Πανεπιστήμιο Πειραιώς
Τμήμα Διδακτικής της Τεχνολογίας και Ψηφιακών Συστημάτων

Μελέτη επίδοσης DCA σε κυψελοειδές δίκτυο

ΠΗΧΑΣ ΣΤΕΦΑΝΟΣ

Αθήνα, 22 Μαρτίου, 2009

ΠΕΡΙΕΧΟΜΕΝΑ

Κατάλογος Σχημάτων

ΠΕΡΙΛΗΨΗ

Σκοπός της παρούσης εργασίας είναι η ανάλυση της τεχνικής δυναμικής εκχώρησης καναλιού (dynamic channel allocation - DCA) σε ένα κυψελοειδές τηλεπικοινωνιακό δίκτυο και η μελέτη της επίδοσης του συστήματος αυτού με τη χρήση της παραπάνω τεχνικής..

Στο πρώτο κεφάλαιο θα αναλύσουμε την τεχνική αυτή η οποία χρησιμοποιείται ευρέος στα κινητά συστήματα WiMAX. Επίσης θα αναφέρουμε τις τεχνικές beamforming και power control, τεχνικές οι οποίες χρησιμοποιούνται σε συνδυασμό με την DCA για την περεταίρω βελτίωση της επίδοσης των τηλεπικοινωνικών συστημάτων. Τέλος για κάθε μία από τις τεχνικές αυτές θα παραθέσουμε τις σχέσεις υπολογισμού του λόγου του σήματος προς τον θόρυβο συν την παρεμβολή (CNIR)

Στο δεύτερο κεφάλαιο θα περιγράψουμε το βασικό πρόγραμμα προσομοίωσης για την υλοποίηση του DCA αλγόριθμου καθώς επίσης και τις τεχνικές beamforming και power control. Επίσης θα αναλύσουμε λεπτομερώς όλες τις συναρτήσεις του MATLAB από τις οποίες συνθέτονται τα πρόγραμματα προσομοίωση μας, όπως θα δούμε πιο κάτω.

Τέλος στο τρίτο κεφάλαιο θα παραθέσουμε και θα σχολιάσουμε τα αποτελέσματα από την εκτέλεση των προγραμμάτων προσομοίωσης. Με βάση τα αποτελέσματα αυτά θα αναλύσουμε την επίδοση του συστήματος μας μέσω των τιμών της πιθανότητας απόρριψης εισερχομένων κλήσεων (blocking probability) και της πιθανότητας εξαναγκασμένου τερματισμού μιας εν εξελίξει κλήσης (forced terminated probability).

ΚΕΦΑΛΑΙΟ 1
ΚΑΤΑΝΟΜΗ ΣΥΧΝΟΤΗΤΩΝ ΣΕ ΚΥΨΕΛΩΤΑ ΔΙΚΤΥΑ

1.1 Εισαγωγή

Η τεχνολογία WiMAX βασίζεται στο πρότυπο IEEE 802.16, ένα ανοικτό παγκόσμιο πρότυπο για μητροπολιτικά ευρυζωνικά ασύρματα δίκτυα, το οποίο αποτελείται από δύο προσθήκες την 802.16-2004 και την 802.16-2005. Η πρώτη που είναι γνωστή ως fixed WiMAX παρέχει υπηρεσίες παρόμοιες με τις ευρυζωνικές υπηρεσίες ADSL χρησιμοποιώντας όμως ασύρματο μέσο μετάδοσης. Η δεύτερη που είναι γνωστή ως mobile WiMAX αποτελεί βελτιωμένη έκδοση του fixed WiMAX παρέχοντας επιπλέον υπηρεσίες για κινητούς σταθμούς. Ο όρος WiMAX προέκυψε από την ονομασία του φόρουμ το οποίο συγκροτήθηκε με σκοπό να δημιουργήσει ένα σύνολο πιστοποιημένων προϊόντων και υπηρεσιών συμβατά με το παραπάνω πρότυπο [1].

Ένα από τα σημαντικότερα προβλήματα στη λειτουργία ενός συστήματος mobile WiMAX, αλλά και γενικότερα ενός κυψελοειδούς συστήματος, είναι το πώς να χρησιμοποιήσει αποτελεσματικά το διαθέσιμο εύρος ζώνης για να παρέχει την καλύτερη δυνατή υπηρεσία (quality of service) σε όσο το δυνατόν περισσότερους χρήστες (πελάτες). Προκειμένου να παρέχουν όσο γίνεται καλύτερη κατανομή των φασματικών πόρων τα κυψελωτά συστήματα εκμεταλλεύονται το γεγονός ότι ένα κανάλι επικοινωνίας (μια ζώνη συχνοτήτων) μπορεί να χρησιμοποιηθεί ταυτόχρονα από πολλούς χρήστες εάν η μεταξύ τους απόσταση είναι τέτοια, ώστε οι κλήσεις τους να μην αλληλοπαρεμβάλλονται μεταξύ τους (απόσταση επαναχρησιμοποίησης συχνοτήτων) [2]-[5].

Με βάση το παραπάνω δεδομένο τα κανάλια διατίθενται στα κελιά και στις κλήσεις που γίνονται μέσα σε αυτά με τέτοιο τρόπο, ώστε να εξυπηρετούνται οι ανάγκες κάθε χρήστη ανάλογα με το βαθμό ποιότητας της απαιτούμενης υπηρεσίας, διατηρώντας παράλληλα το επίπεδο παρεμβολής σε ένα ανεκτό όριο.

Οι δύο βασικές τεχνικές ανάθεσης καναλιών που χρησιμοποιούνται είναι η τεχνική Αμετάβλητης Εκχώρησης Καναλιού (Fixed Channel Assignment - FCA) και η Δυναμική Εκχώρηση Καναλιού (Dynamic Channel Assignment - DCA).

Κατά την **FCA** τεχνική [6], στην κάθε κυψέλη εκχωρείται μόνιμα ένας συγκεκριμένος αριθμός καναλιών βάση του φορτίου κίνησης και του μεγέθους της κυψέλης. Το βασικότερο μειονέκτημα αυτής της τεχνικής είναι η χαμηλή επίδοση της κατά την περίπτωσή που η υπάρχουσα συνολική κίνηση δεν είναι ομοιόμορφη αλλά ακολουθεί σημαντικές διακυμάνσεις. Αυτό έχει σαν αποτέλεσμα την μη βέλτιστη αξιοποίηση της χωρητικότητας του συστήματος και κατ' επέκταση την μη βέλτιστη ποιότητα στις παρεχόμενες υπηρεσίες

Κατά την **DCA** τεχνική [7]-[10], στον κάθε σταθμό βάσης της κάθε κυψέλης ανατίθεται όλο το σύνολο των διαθέσιμων καναλιών. Στη συνέχεια ένα ή περισσότερα από αυτά διατίθενται δυναμικά στο κάθε χρήστη προκειμένου να ικανοποιηθούν οι απαιτήσεις της παρεχόμενης υπηρεσίας όπως αυτές καθορίζονται από το βαθμό ποιότητας της. Η μεγάλη προσαρμοστικότητα που παρουσιάζει η DCA τεχνική στις μεταβολές της τηλεπικοινωνιακής κίνησης την κάνει πολύ πιο αξιόπιστη (μείωση της πιθανότητας απόρριψης κλήσεων, αύξηση της χωρητικότητας του συστήματος, καλύτερη ποιότητα υπηρεσιών) από την αντίστοιχη FCA, υπό το κόστος της αύξησης της περιπλοκότητας στη σχεδίαση του συστήματος.

Η εφαρμογή της DCA τεχνικής παράλληλα με την τεχνική του **beamforming** και του **power control** εξασφαλίζει ακόμα μεγαλύτερη βελτίωση στην επίδοση των τηλεπικοινωνιακών συστημάτων.

1.2 Dynamic channel allocation (DCA)

Κατά την τεχνική αυτή δεν υπάρχει καμία μόνιμη κατανομή των καναλιών στις κυψέλες του συστήματος. Όλο το σύνολο τους είναι διαθέσιμο για αυτές, στις οποίες ανατίθενται με τρόπο δυναμικό κατά "κλήση προς κλήση". Το μεγάλο πλεονέκτημα της DCA σε σχέση με την αντίστοιχη τεχνική FCA είναι η προσαρμοστικότητά της στις μεταβολές της τηλεπικοινωνιακής κίνησης και του συνολικού αριθμού των χρηστών [6].

Ο βασικότερος στόχος της τεχνικής DCA είναι να αναπτυχθεί μια στρατηγική ανάθεσης καναλιών, η οποία να ελαχιστοποιεί το συνολικό αριθμό των απορριπτόμενων κλήσεων [7].

Υπάρχουν δύο βασικές μορφές υλοποίησης του DCA αλγόριθμου, η συγκεντρωτική (centralized) και η κατανεμημένη (distributed). Στην πρώτη περίπτωση, όλα τα αιτήματα για την διανομή των καναλιών διαχειρίζονται από έναν κεντρικό ελεγκτή καναλιών ο οποίος είναι και ο υπεύθυνος για τη διαχείριση όλων των καναλιών του συστήματος έχοντας σαν βασικό σκοπό την διασφάλιση της απαιτούμενης ποιότητας του σήματος. Στην δεύτερη περίπτωση, η αντίστοιχη διαχείριση πραγματοποιείται από έναν αριθμό ελεγκτών οι οποίοι είναι κατανεμημένοι σε όλο το εύρος του δικτύου, καθένας από τους οποίους είναι υπεύθυνος για την διαχείριση και διανομή των καναλιών όσον αφορά τις περιβάλλουσες σε αυτόν κυψέλες [7]. Ο centralized DCA αλγόριθμος είναι αυτός που εξασφαλίζει την βέλτιστη απόδοση του συστήματος αφού η υλοποίηση του βασίζεται στη "σφαιρική" γνώση του τηλεπικοινωνιακού συστήματος. Καθίσταται όμως μη εφαρμόσιμος, εξαιτίας των μεγάλων καθυστερήσεων στις οποίες υπόκειται το σύστημα λόγω της αύξησης τόσο της υπολογιστικής ισχύος, όσο και της επικοινωνίας μεταξύ των σταθμών βάσης [7].

Ο distributed DCA αλγόριθμος βασίζεται κυρίως σε μετρήσεις της ποιότητας του σήματος και ειδικότερα σε μέτρησεις της τιμής του λόγου της ισχύος του σήματος ως προς τον θόρυβο και τις παρεμβολές (CNIR) για κάθε νέα εισερχόμενη κλήση. Ο πιο γνωστός distributed DCA αλγόριθμος είναι ο LOLIA (Local Optimized Least Interference Algorithm). Αυτός βασίζεται στον υπολογισμό της παρεμβολής μεταξύ των χρηστών που χρησιμοποιούν τον ίδιο ραδιοδίαυλο και στη συνέχεια στην σύγκριση της τιμής του λόγου CNIR με μία προκαθορισμένη τιμή κατωφλίου [5], [7].

1.3 Τεχνική beamforming

Η τεχνική του **beamforming** στηρίζεται στη χρήση κατευθυντικών κεραιών οι οποίες είναι γνωστές ως έξυπνες κεραίες (smart antennas). Πιο συγκεκριμένα είναι μία τεχνική επεξεργασίας σήματος που χρησιμοποιείται στις σειρές των στοιχεικοκεραιών για την κατευθυντική μετάδοση ή λήψη των σημάτων [1], [11].

Με την τεχνική αυτή μεταβάλλοντας τα βάρη της στοιχειοκεραίας στον κάθε σταθμό βάσης μπορούμε είτε να εστιάσουμε την δέσμη της προς εκείνον τον κινητό σταθμό τον οποίο εμείς θέλουμε (downlink), είτε να εντοπίσουμε την δέσμη που προέρχεται από εκείνον τον σταθμό τον οποίο εμείς επιθυμούμε (uplink). Με τον

τρόπο αυτό επιτυγχάνουμε τη στροφή του διαγράμματος ακτινοβολίας της κεραίας προς την επιθυμητή κατεύθυνση εξασφαλίζοντας έτσι, από την μία την ενίσχυση του κύριου σήματος και από την άλλη την καταστολή των παρεμβολών.

Η παραπάνω κατεύθυνση η οποία μπορεί να είναι είτε φυσική είτε υπό μια έννοια μαθηματική χαρακτηρίζει τις δύο βασικές μορφές της τεχνικής αυτής. Η πρώτη είναι η (DOA)-based beamforming (φυσική κατεύθυνση) η οποία βασίζεται στον υπολογισμό της γωνίας άφιξης (direction of arrival) του σήματος στην κεραία του σταθμού βάσης με τη βοήθεια διαφόρων τεχνικών επεξεργασίας σήματος. Με βάση τη γωνία αυτή ο beamformer δημιουργεί για όλα τα στοιχεία της κεραίας τον πίνακα weight vector τον οποίο και χρησιμοποιεί για να λάβει ή να μεταδώσει εκείνο το σήμα το οποίο προέρχεται ή προορίζεται από ή για τον επιθυμητό χρήστη. Η δεύτερη τεχνική είναι η eigen-beamforming (μαθηματική κατεύθυνση), η οποία εκμεταλλεύεται την κρουστική απόκριση (channel impulse response) για καθένα από τα στοιχεία της κεραίας προκειμένου να βρει τα κατάλληλα βάρη τα οποία ικανοποιούν κάποια προκαθορισμένα κριτήρια επίδοσης (μεγιστοποίηση SNR, ελαχιστοποίηση MSE) [1].

Η τεχνική beamforming μπορεί να εφαρμοστεί τόσο στην άνω (uplink), όσο και στην κάτω ζεύξη (downlink) και είναι πολύ διαφορετική για καθεμία από αυτές. Στην πρώτη περίπτωση εφαρμόζεται κατά τη λήψη του σήματος στην κεραία του σταθμού βάσης, δηλαδή αφού το σήμα έχει μεταδοθεί μέσω του καναλιού διάδοσης, ενώ στη δεύτερη περίπτωση εφαρμόζεται πριν από την εκπομπή του. Αυτό σημαίνει ότι κατά τη σύνθεση του διαγράμματος ακτινοβολίας θα πρέπει να λάβουμε υπόψη και το περιβάλλον διάδοσης. Στην περίπτωσή αυτής της εργασίας, για λόγους απλότητας, θα επικεντρωθούμε στην βελτίωση μόνο της άνω ζεύξης (uplink).

Η τεχνική αυτή τις περισσότερες φορές συνδυάζεται παράλληλα και με τον έλεγχο ισχύος στους κινητούς σταθμούς των χρηστών εξασφαλίζοντας ακόμα καλύτερα αποτελέσματα όσον αφορά την επίδοση του συστήματος [1], [5]. Εφαρμόζοντας τον έλεγχο ισχύος ρυθμίζουμε το μέγεθος της εκπεμπόμενης ισχύος από τον κάθε κινητό σταθμό σε σχέση με ένα μέγιστο επιτρεπόμενο όριο ισχύος, έτσι ώστε το επίπεδο της λαμβανόμενης ισχύος στον κάθε σταθμό βάσης να παραμένει σταθερό ανεξαρτήτου της μεταξύ τους απόστασης. Το αποτέλεσμα που πετυχαίνεται είναι η μείωση της συνολικής ισχύος των παρεμβολών και κατά συνέπεια η βελτίωση του λόγου CNIR.

1.4 Σκοπός της εργασίας

Για να μελετήσουμε την επίδοση του κυψελοειδούς WiMAX συστήματος με τη χρήση του DCA αλγόριθμου θα χρησιμοποιήσουμε το **MATLAB**. Με αυτό θα γράψουμε το πρόγραμμα προσομοίωσης, μέσω του οποίου θα υλοποιήσουμε το σύστημά μας.

Αρχικά θα υλοποιήσουμε τον DCA LOLIA αλγόριθμο σε ένα κυψελοειδές τηλεπικοινωνιακό σύστημα λαμβάνοντας ως δεδομένα διάφορες παραμέτρους, όπως το μέγεθος της κάθε κυψέλης, τον αριθμό των διαθέσιμων καναλιών, το περιβάλλον διάδοσης, κ.λ.π., και θα υπολογίσουμε εκείνες τις παραμέτρους οι οποίες σχετίζονται άμεσα με την επίδοση του συστήματος μας, οι οποίες είναι

- η πιθανότητα απόρριψης εισερχομένων κλήσεων "blocking probability" και
- η πιθανότητα εξαναγκασμένου τερματισμού μιας εν εξελίξει κλήσης "forced terminated probability".

Η επεξεργασία των αποτελεσμάτων γίνεται μέσω γραφικών παραστάσεων και πινάκων που θα δημιουργήσουμε βάσει των αποτελεσμάτων που θα συλλέξουμε από την εκτέλεση του προγράμματος προσομοίωσης. Για παράδειγμα θα εξετάσουμε το πώς μεταβάλλεται η πιθανότητα απόρριψης κλήσεων συναρτήσει του αριθμού των χρηστών που μπορεί να εξυπηρετήσει η κάθε κυψέλη.

Στη συνέχεια θα υλοποιήσουμε τον DCA αλγόριθμο εφαρμόζοντας παράλληλα την τεχνική beamforming με την χρήση έξυπνων κεραιών (smart antennas) στους σταθμούς βάσης. Εδώ θα διαπιστώσουμε την επίδραση της τεχνικής αυτής αρχικά στην βελτίωση της τιμής του CNIR και στη συνέχεια στη βελτίωση της επίδοσης του συστήματός μας.

Τέλος, θα δούμε το πως βελτιώνεται περαιτέρω η επίδοση του συστήματος μας, εφαρμόζοντας παράλληλα και τον έλεγχο ισχύος (Power Control) για τον κάθε κινητό σταθμό.

1.5 Σχέσεις CINR για κυψελωτά συστήματα

Στο πρόγραμμα προσομοίωσης που θα πραγματοποιήσουμε στη συνέχεια, για τον υπολογισμό του λόγου CNIR στην κεραία του κάθε σταθμού βάσης (uplink) θα χρησιμοποιήσουμε τις ακόλουθες σχέσεις.

1. Η τιμή του λόγου C / (N + I) στην κεραία του κάθε σταθμού βάσης υπολογίζεται από την παρακάτω εξίσωση [5]:

$$CNIR = \frac{P_0 \cdot d_0^{-a} \cdot 10^{\frac{\xi_0}{10}}}{\mathrm{N} + \sum_{i=1}^{18} \sum_{j=1}^{user} P_{ij} \cdot d_{ij}^{-a} \cdot 10^{\frac{\xi_{ij}}{10}}}$$

όπου α είναι ο συντελεστής απωλειών διάδοσης, P_{ij} είναι η εκπεμπόμενη ισχύς από τον j χρήστη της i κυψέλης, d_{ij} είναι η απόσταση μεταξύ του j χρήστη της i κυψέλης και του κύριου σταθμού βάσης, και ξ_{ij} είναι η τιμή της εξασθένισης η οποία προκαλείται εξαιτίας της σκέδασης μεταξύ του j χρήστη της i κυψέλης και του κύριου σταθμού βάσης

2. Η τιμή του λόγου C / (N + I) στην κεραία του κάθε σταθμού βάσης (στοιχειοκεραία) εφαρμόζοντας την τεχνική beamforming υπολογίζεται από την παρακάτω εξίσωση [5]:

$$CNIR = \frac{P_0 \cdot d_0^{-a} \cdot 10^{\frac{\xi_0}{10}} \times 10^{\frac{G_{HB_0}(\theta_{HM_0})}{10}} \times 10^{\frac{G_{VB_0}(\theta_{VM_0})}{10}}}{N + \sum_{i=1}^{18} \sum_{j=1}^{user} P_{ij} \cdot d_{ij}^{-a} \cdot 10^{\frac{\xi_{ji}}{10}} \times 10^{\frac{G_{HB_0}(\theta_{HM_{ji}})}{10}} \times 10^{\frac{G_{VB_0}(\theta_{VM_{ij}})}{10}}}$$

όπου α είναι ο συντελεστής απωλειών διάδοσης, P_{ij} είναι η εκπεμπόμενη ισχύς από τον j χρήστη της i κυψέλης, d_{ij} είναι η απόσταση μεταξύ του j χρήστη της i κυψέλης και του κύριου σταθμού βάσης, και ξ_{ij} είναι η τιμή της εξασθένισης η οποία προκαλείται εξαιτίας της σκέδασης μεταξύ του j χρήστη της i κυψέλης και του κύριου σταθμού βάσης.

Ο όρος $G_{HB0}(\theta_{HMij})$ αντιστοιχεί στο οριζόντιο κέρδος της κεραίας του σταθμού βάσης B_0 προς τη θ_{HMij}, όπου θ_{HMij} είναι η κατεύθυνση του κινητού

σταθμού M_{ij} από τον σταθμό βάσης B_0 στο οριζόντιο επίπεδο και ο όρος $G_{VB0}(\theta_{HMij})$ στο κατακόρυφο κέρδος κεραίας του κεντρικού σταθμού βάσης B_0 προς τη θ_{VMij}, όπου θ_{VMij} είναι η κατεύθυνση του κινητού σταθμού M_{ij} από τον σταθμό βάσης B_0 στο κατακόρυφο επίπεδο.

Θα πρέπει εδώ να επισημάνουμε ότι το συνολικό κέρδος της κεραίας του σταθμού βάσης B_0 ισούται με το άθροισμα του οριζόντιου και κατακόρυφου κέρδους (σε dB). Το πρώτο αντιστοιχεί στο κέρδος της κεραίας ως προς το οριζόντιο επίπεδο, ενώ το δεύτερο ως προς το κατακόρυφο.

Η μαθηματική συνάρτηση, η οποία μας δίνει τις τιμές του κέρδους συναρτήσει της γωνίας έλευσης φ για το συγκεκριμένο τύπο κεραίας (στοιχειοκεραία) είναι η παρακάτω [5]:

$$G_{HB_i}(\phi) = \begin{cases} 10\log_{10}\cos^n(\phi) & -\pi/2 \le \varphi \le \pi/2 \\ \chi & \text{σε κάθε άλλη περίπτωση} \end{cases}$$

Στην παραπάνω εξίσωση οι μεταβλητές **φ** και **x** αντιστοιχούν στην γωνία έλευσης και στην τιμή κέρδους κατά την αντίθετη κατεύθυνση. Τέλος η μεταβλητή **n** είναι μία παράμετρος μέσω της οποίας υπολογίζεται η τιμή του πλάτους της δέσμης του κύριου λοβού της κεραίας βάση της ακόλουθης σχέσης.

$$n = \frac{\log_{10}\sqrt{1/2}}{\log_{10}\cos\left(\theta\pi/180\right)}$$

3. Η τιμή του λόγου C / (N + I) στην κεραία του κάθε σταθμού βάσης εφαρμόζοντας επιπλέον και τον έλεγχο ισχύος για το κάθε χρήστη του συστήματος υπολογίζεται από την παρακάτω σχέση [5]:

$$CNIR = \frac{\frac{P_{\max}\cdot d_0}{R}\times d_0^{-a}\times 10^{\frac{\xi_0}{10}}\times 10^{\frac{G_{HB_0}(\theta_{HM_0})}{10}}\times 10^{\frac{G_{VB_0}(\theta_{VM_0})}{10}}}{N+\sum_{i=1}^{18}\sum_{j=1}^{user}\frac{P_{\max}\cdot d_{ij}}{R}\times d_{ij}^{-a}\times 10^{\frac{\xi_{ji}}{10}}\times 10^{\frac{G_{HB_0}(\theta_{HM_{ji}})}{10}}\times 10^{\frac{G_{VB_0}(\theta_{VM_{ij}})}{10}}}$$

όπου P_{max} η μέγιστη τιμή της ισχύος η οποία μπορεί να εκπεμφθεί από τον κάθε κινητό σταθμό (αντιστοιχεί στην τιμή της ισχύος στα όρια της κυψέλης) και R η ακτίνα της κυψέλης.

