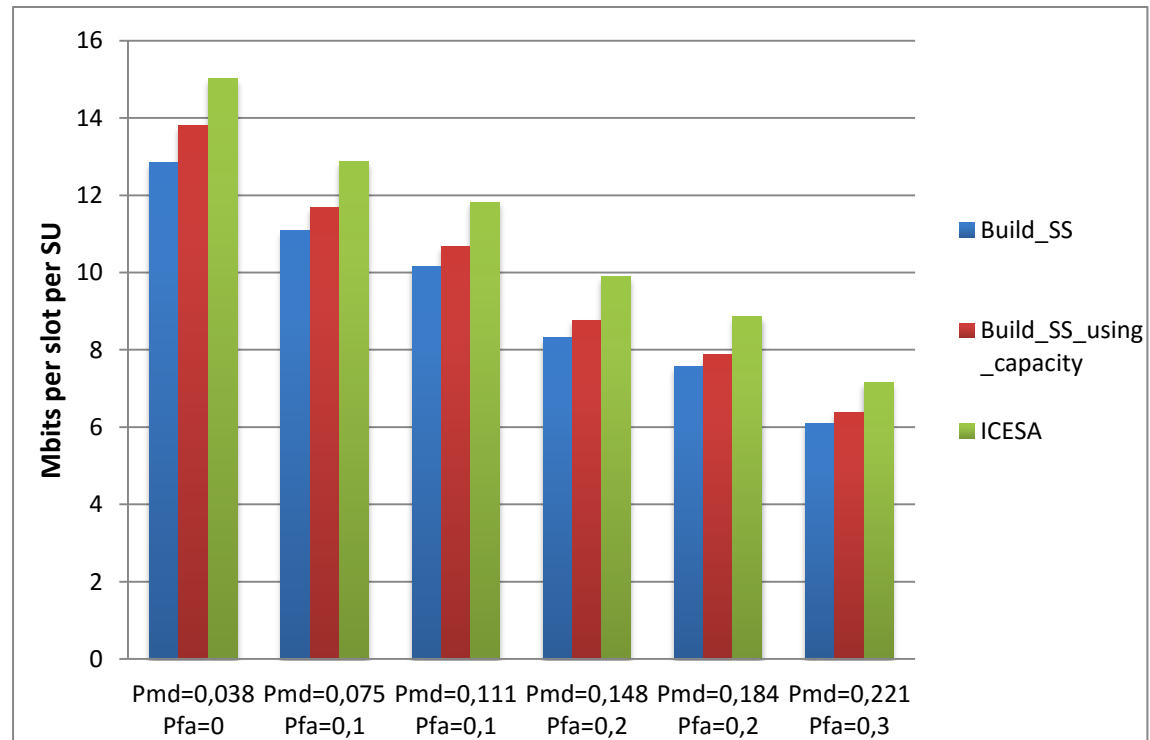
Από τα αποτελέσματα στα σχήματα 3.12-3.14 παρατηρούμε τα εξής. Ο αλγόριθμος ICESA οδηγεί σε μεγαλύτερη πιθανότητα σύγκρουσης σε ένα slot (μεταξύ SU-SU ή SU-PU) αφού η κίνηση είναι χαμηλή και υπάρχουν πολλά κανάλια, με αποτέλεσμα η ελευθερία που δίνεται στους χρήστες στους αλγορίθμους τύπου Build SS για επιλογή καναλιού, να ενεργεί θετικά ως προς τις συγκρούσεις. Όταν υπάρχει υψηλή κίνηση στο σύστημα ο ICESA πετυχαίνει χαμηλότερη πιθανότητα σύγκρουσης σε ένα slot αφού πλέον η τυχαία επιλογή καναλιού δεν είναι αποδοτική καθώς όλα τα κανάλια είναι υψηλά φορτωμένα. Επιπλέον βάσει των αποτελεσμάτων του σχήματος 3.14, ο αλγόριθμος ICESA επιτυγχάνει μεν μικρότερη πιθανότητα επιτυχημένης μετάδοσης δευτερεύοντα χρήστη σε ένα slot από αυτήν του αλγορίθμου Build SS using capacity, όμως στον αλγόριθμο ICESA οι SUs επιτυγχάνουν μεταδόσεις σε κανάλια υψηλότερης χωρητικότητας. Έτσι στον αλγόριθμο ICESA καταφέρνει κάθε SU με λιγότερες συνολικά επιτυχημένες μεταδόσεις να στείλει μεγαλύτερο όγκο δεδομένων.



### 3.4.3.5 Αποτελέσματα προσομοιώσεων για $N_p < N_s$

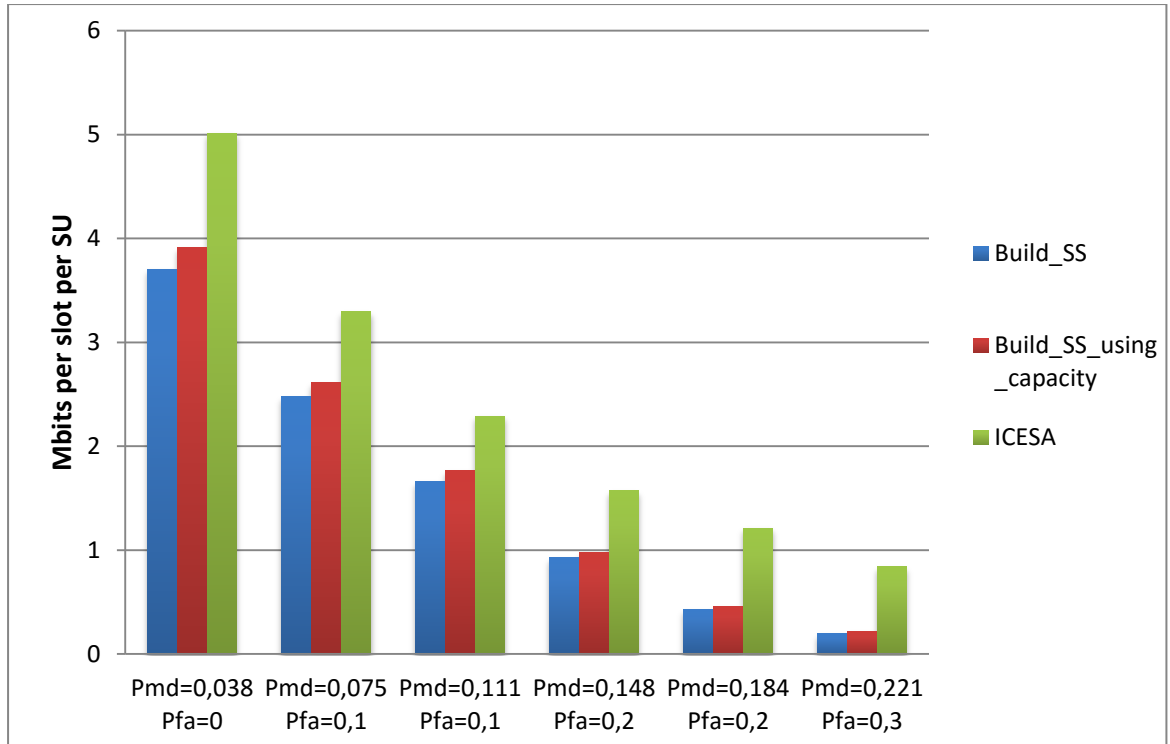
Για τις παρακάτω προσομοιώσεις ισχύει για τις παραμέτρους ( $N_p$ ,  $N_s$  και  $\lambda_i$ ) ότι ισχύει στις προσομοιώσεις της ενότητας 3.4.3.4.