ΚΕΦΑΛΑΙΟ 2
ΠΡΟΣΟΜΟΙΩΣΗ ΚΥΨΕΛΩΤΟΥ ΣΥΣΤΗΜΑΤΟΣ

2.1 Περιγραφή συστήματος προσομοίωσης

Η προσομοίωσή του κυψελωτού συστήματος αποτελείται από τις ακόλουθες συναρτήσεις όπως φαίνονται στον παρακάτω πίνακα:

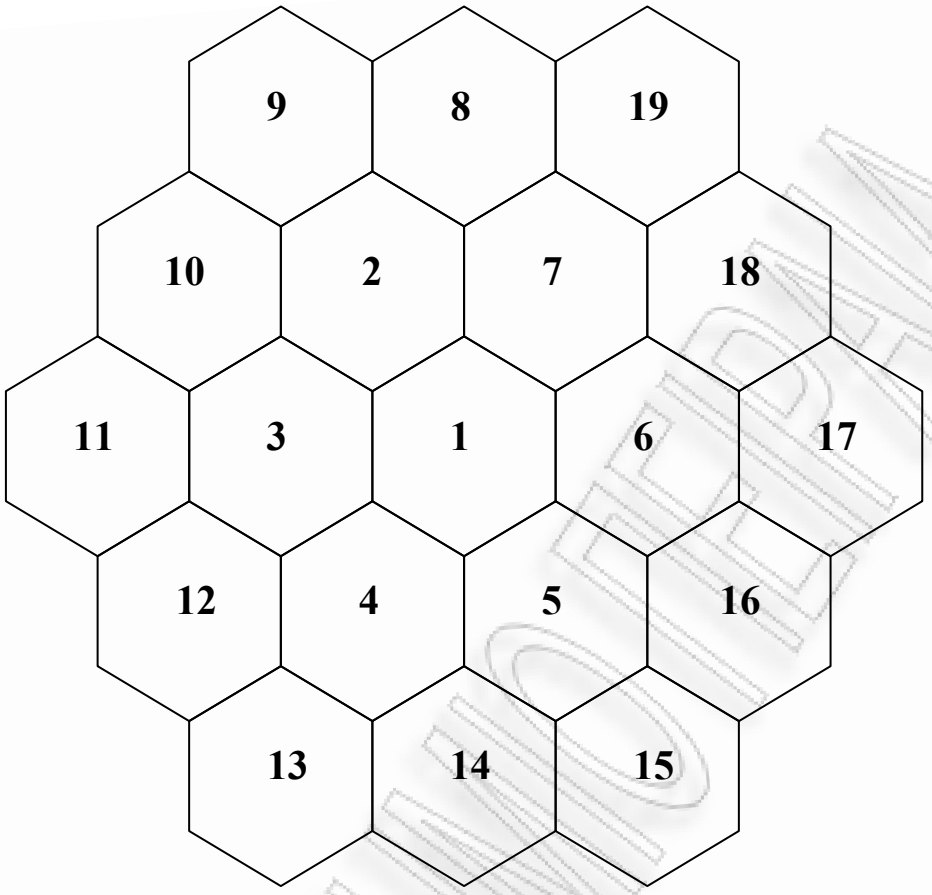
dcamain.m	Το βασικό πρόγραμμα
basest.m	Καθορίζει τη θέση του κάθε σταθμού βάσης
wrap.m	Καθορίζει ποιες είναι οι γειτονικές κυψέλες για κάθε μία από τις κυψέλες του συστήματος σύμφωνα με τεχνική cell-wrapping
cellmesh.m	Χωρίζει την κυψέλη σε σημεία τα οποία αποτελούν τις πιθανές θέσεις για κάθε έναν από τους χρήστες και αποθηκεύει τις συντεταγμένες τους σε έναν πίνακα
holdtime.m	Υπολογίζει τη συνολική διάρκεια μιας κλήσης (ακολουθεί εκθετική κατανομή με μέση τιμή ht)
shadow.m	Υπολογίζει την εξασθένιση του σήματος λόγω του φαινομένου της σκίασης (ακολουθεί λογαριθμο-κανονική κατανομή με τυπική απόκλιση σ)
dist.m	Υπολογίζει την εξασθένιση του σήματος λόγω της απόστασης (path loss)
DCA_theory.m	Συγκρίνει την πειραματική τιμή της blocking probability με τη θεωρητική της τιμή
block_prob.m	Υπολογίζει την θεωρητική τιμή της blocking probability βάση της

	εξίσωσης του Engset.
BeamformingCNIR.m	Υπολογίζει την βελτίωση του λόγου CNIR με τη χρήση της τεχνικής Beamforming για διάφορες τιμές του ανοίγματος δέσμης του κηρίου λοβού της κεραίας
Antgain.m	Υπολογίζει την τιμή κέρδους στην κεραία του σταθμού βάσης για όλες τις γωνίες έλευσης
DCABeamforming.m	Υλοποίηση του DCA αλγόριθμου εφαρμόζοντας παράλληλα την τεχνική Beamforming
Antgain_Interf.m	Συνάρτηση η οποία υπολογίζει το κέρδος του λαμβανόμενου σήματος στην κεραία του ΣΒ για τον κάθε χρήστη του συστήματος (χρησιμοποιείται από το πρόγραμμα DCABeamforming.m)
PowerControlCNIR.m	Υπολογίζει την βελτίωση του λόγου CNIR με τη χρήση της τεχνικής Beamforming και Power Control για διάφορες τιμές του ανοίγματος δέσμης του κηρίου λοβού της κεραίας
DCAPowerControl.m	Υλοποίηση του DCA αλγόριθμου εφαρμόζοντας παράλληλα τις τεχνικές Beamforming & Power Control

ΠΙΝΑΚΑΣ 1

Το κυψελοειδές σύστημα που θα προσομοιώσουμε αποτελείται από 19 κυψέλες (σχήμα 2.1). Κατά τον υπολογισμό των παρεμβολών λαμβάνουμε υπόψη μόνο τις κυψέλες της $1^{ης}$ ζώνης. Οι κυψέλες της $2^{ης}$ ζώνης αγνοούνται θεωρώντας ότι η απόστασή τους από την κύρια κυψέλη είναι τέτοια ώστε η παρεμβολή τους να θεωρείται αμελητέα.

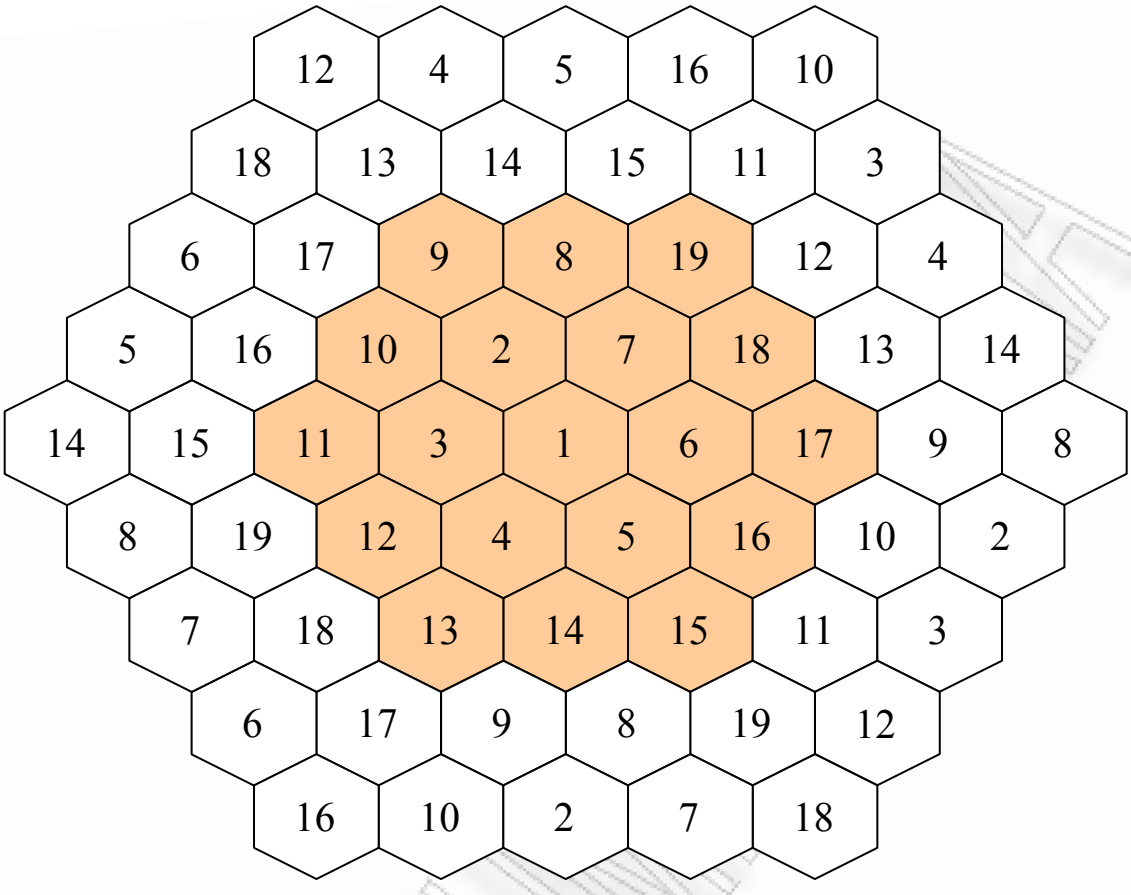
Έτσι, π.χ. για την κυψέλη 1 οι παρεμβολές της είναι οι κυψέλες 2, 3, 4, 5, 6 και 7.



ΣΧΗΜΑ 2.1
Κυψελοειδές σύστημα

Παρατηρώντας το παραπάνω σχήμα βλέπουμε ότι οι κυψέλες που βρίσκονται στα όρια του σχεδιαγράμματος (π.χ. οι κυψέλες 13, 14, και 15) συνορεύουν με μικρότερο αριθμό κυψελών από ότι κεντρικές. Αυτό έχει σαν αποτέλεσμα η συνολική παρεμβολή που δέχονται να είναι μικρότερη, γεγονός που μπορεί να οδηγήσει σε εσφαλμένα αποτελέσματα.

Για την επίλυση αυτού του προβλήματος θα χρησιμοποιήσουμε την τεχνική **cell – wrapping**. Η μορφή του κυψελοειδούς συστήματος με την τεχνική αυτή φαίνεται στο σχήμα 2.2.



ΣΧΗΜΑ 2.2
Τεχνική cell - wrapping

Με αυτή τεχνική θεωρούμε ότι οι κυψέλες που βρίσκονται στην άκρη του σχεδιαγράμματος είναι γειτονικές αυτών που βρίσκονται ακριβώς απέναντι τους. Έτσι για τις κυψέλες 13, 14 και 15 θεωρούμε ότι γειτονικές τους κυψέλες είναι οι 9, 8 και 19 αντίστοιχα. Ομοίως για τις 11, 12 και 13 οι γειτονικές τους είναι οι 19, 18 και 17. Αυτό έχει σαν αποτέλεσμα ο συνολικός αριθμός των παρεμβολέων για κάθε κυψέλη να είναι 6 και επομένως η συνολική παρεμβολή να είναι ίδια για όλες τις κυψέλες.

Τα δεδομένα που θα χρησιμοποιήσουμε για το σχεδιασμό του συστήματός μας παρουσιάζονται στον πίνακα 2.

Μέγεθος κυψέλης	Εκφράζεται μέσω της ακτίνα της
Αριθμός χρηστών	Ο μέγιστος αριθμός χρηστών που μπορεί να εξυπηρετήσει η κάθε κυψέλη
Αριθμός καναλιών	Το σύνολο των διαθέσιμων καναλιών σε κάθε κυψέλη
Συντελεστής απωλειών διάδοσης	Χαρακτηρίζει το περιβάλλον διάδοσης
Τυπική απόκλιση συντελεστή σκέδασης	Για τον υπολογισμό του συντελεστή σκέδασης (ακολουθεί λογαριθμοκανονική κατανομή με τυπική απόκλιση σ)
Μέσος χρόνος διάρκειας μια κλήσης (sec)	Για τον υπολογισμό της διάρκειας των κλήσεων (ακολουθεί εκθετική κατανομή με μέση τιμή ht)
Μέσος ρυθμός άφιξης κλήσεων (κλήσεις / hour)	Για τον υπολογισμό του ρυθμού άφιξης των κλήσεων (ακολουθεί κατανομή Poison με μέση τιμή λ)
C / N maximum (dB)	Καθορίζει τη μέγιστη τιμή της εκπεμπόμενης ισχύος από τον κινητό σταθμό (όταν αυτός βρίσκεται στα όρια της κυψέλης)
C / (N * I) threshold (dB)	Η ελάχιστη τιμή του λόγου CNIR στην κεραία του σταθμού βάσης ώστε η κλήση να μη διακοπεί
Συνολική διάρκεια της προσομοίωσης (sec)	Σε συνδυασμό με το βήμα της προσομοίωσης καθορίζει τον αριθμό επαναλήψεων.

ΠΙΝΑΚΑΣ 2

Μεταβάλλοντας ένα ή περισσότερα από αυτά τα δεδομένα μπορούμε να δούμε πώς επηρεάζεται η συμπεριφορά του συστήματος και η εν γένει συνολική του απόδοση.

Η πιθανότητα απόρριψης μιας εισερχόμενης κλήσης (blocking probability) και η πιθανότητα εξαναγκασμένου τερματισμού μιας εν εξελίξει κλήσης (forced terminated probability) είναι οι δύο βασικές παράμετροι οι οποίες σχετίζονται άμεσα με την επίδοση του συστήματος.

Η **blocking probability** εκφράζει την πιθανότητα μία εισερχόμενη κλήση να μην εξυπηρετηθεί εξαιτίας της μη ύπαρξης είτε ενός ελεύθερου καναλιού, είτε ενός καναλιού που να εξασφαλίζει την απαιτούμενη ποιότητα επικοινωνίας.

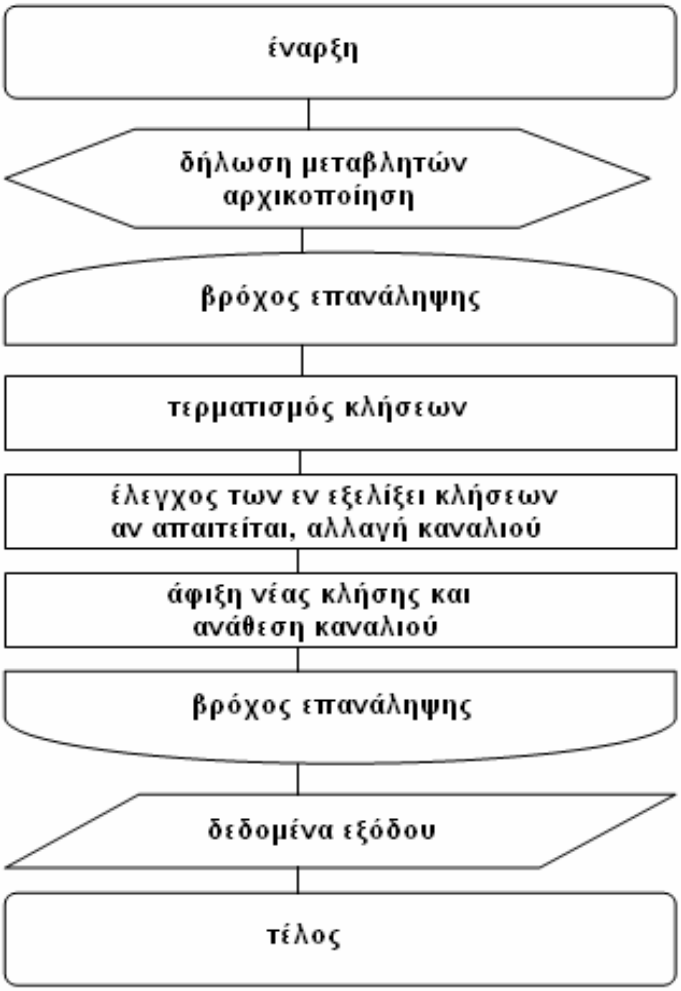
Η **forced terminated probability** εκφράζει την πιθανότητα μία εν εξελίξει κλήση να τερματιστεί λόγω της χαμηλής ποιότητας της ραδιοζεύξης ($CNIR < CNIR_{threshold}$) την δεδομένη χρονική στιγμή, χωρίς παράλληλα να βρεθεί ένας άλλος ελεύθερος ραδιοδίαυλος ο οποίος να εξασφαλίσει την απαιτούμενη ποιότητα επικοινωνίας.

Το πρόγραμμα για την μελέτη της απόδοσης του DCA αλγόριθμου περιέχει 3 βασικά στάδια

1. εξετάζεται ποιες από τις εν εξελίξει κλήσεις έχουν ολοκληρωθεί.
2. ελέγχεται για κάθε συνδεδεμένο χρήστη η ποιότητα του λόγου CNIR. Εάν αυτός είναι μικρότερος από την τιμή του κατωφλίου $CNIR_{threshold}$, τότε η ραδιοζεύξη θεωρείται χαμηλής ποιότητας και αναζητείται άμεσα κάποιο άλλο διαθέσιμο κανάλι που να ικανοποιεί τον παραπάνω περιορισμό. Στην περίπτωση που δεν βρεθεί ένα τέτοιο, τότε η κλήση τερματίζεται απότομα.
3. στο τελευταίο στάδιο εξετάζεται η περίπτωση άφιξης νέων κλήσεων. Για μια εισερχόμενη κλήση αναζητείται ένας ελεύθερος δίαυλος, για τον οποίο υπολογίζεται η τιμή του λόγου CNIR συγκρινόμενη με την τιμή του κατωφλίου $CNIR_{threshold}$. Εάν δεν βρεθεί ένας τέτοιος ο οποίος να εξασφαλίζει την απαιτούμενη τιμή για τον παραπάνω λόγο, τότε η κλήση απορρίπτεται.

2.2 DCAmain.m

Αποτελεί το κύριο πρόγραμμα προσομοίωσης, το οποίο για την ευκολότερη επεξεργασία του το έχουμε χωρίσει σε πέντε μέρη. Το πρώτο μέρος περιλαμβάνει τη δήλωση των μεταβλητών και την αρχικοποίηση τους, το δεύτερο εξετάζει την περίπτωση τερματισμού μιας κλήσης, το τρίτο ελέγχει εάν απαιτείται αλλαγή καναλιού, το τέταρτο αφορά την περίπτωση άφιξης νέων κλήσεων και το πέμπτο μέρος ασχολείται με τον υπολογισμό των μεταβλητών εξόδου και την εμφάνιση τους. Το διάγραμμα ροής του προγράμματος φαίνεται στο σχήμα 2.3



ΣΧΗΜΑ 2.3
Διάγραμμα ροής DCA αλγόριθμου

Δήλωση μεταβλητών & αρχικοποίηση:

Οι μεταβλητές που δηλώνουμε σε αυτό το μέρος αναφέρονται και στον πίνακα 2.

- Η **cnedge** (σε dB) είναι το Carrier to Noise ratio στα άκρα της κυψέλης και αποτελεί το ανώτατο όριο για την τιμή της ισχύος που μπορεί να εκπέμψει ένας κινητός σταθμός.
- Η **cnirth** (σε dB) είναι η τιμή κατωφλίου του λόγου CNIR (Carrier to Noise plus Interference ratio) στην κεραία του σταθμού βάσης, κάτω από την οποία η επικοινωνία θεωρείται μη αξιόπιστη.
- Η **lambda** (κλήσεις / ώρα) είναι η μέση τιμή του ρυθμού άφιξης κλήσεων για τον κάθε χρήστη του συστήματος.
- Η **ht** (σε sec) είναι η μέση τιμή της διάρκειας μίας κλήσης.
- Η **timestep** είναι το βήμα αύξησης της προσομοίωσης.
- Η **timend** είναι η συνολική διάρκεια της προσομοίωσης.
- Η **chnum** είναι ο συνολικός αριθμός των καναλιών που διατίθενται στο σταθμό βάσης της κάθε κυψέλης.
- Η **alpha** είναι ο συντελεστής απωλειών διάδοσης. Χαρακτηρίζει το περιβάλλον διάδοσης
- Η **sigma** είναι η τυπική απόκλιση του συντελεστή σκέδασης
- Η **usernum** είναι ένας vector 1 x 5, κάθε στήλη του οποίου περιέχει τον μέγιστο αριθμό χρηστών που μπορεί να εξυπηρετήσει η κάθε κυψέλη (5, 10, 15, 20 και 25). Η προσομοίωση επαναλαμβάνεται για όλες τις τιμές του πίνακα usernum
- Η **parameter** χρησιμοποιείται για την προσπέλαση των στοιχείων του πίνακα usernum
- Η **userinfo** είναι ένας πίνακας (19 x usernum x 6). Η κάθε γραμμή του αντιστοιχεί σε κάθε μία από τις 19 κυψέλες του συστήματος, η κάθε στήλη του αντιστοιχεί σε κάθε έναν από τους χρήστες της αντίστοιχης κυψέλης και τέλος η τρίτη διάσταση του πίνακα περιέχει πληροφορίες για την κατάσταση του κάθε χρήστη σύμφωνα με τον πίνακα 3.

userinfo(:, :, 1)	Η x συντεταγμένη του χρήστη
userinfo(:, :, 2)	Η y συντεταγμένη του χρήστη
userinfo(:, :, 3)	Η τιμή της εξασθένισης λόγω του περιβάλλοντος διάδοσης (Path Loss & Shadowing)
userinfo(:, :, 4)	Εάν ο χρήστης είναι συνδεδεμένος (έχει την τιμή 0) ή όχι (έχει την τιμή 1)
userinfo(:, :, 5)	Η συνολική διάρκεια της κλήσης
userinfo(:, :, 6)	Ο αριθμός του καναλιού που έχει ανατεθεί στο χρήστη

ΠΙΝΑΚΑΣ 3

- Η **user** είναι ο αριθμός των χρηστών σε κάθε κυψέλη
- Η **timenow** είναι η τρέχουσα τιμή του χρόνου προσομοίωσης.
- Η **blocknum** είναι ο αριθμός των απορριφθέντων κλήσεων.
- Η **forcenum** είναι ο αριθμός των βίαια τερματισμένων κλήσεων
- Η **callnum** είναι ο αριθμός των συνολικών κλήσεων
- Η **users** είναι ο αριθμός των συνδεδεμένων χρηστών
- Η **baseinfo** είναι ένας (19 x 2) πίνακας ο οποίος περιέχει τις x και y συντεταγμένες για κάθε έναν από τους 19 σταθμούς βάσης.
- Η **wrapinfo** είναι ένας (19 x 19) πίνακας, ο οποίος περιέχει τις γειτονικές κυψέλες (1ης και 2ης ζώνης) για κάθε μία από τις 19 κυψέλες του συστήματος λαμβάνοντας υπόψη την τεχνική cell – wrapping.
- Η **meshnum** περιέχει τον συνολικό αριθμό των πιθανών θέσεων που μπορεί να έχει ο κάθε χρήστης μέσα στην κάθε κυψέλη
- Η **msehpostion** είναι ένας (meshnum x 2) πίνακας που περιέχει τις x και y συντεταγμένες για κάθε μία από τις πιθανές θέσεις του κάθε χρήστη.
- Η **output** είναι ένας (5 x 5) πίνακας. Η κάθε μία από τις 5 γραμμές του περιέχει τις τελικές τιμές των μεταβλητών callnum, blocknum, forcenum, blocking probability και forced termination probability αντίστοιχα Η κάθε στήλη του περιέχει τα παραπάνω αποτελέσματα για αριθμό χρηστών 5, 10, 15, 20 και 25 αντίστοιχα (τιμές του πίνακα usernum).
- Η **check** είναι ένας πίνακας που σε κάθε μία από τις στήλες του αποθηκεύεται η τιμή της blocking probability για τη τρέχουσα χρονική στιγμή της προσομοίωσης. Η κάθε γραμμή του αντιστοιχεί σε αριθμό χρηστών 5, 10, 15, 20 και 25.

- Η **check2** είναι ένας πίνακας που σε κάθε μία από τις στήλες του αποθηκεύεται η τιμή της forced termination για τη τρέχουσα χρονική στιγμή της προσομοίωσης. Η κάθε γραμμή του αντιστοιχεί σε αριθμό χρηστών 5, 10, 15, 20, και 25.

Κατά το στάδιο της αρχικοποίησης οι πίνακες **output** και **userinfo** μηδενίζονται με τη βοήθεια της συνάρτησης **zero**. Το ίδιο ισχύει και για τους πίνακες **ckeck** και **check2**. Η συνάρτηση **floor** στρογγυλοποιεί το πηλίκο (timend / timestep) στο μικρότερο πλησιέστερο ακέραιο. Το πηλίκο αυτό αποτελεί τον αριθμό των επαναλήψεων της προσομοίωσης και στην περίπτωση που είναι δεκαδικός αριθμός θα πρέπει να στρογγυλοποιηθεί, αφού παράλληλα καθορίζει και τον αριθμό των στηλών για τους παραπάνω πίνακες.

Ο πίνακας **baseinfo** αφού αρχικά μηδενιστεί, στη συνέχεια με τη βοήθεια της συνάρτησης **base()** συμπληρώνει την $1^{η}$ και $2^{η}$ στήλη του με τις x και y συντεταγμένες του καθενός από τους 19 σταθμούς βάσης . Πχ τα στοιχεία (3, 1) και (3, 2) του πίνακα baseinfo περιέχουν τις x και y συντεταγμένες του σταθμού βάσης της $3^{ης}$ κυψέλης.

Ο πίνακας **wrapinfo** με τη βοήθεια της συνάρτησης **wrap()** συμπληρώνει και αυτός τις 19 στήλες του με τους αριθμούς των γειτονικών κυψελών που αντιστοιχούν σε καθεμία από τις 19 κυψέλες του συστήματος. Η $1^{η}$ στήλη της κάθε γραμμής περιέχει τον αριθμό της κύριας κυψέλης, οι στήλες 2 έως και 7 περιέχουν τις γειτονικές κυψέλες της $1^{ης}$ ζώνης κινούμενοι αριστερόστροφα και οι στήλες 8 έως και 19 τις γειτονικές κυψέλες της $2^{ης}$ ζώνης κινούμενοι και εδώ αριστερόστροφα. Π.χ. η $2^{η}$ γραμμή του πίνακα wrapinfo που περιέχει τις γειτονικές κυψέλες για τη $2^{η}$ κυψέλη του συστήματος είναι η `[2, 9, 10, 3, 1, 7, 8, 14, 13, 17, 16, 11, 12, 4, 5, 6, 18, 19, 15]`

Στο στάδιο αυτό μηδενίζονται και οι μεταβλητές **timenow**, **callnum**, **blocknum**, **forcenum**, **callnum** και **user.**

Όσον αφορά τη μεταβλητή **user** αυτή παίρνει εκείνη την τιμή από τον πίνακα usernum η οποία αντιστοιχεί στη συγκεκριμένη τιμή του βρόχου επανάληψης. Στο στάδιο της αρχικοποίησης επίσης αρχικοποιούμε και τις συναρτήσεις rand() και randn().

Με τον τρόπο αυτό, οι συναρτήσεις αυτές παράγουν την ίδια ακολουθία τυχαίων αριθμών για όλες τις τιμές του πίνακα usernum, δίνοντας τα ίδια

αποτελέσματα για όλους τους αριθμούς χρηστών για τους οποίους εκτελείται η προσομοίωση.

Η συνάρτηση **rand()** είναι μια γεννήτρια ψευδοτυχαίων αριθμών στο διάστημα [0, 1] οι οποίοι ακολουθούν σταθερή κατανομή. Κατά την αρχικοποίηση της μέσω της εντολής `rand('state', 5)` ορίζουμε κατά πρώτον ποια μέθοδο θα χρησιμοποιήσει και κατά δεύτερον την κατάστασή της. Στο πρόγραμμα μας χρησιμοποιείται η μέθοδος **state**, η οποία βασίζεται σε αλγόριθμο ο οποίος μπορεί να. παράγει 2^{1492} αριθμούς στο διάστημα $[2^{-53}, 1-2^{-53}]$ μέχρι να επαναλάβει τον εαυτό του. Η **κατάσταση** της γεννήτριας παίρνει τιμές από 0 έως $2^{31}-2$ και καθορίζει ποια ακολουθία τυχαίων αριθμών θα παράγει η γεννήτρια. Στο πρόγραμμα μας θέτεται ίση με 5

Ομοίως και η συνάρτηση **randn()** είναι μια γεννήτρια ψευδοτυχαίων αριθμών στο διάστημα [0, 1] οι οποίοι ακολουθούν κανονική κατανομή με μέση τιμή 0 και τυπική απόκλιση 1. Με την εντολή `randn('state', 1)` χρησιμοποιούμε και εδώ τη μέθοδο **state** ενώ θέτουμε τη γεννήτρια στην κατάσταση 1

Όπως θα δούμε και στη συνέχεια η συνάρτηση rand() χρησιμοποιείται από το κύριο πρόγραμμα (dcamain.m) για την δημιουργία των τυχαίων θέσεων του κάθε χρήστη μέσα στην κάθε κυψέλη καθώς επίσης και από τη συνάρτηση **holdtime** για τον υπολογισμό του συνολικού χρόνου διάρκειας των κλήσεων. Από την άλλη η συνάρτηση randn() χρησιμοποιείται από την συνάρτηση shadow() για τον υπολογισμό της εξασθένισης λόγω του φαινομένου της σκέδασης.

Τερματισμός κλήσεων

Κατά το στάδιο αυτό εξετάζουμε για όλους τους χρήστες όλων των κυψελών κατά πρώτον, εάν είναι συνδεδεμένοι και κατά δεύτερον, εάν ο χρόνος τερματισμού της κλήσης τους είναι μικρότερος του τρέχοντα χρόνου προσομοίωσης .

Αυτό γίνεται μέσω της εντολής `if userinfo(numcell, numuser, 4) == 1 & userinfo(numcell, numuser, 5) < timenow`

Όπως παρατηρούμε και από τον πίνακα 3, εάν η τιμή του στοιχείου `userinfo(numcell, numuser, 4)` ισούται με 1 τότε ο χρήστης είναι συνδεδεμένος. Επίσης από τον παραπάνω πίνακα παρατηρούμε ότι η τιμή του στοιχείου `userinfo(numcell, numuser, 5)` ισούται με το χρόνο τερματισμού της κλήσης του χρήστη. Εάν αυτός είναι μικρότερος του τρέχοντα χρόνου προσομοίωσης, τότε

θεωρούμε ότι η κλήση έχει τερματιστεί. Πχ εάν η τρέχουσα τιμή του χρόνου προσομοίωσης είναι *timenow = 200 sec* και ο χρόνος τερματισμού της κλήσης είναι 196 sec τότε θεωρούμε ότι η κλήση έχει τερματιστεί

Για την προσπέλαση των 19 κυψελών του συστήματος χρησιμοποιούμε το βρόχο επανάληψης **for** *(for numcell = 1:19)*. Η μεταβλητή **numcell** περιέχει το νούμερο εκείνης της κυψέλης η οποία αντιστοιχεί στη τρέχουσα τιμή του βρόχου επανάληψης.

Ομοίως για την προσπέλαση όλων των χρηστών που βρίσκονται στη κάθε κυψέλη χρησιμοποιούμε έναν εσωτερικό βρόχο επανάληψης **for** *(for numuser = 1:user).* Η μεταβλητή **numuser** περιέχει το αριθμό του χρήστη που αντιστοιχεί στην τρέχουσα τιμή του βρόχου επανάληψης Τέλος για καθέναν από τους χρήστες που εξυπηρετεί τα παραπάνω κριτήρια αλλάζουμε την κατάσταση του σε μη συνδεδεμένο (*userinfo(numcell, numuser, 4) == 0*), ενώ παράλληλα ελαττώνουμε τον αριθμό των συνδεδεμένων χρηστών κατά 1 (*users = users - 1*)

Έλεγχος για αναδιανομή καναλιών

Και στο στάδιο αυτό εξετάζουμε όλους τους χρήστες όλων των κυψελών χρησιμοποιώντας και εδώ δύο βρόχους επανάληψης.
Αρχικά ελέγχουμε και εδώ ποιοι χρήστες είναι συνδεδεμένοι
userinfo(numcell, numuser, 4) == 1.
Για κάθε έναν από αυτούς θα εξετάσουμε, εάν ο λόγος CNIR που λαμβάνεται από τον αντίστοιχο σταθμό βάσης είναι μικρότερος από την τιμή κατωφλίου και αν ισχύει κάτι τέτοιο θα αναζητήσουμε ένα νέο κανάλι που να εξασφαλίζει την απαιτούμενη τιμή για το λόγο αυτόν.

Για κάθε συνδεδεμένο χρήστη αρχικοποιούμε τη μεταβλητή **reallo** και την θέτουμε ίση με μηδέν. Η μεταβλητή αυτή χρησιμοποιείται για να καθορίσουμε εάν χρειάζεται η αναδιανομή καναλιού (`reallo = 0`) ή όχι (`reallo = 1`). Παράλληλα αρχικοποιούμε και την μεταβλητή **cnirdb** την οποία και αυτή τη θέτουμε ίση με μηδέν. Στην μεταβλητή αυτή αποθηκεύεται η τιμή του λόγου CNIR για τον κάθε χρήστη.

Στη συνέχεια για τον αντίστοιχο χρήστη υπολογίζουμε τη τιμή του λόγου CNR που λαμβάνεται από το σταθμό βάσης του. Ο λόγος αυτός ισούται με την τιμή της μεταβλητής **cnedge** πολλαπλασιαζόμενη με την τιμή της εξασθένισης λόγω του περιβάλλοντος διάδοσης .

Για τον υπολογισμό του στο MATLAB χρησιμοποιούμε την εντολή

```
cn = power(10.0, cnedge / 10.0) * dwave
```

η οποία αντιστοιχεί στην μαθηματική εξίσωση

$$\frac{C}{N} = 10^{\frac{cnedge}{10}} * attenuation$$

Ο λόγος αυτός εκφράζεται σε Watt και αποθηκεύεται στη μεταβλητή **cn.**

Η μεταβλητή **cnedge** όπως είδαμε και στον πίνακα 2 αντιστοιχεί στην τιμή του λόγου CNR στα όρια της κυψέλης**.** Η μεταβλητή **dwave** παίρνει την τιμή της από την αντίστοιχη στήλη του πίνακα userinfo και αντιστοιχεί στην εξασθένιση που υφίσταται το εκπεμπόμενο σήμα εξαιτίας του περιβάλλοντος διάδοσης (path loss & shadowing).

```
dwave = userinfo(numcell, numuser, 3)
```

Επειδή η εξασθένιση εκφράζεται σε watt και η cnedge σε dB, για τον υπολογισμό του CNR θα πρέπει να μετατρέψουμε την cnedge από dB σε watt ($cne\deg e(watt) = 10^{\frac{cnedge(dB)}{10}}$).

Στη συνέχεια υπολογίζουμε την συνολική παρεμβολή που φτάνει στον σταθμό βάσης από όλους τους χρήστες των γειτονικών κυψελών που χρησιμοποιούν το ίδιο κανάλι.

Η τιμή της συνολικής παρεμβολής θα ισούται με:

$$I_{total} = \sum_{j=1}^{N} I_j = \sum_{j=1}^{N} power_j * attenuation_j = power * \sum_{j=1}^{N} attenuation_j = power * attenuation_{total}$$

Στην παραπάνω εξίσωση power είναι η εκπεμπόμενη ισχύς από τον κάθε παρεμβολέα (είναι η ίδια για όλους), N ο συνολικός αριθμός των παρεμβολέων και $attenuation_{total}$ η συνολική εξασθένιση της εκπεμπόμενης ισχύος από όλους τους παρεμβολείς.

Αρχικά αρχικοποιούμε την τιμή της εξασθένισης της συνολικής παρεμβολής θέτοντας την ίση με μηδέν (*uwave = 0.0)* και αποθηκεύουμε την τιμή του καναλιού, που ήδη έχει ανατεθεί στον αντίστοιχο χρήστη κατά το στάδιο έναρξης της κλήσης στην μεταβλητή **ch.**

Η τιμή του καναλιού αυτού βρίσκεται στο στοιχείο `userinfo (numcell, numuser, 6)` του πίνακα `userinfo`.

```
ch = userinfo(numcell, numuser, 6);
```

Έπειτα για όλους τους χρήστες των 6 γειτονικών κυψελών αναζητούμε αυτούς που είναι συνδεδεμένοι και τους έχει ανατεθεί το ίδιο κανάλι. Αυτοί είναι και οι χρήστες που προκαλούν την παρεμβολή.

```
if userinfo(othercell, other, 4) == 1 & userinfo(othercell, other, 6)
== ch
```

Η προσπέλαση των γειτονικών κυψελών και των χρηστών που βρίσκονται σε αυτές γίνεται μέσω των 2 βρόχων επανάληψης.

`for around = 2:7` και `for other = 1:user`

Η μεταβλητή **other** περιέχει τον αριθμό του χρήστη της γειτονικής κυψέλης που αντιστοιχεί στη τρέχουσα τιμή του βρόχου επανάληψης.

Η μεταβλητή **around** χρησιμοποιείται για την προσπέλαση των στηλών του πίνακα wrapinfo. Το νούμερο της καθεμίας από αυτές αποθηκεύεται στην μεταβλητή **othercell .**

```
othercell = wrapinfo(numcell, around)
```

Ο συνολικός αριθμός των γειτονικών κυψελών, όπως είδαμε και πιο πριν είναι 6 αφού λαμβάνουμε υπόψη μόνο τις κυψέλες της $1^{ης}$ ζώνης. Ο βρόχος επανάληψης αρχίζει από το δεύτερο στοιχείο της αντίστοιχης γραμμής του πίνακα wrapinfo, αφού το πρώτο περιέχει την κύρια κυψέλη.

Για κάθε έναν από τους χρήστες, οι οποίοι προκαλούν διακαναλική παρεμβολή υπολογίζουμε την τιμή της εξασθένισης που υφίσταται η παρεμβολή αυτή. Η εξασθένιση αυτή ισούται με την εξασθένιση που υφίσταται το σήμα λόγω της απόστασης πολλαπλασιαζόμενη με την τιμή της εξασθένισης λόγο του φαινομένου της σκέδασης. Μετράται σε watt και αποθηκεύεται στη μεταβλητή **uwave** (ισχύει uwave = $\text{attenuation}_{\text{total}}$)

```
uwave = dist (here, there, alpha) * shadow (sigma).
```

Η τιμή της εξασθένισης λόγω της απόστασης υπολογίζεται μέσω της συνάρτησης **dist()**, η οποία χρησιμοποιεί σαν ορίσματα της τις μεταβλητέ here, there και alpha**.**

Η μεταβλητή **alpha** περιέχει την τιμή του συντελεστή εξασθένισης α.

Η μεταβλητή **here** είναι 1 x 2 πίνακας ο οποίος περιέχει τις x και y συντεταγμένες του σταθμού βάσης που υφίσταται την παρεμβολή. Τις τιμές της αυτές τις λαμβάνει από την αντίστοιχη γραμμή του πίνακα baseinfo.

```
here = baseinfo(numcell, :)
```

Τέλος, η μεταβλητή **there** είναι 1 x 2 πίνακας ο οποίος περιέχει τις x και y συντεταγμένες του παρεμβολέα, οι οποίες εκφράζουν την εικονική του θέση στο κυψελοειδές σύστημα λαμβάνοντας υπόψη τη τεχνική cell – wrapping (σχήμα 4).

```
there = userposi - baseinfo(othercell, :) + baseinfo(around, :) +
baseinfo(numcell, :)
```

Στην παραπάνω εξίσωση οι τρεις τελευταίοι όροι **(baseinfo(othercell, :), baseinfo(around, :)** και **baseinfo(numcell, :))** παίζουν έναν αντισταθμιστικό ρόλο, ώστε να μην έχουμε εσφαλμένα αποτελέσματα κατά τον υπολογισμό της εξασθένισης uwave

Η μεταβλητή **userposi** είναι ένας 1 x 2 πίνακας, ο οποίος περιέχει τις x και y συντεταγμένες του παρεμβολέα εάν θεωρήσουμε ότι αυτός βρίσκεται σε εκείνη την κυψέλη του συστήματος (μία από τις 19), η οποία προκύπτει από την αντίστοιχη στήλη του πίνακα wrapinfo. Τις τιμές του τις λαμβάνει από τις 2 πρώτες στήλες του πίνακα userinfo

```
userposi(1, 1:2) = userinfo(othercell, other, 1:2)
```

Η μεταβλητή **baseinfo(othercell, :)** είναι ένας 1 x 2 πίνακας ο οποίος περιέχει τις x και y συντεταγμένες του σταθμού βάσης της παραπάνω κυψέλης

Η μεταβλητή **baseinfo(around, :)** είναι ένας 1 x 2 πίνακας ο οποίος περιέχει τις x και y συντεταγμένες του σταθμού βάσης εκείνης της κυψέλης της οποίας ο αριθμός ισούται με τη τιμή τη μεταβλητής around

Τέλος η μεταβλητή **baseinfo(numcell, :)** είναι ένας 1 x 2 πίνακας ο οποίος περιέχει τις x και y συντεταγμένες του σταθμού βάσης ο οποίος υφίσταται την παρεμβολή.

.

Έστω ότι θέλουμε να υπολογίσουμε την εξασθένιση της παρεμβολής που δέχεται ο σταθμός βάσης της κυψέλης **19** (πορτοκαλί κυψέλη) από έναν χρήστη στη κυψέλη **15** (κίτρινη κυψέλη).

Σε αυτή την περίπτωση θα ισχύει:

numcell = 19, othercell = 15 και around = 2

Αν για τον υπολογισμό της uwave δεν χρησιμοποιούσαμε τους 3 αντισταθμιστικούς παράγοντες τότε θα είχαμε:

```
here = baseinfo (19, :)
there = userposi = userinfo(othercell, other, 1:2) και
uwave = dist(here, there, alpha) * shadow (sigma);
```

γεγονός που θα μας οδηγούσε σε εσφαλμένο αποτέλεσμα.

Το παραπάνω είναι εύκολο να το διαπιστώσουμε εάν δούμε και το σχήμα 2.4. Παρατηρώντας το βλέπουμε ότι η απόσταση μεταξύ του σταθμού βάσης που δέχεται

την παρεμβολή (μεταβλητή here) και του παρεμβολέα (μεταβλητή there) είναι μεγαλύτερη από την πραγματική.

Αυτό οφείλεται στο γεγονός ότι κατά τον υπολογισμό των συντεταγμένων του παρεμβολέα θεωρήσαμε ότι η κυψέλη στην οποία αυτός ανήκει, είναι η μία από τις 19 κυψέλες του συστήματος της οποίας ο αριθμός της ισούται με την αντίστοιχη στήλη του πίνακα **wrapinfo** (κυψέλη με το πράσινο χρώμα) και όχι η εικονική κυψέλη που προκύπτει από την τεχνική cell – wrapping (κυψέλη με το κίτρινο χρώμα).

Αντίθετα εάν λάβουμε υπόψη τους 3 αντισταθμιστικούς παράγοντες έχουμε

```
here = baseinfo(19, :)
there = userposi - baseinfo(15, :) + baseinfo(2, :) + baseinfo(19, :)
uwave = dist(here, there, alpha) * shadow (sigma);
```

οι οποίοι μας οδηγούν στο σωστό αποτέλεσμα.

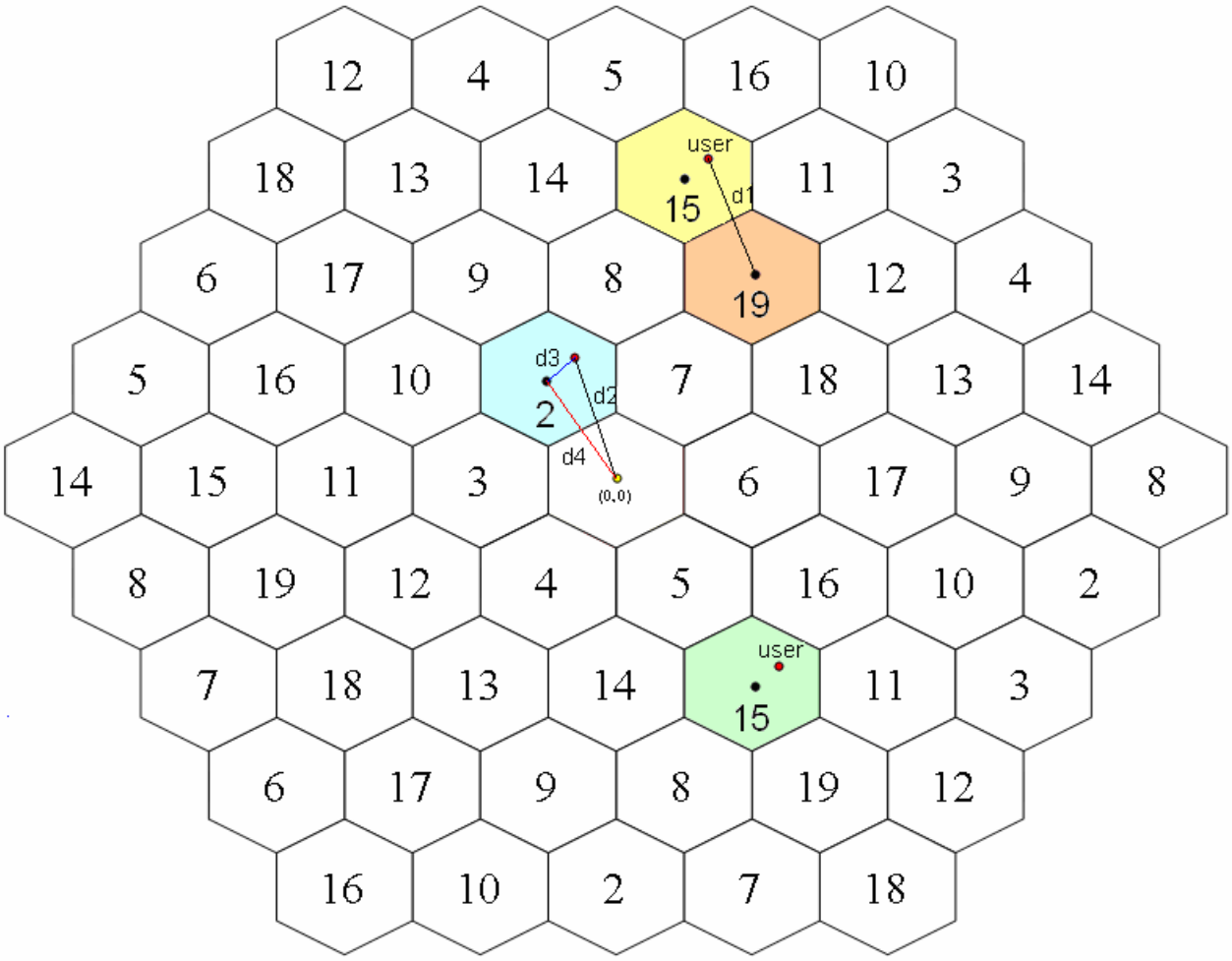
Από σχήμα 2.4 μπορούμε επίσης να παρατηρήσουμε ότι η απόσταση μεταξύ του παρεμβολέα και του σταθμού βάσης (απόσταση d1) είναι η ίδια με την απόσταση μεταξύ της αντίστοιχης θέσης του παρεμβολέα στη κυψέλη around (κυψέλη 2) και της αρχής των αξόνων (απόσταση d2). Αυτό αποδεικνύεται ως εξής:

Όπως βλέπουμε από το σχήμα 2.4 το μέτρο της διανυσματικής διαφοράς userposi - baseinfo(15, :) ισούται με την απόσταση μεταξύ του παρεμβολέα και του σταθμού βάσης του. Η απόσταση αυτή είναι η ίδια με αυτήν μεταξύ της αντίστοιχης θέσης του παρεμβολέα στην κυψέλη 2 και του σταθμού βάσης αυτής (απόσταση d3).

Επομένως η διαφορά μεταξύ των διανυσμάτων here και there είναι:

```
there – here = userposi - baseinfo(15, :) + baseinfo(2, :) +
baseinfo(19, :) - baseinfo(19, :) = userposi - baseinfo(15, :) +
baseinfo(2, :) = d3 + d4 = d2
```

Δηλαδή το μέτρο της διανυσματικής διαφοράς there – here ισούται με την απόσταση μεταξύ της θέσης του παρεμβολέα στο αντίστοιχο σημείο εκείνης της κυψέλης η οποία καθορίζεται από την μεταβλητή around και της αρχής των αξόνων.



ΣΧΗΜΑ 2.4
Υπολογισμός απόστασης με τη τεχνική cell - wrapping

Η εξασθένιση που οφείλεται στο φαινόμενο της σκέδασης υπολογίζεται μέσω της συνάρτησης **shadow(),** η οποία δέχεται σαν όρισμα της τη τυπική απόκλιση του συντελεστή σκέδασης σ.

Μετά τον υπολογισμό της συνολικής εξασθένισης της παρεμβολής υπολογίζουμε τη τιμή του λόγου CNIR στο δέκτη του σταθμού βάσης:

$$CNIR(dB) = \frac{C}{N+I} = \frac{C/C}{N/C + I/C} = \frac{1}{(C/N)^{-1} + (C/I)^{-1}} = \frac{1}{(cnedge * dwave)^{-1} + \left(\frac{cnedge * N * dwave}{cnedge * N * uwave}\right)^{-1}}$$

$$\Rightarrow CNIR(dB) = \frac{1}{\frac{1}{cn} + \frac{uwave}{dwave}}$$

Η εξίσωση αυτή στο MATLAB αντιστοιχεί στην παρακάτω γραμμή κώδικα:

```
cnirdb = 10.0 * log10 (1 / ((uwave / dwave) + (1 / cn)))
```

Επειδή οι τιμές των μεταβλητών dwave, uwave και cn, όπως έχουμε δει και πιο πάνω έχουν υπολογιστεί σε watt ο λόγος cnirdb εκφράζεται και αυτός σε watt. Για να τον μετατρέψουμε σε dB χρησιμοποιούμε την παρακάτω σχέση.

$$CNIR(dB) = 10 * \log_{10}(CNIR(watt))$$

Στη συνέχεια συγκρίνουμε τη τιμή του λόγου αυτού με την τιμή του κατωφλίου (μεταβλητή **cnirth**).

Εάν η τιμή του είναι μικρότερη από την τιμή του κατωφλίου τότε χρειάζεται η αναδιανομή καναλιού. Στην περίπτωση αυτή η μεταβλητή reallo παίρνει την τιμή 1.

```
if cnirdb < cnirth
reallo = 1
```

Το επόμενο στάδιο που ακολουθεί, αφορά την περίπτωση που απαιτείται η αναδιανομή καναλιού για το συγκεκριμένο χρήστη.

Αρχικά η κατάσταση του χρήστη αυτού αλλάζει σε μη συνδεδεμένος μέσω του πίνακα userinfo

```
userinfo (numcell, numuser, 4) = 0
```

και παράλληλα ο αριθμός των συνδεδεμένων χρηστών μειώνεται κατά ένας

```
users = users - 1
```

Η μεταβλητή **succeed** χρησιμοποιείται για να καθορίσουμε, εάν η αναδιανομή καναλιού ήταν επιτυχής (succeed = 1) ή όχι (succeed = 0).

Αρχικά παίρνει την τιμή 0.

Εκτός από αυτήν αρχικοποιούμε και τη μεταβλητή cnirdb, η οποία αποθηκεύει την νέα τιμή του λόγου CNIR για το νέο κανάλι.