#### 3.4.3.5.1 Περίπτωση όπου τα κανάλια είναι χαμηλά φορτωμένα



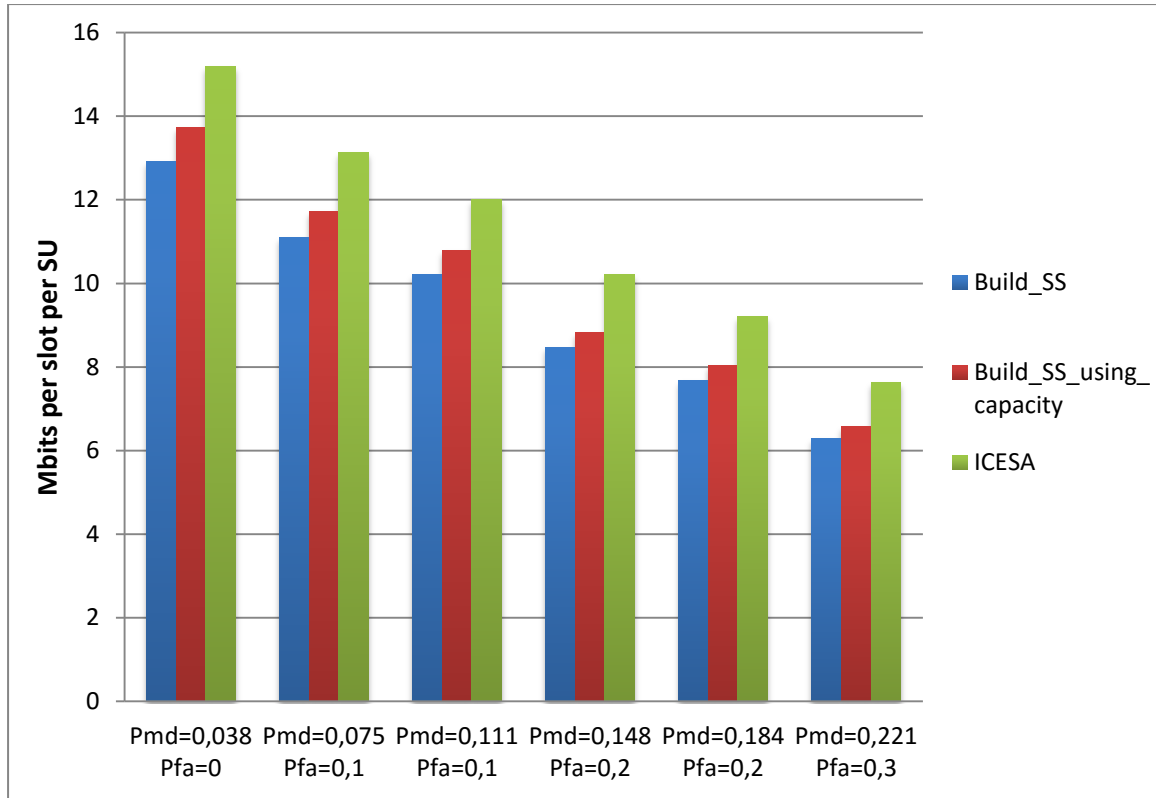
**Figure 3.15** Απόδοση αλγορίθμων σε χαμηλά φορτωμένο δίκτυο σαν συνάρτηση των πιθανοτήτων λανθασμένης ανίχνευσης,  $P_{md}$ , και λανθασμένου συναγερμού,  $P_{fa}$ .