Στη συνέχεια αναζητούμε ένα από τα κανάλια του σταθμού βάσης το οποίο να μη χρησιμοποιείται από κάποιον άλλο χρήστη στη κυψέλη

Η προσπέλαση των καναλιών αυτών γίνεται μέσω του βρόχου επανάληψης **for**

```
for ch = 1:chnum
```

Η μεταβλητή **ch** περιέχει τον αριθμό του καναλιού που αντιστοιχεί στη τρέχουσα τιμή του βρόχου επανάληψης. Μέσω της μεταβλητής **available** καθορίζουμε εάν το παραπάνω κανάλι είναι ελεύθερο (available = 1) ή όχι (available = 0). Πριν από τον έλεγχο αυτόν την αρχικοποιούμε θέτοντας την ίση με 1 (`available = 1`). Στη συνέχεια ελέγχουμε εάν το συγκεκριμένο κανάλι έχει ήδη ανατεθεί σε κάποιον άλλο συνδεδεμένο χρήστη της κυψέλης.

```
if userinfo(numcell, other, 4) == 1 & userinfo(numcell, other, 6) == ch
```

Στην περίπτωση που ισχύει κάτι τέτοιο η μεταβλητή available παίρνει την τιμή 0 και επαναλαμβάνουμε τον έλεγχο για το επόμενο κανάλι. Για τη προσπέλαση των άλλων χρηστών της κυψέλης χρησιμοποιούμε και εδώ τον βρόχο επανάληψης **for**

```
for other = 1:user
```

Η μεταβλητή **other** περιέχει τον αριθμό του χρήστη που αντιστοιχεί στη τρέχουσα τιμή του βρόχου επανάληψης. Στην περίπτωση που δεν βρεθεί κανένα ελεύθερο κανάλι η τιμή της μεταβλητής cnirdb ισούται με μηδέν, η οποία όπως είναι φυσικό είναι μικρότερη της τιμής κατωφλίου.

```
cnirdb = 0.0
```

Μόλις βρεθεί ένα διαθέσιμο κανάλι, τότε για αυτό υπολογίζουμε εκ νέου την τιμή του λόγου CNIR και τη συγκρίνουμε με την τιμή του κατωφλίου, όπως ακριβώς κάναμε και πιο πριν. Αυτό το κανάλι, το οποίο εξασφαλίζει τιμή του λόγου CNIR μεγαλύτερη από την τιμή κατωφλίου είναι αυτό που διατίθεται στον συγκεκριμένο χρήστη.

```
userinfo(numuser, numuser, 6) = ch
```

Στην περίπτωση αυτή η κατάσταση του χρήστη αλλάζει σε συνδεδεμένος

```
userinfo(numcell, numuser, 4) = 1
```

και παράλληλα ο αριθμός των συνδεμένων χρηστών αυξάνεται κατά ένας

```
users = users + 1
```

Τέλος η μεταβλητή succeed μετατρέπεται από 0 σε 1 (`suceed = 1`).

Το τελευταίο μέρος του σταδίου αναδιανομής περιλαμβάνει την περίπτωση που δεν βρεθεί κανένα κανάλι το οποίο να εξασφαλίζει μία επαρκή τιμή για το λόγο CNIR (`if suceed == 0`)

Στην περίπτωση αυτή έχουμε διακοπή της υπάρχουσας σύνδεσης του χρήστη και ο αριθμός των βίαια τερματισμένων κλήσεων αυξάνει κατά ένα

```
forcenum = forcenum + 1
```

Άφιξη νέων κλήσεων

Και στο στάδιο αυτό όπως και στα προηγούμενα εξετάζουμε όλους τους χρήστες όλων των κυψελών του συστήματος.

Ο ρυθμός άφιξης κλήσεων για τον κάθε χρήστη ακολουθεί κατανομή Poison με μέση τιμή λ (κλήσεις/ώρα). Για να εξετάσουμε εάν ο κάθε χρήστης του συστήματος μπορεί να ξεκινήσει μια νέα κλήση μέσα στο διάστημα που διαρκεί η κάθε επανάληψη της προσομοίωσης (timestep), παράγουμε μια τυχαία τιμή στο διάστημα [0, 1] με τη βοήθεια της συνάρτησης **rand().** Η τιμή αυτή αντιστοιχεί στο ρυθμό άφιξης κλήσεων εκείνο το χρονικό διάστημα.
Στη συνέχεια συγκρίνουμε τον τυχαίο αυτόν αριθμό με την τιμή του ρυθμού άφιξης κλήσεων για το ίδιο χρονικό διάστημα όπως καθορίζεται από τη μεταβλητή λ:
new call arrival rate = λ * t = λ * (simulation step / 3600).
Εάν ισχύει rand() < new call arrival rate, τότε θεωρούμε ότι ο χρήστης μπορεί να ξεκινήσει μία νέα κλήση.

Στο MATLAB ο έλεγχος αυτός αντιστοιχεί στην παρακάτω εντολή:

```
if userinfo(numcell, numuser, 4) == 0 & rand <= lambda * (timestep / 3600)
```

Αρχικά ελέγχουμε ποιοι χρήστες δεν είναι συνδεδεμένοι (`userinfo(numcell, numuser, 4) == 0`) και στη συνέχεια συγκρίνουμε τον τυχαίο αριθμό με την αντίστοιχη τιμή του ρυθμού άφιξης κλήσεων.
Επειδή η μέση τιμή του ρυθμού άφιξης κλήσεων λ εκφράζεται σε κλήσεις / ώρα και η συνολική διάρκεια του κάθε βήματος της προσομοίωσης μετράται σε sec, για το λόγο αυτό μετατρέπουμε τη δεύτερη από sec σε ώρες

Για κάθε έναν από τους χρήστες που ξεκινά μια νέα κλήση αυξάνουμε τον αριθμό των παραγόμενων κλήσεων κατά 1 (`callnum = callnum + 1`).
Επίσης και πάλι με τη βοήθεια της συνάρτησης **rand()** παράγουμε έναν τυχαίο αριθμό στο διάστημα [0, 1]. Αυτόν τον πολλαπλασιάζουμε με την τιμή της μεταβλητής **meshnum,** η οποία περιέχει τον αριθμό των πιθανών θέσεων (meshes) του κάθε χρήστη μέσα στις κυψέλες και στη συνέχεια με τη βοήθεια της συνάρτησης **floor** στρογγυλοποιούμε το αποτέλεσμα αυτό στο μικρότερο πλησιέστερο ακέραιο .

Με τον τρόπο αυτό δημιουργούμε μια τυχαία θέση για το κάθε χρήστη μέσα στην κυψέλη, η οποία καθορίζεται από το αποτέλεσμα του παραπάνω πολλαπλασιασμού και αποθηκεύεται στη μεταβλητή **mesh.**

```
mesh = floor(meshnum .* rand) + 1
```

Στην παραπάνω σχέση βλέπουμε ότι κατά τον υπολογισμό της μεταβλητής mesh προσθέτουμε και την μονάδα. Με αυτόν τον τρόπο πετυχαίνουμε η μικρότερη τιμή της να ισούται με 1, έτσι ώστε να είμαστε σε θέση να προσπελάσουμε όλες τις γραμμές του πίνακα meshposition.

Αυτό βέβαια από την άλλη έχει σαν αποτέλεσμα η μέγιστη τιμή της να υπερβαίνει τον αριθμό meshnum κατά 1 ($mesh_{max}$ = meshnum + 1).

Για να αποφύγουμε αυτήν την περίπτωση χρησιμοποιούμε τον βρόχο while, ο οποίος επαναϋπολογίζει τη μεταβλητή mesh στην περίπτωση που αυτή είναι μεγαλύτερη από την μεταβλητή meshnum.

Για το κάθε χρήστη που μόλις ξεκίνησε μια νέα κλήση, καθορίζουμε τις x και y συντεταγμένες του, τις οποίες στη συνέχεια τοποθετούμε στις 2 πρώτες στήλες του πίνακα userinfo

```
userinfo(numcell, numuser, 1:2) = baseinfo(numcell, :) + meshposition(mesh, :)
```

Ο πίνακας baseinfo είναι 1 x 2 πίνακας, ο οποίος περιέχει τις συντεταγμένες του σταθμού βάσης της κυψέλης στην οποία βρίσκεται ο χρήστης.

Ομοίως ο πίνακας meshposition είναι 1 x 2 πίνακας, ο οποίος περιέχει τις συντεταγμένες του χρήστη σε σχέση με τον παραπάνω σταθμό βάσης.

Επομένως οι x και y συντεταγμένες του χρήστη σε σχέση με την αρχή των αξόνων προέρχονται από το άθροισμα των αντίστοιχων στηλών των δύο πινάκων baseinfo και meshposition

Στο επόμενο βήμα αναζητούμε ένα ελεύθερο κανάλι για να το διαθέσουμε στον παραπάνω χρήστη και στη συνέχεια υπολογίζουμε για αυτό την τιμή του λόγου CNIR που λαμβάνεται από το σταθμό βάσης.

Αρχικά αρχικοποιούμε τη μεταβλητή **succeed** (succeed = 0)**,** η οποία καθορίζει εάν βρέθηκε ένα τέτοιο κανάλι το οποίο να εξασφαλίζει τιμή του λόγου CNIR τουλάχιστον ίση με την τιμή κατωφλίου.

Εκτός από αυτήν αρχικοποιούμε και την μεταβλητή **cnirdb**, την οποία και αυτή τη θέτουμε ίση με μηδέν.

Όπως και στο στάδιο αναδιανομής καναλιού έτσι και εδώ, πρώτα υπολογίζουμε τη τιμή του λόγου CNR που λαμβάνεται από τον σταθμό βάσης.(μεταβλητή **cn**).

Και εδώ η τιμή του ισούται με την τιμή του λόγου CNR στα όρια της κυψέλης πολλαπλασιαζόμενη με την τιμή της εξασθένιση λόγο του περιβάλλοντος διάδοσης.

```
cn = power(10, cnedge / 10.0) * dwave
```

Ο υπολογισμός της εξασθένισης dwave είναι παρόμοιος με αυτόν της uwave που είδαμε πιο πάνω. Η μόνη διαφορά είναι ότι κατά τον υπολογισμό της απόστασης μεταξύ του χρήστη και του σταθμού βάσης μέσω της συνάρτησης **dist(),** η μεταβλητή there που περιέχει τις συντεταγμένες του χρήστη και χρησιμοποιείται σαν όρισμα της παραπάνω συνάρτησης δεν περιλαμβάνει τους 3 αντισταθμιστικούς όρους (baseinfo(othercell, :), baseinfo(around, :) και baseinfo(numcell, :)).

Αυτό είναι φυσικό, εάν σκεφτούμε ότι ο χρήστης βρίσκεται στην ίδια κυψέλη με το σταθμό βάσης και όχι σε κάποια γειτονική, οπότε και η τεχνική cell –wrapping δεν έχει καμία επιρροή.

```
here = baseinfo (numcell, :) και there = userposi
```

Στη συνέχεια ακολουθεί η αναζήτηση ενός διαθέσιμου καναλιού, ο υπολογισμός της τιμής του λόγου CNIR για αυτό το κανάλι και τέλος η σύγκριση του με την τιμή του κατωφλίου cnirth.

Ο κώδικας που αντιστοιχεί στην παραπάνω περίπτωση είναι ακριβώς ο ίδιος με αυτόν που χρησιμοποιήσαμε στο στάδιο έλεγχου αναδιανομής καναλιών

Στην περίπτωση που βρεθεί ένα ελεύθερο κανάλι το οποίο να εξασφαλίζει την απαιτούμενη τιμή του λόγου CNIR, αυτό διατίθεται στον χρήστη και αποθηκεύεται στο αντίστοιχο στοιχείο του πίνακα userinfo .

```
userinfo(numcell, numuser, 6) = ch
```

Επίσης η κατάσταση του χρήστη αλλάζει σε συνδεδεμένος

```
userinfo(numcell, numuser, 4) = 1
```

και ο αριθμός των συνδεδεμένων χρηστών αυξάνει κατά ένας.

Παράλληλα στον πίνακα userinfo αποθηκεύεται ο χρόνος τερματισμού της κλήσης του χρήστη. Αυτός ισούται με τη τρέχουσα τιμή του χρόνου προσομοίωσης συν τη συνολική διάρκεια της κλήσης.

Η συνολική διάρκεια της κλήσης του κάθε χρήστη είναι μια τυχαία τιμή η οποία ακολουθεί εκθετική κατανομή με μέση τιμή h και λαμβάνεται από τη συνάρτηση

holdtime(). Η συνάρτηση αυτή δέχεται σαν όρισμα της τη μέση τιμή της διάρκειας της κλήση **ht.**

```
userinfo (numcell, numuser, 5) = timenow + holdtime (ht)
```

Τέλος στον πίνακα userinfo αποθηκεύεται και η τιμή της εξασθένισης λόγο του περιβάλλοντος διάδοσης (μεταβλητή dwave)

```
userinfo(numcell, numuser, 3) = dwave
```

Στην περίπτωση που δεν βρεθεί ένα διαθέσιμο κανάλι το οποίο να εξασφαλίζει μία επιθυμητή τιμή για το λόγο CNIR, τότε η κλήση απορρίπτεται και ο αριθμός των μπλοκαρισμένων κλήσεων αυξάνεται κατά ένας.

```
blocknum = blocknum + 1
```

Εμφάνιση των αποτελεσμάτων

Αρχικά μέσω της εντολής **frintf** κατά την εκτέλεση του προγράμματος εμφανίζονται στην οθόνη για το κάθε βήμα της προσομοίωσης οι τιμές των μεταβλητών **parameter**, **timenow** (τρέχουσα τιμή του χρόνου προσομοίωσης), **callnum – callnumold** (αριθμός των παραγόμενων κλήσεων για το κάθε βήμα της προσομοίωσης), **blocknum – blocknumold** (αριθμός των απορριφθέντων κλήσεων για το κάθε βήμα της προσομοίωσης) και **blocknum / callnum** (τιμή της πιθανότητας απόρριψης κλήσεων για τη τρέχουσα τιμή του χρόνου προσομοίωσης).

```
fprintf('%d\t%d\t%d\t%d\t%e\n', parameter, timenow, callnum -
callnumold, blocknum - blocknumold, blocknum / callnum)
```

Οι μεταβλητές **callnumold** και **blocknumold** περιέχουν τις τιμές των μεταβλητών **callnum** και **blocknum** στην αρχή του κάθε βήματος της προσομοίωσης, προτού υπολογιστούν οι νέες τιμές τους στο τέλος του αντίστοιχου βήματος. Με τον τρόπο αυτό η διαφορά **callnum – callnumold** και **blocknum – blocknumold** μας δίνει τις τρέχουσες τιμές των παραπάνω μεταβλητών για το αντίστοιχο βήμα της προσομοίωσης.

Οι τέσσερις πρώτες μεταβλητές (**parameter, timenow, callnum – callnumold, blocknum – blocknumold**) εκφράζονται σε δεκαδική μορφή (%d) και η πέμπτη (**blocknum / callnum**) σε εκθετική (%e). Οι παραπάνω μεταβλητές απέχουν μεταξύ τους απόσταση ενός στηλοθέτη (\t). Μετά την εμφάνιση των μεταβλητών στο κάθε βήμα της προσομοίωσης ακολουθεί αλλαγή γραμμής (\n).

Στο τέλος του κάθε βήματος της προσομοίωσης συμπληρώνονται οι αντίστοιχες στήλες των πινάκων **check** και **check2** με τις τρέχουσες τιμές της πιθανότητας απόρριψης κλήσεων (blocking probability) και της πιθανότητας εξαναγκασμένου τερματισμού των εν εξελίξει κλήσεων (forced terminated probability) αντίστοιχα. Οι παραπάνω στήλες ανήκουν σε εκείνη τη γραμμή, η οποία αντιστοιχεί σε εκείνον τον αριθμό χρηστών ο οποίος καθορίζεται από την τρέχουσα τιμή της μεταβλητής **parameter.**

Επίσης, μετά το τέλος του συνολικού χρόνου της προσομοίωσης συμπληρώνεται η $1^{η}$, $2^{η}$, $3^{η}$, $4^{η}$ και $5^{η}$ γραμμή του πίνακα output με τις τελικές τιμές των μεταβλητών **callnum, blocknum, forcenum, blocknum / callnum** και **forcenum / (callnum – blocknum)** αντίστοιχα. Η κάθε στήλη του πίνακα αυτού αντιστοιχεί σε αριθμό χρηστών 5, 10, 15 , 20 και 25 αντίστοιχα

Τέλος, τα παραπάνω δεδομένα γράφονται και σε ένα αρχείο txt το το οποίο έχει την ονομασία **data.txt** Για να ανοίξουμε το παραπάνω αρχείο για εγγραφή χρησιμοποιούμε την εντολή **fopen**

```
fid = fopen('data.txt', 'w')
```

Το όρισμα ‘**w**’ καθορίζει ότι το αρχείο είναι μόνο για εγγραφή.

Η μεταβλητή **fid** είναι ένας ακέραιος, ο οποίος χρησιμοποιείται σαν δείκτης για τον καθορισμό του παραπάνω αρχείου. Εάν το αρχείο δεν μπορεί να ανοιχτεί τότε επιστρέφεται η τιμή -1.

Για να γράψουμε μέσα στο αρχείο χρησιμοποιούμε τη μεταβλητή **fprintf.**

Μέσω της εντολής

```
fprintf(fid, 'UserNumber\t')
```

μπορούμε να γράψουμε μέσα στο αρχείο το οποίο σχετίζεται με το δείκτη fid (δηλαδή το data.txt) τη λέξη UserName αφήνοντας διάστημα ενός στηλοθέτη (\t).

Η εντολή

```
fprintf(fid, '%g\t%g\t%g\t%g\t%g\n', usernum(1,:))
```

γράφει τα δεδομένα της κάθε στήλης του πίνακα **usernum** (δηλαδή τον αριθμό των χρηστών) σε εκθετική μορφή (%g) αφήνοντας μεταξύ τους απόσταση ενός στηλοθέτη.

Ομοίως η εντολή

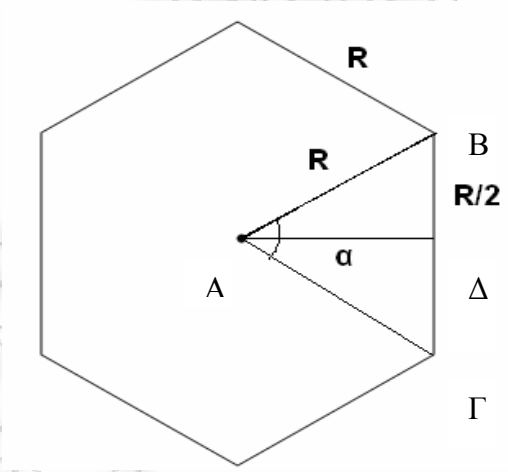
```
fprintf(fid, '%g\t%g\t%g\t%g\t%g\n', output(1,:));
```

γράφει τα δεδομένα της κάθε στήλης της $1^{ης}$ γραμμή του πίνακα **output** σε εκθετική μορφή (%g) αφήνοντας μεταξύ τους απόσταση ενός στηλοθέτη. Τα δεδομένα αυτά αποτελούν την τελική τιμή των παραγόμενων κλήσεων για αριθμό χρηστών 5, 10, 15, 20 και 25 αντίστοιχα.

Με τον ίδιο ακριβώς τρόπο γράφουμε στο παραπάνω αρχείο τον αριθμό των απορριφθέντων κλήσεων, τον αριθμό των βίαια τερματισμένων κλήσεων, την τιμή της blocking probability και forced terminated probability ($2^{η}$, $3^{η}$, $4^{η}$ και $5^{η}$ γραμμή του πίνακα output αντίστοιχα) για αριθμό χρηστών 5, 10, 15, 20 και 25.

2.3 basest.m

Η συνάρτηση αυτή υπολογίζει τις συντεταγμένες των 19 σταθμών βάσης του συστήματος οι οποίοι βρίσκονται στο κέντρο των αντίστοιχων κυψελών. Για το καθορισμό των x και y συντεταγμένων θα χρησιμοποιήσουμε 2 μεταβλητές. Την ακτίνα R της κυψέλης και την απόσταση α όπως φαίνεται και στο παρακάτω σχήμα.



ΣΧΗΜΑ 2.5
Υπολογισμός απόστασης α

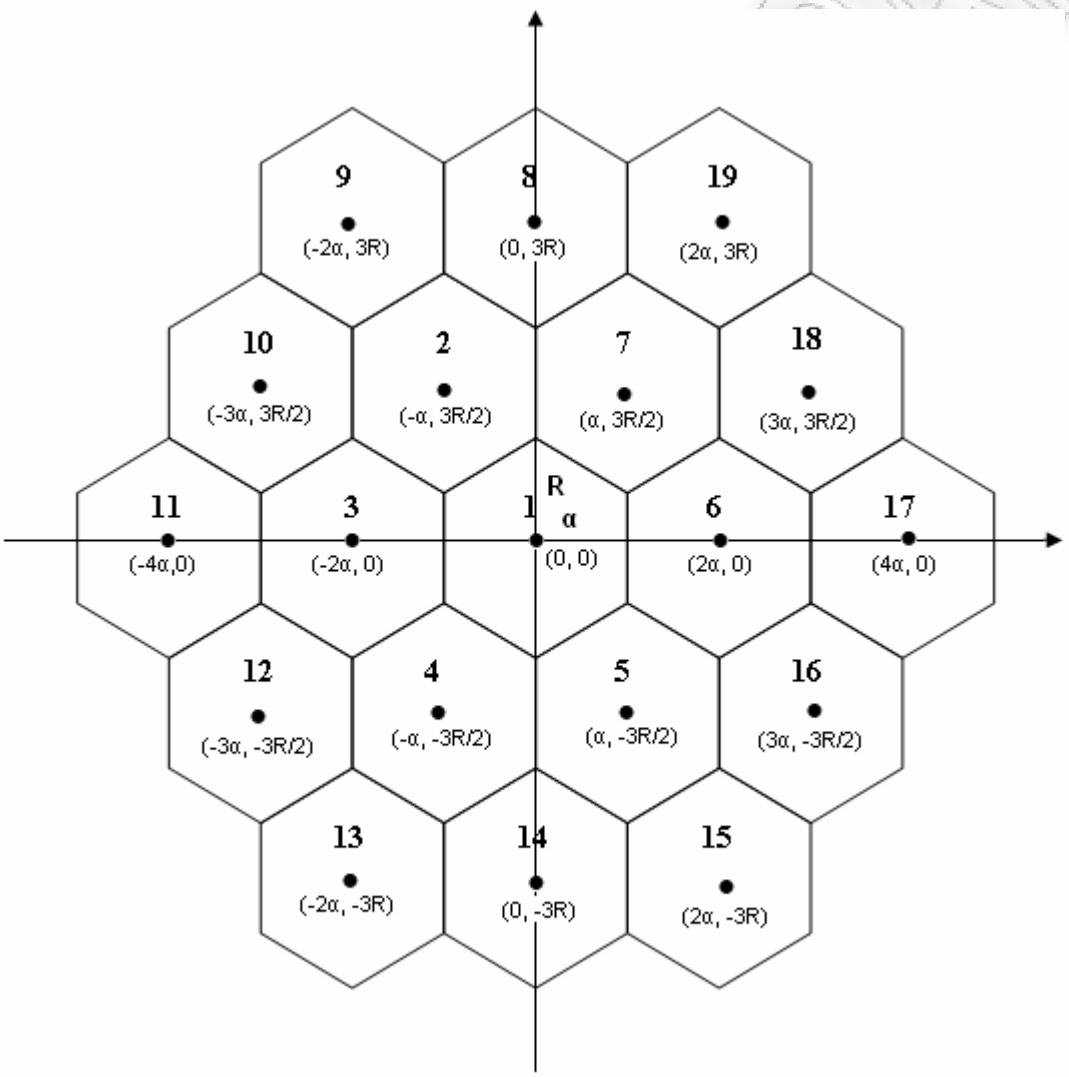
Η κάθε κυψέλη όπως ξέρουμε, αναπαριστάται με την μορφή ενός κανονικού εξαγώνου. Από τη γεωμετρία γνωρίζουμε ότι η πλευρά του κανονικού εξαγώνου ισούται με την ακτίνα του R.

Για τον υπολογισμό της απόσταση α χρησιμοποιούμε το πυθαγόρειο θεώρημα.

$$a = (\text{AΔ}) = \sqrt{(\text{AB})^2 - (\text{BΔ})^2} = \sqrt{R^2 - \left(R/2\right)^2} = \frac{R}{2}\sqrt{3}$$

Στο παραπάνω σχήμα (σχήμα 2.5) επειδή το τρίγωνο ΑΒΓ είναι ισόπλευρο (όλες οι πλευρές του είναι ίσες με R) η κάθετος ΑΔ είναι και διάμεσος του, οπότε το τμήμα ΒΔ ισούται με R/2.

Με τη βοήθεια του σχεδιαγράμματος του κυψελοειδές συστήματος (σχήμα 2.1) και λαμβάνοντας υπόψη τις παραπάνω μεταβλητές είναι πλέον εύκολο να υπολογίσουμε τις συντεταγμένες όλων των σταθμών βάσης όπως φαίνεται και σχήμα 2.6. Ως αρχή των αξόνων θεωρούμε το κέντρο της κυψέλης 1.



ΣΧΗΜΑ 2.6
Θέσεις των σταθμών βάσης

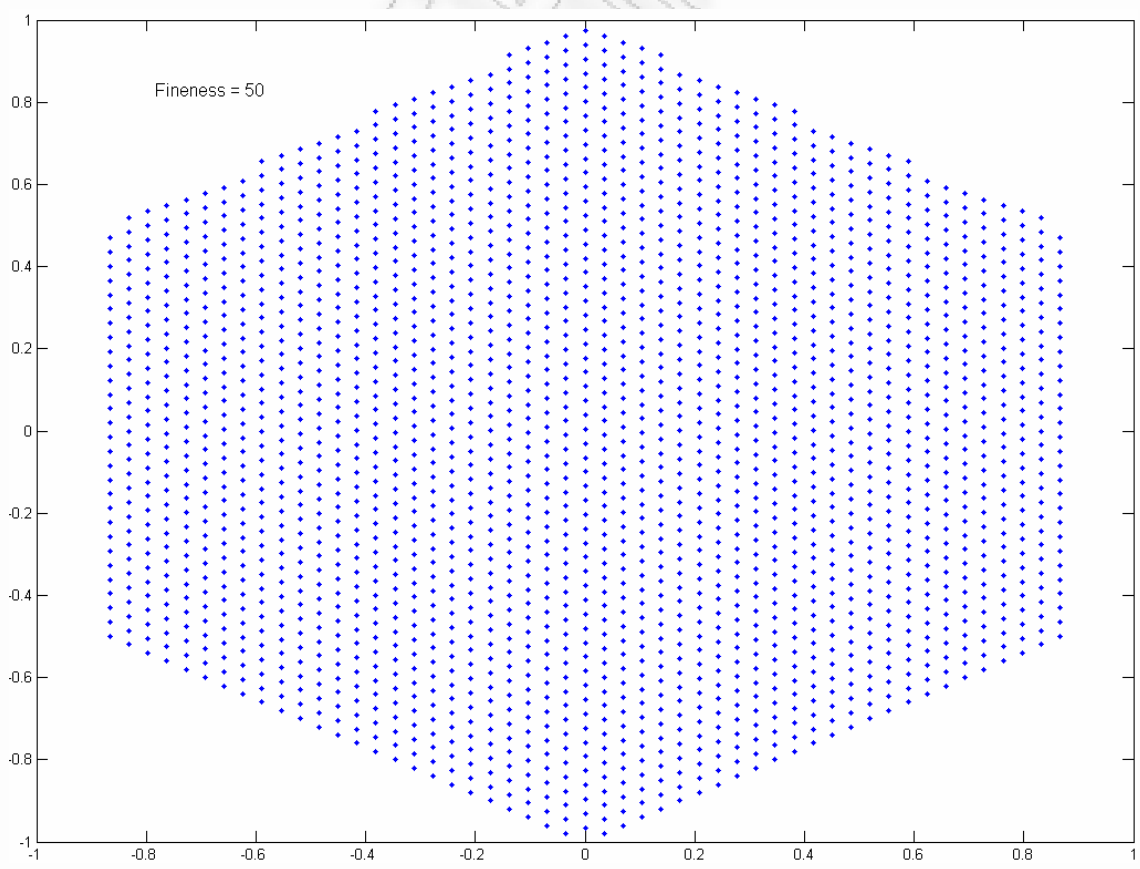
Οι συντεταγμένες των 19 σταθμών βάσης αποθηκεύονται στον πίνακα **baseinfo,** ο οποίος είναι ένας 19 x 2 πίνακας. Ο αριθμός της κάθε γραμμής του αντιπροσωπεύει τον αντίστοιχο σταθμό βάσης, ενώ η $1^{η}$ και $2^{η}$ στήλη του περιέχουν τις x και y συνταγμένες του αντίστοιχα.

Έτσι πχ τα στοιχεία baseinfo(8,1) = 0 και baseinfo(8,2) = 3R περιέχουν τι x και y συντεταγμένες του σταθμού βάσης της κυψέλης 8.

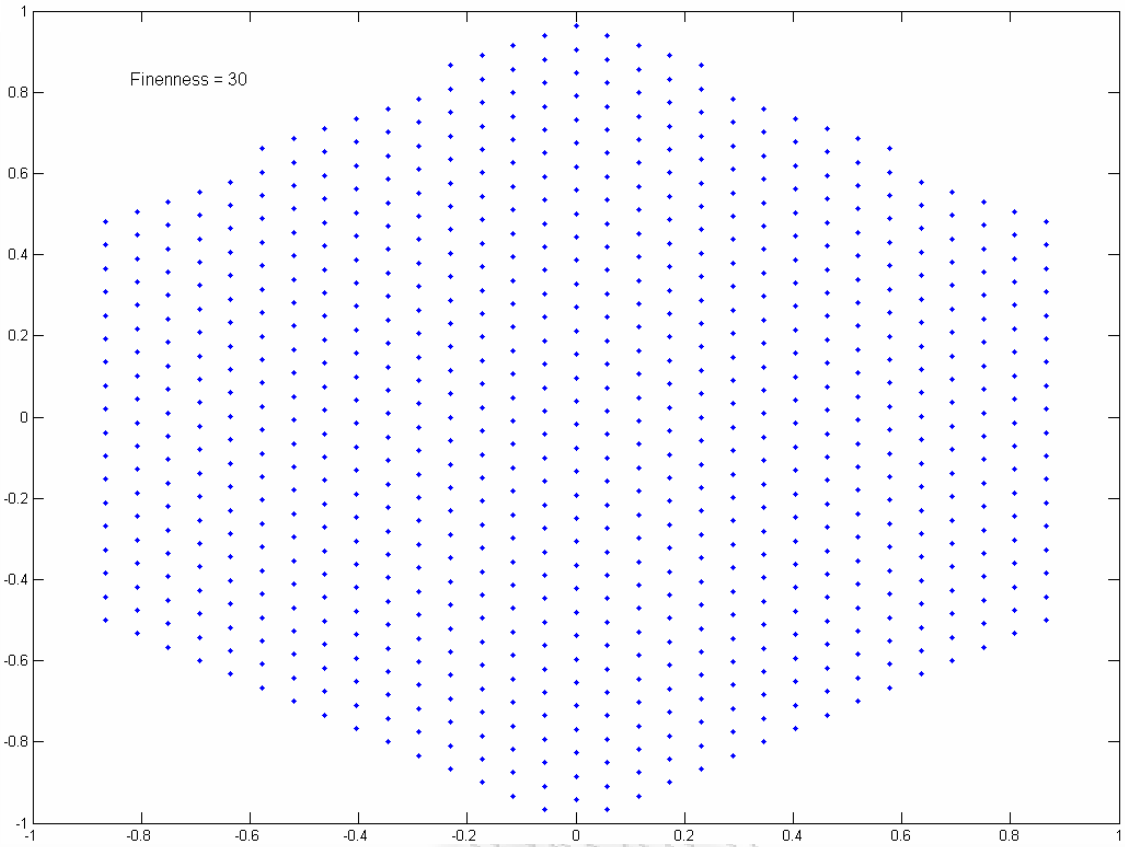
2.4 cellmesh.m

Η συνάρτηση cellmesh δημιουργεί τις πιθανές θέσεις ενός χρήστη μέσα στην κυψέλη (meshes) υπολογίζοντας πόσες είναι αυτές και τι συντεταγμένες έχουν. Η παραπάνω συνάρτηση επιστρέφει 2 μεταβλητές. Την **meshnum,** η οποία είναι ένας ακέραιος που αντιστοιχεί στον αριθμό των meshes και την **cellmseh,** η οποία είναι ένας πίνακας 2 στηλών και τόσων γραμμών όσος και ο αριθμός των meshes (meshnum x 2). Η κάθε γραμμή του πίνακα αντιστοιχεί στον αριθμό ενός από τα meshes και η κάθε στήλη του περιέχει τις x και y συντεταγμένες του αντίστοιχου mesh.

Η μεταβλητή **Fineness** είναι πολύ σημαντική, γιατί η τιμή της καθορίζει την πυκνότητα των meshes. Όσο μεγαλύτερη είναι αυτή, τόσο μεγαλύτερος είναι και ο αριθμός των meshes. Στα σχήματα 2.7 & 2.8 φαίνεται εμφανώς η διαφορά μεταξύ της πυκνότητας των meshes για τιμή της μεταβλητής Fineness = 50 και Fineness = 30



ΣΧΗΜΑ 2.7
Κατανομή των meshes για Fineneness = 30



ΣΧΗΜΑ 2.8
Κατανομή των meshes για Fineneness = 50

Οι μεταβλητές **j** και **κ** είναι δύο μετρητές για τον αριθμό των meshes και τον αριθμό των μετατοπίσεων κατά τον άξονα x αντίστοιχα. Οι αρχικές τιμές τους ισούνται με 0 και 1.

Η απόσταση μεταξύ των meshes υπολογίζεται μέσω της "διαμέτρου" της κυψέλης, εάν λάβουμε υπόψη ότι η μεταξύ τους απόσταση επί τη πυκνότητα Fineness ισούται με το συνολικό μήκος της διαμέτρου :

$$d = 2a / Fineness = \frac{2 \cdot R \cdot \sqrt{3}/2}{Fineness} = \frac{R\sqrt{3}}{Fineness}$$

ή `ds = sqrt(3.0) / Fineness`

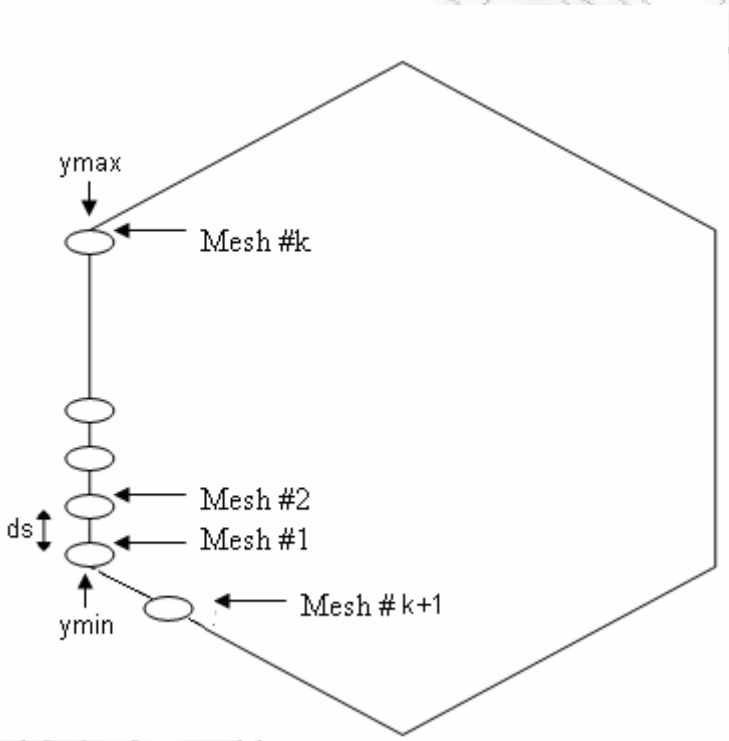
.

Η απόσταση αυτή καταχωρείται στη μεταβλητή **ds.**

Η θέση του 1ου mesh (Mesh #1) φαίνεται στο σχήμα 2.9 και αποτελεί την αρχή έναρξης για τον υπολογισμό των θέσεων και των επόμενων meshes.

Όπως μπορούμε να δούμε και από το ίδιο σχήμα, ο καθορισμός των θέσεων τους γίνεται κινούμενοι κατά μήκος του άξονα y από κάτω προς τα πάνω μεταξύ των σημείων y_{min} και y_{max} και κατά μήκος του άξονα x από αριστερά προς τα δεξιά εντός της "διαμέτρου" 2α.

Πιο συγκεκριμένα για κάθε τιμή της μεταβλητής xmesh, η οποία αντιστοιχεί στην τρέχουσα τιμή του βρόχου επανάληψης while υπολογίζουμε όλες τις τιμές της μεταβλητής ymesh κατά μήκος του άξονα y, οι οποίες βρίσκονται εντός του διαστήματος y_{min} και y_{max}.



ΣΧΗΜΑ 2.9
Προσδιορισμός των θέσεων των meshes

Η κίνηση κατά μήκος του άξονα x γίνεται μέσω του βρόχου while κάθε επανάληψη του οποίου αντιστοιχεί και σε μία νέα μετατόπιση προς τα δεξιά. Η μετατόπιση αυτή η οποία αντιστοιχεί στη θέση ενός νέου mesh καθορίζει και την τιμή της x συντεταγμένης του.

Για να μην ξεπεράσουμε τα όρια της κυψέλης και βγούμε έξω από αυτήν, η τιμή της μεταβλητής xmesh πρέπει να είναι μικρότερη από τη θέση του δεξιού άκρου της "διαμέτρου".

```
while xmesh < 0.5 * sqrt(3.0)
```

Η σχέση που μας δίνει τη θέση του κάθε mesh στον άξονα x για την κάθε επανάληψη του βρόχου while είναι η:

```
xmesh = (k-1) * ds - 0.5 * sqrt(3.0)
```

Με βάση την παραπάνω εξίσωση, η τιμή της μεταβλητής xmesh η οποία αντιστοιχεί στη x συντεταγμένη του 1ου mesh θα είναι: xmesh = - 0.5 * sqrt (3.0).

Αυτή αποτελεί και την αρχική τιμή της παραπάνω μεταβλητής (k = 1).

Πράγματι όπως φαίνεται και από το σχήμα 2.9, η x συντεταγμένη του 1ου mesh ισούται με $x = -\alpha = -R\sqrt{3}/2$

Στη συνέχεια υπολογίζουμε για κάθε μετατόπιση στον άξονα x την ελάχιστη και μέγιστη τιμή (y_{min} και y_{max}) των meshes κατά τον άξονα y.

Χρησιμοποιώντας την συνθήκη `if xmesh < 0.0` αναφερόμαστε σε εκείνα τα meshes τα οποία βρίσκονται στο αριστερό τμήμα της διαμέτρου. Στην περίπτωση αυτή ισχύει

```
ymin = -xmesh/sqrt(3.0) - ds - 1.0
ymax = xmesh/sqrt(3.0) + 1.0
```

Από την άλλη, χρησιμοποιώντας την συνθήκη `if xmesh > 0.0` αναφερόμαστε σε εκείνα τα meshes τα οποία βρίσκονται στο δεξιό τμήμα της διαμέτρου. Στην περίπτωση αυτή ισχύει

```
ymin = xmesh/sqrt(3.0) - ds - 1.0
ymax = -xmesh/sqrt(3.0) + 1.0
```

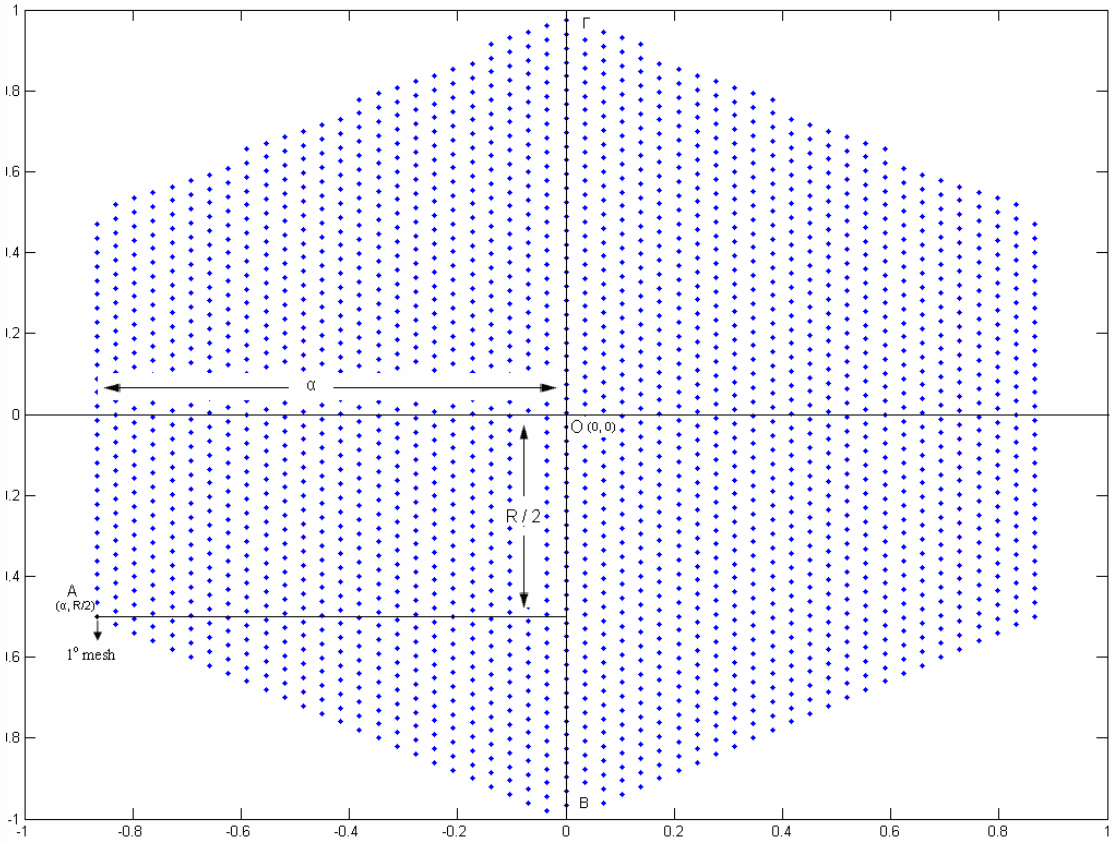
Τέλος χρησιμοποιώντας την συνθήκη `if xmesh == 0.0` αναφερόμαστε σε εκείνο το mesh το οποίο βρίσκεται στην αρχή των αξόνων. Στην περίπτωση αυτή ισχύει

```
ymin = -1
ymax = 1
```

Η μεταβλητή ds κατά τον υπολογισμό της μεταβλητής y_{min} προστίθεται για να απαλείψει τον εαυτό της κατά τον υπολογισμό της μεταβλητής ymesh (`ymesh = ymesh + ds`) πιο κάτω.

Η μεταβλητή y_{min} με αρχική τιμή ίση με -R / 2 = -0.5 (σχήμα 2.10 - σημείο A) σε κάθε βήμα της επανάληψης (τιμή της μεταβλητής xmesh) αυξάνεται κατά ds όσο βρισκόμαστε στο αριστερό τμήμα της "διαμέτρου", μέχρι την ελάχιστη τιμή που μπορεί να πάρει και η οποία ισούται με -1 (σημείο B, xmesh = 0.0). Στη συνέχεια εβρισκόμενη πλέον στο δεξιό τμήμα της διαμέτρου μειώνεται κατά ymesh μέχρι την τελική της τιμή η οποία ισούται με -0.5 .

Ομοίως η μεταβλητή y_{max} με αρχική τιμή την πλησιέστερη στο 0.5 σε κάθε βήμα της επανάληψης αυξάνεται κατά ymesh μέχρι την μέγιστη τιμή της που ισούται με 1 (σημείο Γ, xmesh = 0). Στη συνέχεια εβρισκόμενη πλέον στο δεξιό τμήμα της διαμέτρου μειώνεται κατά ymesh μέχρι την τελική της τιμή η οποία ισούται με την πλησιέστερη στο 0.5.



ΣΧΗΜΑ 2.10
Προσδιορισμός του ymesh

Όλες οι τιμές της μεταβλητής ymesh κατά τον κατακόρυφο άξονα y (αντιστοιχούν στο ίδιο xmesh) υπολογίζονται μέσω του εσωτερικού βρόχου while, ο οποίος διαρκεί μέχρι η μεταβλητή ymesh να ξεπεράσει την μέγιστη τιμή y_{max}

```
while ymesh < ymax
```

Πριν από αυτόν η μεταβλητή ymesh έχει ήδη αρχικοποιηθεί και έχει τεθεί ίση με την ελάχιστη τιμή y_{min}

Σε κάθε επανάληψη του βρόχου η τιμή της μεταβλητή ymesh αυξάνει κατά ds, όση είναι δηλαδή και η απόσταση μεταξύ των meshes, ενώ παράλληλα αυξάνεται και ο μετρητής j, δηλαδή ο αριθμός των meshes κατά 1 .

```
ymesh = ymesh + ds
```

Τέλος συμπληρώνονται οι στήλες του πίνακα meshposition με τις x και y συντεταγμένες του κάθε mesh που δημιουργήθηκε.

2.5 wrap.m

Με τη βοήθεια της παραπάνω συνάρτησης υπολογίζουμε τις γειτονικές κυψέλες για κάθε μία από τις 19 κυψέλες του συστήματος λαμβάνοντας υπόψη τη τεχνική cell – wrapping που αναλύσαμε πιο πάνω.

Με τον τρόπο αυτό συμπληρώνονται οι αντίστοιχες στήλες του πίνακα **inputmat,** ο οποίος είναι ένας 19 x 19 πίνακας**.** Κάθε γραμμή του αντιστοιχεί σε κάθε μία από τις 19 κυψέλες του συστήματος και κάθε στήλη του περιέχει τον αριθμό των γειτονικών κυψελών της, οι οποίες υπολογίζονται κινούμενοι αριστερόστροφα και περιλαμβάνοντας πρώτα τις κυψέλες της $1^{ης}$ ζώνης και στη συνέχεια αυτές της $2^{ης}$

2.6 holdtime.m

Η συνάρτηση αυτή υπολογίζει τη συνολική διάρκεια του χρόνου κλήσης για το κάθε χρήστη του συστήματος. Όπως αναφέραμε και πιο πάνω, η διάρκεια του χρόνου κλήσης είναι μια τυχαία μεταβλητή η οποία ακολουθεί εκθετική κατανομή με μέση τιμή h.

Το σχήμα 2.11 δείχνει τη συνάρτηση πυκνότητας πιθανότητας pdf (power distribution function) για τη συνολική διάρκεια του χρόνου κλήσης, ο οποίος υπολογίστηκε με την βοήθεια της παραπάνω συνάρτησης. Ο αριθμός των επαναλήψεων για τον υπολογισμό της pdf είναι N = 500 και η μέση τιμή του χρόνου κλήσης είναι ht = 120.

Για τον υπολογισμό της στο MATLAB χρησιμοποιήσαμε τη συνάρτησης **pdf()** και τον παρακάτω κώδικα

```
output = zeros (1, 500);
pdf_out = zeros (1, 500);
for i=1:500
    output (1,i) = holdtime(120.0);
    pdf_out (1, i) = pdf ('exp', output (1, i), 120);
end
plot (output, pdf_out, '.');
```

Με τη βοήθεια της συνάρτησης rand(), την οποία έχουμε αρχικοποιήσει στο κύριο πρόγραμμα (`rand('state', 5))` παράγουμε ένα τυχαίο αριθμό στο διάστημα [0, 1] τον οποίο και αποθηκεύουμε στη μεταβλητή **para.** Η μεταβλητή αυτή αντιστοιχεί στην τιμή της αθροιστικής συνάρτησης κατανομής cdf (cumulative distribution function) για τη συνολική διάρκεια του χρόνου κλήσης, η οποία δίνεται από τη σχέση:

Para = $f(x;\lambda) = 1 - e^{-\lambda x}$ όπου $\lambda = 1 / ht$ και x = call holding time.

Επομένως η μεταβλητή x (call holding time) λαμβάνεται μέσω της αντίστροφης αθροιστικής συνάρτησης κατανομής cdf^{-1} και ισούται με:

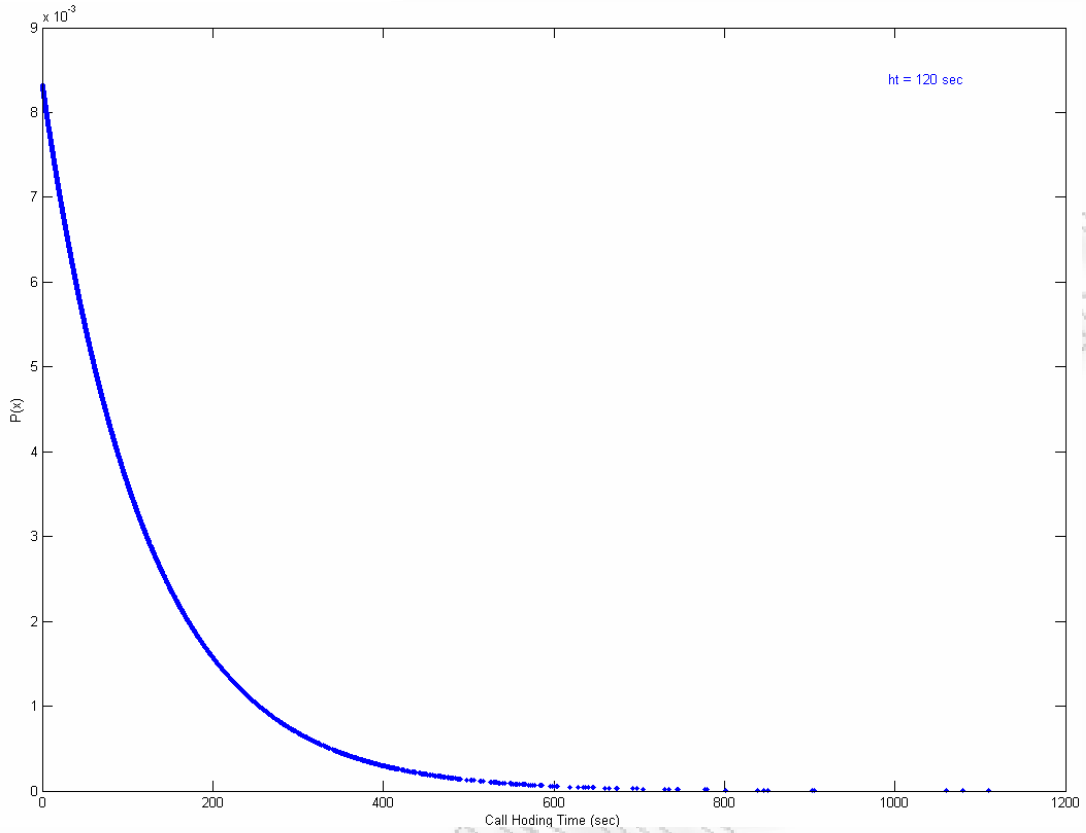
$x = - \ln(1 - p) / \lambda = - ht * \ln(1 - para)$

Η αντίστοιχη εντολή στο MATLAB είναι η παρακάτω:

```
x = ht.*(-log (1 - para))
```

Ο log αντιστοιχεί στο φυσικό λογάριθμο ln

Για να αποφύγουμε την περίπτωση η μεταβλητή para να ισούται με 1 κάτι που έχει σαν αποτέλεσμα το όρισμα του λογάριθμου να είναι 0, (κάτι που δεν ισχύει) χρησιμοποιούμε το βρόχο while ο οποίος επαναϋπολογίζει την παραπάνω μεταβλητή για την περίπτωση αυτή.