#### 3.4.3.5.2 Περίπτωση όπου τα κανάλια είναι υψηλά φορτωμένα

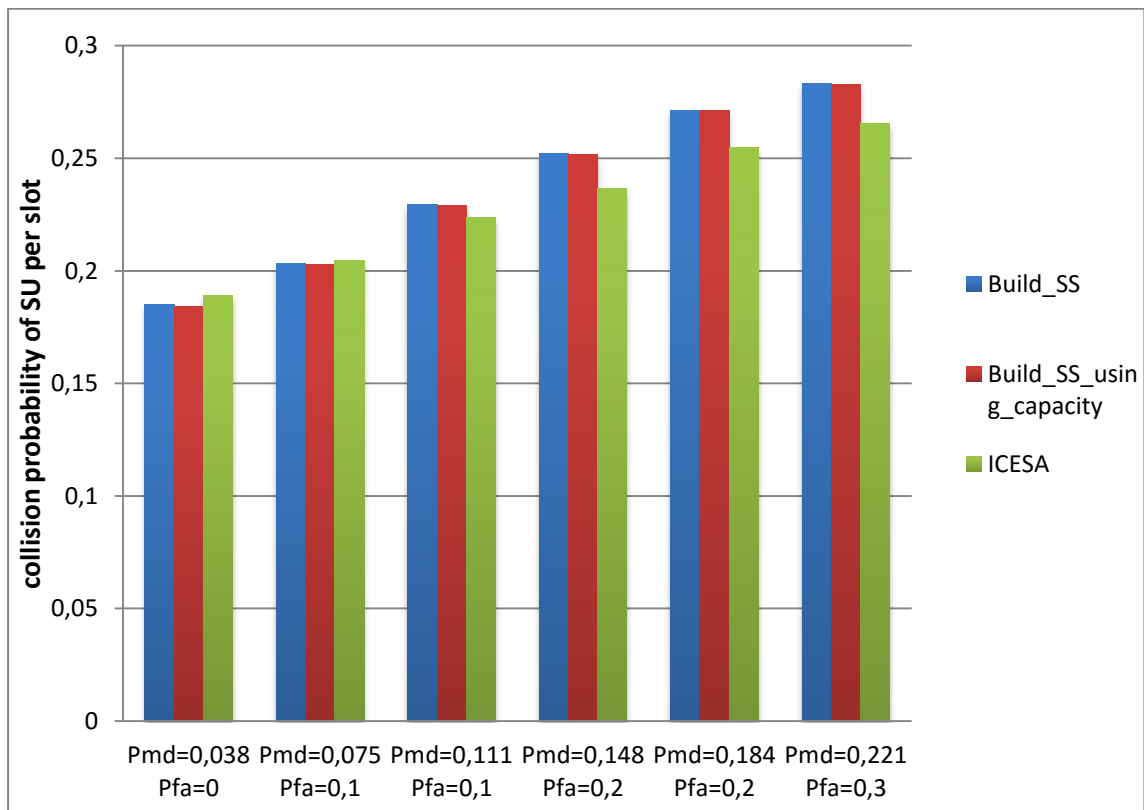


**Figure 3.16** Απόδοση αλγορίθμων σε υψηλά φορτωμένο δίκτυο σαν συνάρτηση των πιθανοτήτων λανθασμένης ανίχνευσης,  $P_{md}$ , και λανθασμένου συναγερμού,  $P_{fa}$ .

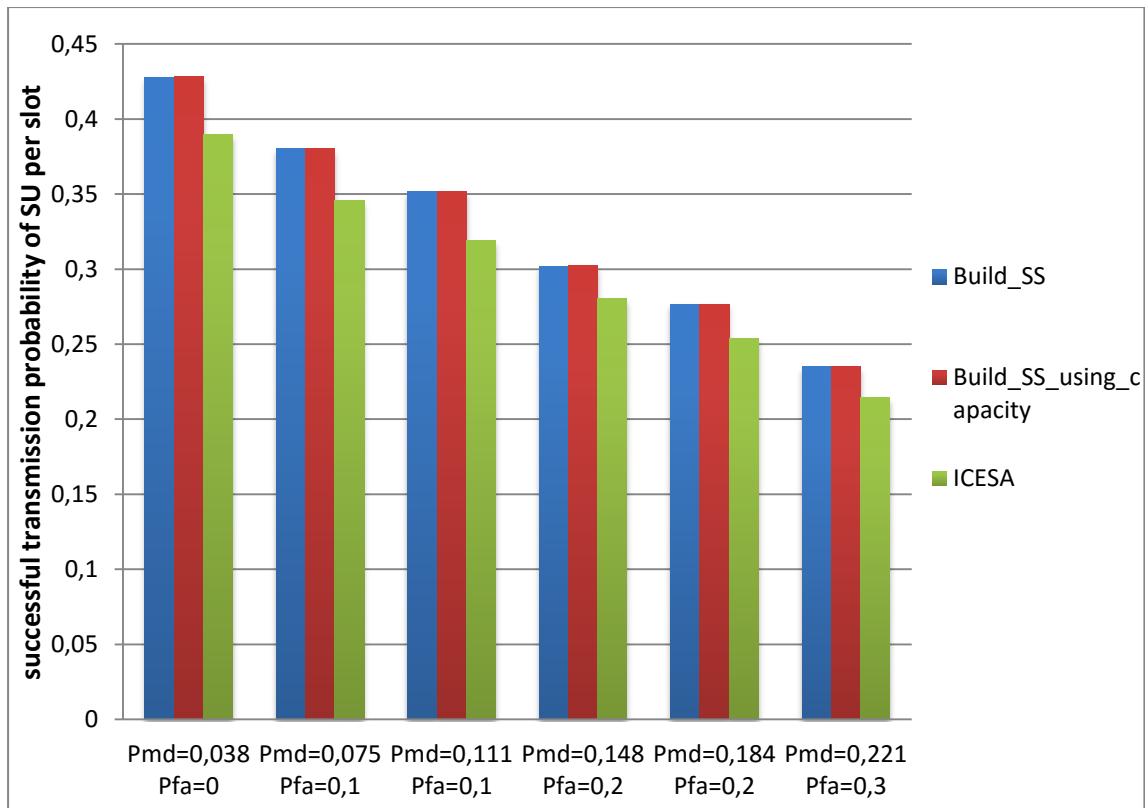
#### 3.4.3.5.3 Περίπτωση όπου τα κανάλια είναι άνισα φορτωμένα



**Figure 3.17** Απόδοση αλγορίθμων σε άνισα φορτωμένο δίκτυο σαν συνάρτηση των πιθανοτήτων λανθασμένης ανίχνευσης,  $P_{md}$ , και λανθασμένου συναγερμού,  $P_{fa}$ .



**Figure 3.18** Πιθανότητα σύγκρουσης δευτερεύοντα χρήστη ανά slot σε άνισα φορτωμένο δίκτυο σαν συνάρτηση των πιθανοτήτων λανθασμένης ανίχνευσης,  $P_{md}$ , και λανθασμένου συναγερμού,  $P_{fa}$ .