ΣΧΗΜΑ 2.11
Call holding time pdf

2.7 shadow.m

Η συνάρτηση αυτή υπολογίζει την εξασθένιση που υφίσταται το εκπεμπόμενο σήμα λόγω του φαινόμενου της σκέδασης. Η εξασθένιση αυτή ακολουθεί λογαριθμο-κανονική κατανομή με μέση τιμή μ και διασπορά σ η οποία χρησιμοποιείται και σαν όρισμα της συνάρτησης.

Στο σχήμα 2.12 φαίνεται η pdf της, η οποία υπολογίστηκε στο MATLAB μέσω του παρακάτω κώδικα

```
output = zeros (1, 500);
pdf_out = zeros(1, 500);
for i=1:500
    Output(1, i) = shadow(6.5);
    pdf_out (1, i) = pdf ('logn', output (1, i),0,1);
end
figure;
plot (output, pdf_out, '.');
```

Αρχικά με τη βοήθεια της συνάρτησης randn() την οποία έχουμε αρχικοποιήσει στο κύριο πρόγραμμα (`randn('state', 1))`, παράγουμε μια τυχαία μεταβλητή η οποία ακολουθεί κανονική κατανομή με μέση τιμή μ = 0 και διασπορά 1. Πολλαπλασιαζόμενη στην συνέχεια με την τιμή sigma και εκφραζόμενη σε dB, η μεταβλητή αυτή ακολουθεί πλέον λογαριθμο-κανονική κατανομή (lognormal distribution) με μέση τιμή μ = 0 και διασπορά σ .

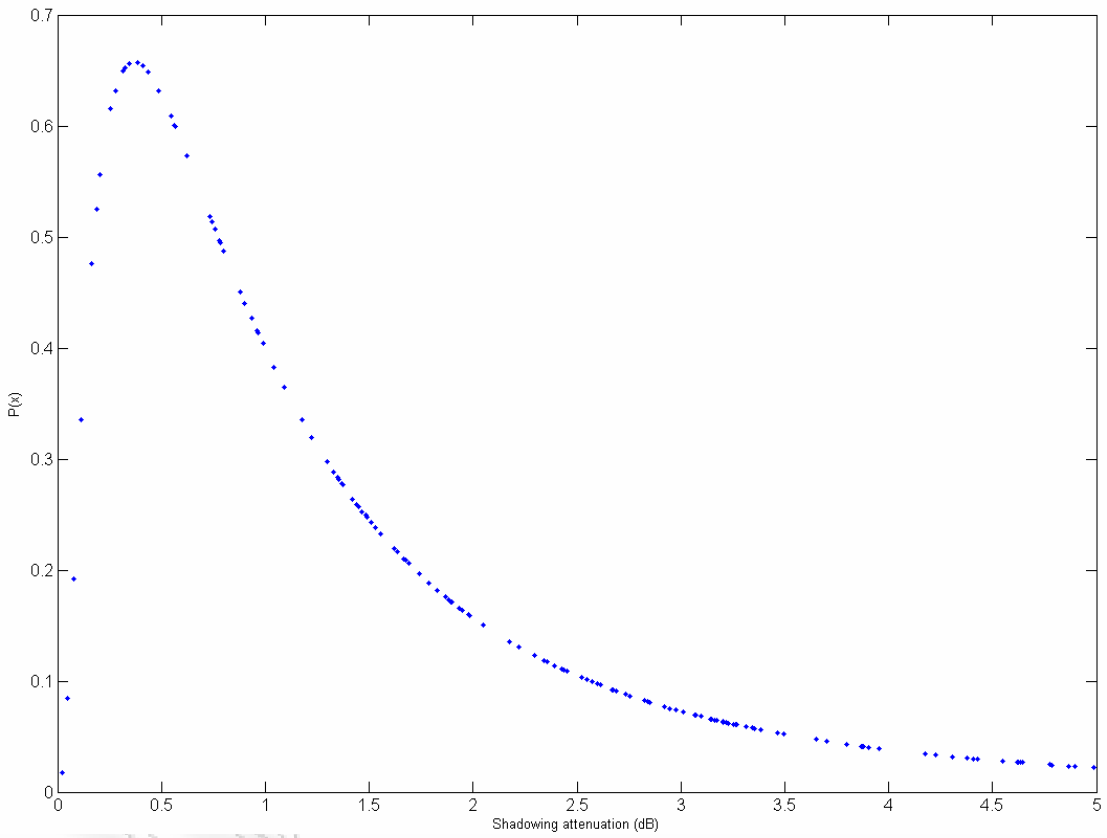
```
anoz = randn;
db = sigma * anoz;
```

Στη συνέχεια μετατρέπουμε την εξασθένιση από dB σε watt μέσω της σχέσης:

ξ (dB) = 10 * $\log_{10}(\xi$ (watt)) = > ξ (watt) = $10^{1/10 * \xi (dB)}$

δηλαδή στο MATLAB

```
x=power(10.0, 0.1 * db)
```



ΣΧΗΜΑ 2.12
Shadowing pdf

2.8 dist.m

Μέσω της συνάρτησης αυτής υπολογίζουμε την εξασθένιση που υφίσταται το εκπεμπόμενο σήμα λόγω της απόστασης. Η παραπάνω συνάρτηση δέχεται τρία ορίσματα: Τα 2 από αυτά αποθηκεύουν τους 2 πίνακες (1 x 2) οι οποίοι περιέχουν τις x και y συντεταγμένες των 2 σημείων για τα οποία θέλουμε να υπολογίσουμε την μεταξύ τους απόσταση. Το τρίτο δέχεται την τιμή του συντελεστή απωλειών (path loss factor), η οποία κυμαίνεται από 2 έως 4 και εξαρτάται από τα χαρακτηριστικά του περιβάλλοντος διάδοσης (για τον ελεύθερο χώρο α = 2).

Γνωρίζουμε ότι η απόσταση μεταξύ 2 σημείων A και B με συντεταγμένες X_A,Y_A και X_B, Y_B αντίστοιχα δίνεται από την παρακάτω σχέση:

$$d = \sqrt{(X_A - X_B)^2 + (Y_A - Y_B)^2}$$

Στο MATLAB η παραπάνω εξίσωση αντιστοιχεί στην εντολή:

```
x = sqrt ((a-b)*(a-b)')
```

όπου (a – b)΄ είναι ο ανάστροφος πίνακας της διαφοράς a – b

Η τελική τιμή της εξασθένισης συμπεριλαμβανομένου και του συντελεστή εξασθένισης α θα είναι:

pathloss (watt) = d^{-a} ή στο MATLAB

```
x = power(x, -1.* alpha)
```

2.9 dca_theory.m

Η συνάρτηση αυτή συγκρίνει την θεωρητική τιμή της πιθανότητας απόρριψης κλήσεων (blocking probability) με αυτήν που προκύπτει από την εκτέλεση του προγράμματος προσομοίωσης.

Η θεωρητική τιμή για την blocking probability υπολογίζεται μέσω της Engset Formula σύμφωνα με την παρακάτω σχέση [2]:

$$P_{bl\_teory} = \frac{\binom{n-1}{s}(vh)^s}{\sum_{i=0}^{s}\binom{n-1}{s}(vh)^i}$$

Στην παραπάνω εξίσωση οι μεταβλητές **n** και **s** αντιστοιχούν στον αριθμό των χρηστών και των διαθέσιμων καναλιών αντίστοιχα, ενώ οι μεταβλητές **v** και **h** αντιστοιχούν στην μέση τιμή του ρυθμού άφιξης κλήσεων και στο μέσο χρόνο διάρκειας κλήσεων αντίστοιχα.

Θα πρέπει να τονίσουμε ότι κατά τον υπολογισμό της blocking probability θεωρήσαμε ότι όλες οι παρεμβολές είναι μηδέν. Μελετώντας την συνάρτηση DCA_theory.m παρατηρούμε ότι είναι ακριβώς η ίδια με την **DCAmain.m** με την μόνη διαφορά ότι αγνοήσαμε το τμήμα που αναφέρεται στον έλεγχο αναδιανομής καναλιού, αφού όπως αναφέραμε και πιο πριν οι παρεμβολές δεν ελήφθησαν υπόψη.

Στο πρόγραμμα αυτό εκτός από την πειραματική τιμή της πιθανότητας απόρριψης κλήσεων υπολογίσαμε και την θεωρητική της τιμή.

Η πρώτη όπως έχουμε ήδη αναφέρει και πιο πριν, προκύπτει από το αποτέλεσμα της διαίρεσης του αριθμού των μπλοκαρισμένων κλήσεων με τον συνολικό αριθμό των παραγόμενων κλήσεων (pblock_simul = blocknum / callnum), ενώ η δεύτερη υπολογίστηκε μέσω της συνάρτησης **block_prob().**

Η συνάρτηση αυτή η οποία υπολογίζει την φόρμουλα του Engset, δέχεται 4 ορίσματα. Τον αριθμό των συνολικών χρηστών (user), τον αριθμό των διαθέσιμων καναλιών (chnum), τη μέση τιμή του ρυθμού άφιξης κλήσεων (lambda) και τη μέση τιμή του χρόνου διάρκειας κλήσεων (ht) αντίστοιχα. Για τον υπολογισμό των παραγοντικών χρησιμοποιήσαμε τη συνάρτηση **factorial().**

Η προσομοίωση εκτελέστηκε για αριθμό χρηστών 10, 15, 20 και 25, όχι όμως και για 5. Ο λόγος που έγινε αυτό είναι για να αποφύγουμε το λάθος που θα προέκυπτε κατά τον υπολογισμό του συνδυασμού $\binom{n-1}{s} = \binom{4}{5}$ ο οποίος ισχύει μόνο για n-1 > s .

Τέλος στον πίνακα output η τελευταία γραμμή αντιστοιχεί στη θεωρητική τιμή της blocking probability για τον αντίστοιχο αριθμό χρηστών και όχι στην τιμή της forced termination probability όπως συνέβη στο πρόγραμμα DCAmain.m.

```
output (5, parameter) = block_prop (user, chnum, lambda, ht)
```

2.10 beamforming

Εδώ θα μελετήσουμε το πως μεταβάλλεται η επίδοση του συστήματος μας εφαρμόζοντας παράλληλα και την τεχνική Beamforming. Το πρόγραμμα προσομοίωσης που θα χρησιμοποιήσουμε για το σκοπό αυτόν είναι το **DCA_Beamforming.m,** το οποίο και θα αναλύσουμε παρακάτω.

Αρχικά με τη βοήθεια του προγράμματος **BeamformingCNIR.m** θα δούμε την επίδραση της τεχνικής αυτής στην βελτίωση του λόγου CNIR. Το πρόγραμμα αυτό το οποίο αναπαριστά ένα κυψελοειδές σύστημα των 19 κυψελών, αρχικά υπολογίζει την τιμή του λόγου CNIR στην κεραία του σταθμού βάσης της κεντρικής κυψέλης (σχήμα 3.1) με και χωρίς την χρήση κατευθυντικών κεραιών ($CNIR_{Beamforning}$ και CNIR αντίστοιχα) και στη συνέχεια τη διαφορά $CNIR_{Beamforning}$ – CNIR. Η διαφορά αυτή υπολογίζεται για ένα εύρος τιμών του πλάτους δέσμης του κύριου λοβού της κεραίας (beamwidth) και απεικονίζεται από την αντίστοιχη γραφική παράσταση, που θα δούμε παρακάτω.

Στο πρόγραμμα αυτό, ο αριθμός των χρηστών ανά κυψέλη ισούται με 1. Επίσης θεωρούμε ότι όλες οι κυψέλες χρησιμοποιούν την ίδια συχνότητα (ραδιοδίαυλο) και επομένως λαμβάνονται υπόψη κατά τον υπολογισμό της παρεμβολής.

2.11 BeamformingCNIR.m

Η τιμή του πλάτους της δέσμης του κύριου λοβού της κεραίας του κεντρικού σταθμού βάσης (στην οριζόντια κατεύθυνση) αντιστοιχεί στη μεταβλητή ***w_HBS*.** Το πρόγραμμα αυτό εκτελείται για τιμές της παραπάνω μεταβλητής μεταξύ 30^o και 120^o με βήμα αύξησης 10^o, μέσω του βρόχου επανάληψης for.

```
for w_HBS = 30:10:120
```

Η μεταβλητή **ι** είναι ένας δείκτης, η τιμή του οποίου αντιστοιχεί στην τρέχουσα επανάληψη του βρόχου **for**.

Οι τιμές της διαφοράς $CNIR_{Beamforning}$ – CNIR για την κάθε τιμή της μεταβλητής ***w_HBS*** αποθηκεύονται στην αντίστοιχη στήλη του πίνακα **output**, ο οποίος είναι ένας (1 x (120 – 30) / 10 + 1) πίνακας

Οι μεταβλητές που χρησιμοποιήσαμε ως δεδομένα στο πρόγραμμα BeamformingCNIR.m αναφέρονται στον πίνακα. 4:

Αριθμός χρηστών	user = 1
Αριθμός κυψελών για τον οποίο θα υπολογιστεί η βελτίωση του CNIR	numcell = 1
C / N maximum (dB)	cnedge = 20
C / (N * I) threshold (dB)	cnirth = 15
Συντελεστής απωλειών διάδοσης	alpha = 3.5
Τυπική απόκλιση συντελεστή σκέδασης	sigma = 6.5
Συνολική διάρκεια της προσομοίωσης (sec)	timend = 5000
Βήμα της προσομοίωσης	timestep = 10
Ύψος κεραίας BS (m)	h = 0
Beamwidth στον BS (οριζόντιο επίπεδο) (^{o})	w_HBS = 30, 40, 50, … , 120
Κέρδος κεραίας στον BS για την αντίθετη κατεύθυνση (οριζόντιο επίπεδο) (dB)	backg_BS = -100
Beamwidth στον BS (κατακόρυφο επίπεδο) (^{o})	w_VBS = 360
CNIR χωρίς την τεχνική Beamforming	cnirdb_all (dB)
CNIR με την τεχνική Beamforming	cnirdb_BF_all (dB)

ΠΙΝΑΚΑΣ 4

Θα πρέπει να τονίσουμε ότι στο σύστημα μας εφαρμόσαμε το macrocell μοντέλο για τη κάθε κυψέλη, το οποίο προϋποθέτει ύψος κεραίας για το σταθμό βάσης ίσο με μηδέν (h = 0).

1. Κατά το στάδιο αρχικοποίησης καθορίσαμε τις θέσεις των 19 σταθμών βάσης καθώς και τις πιθανές θέσεις των χρηστών μέσα στις κυψέλες, μέσω των συναρτήσεων **baseinfo()** και **cellmesh()** αντίστοιχα. Ο πίνακας **baseinfo_new** είναι ένας πίνακας (19 x 1) κάθε γραμμή του οποίου περιέχει τις συντεταγμένες του κάθε σταθμού βάσης σε μιγαδική μορφή. Γνωρίζουμε ότι οι x και y συντεταγμένες ενός σημείου (a, b) σε μιγαδική μορφή δίνονται από τη σχέση a + j * b.

   Με βάση τα παραπάνω εάν λάβουμε λοιπόν υπόψη, ότι η πρώτη στήλη του πίνακα baseinfo περιέχει τις x συντεταγμένες των σταθμών βάσης και η δεύτερη τις y τότε θα ισχύει:

   ```
   baseinfo_new = baseinfo(:, 1) + baseinfo(:, 2) * j;
   ```

   Ο λόγος που χρησιμοποιήσαμε αυτή τη μετατροπή είναι για να μπορέσουμε στη συνέχεια να υπολογίσουμε το μέτρο και την φάση μεταξύ των συντεταγμένων αυτών, όπως θα δούμε πιο κάτω.

2. Στο στάδιο αυτό επίσης υπολογίσαμε το οριζόντιο και κατακόρυφο κέρδος της κεραίας του σταθμού βάσης για όλες τις γωνίες έλευσης από 0^o έως 360^o μέσω της συνάρτησης **antgain()**. Οι τιμές αυτές αποθηκεύονται στις αντίστοιχες στήλες των πινάκων **g_HBS** και **g_VBS** αντίστοιχα. Πιο συγκεκριμένα η καθεμία από τις 360 στήλες του κάθε πίνακα περιέχει εκείνη τη τιμή του κέρδους που αντιστοιχεί στη αντίστοιχη γωνία έλευσης μεταξύ πομπού και δέκτη. Η μέγιστη τιμή κέρδους αντιστοιχεί σε γωνία ίση με 0^o

   ```
   g_HBS = antgain(w_HBS, backg_BS);
   g_VBS = antgain(w_VBS, 0);
   ```

   Τη συνάρτηση αυτή η οποία δέχεται σαν ορίσματα της τη τιμή του πλάτους της δέσμης του κύριου λοβού και την τιμή του κέρδους κατά την αντίθετη κατεύθυνση θα την αναλύσουμε στη συνέχεια..

   Κατά τον υπολογισμό του κατακόρυφου κέρδους g_VBS η τιμή του κέρδους κατά την αντίθετη κατεύθυνση ισούται με μηδέν, αφού το vertical beamwidth ισούται με 360^o.

3. Στο 3^o στάδιο ακολουθεί ο καθορισμός των θέσεων του κάθε χρήστη για κάθε μία από τις 19 κυψέλες του συστήματος με τον ίδιο ακριβώς τρόπο τον οποίο είδαμε και στο πρόγραμμα DCAmain.m.

Ο πίνακας **userposi_new** είναι ένας (19 x 1) πίνακας, ο οποίος όπως και ο baseinfo_new περιέχει τις συντεταγμένες του κάθε χρήστη σε μιγαδική μορφή.

```
userposi_new = userposi(:, 1) + userposi(:, 2) * j;
```

Ομοίως ο πίνακας **mesh_position_new** είναι και αυτός ένας (19 x 1) πίνακας, ο οποίος περιέχει τις συντεταγμένες του κάθε χρήστη σε σχέση με τον αντίστοιχο σταθμό βάσης σε μιγαδική μορφή.

Ο πίνακας αυτός υπολογίζεται από τη διαφορά userposi_new (συντεταγμένες του κάθε χρήστη σε σχέση με την αρχή των αξόνων) - baseinfo_new (συντεταγμένες του κάθε BS σε σχέση με την αρχή των αξόνων).

```
meshposition_new = userposi_new - baseinfo_new;
```

Τέλος ο πίνακας z είναι ένας (2 x 19) πίνακας, του οποίου η πρώτη γραμμή αντιστοιχεί στις συντεταγμένες του κάθε χρήστη σε σχέση με τον αντίστοιχο σταθμό βάσης του και η δεύτερη γραμμή του στις συντεταγμένες του κάθε χρήστη σε σχέση με την αρχή των αξόνων (κεντρικός σταθμός βάσης).

```
z = [meshposition_new'; userposi_new'];
```

Όπως βλέπουμε ο πίνακας αυτός, ο οποίος περιέχει τις συντεταγμένες των 19 χρηστών σε μιγαδική μορφή, προκύπτει από τον συνδυασμό των πινάκων meshposition_new και userposi_new.

Ο **meshposition_new'** είναι ο ανάστροφος του meshposition_new, ομοίως και ο **userposi_new'** είναι ο ανάστροφος του userposi_new**.**

4. Το 4^{ο} στάδιο περιλαμβάνει τον υπολογισμό του κέρδους για την κεραία του κεντρικού σταθμού βάσης.

   Ο πίνακας **d** είναι ένας (2 x 19) πίνακας. Η 1^{η} γραμμή του περιέχει το μέτρο καθεμίας από τις 19 μιγαδικές στήλες του. Το μέτρο αυτό αντιστοιχεί στην απόσταση κατά το οριζόντιο επίπεδο του κάθε χρήστη από τον σταθμό βάσης του. Ομοίως η 2^{η} γραμμή του περιέχει την απόσταση του κάθε χρήστη από την αρχή των αξόνων, δηλαδή από τον κεντρικό σταθμό βάσης.

   ```
   d(1, : ) = abs(z(1, :));
   d(2, : ) = abs(z(2, :));
   ```

   Για τον υπολογισμό των παραπάνω αποστάσεων χρησιμοποιήσαμε τη συνάρτηση **abs()** του MAT LAB.

   Κατά ανάλογο τρόπο ο πίνακας **phai** είναι ένας (2 x 19) πίνακας, του οποίου η 1^{η} γραμμή περιέχει τη γωνία κατά το οριζόντιο επίπεδο μεταξύ του κάθε

χρήστη και του σταθμού βάσης της κυψέλης στην οποία αυτός ανήκει, ενώ η δεύτερη γραμμή περιέχει τη γωνία μεταξύ του κάθε χρήστη και του κεντρικού σταθμού βάσης.

Ο υπολογισμός των παραπάνω γωνιών οι οποίες εκφράζονται σε rad έγινε μέσω της συνάρτησης **angle()** του MATLAB.

```
phai(1, :) = angle(z(1, :));
phai(2, :) = angle(z(2, :));
```

Τέλος ο πίνακας **deg** είναι και αυτός ένας (2 x 19) πίνακας, ο οποίος περιέχει τις τιμές του πίνακα phai εκφρασμένες σε μοίρες. Η μετατροπή των ακτίνιων (rad) σε μοίρες (degrees) έγινε μέσω της σχέσης:

$angle_{degrees} = angle_{rad} * (\pi / 180)$.

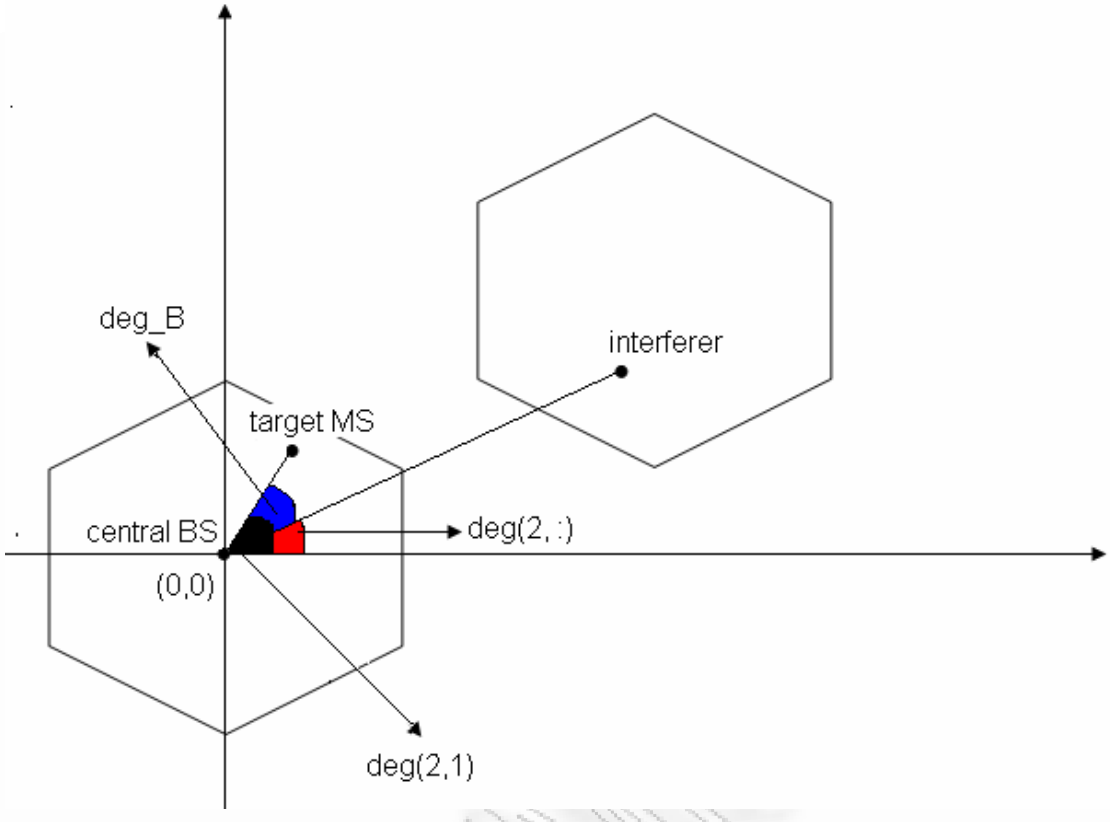
ή στο MATLAB *deg = phai * (180 / pi);*

Ο πίνακας **deg_B** είναι ένας (1 x 19) πίνακας, ο οποίος προκύπτει από τη διαφορά της γωνίας του κεντρικού σταθμού βάσης και του χρήστη της κεντρικής κυψέλης (deg(2, 1)) και της γωνίας του κεντρικού σταθμού βάσης και των υπόλοιπων χρηστών των άλλων κυψελών (deg(2, :)).

```
deg_B = deg(2, 1) - deg(2, :);
```

Ο παράγοντας deg(2, 1) αντιστοιχεί στη γωνία αναφοράς, για την οποία ο κεντρικός σταθμός βάσης ρυθμίζει τα βάρη της στοιχειοκεραίας του, ώστε η λαμβανόμενη τιμή του λόγου CNIR να είναι η μέγιστη. Η γωνία αυτή υπολογίζεται σε σχέση με την αρχή των αξόνων και αντιστοιχεί στη μέγιστη τιμή κέρδους.

Ο παράγοντας deg(2, :) αντιστοιχεί στη γωνία μεταξύ του κεντρικού σταθμού και των κινητών σταθμών των υπολοίπων κυψελών (παρεμβολέων) θεωρώντας ως σημείο 0^o την αρχή των αξόνων. Επομένως η διαφορά deg(2, 1) - deg(2, :) αντιστοιχεί στη γωνία που σχηματίζεται μεταξύ του κάθε παρεμβολέα και του "κύριου" κινητού σταθμού (κυψέλη 1) (σχήμα 2.13). Η γωνία αυτή μας δίνει την τιμή του οριζόντιου κέρδους της κεραίας του κεντρικού σταθμού βάσης για ην παραπάνω κατεύθυνση, η οποία αντιστοιχεί στη θέση του αντίστοιχου παρεμβολέα. Την τιμή αυτή την λαμβάνουμε από εκείνη την στήλη του πίνακα g_HBS, η οποία ισούται με την τιμή της γωνίας αυτής.



ΣΧΗΜΑ 2.13
Υπολογισμός της γωνίας έλευσης

Ομοίως ο πίνακας **degHBS** είναι και αυτός ένας (1 x 19) πίνακας, ο οποίος περιέχει τις τιμές του πίνακα deg_B προσαρμοσμένες στο διάστημα από 0 έως 360 μοίρες.

Η μετατροπή αυτή προκύπτει από το υπόλοιπο της διαίρεσης των τιμών του πίνακα deg_B στρογγυλοποιημένων στον πλησιέστερο ακέραιο αριθμό, με το 360 (αντιστοιχεί σε έναν πλήρη κύκλο) .

```
degHBS = mod(round(deg_B), 360);
```

Για το κατακόρυφο επίπεδο η γωνία μεταξύ των κινητών σταθμών και του κεντρικού σταθμού βάσης είναι η ίδια και ισούται με 90^{o}. Οι τιμές αυτές αποθηκεύονται στις αντίστοιχες στήλες του πίνακα **degH** ο οποίος είναι επίσης ένας (1 x 19) πίνακας.

```
degH = 90 * ones(1, 19);
```

Η τιμή κέρδους της κεραίας του κεντρικού σταθμού βάσης που αντιστοιχεί τόσο στον "κύριο" κινητό σταθμό, όσο και στους γειτονικούς παρεμβολείς,

υπολογίζεται από το άθροισμα των κερδών των πινάκων g_HBS και g_VBS για τις αντίστοιχες γωνίες έλευσης που υπολογίσαμε πιο πάνω.

Οι τιμές αυτές στη συνέχεια αποθηκεύονται στον πίνακα **gain** ο οποίος είναι ένας (1 x 19) πίνακας, κάθε στήλη του οποίου περιέχει την τιμή κέρδους για τον αντίστοιχο χρήστη

```
gain = g_HBS(degHBS(1:19) + 1) + g_VBS(degVBS(1:19) + 1);
```

Στην παραπάνω εξίσωση προσθέτουμε το 1 για να μπορούμε να προσπελάσουμε όλα τα στοιχεία του πίνακα g_HBS, ακόμα και κατά την περίπτωση που κάποιο από τα στοιχεία του πίνακα degHBS ισούται με μηδέν (degHBS(i) = 0)

Ο πίνακας **gain_W** περιέχει τις τιμές των κερδών αυτών εκφρασμένες σε watt

```
gain_W = 10 .^ ( gain ./ 10);
```

5. Στο 5^{o} και τελευταίο στάδιο ακολουθεί ο υπολογισμός της τιμής του λόγου CNIR που λαμβάνεται από το κεντρικό σταθμό βάσης για δύο περιπτώσεις. Με και χωρίς τη χρήση κατευθυντικών κεραιών.

   Οι παραπάνω τιμές υπολογίζονται με τον ίδιο ακριβώς τρόπο, τον οποίο χρησιμοποιήσαμε και στο πρόγραμμα DCAmain.m .

   Η μόνη διαφορά είναι ότι κατά τον υπολογισμό της ισχύος της συνολικής παρεμβολής με τη χρήση της τεχνικής Beamforming, η τιμή της παρεμβολής u_wave_BF, που προκαλείται από καθέναν από τους γειτονικούς χρήστες πολλαπλασιάζεται με την αντίστοιχη τιμή κέρδους για τον χρήστη αυτόν.

```
uwave_BF = uwave_BF + dist(here, there, alpha) * shadow(sigma)
* grain(1, othercell);
```

   Εδώ θα πρέπει να προσέξουμε μια σημαντική λεπτομέρεια κατά τον υπολογισμό των μεταβλητών uwave και uwave_BF. Επειδή κατά τον υπολογισμό τους χρησιμοποιείται η συνάρτηση shadow() δύο φορές (μία κατά τον υπολογισμό της uwave και μία κατά τον υπολογισμό της uwave_BF) το αποτέλεσμά της θα είναι διαφορετικό για κάθε μία από αυτές τις δύο μεταβλητές. Ο λόγος οφείλεται στο ότι η συνάρτηση αυτή παράγει μια τυχαία τιμή, η οποία είναι διαφορετική σε κάθε μία από τις δύο κλήσεις της. Για αποφύγουμε το σφάλμα αυτό, καλούμε την συνάρτηση αυτή μια φορά πριν από τον υπολογισμό των παραπάνω μεταβλητών και το αποτέλεσμα που μας

επιστρέφει το αποθηκεύομαι στη μεταβλητή **shadowing** Αυτή είναι και η μεταβλητή που χρησιμοποιούμε για τον υπολογισμό της σκέδασης.

```
shadowing = shadow(sigma);
uwave = uwave + dist(here, there, alpha) * shadowing;
uwave_BF = uwave_BF + dist(here, there, alpha) * shadowing *
gain_W(1, othercell);
```

Επίσης η λαμβανόμενη τιμή της ισχύος από εκείνον τον χρήστη τον οποίο επιθυμούμε (κυψέλη 1) πολλαπλασιάζεται και αυτή με την αντίστοιχη τιμή κέρδους. Η τιμή αυτή όπως είδαμε πιο πάνω αντιστοιχεί στη μέγιστη τιμή κέρδους η οποία ισούται με 1 watt ή 0 dB. Για το λόγο αυτό την παραλείπουμε κατά τον υπολογισμό της μεταβλητής **cn**.

2.12 antgain.m

Η συνάρτηση αυτή η οποία υπολογίζει το κέρδος της κεραίας για όλες της γωνίες έλευσης, δέχεται δύο ορίσματα. Την τιμή του πλάτους της δέσμης του κύριου λοβού της κεραίας (όρισμα **width**) και το κέρδος της κεραίας κατά την αντίθετη κατεύθυνση (όρισμα **back)**.

Οι μεταβλητές **theta** και **gain** είναι δύο διανύσματα 1 x 360. Οι τιμές του πρώτου αντιστοιχούν σε έναν πλήρη κύκλο (σε μοίρες), ενώ του δεύτερου στις τιμές των κερδών για τις αντίστοιχες γωνίες έλευσης (τιμές του διανύσματος **theta**).
Όσον αφορά το διάνυσμα gain, η πρώτη τιμή του (theta(1)) περιέχει την τιμή κέρδους για εκείνη τη γωνία έλευσης η οποία αντιστοιχεί στη γωνία μεταξύ του κύριου σταθμού βάσης και του χρήστη που επιθυμούμε (κύριου χρήστη). Η γωνία αυτή ισούται μηδέν μοίρες για τον πίνακα gain και αντιστοιχεί σε τιμή κέρδους ίση με 0 dB ή 1 watt. Οι υπόλοιπες τιμές του πίνακα gain, περιέχουν τις τιμές των κερδών για γωνίες έλευσης υπολογισμένες σε σχέση με τη γωνία αναφοράς.

Η μαθηματική συνάρτηση, η οποία μας δίνει τις τιμές του κέρδους συναρτήσει της γωνίας έλευσης φ για το συγκεκριμένο τύπο κεραίας (στοιχειοκεραία) είναι η παρακάτω:

$$G_{HB_i}(\phi) = \begin{cases} 10\log_{10}\cos^{n}(\phi) & -\pi/2 \le \varphi \le \pi/2 \\ \chi & \text{σε κάθε άλλη περίπτωση} \end{cases}$$

Από αυτήν παρατηρούμε ότι για γωνίες έλευσης εντός του διαστήματος [-π/2, π/2] ή [0^o, 90^o] U [271^o, 360^o] η τιμή κέρδους υπολογίζεται μέσω της εξίσωσης **$\log_{10}\cos(\theta\pi/180)$**

ή στο MATLAB

```
gain(1:89) = 10 * log10(cos(theta(1:89) * (pi/180)) .^ n);
gain(272:360) = 10 * log10(cos(theta(272:360) * (pi/180)) .^ n);
```

Αντιθέτως για γωνίες έλευσης εκτός του διαστήματος αυτού, η τιμή κέρδους ισούται με την τιμή κέρδους κατά την αντίθετη κατεύθυνση

```
if back ~=0
gain(90:271) = back;
```

Θα πρέπει επίσης να αναφέρουμε ότι η παραπάνω σχέση μπορεί να χρησιμοποιηθεί κατά τον υπολογισμό του αντίστοιχου κέρδους και κατά την κατακόρυφη κατεύθυνση **$G_{VBi}(\varphi)$.**

Στην παραπάνω εξίσωση οι μεταβλητές **φ** και **x** αντιστοιχούν στην γωνία έλευσης και στην τιμή κέρδους κατά την αντίθετη κατεύθυνση. Τέλος η μεταβλητή **n** είναι μία παράμετρος μέσω της οποίας υπολογίζεται η τιμή του πλάτους της δέσμης του κύριου λοβού της κεραίας βάση της ακόλουθης σχέσης.

$$n = \frac{\log_{10}\sqrt{1/2}}{\log_{10}\cos\left(\theta\pi/180\right)}$$

Η συνάρτηση antgain.m ισχύει για κατευθυντικές κεραίες και όχι για πολυκατευθυντικές (beamwidth = 360^o), όπως καθορίζεται μέσω της συνθήκης ελέγχου if.

```
if width ~= 360
```

Πράγματι για τιμή της μεταβλητής width = 360 = 2π, προκύπτει ότι κατά τον υπολογισμό της παραμέτρου n, ο παρανομαστής $\log_{10}\cos(\theta\pi/180)$ ισούται με μηδέν γεγονός που οδηγεί σε σφάλμα.

2.13 DCABeamforming.m

Στο πρόγραμμα αυτό θα μελετήσουμε πως βελτιώνεται η επίδοση του συστήματος μας χρησιμοποιώντας επιπλέον και την τεχνική beamforming. Ο κώδικας που θα χρησιμοποιήσουμε είναι ο ίδιος με αυτόν που χρησιμοποιήσαμε και στο πρόγραμμα **DCAmain.m** με τη μόνη διαφορά τον υπολογισμό της μεταβλητής **uwave_BF**. Η τιμή της ισούται με το άθροισμα των εξασθενίσεων των επιμέρους παρεμβολών που φτάνουν στην κεραία του σταθμού βάσης από τους αντίστοιχους χρήστες πολλαπλασιαζόμενες με τις αντίστοιχες τιμές κερδών. Για τον υπολογισμό των κερδών αυτών χρησιμοποιείται συνάρτηση **antgain_interf()**.

```
ant_gain = antgain_interf (here(1, 1), here(1, 2), userinfo(namely,
numuser, 1), userinfo(numcell, numuser, 2), there(1, 1), there(1, 2),
w_HBS);
uwave = uwave + dist(here, there, alpha) * shadow(sigma) * ant_gain;
```

Η συνάρτηση αυτή δέχεται 7 ορίσματα. Τις x και y συντεταγμένες του "κύριου" χρήστη (μεταβλητές X_user και Y_user), τις x και y συντεταγμένες των γειτονικών χρηστών (παρεμβολέων) (X_otheruser και Y_otheruser), τις x και y συντεταγμένες του κύριου σταθμού βάσης (X_BS και Y_BS) και τη τιμή του πλάτους της δέσμης του κύριου λοβού της κεραίας (width).

Στη συνέχεια, μέσω της συνάρτησης **antgain ()** συμπληρώνονται οι στήλες του πίνακα g_HBS με τις αντίστοιχες τιμές κέρδους όπως ήδη είδαμε πιο πάνω

```
g_HBS = antgain(width, -100);
```

Οι συντεταγμένες που δέχεται σαν ορίσματα η συνάρτηση μας μετατρέπονται σε μιγαδική μορφή.

```
user_coord = X_user + Y_user * j;
otheruser_coord = X_otheruser + Y_oheruser * j;
BS_coord = X_BS + Y_BS * j;
```

Αυτό γίνεται για να μπορέσουμε στη συνέχεια να υπολογίσουμε τη γωνία (ως προς το οριζόντιο επίπεδο), κατά πρώτον μεταξύ του κύριου χρήστη και του κύριου σταθμού βάσης (μεταβλητή **phai1**) και κατά δεύτερον μεταξύ του γειτονικού χρήστη και του κύριου σταθμού βάσης (μεταβλητή **phai2**).

```
phai1 = angle(user_coord - BS_coord);
phai2 = angle(otheruser_coord);
```

Στη συνέχεια, αφού οι γωνίες αυτές μετατραπούν από ακτίνια σε μοίρες

```
deg1 = phai1 * (180 / pi);
deg2 = phai2 * (180 / pi);
```

υπολογίζεται η γωνία μεταξύ του γειτονικού χρήστη και του κύριου χρήστη πάντα ως προς το οριζόντιο επίπεδο.

```
deg_B = deg1 - deg2;
```

Η οποία στη συνέχεια προσαρμόζεται στο διάστημα από 0^o έως 360^o

```
degHBS = mod(round(deg_B), 360);
```

Τέλος μέσω του πίνακα g_HBS υπολογίζεται η τιμή κέρδους η οποία αντιστοιχεί στη γωνία έλευσης deg_B που υπολογίσαμε πιο πάνω

```
gain = g_HBS(degHBS + 1);
```

και το αποτέλεσμα μετατρέπεται από dB σε watt

```
out = 10 .^ ( gain ./ 10);
```

Θα πρέπει τέλος να αναφέρουμε ότι για το κατακόρυφο επίπεδο θεωρούμε ότι το πλάτος της δέσμης του κύριου λοβού της κεραίας ισούται με 360^o(πολυκατευθυντική κεραία), γεγονός που σημαίνει ότι η τιμή του κατακόρυφου κέρδους είναι η ίδια για όλο το εύρος των γωνιών και ίση με 1 watt ή 0 dB. Για το λόγο αυτό και δεν την συμπεριλαμβάνουμε στους υπολογισμούς μας για λόγους απλότητας.

2.14 power control

Εδώ θα μελετήσουμε το πως μεταβάλλεται η επίδοση του συστήματος μας εφαρμόζοντας, εκτός από τη χρήση των κατευθυντικών κεραιών που αναλύσαμε πιο πάνω και τον έλεγχο ισχύος. Το πρόγραμμα προσομοίωσης που θα χρησιμοποιήσουμε για το σκοπό αυτόν είναι το **DCA_PowerControl.m,** το οποίο και θα αναλύσουμε παρακάτω.

Ο έλεγχος ισχύος που εφαρμόζεται στου κινητούς σταθμούς των χρηστών του συστήματος, έχει σαν στόχο η τιμή του λόγου CNR που λαμβάνεται από τον κάθε σταθμό βάσης να παραμένει σταθερή για όλους τους χρήστες ανεξαρτήτου της απόστασης τους από αυτόν. Αυτό γίνεται μειώνοντας την τιμή της εκπεμπόμενης ισχύος σε σχέση με την τιμή αναφοράς, την τιμή δηλαδή του λόγου CNR στα όρια της κυψέλης (αντιστοιχεί στην μέγιστη τιμή της ισχύος που μπορεί να εκπέμψει ένας κινητός σταθμός.) ανάλογα με την απόσταση μεταξύ του κάθε χρήστη και του σταθμού βάσης του.

Με βάση τα παραπάνω η τιμή της εκπεμπόμενης ισχύος από τον κάθε χρήστη μειώνεται όσο αυτός βρίσκεται πλησιέστερα στο σταθμο βάσης του και υπολογίζεται μέσω της παρακάτω σχέσης.

Για απόσταση R = 1 Km (όρια της κυψέλης) $CNR = CNR_{max} = cnedge\,(watt)$

Για απόσταση d (θέση του χρήστη) CNR ;

$$CNR = (d * CNR_{max}) / 1km\,(watt) = d * cnedge$$

Πριν απ' όλα αυτά, όπως και κατά την ανάλυση της τεχνικής Beamforming, πρώτα θα δούμε την επίδραση της τεχνικής αυτής στην βελτίωση του CNIR λόγου μέσω της συνάρτησης **PowerControlCNIR.m.**

Η συνάρτηση αυτή είναι ακριβώς η ίδια με την αντίστοιχη **BamformingCNIR.m** με τη διαφορά ότι κατά τον υπολογισμό της συνολικής παρεμβολής και της εξασθένισης του εκπεμπόμενου σήματος λόγο του περιβάλλοντος διάδοσης (path loss και shadowing) η τιμή της εκπεμπόμενης ισχύος τροποποιήθηκε βάση της παραπάνω σχέσης**.**

Έτσι η τιμή της συνολικής παρεμβολής και εν τέλει του λόγου CNIR στην κεραία του σταθμού βάσης για αυτήν την περίπτωση θα ισούται με:

$$I_{total} = \sum_{j=1}^{N} I_j = \sum_{j=1}^{N} power_j * attenuation_j = \sum_{j=1}^{N} power_{\max} \cdot d_j \cdot attenuation_j = power_{\max} * \sum_{j=1}^{N} d_j \cdot attenuation_j$$

και

$$CNIR(dB) = \frac{C}{N+I} = \frac{C/C}{N/C + I/C} = \frac{1}{(C/N)^{-1+}(C/I)^{-1}} =$$

$$\frac{1}{(cnedge * d_{user} * dwave) + (\frac{cnedge * d_{user} * N * dwave}{cnedge * d_{other} * N * attenuation_{other}})}$$

$$\Rightarrow CNIR(dB) = \frac{1}{\frac{1}{cn\_pc} + \frac{uwave\_pc}{dwave}}$$

ή στο MATLAB

```
uwave_PC = uwave_PC + dist(here, there, alpha) * shadowing *
gain_W(1, othercell) * d(1, around);
cn_PC = power(10, cnedge / 10.0) * dwave * d(1, 1);
cnirdb_PC = 10.0 * log10(1 / ((uwave_PC / dwave) + (1 / cn_PC)));
```

Όπως ήδη έχουμε δη, η τιμή d(1,1) αντιστοιχεί στην απόσταση του κύριου χρήστη από τον κύριο σταθμό βάσης του, ενώ η τιμή d(1, around) στην απόσταση του γειτονικού χρήστη από τον σταθμό βάσης του.

Ομοίως και το πρόγραμμα **DCAPowerControl.m** μέσω του οποίου μελετάμε πως βελτιώνεται η απόδοση του συστήματος μας με τη χρήση επιπλέον και της τεχνικής ελέγχου ισχύος, είναι ακριβώς το ίδιο με το αντίστοιχο **DCABeamforming.m** που αναλύσαμε πιο πάνω.
Η μόνη διαφορά και εδώ αφορά τον υπολογισμό των μεταβλητών **cn** και **uwave** κατά τον υπολογισμό των οποίων λαμβάνουμε υπόψη και τον έλεγχο ισχύος. Έτσι κατά το στάδιο ελέγχου αναδιανομής καναλιού, για τον υπολογισμό του λόγου C/N που φτάνει στην κεραία του κύριου σταθμού βάσης του θα ισχύει:

```
distance = abs(userinfo(numcell, numuser, 1) - baseinfo(numcell, 1) +
(userinfo(numcell, numuser, 2) - baseinfo(numcell, 2)) * j);
cn = power(10.0, cnedge / 10.0) * distance * dwave;
```

όπου distance είναι η απόσταση μεταξύ του κύριου χρήστη και του αντίστοιχου σταθμού βάσης του. Η απόσταση αυτή αντιστοιχεί στο μέτρο του μιγαδικού αριθμού ο οποίος προκύπτει από την αφαίρεση εκείνων των διανυσμάτων τα οποία περιέχουν τις θέσεις του κύριου χρήστη και του σταθμού βάσης του αντίστοιχα.
Ενώ για τον υπολογισμό της συνολικής παρεμβολής που λαμβάνεται από τον κύριο σταθμό βάσης θα ισχύει αντίστοιχα

```
distance = abs(userposi(1) - baseinfo (othercell, 1) + (userposi(2) -
baseinfo(othercell, 2)) * j);
uwave = uwave + dist(here, there, alpha) * shadow(sigma) * distance *
ant_gain;
```

όπου εδώ η μεταβλητή distance αντιστοιχεί στο μέτρο της απόστασης μεταξύ του γειτονικού χρήστη και του αντίστοιχου σταθμού βάσης του.
Ομοίως κατά το στάδιο άφιξης νέων κλήσεων για τον υπολογισμό του λόγου C/N και της συνολικής παρεμβολής θα έχουμε αντίστοιχα:

```
distance = abs(there(1) - here(1) + (there(2) - here(2)) * j);
cn = power(10.0, cnedge / 10.0) * distance * dwave;
```

και

```
distance = abs(userposi(1) - baseinfo (othercell, 1) + (userposi(2) -
baseinfo(othercell, 2)) * j);
```

```
uwave = uwave + dist(here, there, alpha) * shadow(sigma) * distance *
ant_gain;
```

ΚΕΦΑΛΑΙΟ 3
ΑΠΟΤΕΛΕΣΜΑΤΑ ΠΡΟΣΟΜΟΙΩΣΗΣ

3.1 DCA – θεωρητική και με προσομοίωση επίδοση συναρτήσει του αριθμού των χρηστών.

Στο σχήμα 3.1 φαίνονται οι δύο γραφικές παραστάσεις της πιθανότητας απόρριψης κλήσεων (simulation & theory) συναρτήσει του αριθμού χρηστών όπως σχεδιάστηκαν μέσω της συνάρτησης DCA_theory.m, από τις οποίες είναι εμφανής η ταύτιση μεταξύ των θεωρητικών και πειραματικών τιμών.

Για το σχεδιασμό τους χρησιμοποιήσαμε στο MATLAB τις ακόλουθες εντολές:

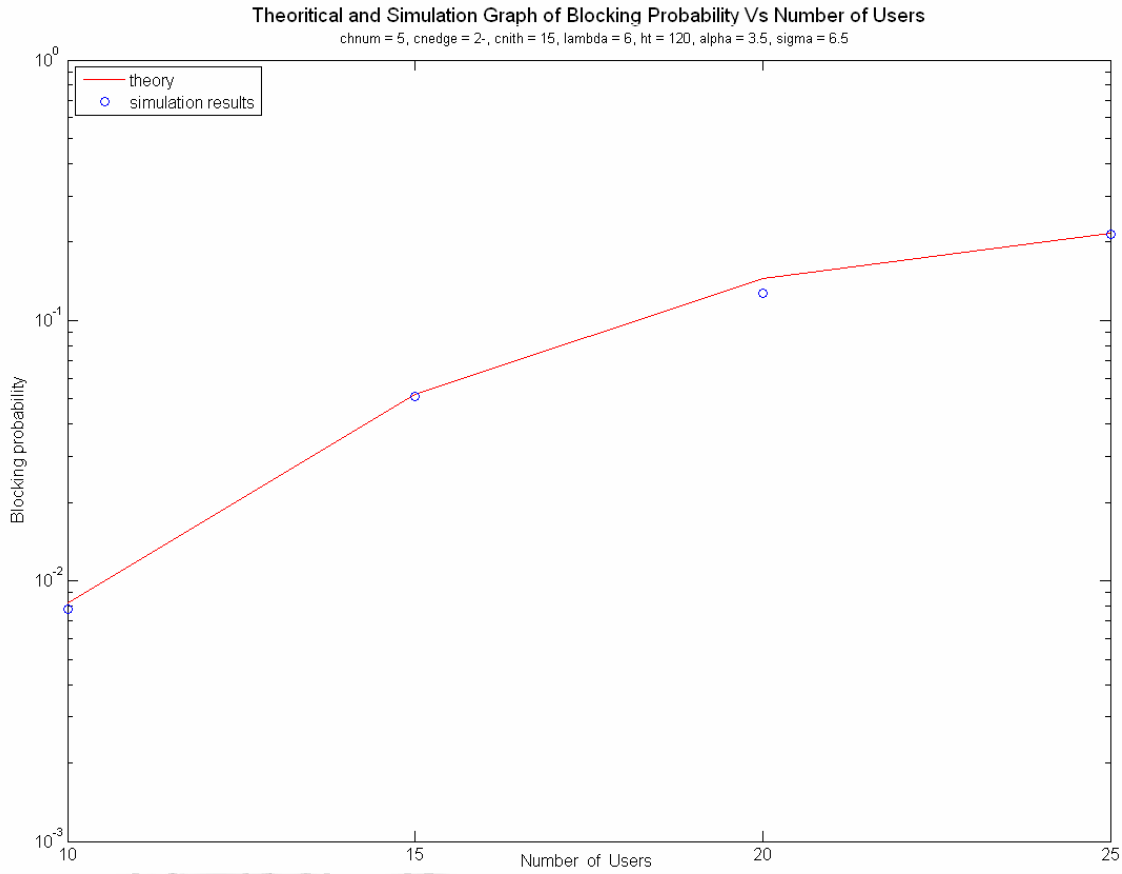
```
semilogy (usernum (1, :), output (5, :), 'r-', usernum (1, :),
output (3, :), 'bo');
legend ('theory', 'simulation results', 2)
xlabel ('Number of users (users/cell)');
ylabel ('Blocking probability');
title ('Graph for chnum=5, cnirth=15(dB) and cnedge=20(dB)');
```

Με την εντολή **semilogy** η οποία χρησιμοποιεί λογαριθμική κλίμακα για τον άξονα y, σχεδιάσαμε τα 2 γραφήματα. Το πρώτο από αυτά αντιστοιχεί στην θεωρητική τιμή της blocking probability και αναπαριστάται από μια κόκκινη καμπύλη (‘r-’), ενώ το δεύτερο αντιστοιχεί στην πειραματική της τιμή και αναπαριστάται με 5 σύμβολα (‘bo’), ένα για το κάθε αριθμό χρηστών..