**Figure 3.19** Πιθανότητα επιτυχημένης μετάδοσης δευτερεύοντα χρήστη ανά slot σε άνισα φορτωμένο δίκτυο σαν συνάρτηση των πιθανοτήτων λανθασμένης ανίχνευσης,  $P_{md}$ , και λανθασμένου συναγερμού,  $P_{fa}$ .

#### 3.4.3.5.4 Σχολιασμός αποτελεσμάτων για την περίπτωση όπου $N_p < N_s$

Από τα αποτελέσματα των προσομοιώσεων για τα σενάρια όπου  $N_p < N_s$  βλέπουμε ότι ο ICESA είναι αποδοτικότερος από τον Build SS και τον Build SS using capacity. Όσον αφορά τη σύγκριση του Build SS με τον Build SS using capacity ισχύει ότι και στην περίπτωση που  $N_p > N_s$ , δηλαδή είναι αναμενόμενο να είναι καλύτερος ο δεύτερος, καθώς ο Build SS επιλέγει ισοπίθانا τα κανάλια σε αντίθεση με τον Build SS\_using\_capacity που τα επιλέγει με πιθανότητες ανάλογες της χωρητικότητας τους. Παρακάτω γίνεται η σύγκριση του αλγορίθμου Build SS\_using\_capacity με τον ICESA.

Από τα σχήματα 3.15 - 3.17 είναι ξεκάθαρο ότι υπερिशύει ο αλγόριθμος ICESA έναντι των άλλων σε οποιαδήποτε περίπτωση κίνησης. Αυτό οφείλεται στο ίδιο γεγονός που αναλύσαμε και στην ενότητα 3.4.3.4.4, ότι στον ICESA αλγόριθμο οι SUs πετυχαίνουν μεταδόσεις στα κανάλια με τη μεγαλύτερη χωρητικότητα.

Από τα αποτελέσματα στα σχήματα 3.17-3.19 παρατηρούμε ότι ο αλγόριθμος ICESA πετυχαίνει πάντα μεγαλύτερη απόδοση από τους υπολοίπους, καθώς κάθε χρήστης μπορεί να στείλει μεγαλύτερο όγκο δεδομένων στη διάρκεια της λειτουργίας του

συστήματος. Παράλληλα όπως φαίνεται από το σχήμα 3.18 στις δύο πρώτες τιμές που διαγράμματος ο ICESA αλγόριθμος έχει μεγαλύτερη πιθανότητα σύγκρουσης σε ένα slot (μεταξύ SU-SU ή SU-PU). Αυτό οφείλεται στο γεγονός ότι στον Build SS using capacity όταν η πιθανότητα λανθασμένης ανίχνευσης και η πιθανότητα λανθασμένου συναγερμού είναι κοντά στο 0, ο χρήστης μπορεί πιο εύκολα να επιλέξει τυχαία κάποιο κανάλι και να μεταδώσει σε αυτό. Αντίθετα στις υπόλοιπες τιμές του διαγράμματος ο ICESA πετυχαίνει μικρότερη πιθανότητα σύγκρουσης σε ένα slot αφού πλέον η τυχαία επιλογή καναλιού δεν είναι αποδοτική, καθώς οι δευτερεύοντες χρήστες επιλέγουν κάποιο κανάλι και με μικρή πιθανότητα μεταδίδουν επιτυχημένα σε αυτό ώστε να το τοποθετήσουν στη λίστα τους.

Τέλος αξίζει να σχολιαστεί ότι για κάποιες τιμές των πιθανοτήτων  $P_{md}$  και  $P_{fa}$  ο αλγόριθμος ICESA έχει μικρότερη πιθανότητα σύγκρουσης σε ένα slot για κάθε SU αλλά και μικρότερη πιθανότητα επιτυχημένης μετάδοσης σε σχέση με τον αλγόριθμο Build SS using capacity. Σε αυτές τις περιπτώσεις τα κανάλια στις λίστες των χρηστών είναι καλύτερα μοιρασμένα από ότι στους αλγορίθμους Build SS και Build SS using capacity. Όσο μεγαλώνουν οι πιθανότητες  $P_{md}$  και  $P_{fa}$  και δεδομένου ότι υπάρχει υψηλός ανταγωνισμός μεταξύ των SUs, τόσο περισσότερο οδηγούνται σε συνεχείς αναζητήσεις καναλιών οι SUs στους αλγορίθμους τύπου Build SS, καθώς δεν αποθηκεύουν με ευκολία κανάλι στη λίστα τους.

## ΚΕΦΑΛΑΙΟ 4<sup>ο</sup>: Επίλογος

Στο κεφάλαιο αυτό αναλύεται η συμβολή της διπλωματικής εργασίας και προτείνονται ιδέες για μελλοντικές επεκτάσεις.

### 4.1. Συνεισφορά της διπλωματικής εργασίας

Στο πρώτο μέρος της διπλωματικής εργασίας, προτείναμε καταναμεημένους (κυρίως) αλγόριθμους, όπου το μόνο που γνωρίζουν οι SUs για το σύστημα είναι το πλήθος των καναλιών. Τα αποτελέσματα των προσομοιώσεων (Κεφάλαιο 1) έδειξαν ότι οι αλγόριθμοι με τις στρατηγικές που προτείναμε βελτιώνουν τη μέση κανονικοποιημένη απόδοση ανά δευτερεύοντα χρήστη σε σχέση με άλλες έρευνες που έχουν γίνει, [1], [2].

Αρχικά προτείναμε τον καταναμεημένο αλγόριθμο Build SS συγκρίναμε τις διάφορες στρατηγικές και καταλήξαμε στο ποιά στρατηγική είναι η αποδοτικότερη, επιλέγοντας ανάλογα και τις κατάλληλες παραμέτρους. Από τη σύγκριση των αποτελεσμάτων με αντίστοιχες έρευνες στην βιβλιογραφία, είδαμε ότι ο αλγόριθμος Build SS επιτυγχάνει κάθε δευτερεύον χρήστη να μεταδίδει μεγαλύτερο όγκο δεδομένων, μειώνει την πιθανότητα σύγκρουσης των SUs και αυξάνει το συνολικό αριθμό των επιτυχημένων μεταδόσεων, σε σύγκριση με τον Distributed algorithm [1] αλλά και με τον PPRA [2].

Πιο συγκεκριμένα, αρχικά δείξαμε ότι ο Build SS αλγόριθμος αποδίδει καλύτερα στα πολλαπλά σενάρια που εξετάστηκαν. Είναι αποδοτικότερο ο δευτερεύον χρήστης να αποθηκεύει ένα κανάλι στη λίστα του μόνο όταν επιτύχει μετάδοση σε αυτό. Έτσι μπορούν τα κανάλια να μοιραστούν κατάλληλα στους χρήστες μειώνοντας τις συνολικές συγκρούσεις στο σύστημα. Παράλληλα, από τα αποτελέσματα συμπεράναμε ότι είναι αποδοτικότερο κάθε SU να διαχειρίζεται ξεχωριστά κάθε κανάλι (δηλαδή τότε πρέπει να το διαγράψει ή να το τοποθετήσει στη λίστα του) και όχι συνολικά όπως στο [1], όπου τοποθετούνται και διαγράφονται ομαδικά τα κανάλια στη λίστα του χρήστη. Επίσης, δείξαμε ότι όταν υπάρχει υψηλός ανταγωνισμός μεταξύ των SUs, είναι προτιμότερο οι δευτερεύοντες χρήστες να επιμένουν περισσότερο στα κανάλια που υπάρχουν στη λίστα τους, και να μην τα διαγράφουν με ευκολία, καθώς το τελευταίο οδηγεί σε μεγαλύτερο ανταγωνισμό μεταξύ των SUs και σε μεγαλύτερη αστάθεια στο σύστημα, υπό την έννοια ότι οι δευτερεύοντες χρήστες αναζητούν συνεχώς νέα κανάλια. Επιπλέον, από τη σύγκριση των αλγορίθμων Build SS και Build SS coop, είδαμε ότι δεν είναι πάντα αποδοτικότερο να υπάρχει συνεργασία μεταξύ των SUs, με τον τρόπο που εμείς την ορίσαμε, καθώς ορισμένες φορές αυτή έχει μη επιθυμητά αποτελέσματα, αφού αυξάνει τον ανταγωνισμό και τις συγκρούσεις μεταξύ των SUs.

Στο δεύτερο μέρος της διπλωματικής εργασίας θεωρήσαμε ένα CRN όπου οι χρήστες είναι ομοιόμορφα κατανομημένοι στο χώρο. Βάσει του θεωρήματος Shannon-Fano, η χωρητικότητα κάθε καναλιού εξαρτάται από την απόσταση που έχει ο πομπός από το δέκτη. Επίσης βάσει της απόστασης δύο χρηστών υπολογίζεται και η πιθανότητα λανθασμένης ανίχνευσης που υπόκειται κάθε χρήστης στην προσπάθεια του να ανιχνεύσει το σήμα κάποιου άλλου χρήστη στο φάσμα. Έτσι προτείναμε τον αλγόριθμο ICESA, ο οποίος τοποθετεί στη λίστα του τα τρία κανάλια με τη μεγαλύτερη εκτιμώμενη χωρητικότητα, αλλά και τον αλγόριθμο Build SS using capacity που η πιθανότητα επιλογής κάθε καναλιού βασίζεται στην εκτιμώμενη χωρητικότητα του από κάθε δευτερεύοντα χρήστη.

Τα αποτελέσματα έδειξαν ότι οι αλγόριθμοι με τις στρατηγικές που προτείναμε βελτιώνουν τη μέση κανονικοποιημένη απόδοση ανά δευτερεύοντα χρήστη σε σύγκριση με τον αλγόριθμο Build SS που προτάθηκε στο πρώτο μέρος της εργασίας άρα και σε σύγκριση με τους αλγορίθμους στα [1] και [2].

Αρχικά, εξετάσαμε τους αλγορίθμους ICESA και ICESA using  $P_{md}$  όπου από τα αποτελέσματα φάνηκε ότι η εκτίμηση της πιθανότητας λανθασμένης ανίχνευσης  $P_{md}$ , κάθε SU για κάθε PU και η χρησιμοποίηση της στη μετρική των βαρών για κάθε κανάλι δεν είναι αρκετά αποδοτική. Αυτό οφείλεται κυρίως στο γεγονός ότι σε κάθε κανάλι μπορεί τη δεδομένη χρονική στιγμή να στέλνει και κάποιος SU με συνέπεια ο χρήστης να πρέπει να ανιχνεύσει το σήμα αυτού του SU, και όχι το σήμα του PU (με βάση το οποίο έχει εκτιμήσει την πιθανότητα λανθασμένης ανίχνευσης,  $P_{md}$ , που χρησιμοποιεί στη μετρική του). Με τον αλγόριθμο ICESA using  $P_{md}$  οι SUs πετυχαίνουν περισσότερες μεταδόσεις αλλά σε κανάλια με μικρότερη χωρητικότητα σε σχέση με τις μεταδόσεις στον αλγόριθμο ICESA. Έτσι η εκτίμηση των πιθανοτήτων αυτών για τα σήματα των PUs και η χρησιμοποίησή τους στη μετρική επιλογής των καναλιών των SUs, είναι και μη αποδοτική πέραν του ότι είναι πολύπλοκη.

Έπειτα, συγκρίνοντας τους αλγορίθμους Build SS και Build SS using capacity δείξαμε ότι η απόδοση του συστήματος βελτιώνεται αισθητά όταν ο δευτερεύων χρήστης χρησιμοποιεί τη χωρητικότητα που έχει εκτιμήσει για κάθε κανάλι για τον υπολογισμό της πιθανότητας επιλογής του κάθε καναλιού. Τέλος, συγκρίνοντας τους αλγορίθμους ICESA και Build SS using capacity είδαμε ότι η στρατηγική του πρώτου στην οποία ο χρήστης επιλέγει τα τρία μεγαλύτερα ως προς τη χωρητικότητα κανάλια πετυχαίνει μεγαλύτερη μέση κανονικοποιημένη απόδοση ανά δευτερεύοντα χρήστη. Αν και σε πολλές περιπτώσεις με τον αλγόριθμο ICESA είχαμε περισσότερες συγκρούσεις μεταξύ των SUs και λιγότερες επιτυχημένες μεταδόσεις, τα συνολικά bits που έστειλε κάθε SU ήταν περισσότερα, καθώς πετύχαινε μεταδόσεις στα κανάλια υψηλής χωρητικότητας.

## 4.2. Ιδέες μελλοντικών επεκτάσεων

Ενδιαφέρον θα παρουσίαζε η μελέτη αλγορίθμων όπου οι δευτερεύοντες χρήστες συνεργάζονται μεταξύ τους, στέλνοντας τις μετρήσεις τους σε ένα κεντρικό συντονιστή ο οποίος θα αποφάσιζε αν τα κανάλια είναι απασχολημένα ή όχι, αλλά και ποιοί χρήστες θα μεταδώσουν σε ποιο κανάλι. Σε αυτήν την περίπτωση μπορούσε να χρησιμοποιηθεί ο αλγόριθμος Build SS σε ένα σενάριο με συνεργασία των SUs ώστε κάθε χρήστης να επιλέγει διαφορετικό κανάλι από κάθε άλλο χρήστη στην ίδια ευκαιρία μετάδοσης, όπως στο [18]. Έτσι θα μειωνόταν ο ανταγωνισμός και οι συγκρούσεις μεταξύ των SUs.

Επιπλέον, θα μπορούσε να δοκιμαστούν οι προτεινόμενοι αλγόριθμοι στο δεύτερο μέρος της εργασίας και να υλοποιηθούν νέοι στην περίπτωση συνεργασίας μεταξύ των SUs, ώστε να δημιουργηθούν clusters με SUs βάσει των αποστάσεων μεταξύ τους, αλλά και στην περίπτωση που αυτοί κινούνται στο χώρο. Στο πρώτο σενάριο οι SUs ομαδοποιούνται (clustering) βάσει της απόστασης που έχουν μεταξύ τους, θα επιλέγουν ένα αρχηγό (cluster head) για κάθε cluster, στον οποίο θα στέλνουν τις μετρήσεις τους, και αυτός θα αποφασίζει ανάλογα για την κατάσταση του κάθε καναλιού και θα ενημερώνει το συντονιστή ο οποίος με τη σειρά του θα αναθέτει κανάλια στους χρήστες, παρόμοια στρατηγική υλοποιείται στα [12]-[15]. Το δεύτερο ενδιαφέρον σενάριο που αναφέραμε και παραπάνω, είναι αυτό όταν οι δευτερεύοντες χρήστες δεν παραμένουν στατικοί στο χώρο. Έτσι οι SUs θα πρέπει συνεχώς να αναπροσαρμόζουν τα κανάλια στη λίστα τους, αφού θα αλλάζει η απόσταση που έχουν από κάθε πρωτεύοντα δέκτη με αποτέλεσμα να αλλάζει συνεχώς η εκτιμώμενη, από τους SUs, χωρητικότητα κάθε καναλιού.



## Βιβλιογραφία

- [1] Theodorou, M., Paterakis M. 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”, Chapter 3, July 2015/
- [2] 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.
- [3] 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.
- [4] FCC, “Spectrum policy task force report,” ET Cocket 02-155, in *Federal Communications Commission*, Nov 2002.
- [5] 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.
- [6] B. Wang and K. J. R. Liu, “Advances in cognitive radio networks: A survey,” *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 1, pp. 5–23, Jan. 2011.
- [7] I. Mitola, J. and J. Maguire, G. Q., “Cognitive radio: making software radios more personal,” *IEEE Personal Commun. Mag.*, vol. 6, no. 4, pp. 13–18, Aug. 1999.
- [8] Y. Zeng, Y. Liang, Z. Lei, S. Oh, F. Chin, and S. Sun, “Worldwide regulatory and standardization activities on cognitive radio,” in *New Frontiers in Dynamic Spectrum*, 2010 IEEE Symposium on. IEEE, pp. 1–9, 2010.
- [9] 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.
- [10] I. F. Akyildiz, W.-Y. Lee, and K. R. Chowdhury, “CRAHNs: Cognitive radio ad hoc networks,” *Ad Hoc Netw.*, vol. 7, no. 5, pp. 810–836, Jul. 2009
- [11] I. Christian, S. Moh, I. Chung, and J. Lee, “Spectrum Mobility in Cognitive Radio Networks,” *IEEE Communications Magazine*, vol. 50, no. 6, pp. 114–121, Jun. 2012

- [12] A. S. B. Kozal, M. Merabti, F. Bouhafs, “*Energy Efficient Clustering Approach for Cooperative Spectrum Sensing in Cognitive Radio Networks*”, in Proc. Of the 13th Annual Post Graduate Symposium on the Convergence of Telecommunications, Networking and Broadcasting (PGNet), 2012.
- [13] A. Malady and C. da Silva, “Clustering methods for distributed spectrum sensing in cognitive radio systems,” in Proc. of the IEEE Military Communications Conference (MILCOM), pp. 1–5, Nov. 17–19, 2008.
- [14] S. Chunhua, et al., "Cluster-Based Cooperative Spectrum Sensing in Cognitive Radio Systems," IEEE International Conference on Communications, ICC '07., pp. 2511-2515, 2007
- [15] B. Zhiquan, et al., "Cluster-based cooperative spectrum sensing for cognitive radio under bandwidth constraints," IEEE International Conference on Communication Systems (ICCS), '10, pp. 569-573, 2010
- [16] J. Zhang and Q. Zhang, “Stackelberg Game for Utility-Based Cooperative Cognitive Radio Networks,” in Proceedings of the tenth ACM international symposium on Mobile ad hoc networking and computing. ACM, pp. 23-31, Sept. 2009
- [17] 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.
- [18] R. Fan and H. Jiang, “Optimal multi-channel cooperative sensing in cognitive radio networks,” IEEE Trans. Wireless Commun., vol. 9, no. 3, pp. 1128–1138, Mar. 2010