Ο άξονας x ο οποίος είναι κοινός και για τα 2 γραφήματα, αντιστοιχεί στον αριθμό των χρηστών και λαμβάνει τα δεδομένα από τον πίνακα usernum, ενώ ο

άξονας y ο οποίος αντιστοιχεί στην θεωρητική και στην πειραματική τιμή της blocking probability λαμβάνει τα δεδομένα από την $5^{η}$ και $3^{η}$ γραμμή του πίνακα output αντίστοιχα.

Με την εντολή **legend** εισαγάγαμε τις 2 ετικέτες για τα δύο γραφήματα, με τις **xlabel** και **ylabel** τις ονομασίες των αξόνων x και y και τέλος με την εντολή **title** το τίτλο του γραφήματος.



ΣΧΗΜΑ 3.1
θεωρητική ανάλυση της blocking probability

3.2 DCA – επίδραση χαρακτηριστικών συστήματος

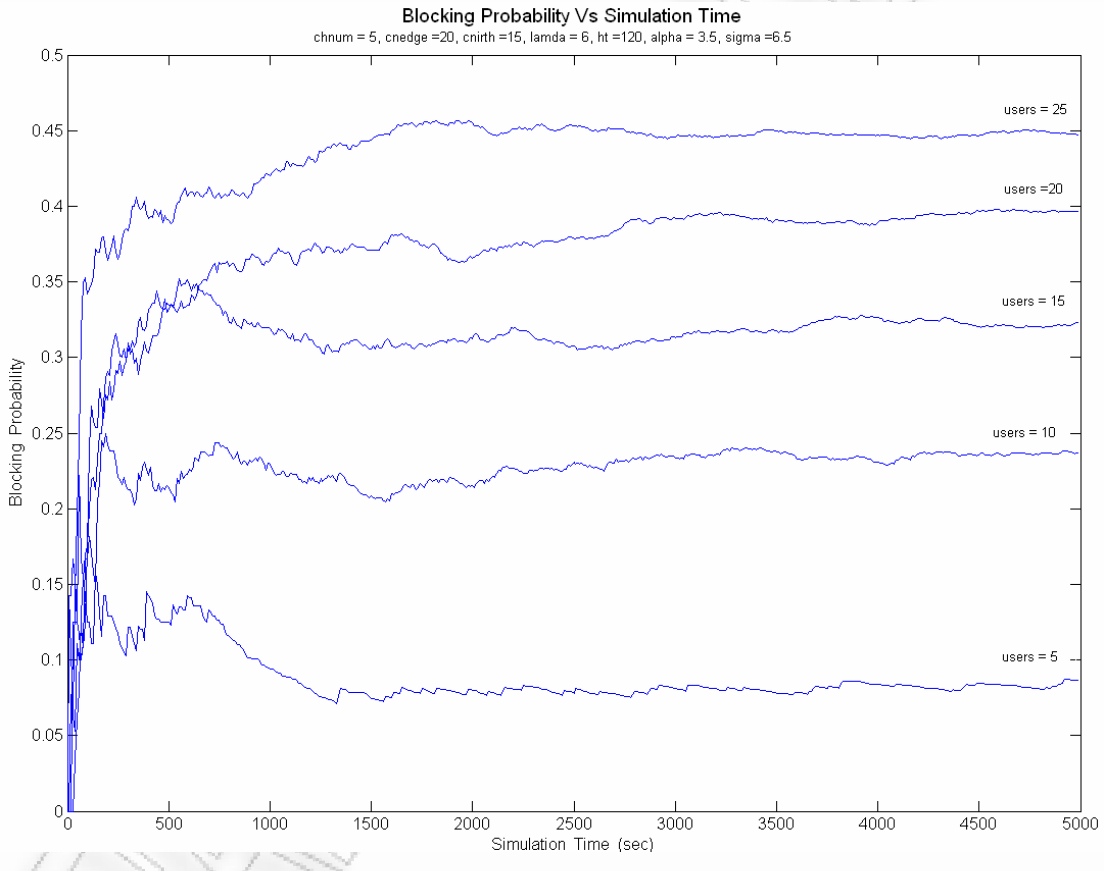
Εδώ θα μελετήσουμε το πώς μεταβάλλεται η επίδοση του συστήματος αν μεταβάλλουμε κάποιο χαρακτηριστικό του, όπως

- ο αριθμός των χρηστών και
- ο αριθμός των διαθέσιμων καναλιών

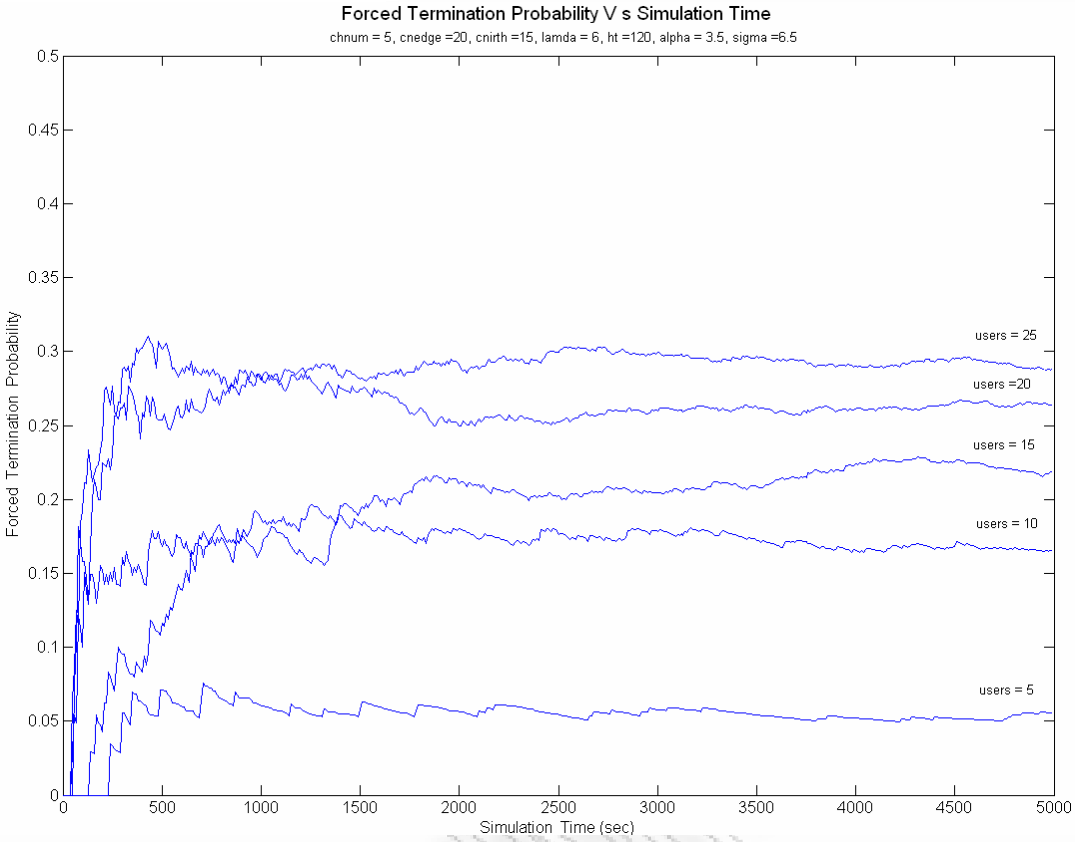
Στα σχήματα 3.2 και 3.3 βλέπουμε τις τιμές της blocking και της forced termination probability συναρτήσει του χρόνου προσομοίωσης timenow για όλες τις τιμές του πίνακα usernum.

Οι γραφικές παραστάσεις σχεδιάστηκαν μέσω των εντολών **plot (check)** και **plot (check2),** αντίστοιχα.

Από τις γραφικές παραστάσεις αυτές διαπιστώνουμε ότι ο συνολικός χρόνος προσομοίωσης που επιλέξαμε (simulation time = 5000 sec) είναι αρκετός ώστε να μας δώσει ικανοποιητικά αποτελέσματα για τι δύο αυτές πιθανότητες .



ΣΧΗΜΑ 3.2
Blocking probability Vs simulation time



ΣΧΗΜΑ 3.3
Forced termination probability Vs Simulation time

Μέσω των γραφικών παραστάσεων των blocking και forced termination probability Vs number of users (σχήματα 3.4 και 3.5 αντίστοιχα) παρατηρούμε την εκθετική αύξηση των παραπάνω πιθανοτήτων σε σχέση με τον αριθμό των χρηστών.

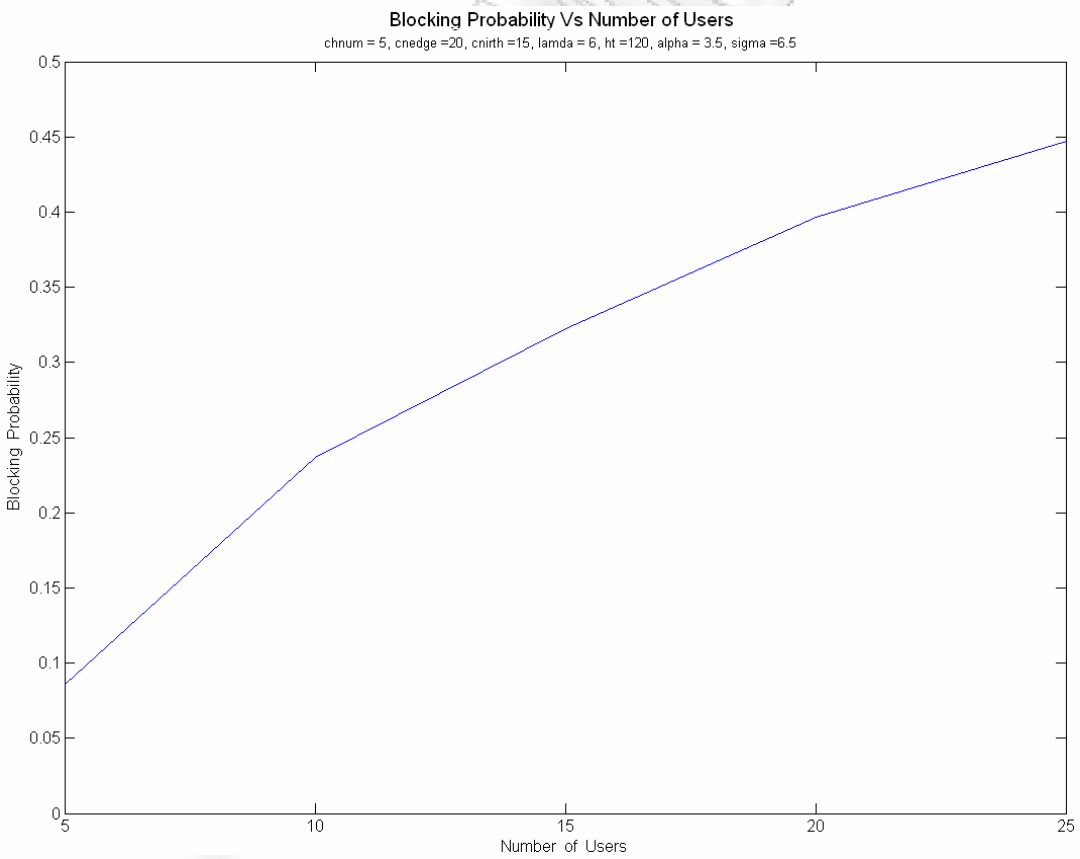
Το γεγονός αυτό οφείλεται σε δύο λόγους. Ο πρώτος είναι ότι όσο αυξάνεται ο αριθμός των χρηστών, τόσο μειώνεται ο αριθμός των ελεύθερων καναλιών που είναι προς διάθεση για τους νέους χρήστες και ο δεύτερος, ότι όσο αυξάνεται ο αριθμός των χρηστών τόσο αυξάνεται και η συνολική παρεμβολή που δέχεται ο κάθε σταθμός βάσης με επακόλουθο την μείωση του λόγου CNIR. Τα παραπάνω έχουν σαν τελικό αποτέλεσμα την αύξηση του αριθμού των μπλοκαρισμένων και απορριπτόμενων κλήσεων και εν τέλει των δύο πιθανοτήτων. Οι παρακάτω γραφικές παραστάσεις σχεδιάστηκαν μέσω των εντολών **plot (usernum, output(3, :)** και **plot (usernum, output(4, :),** αντίστοιχα.

Επίσης από τις γραφικές παραστάσεις του αριθμού των παραγόμενων, των μπλοκαρισμένων και των βίαια τερματισμένων κλήσεων συναρτήσει του αριθμού των χρηστών (σχήμα 3.6) διαπιστώνουμε την γραμμική σχέση αυτών των μεγεθών συναρτήσει του αριθμό των χρηστών.

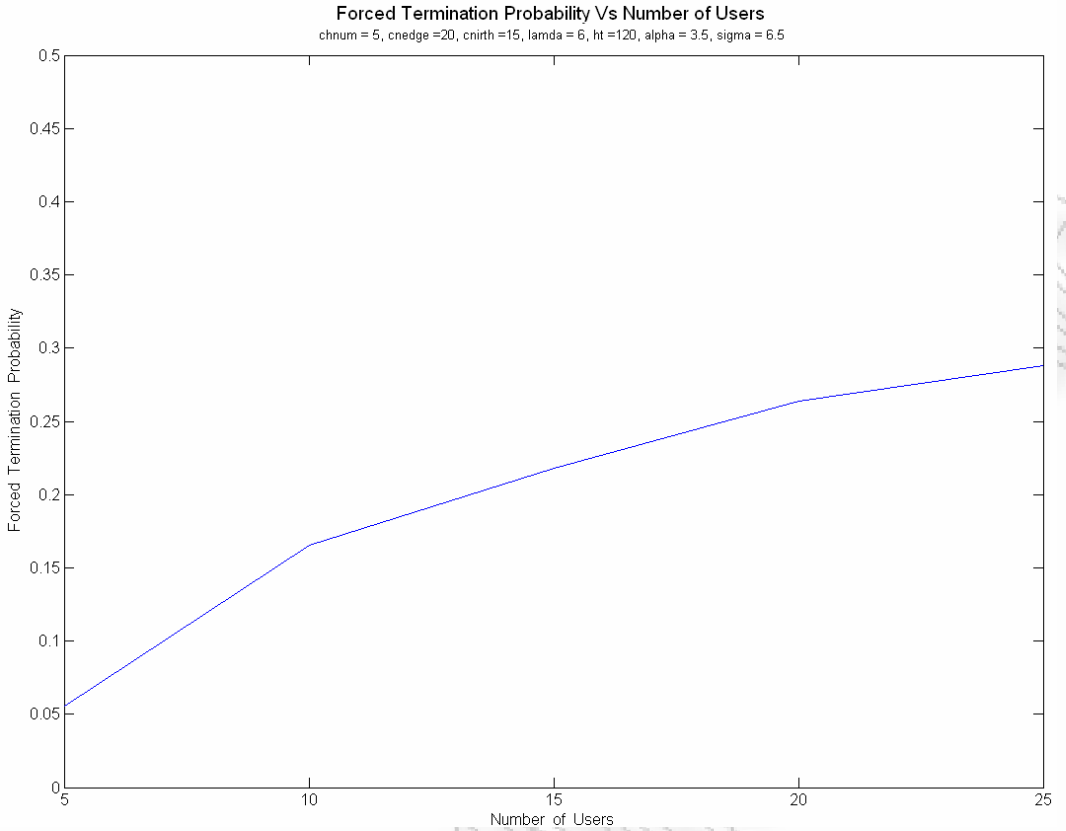
Οι γραφικές παραστάσεις αυτές σχεδιάστηκαν μέσω της ακόλουθης εντολής:

plot (numuser, output (1, :), 'b', numuser, output (2, :), 'r', numuser, output (1, :) - output (2, :), 'g')

UserNumber	**5**	**10**	**15**	**20**	**25**
CallNumber	650	1413	2154	2911	3678
BlockNumber	56	335	695	1155	1645
ForcedNumber	33	178	318	463	585
BlockingProb.	0.0861538	0.237084	0.322656	0.396771	0.447254
ForcedProb.	0.0555556	0.165121	0.217958	0.263667	0.287752

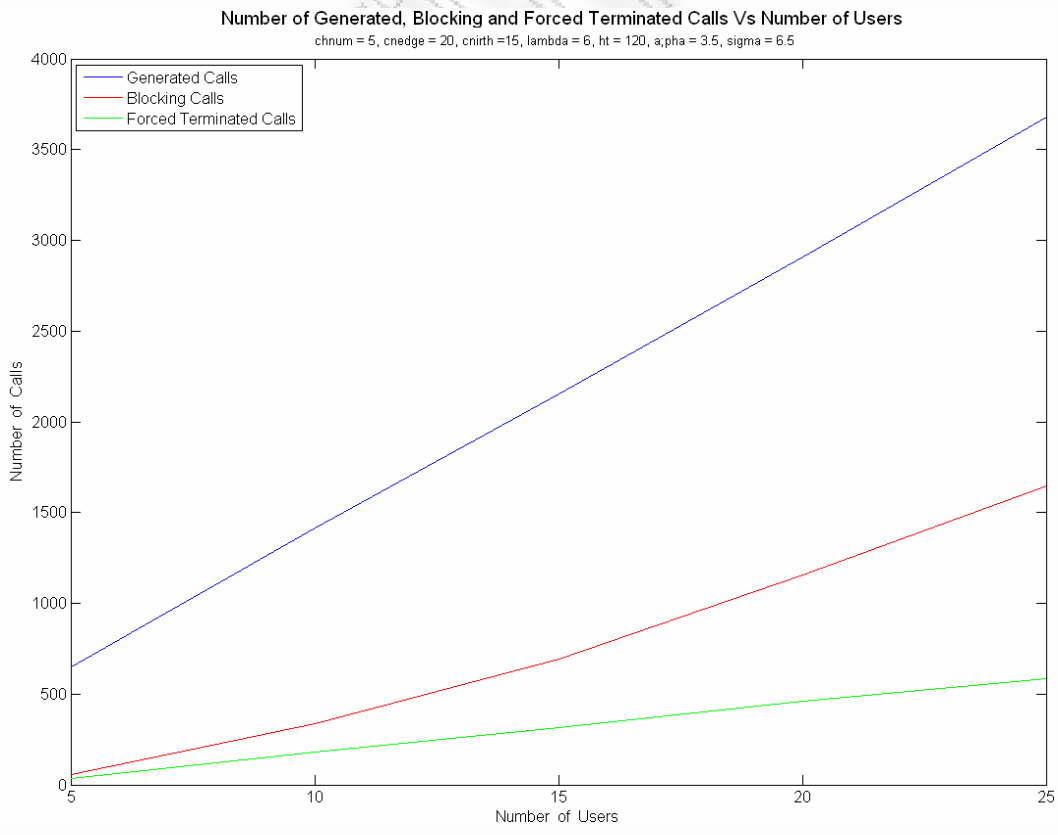


ΣΧΗΜΑ 3.4
Blocking probability Vs number of users



ΣΧΗΜΑ 3.5

Forced termination probability Vs number of users



ΣΧΗΜΑ 3.6

Number of generated, blocking and forced terminated calls VS simulation time

Στις δύο επόμενες γραφικές παραστάσεις (σχήματα 3.7 και 3.8) βλέπουμε την blocking και την forced termination probability vs users number για αριθμό καναλιών n = 3 και n = 5.

Οι γραφικές παραστάσεις αυτές σχεδιάστηκαν μέσω των εντολών **semilogy(usernum, output(3, :))** και **semilogy(usernum, output(4, :)),** αντίστοιχα Τις δύο αυτές εντολές θα τις χρησιμοποιήσουμε και στη συνέχεια για το σχεδιασμό των δύο αυτών πιθανοτήτων και για τις υπόλοιπες περιπτώσεις που ακολουθούν και για το λόγο αυτό δε θα τις αναφέρουμε ξανά.

Παρατηρούμε ότι αυξανομένου του αριθμού των διαθέσιμων διαύλων μειώνεται η τιμή της blocking probability εξαιτίας της μείωσης του αριθμού των μπλοκαρισμένων κλήσεων (blocknumber). Ο λόγος είναι η αύξηση του αριθμού των ελεύθερων διαύλων που είναι προς διάθεση για τις νέες κλήσεις Τα παραπάνω μπορούμε να τα διαπιστώσουμε παρατηρώντας τα παρακάτω αποτελέσματα της προσομοίωσης.

chnum = 5

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber	:	650	1413	2154	2911	3678
BlockNumber	:	56	335	695	1155	1645

chnum = 7

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber	:	643	1379	2096	2844	3600
BlockNumber	:	32	175	460	779	1251

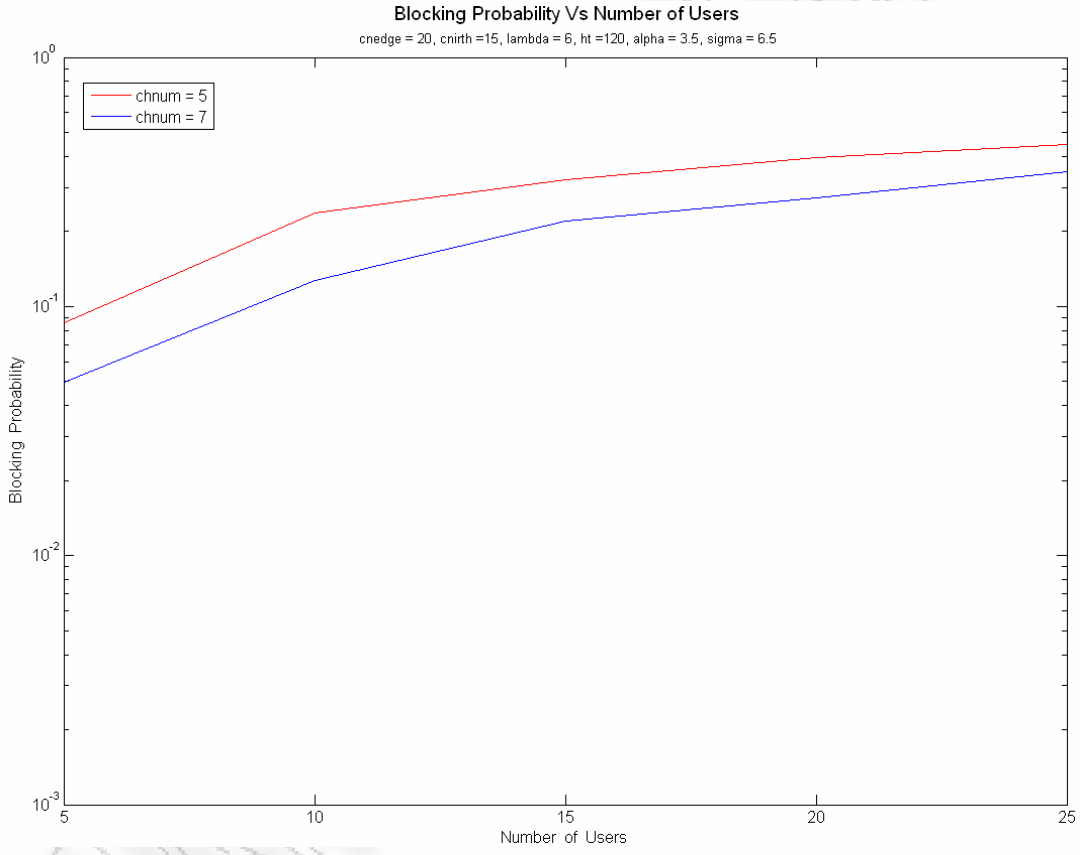
Ομοίως το ίδιο συμβαίνει και για την forced termination probability όπως φαίνεται και στα παρακάτω αποτελέσματα της προσομοίωσης. Εδώ η μείωση του αριθμού των βίαια τερματισμένων κλήσεων (ForcedNumber) οφείλεται στην αύξηση του αριθμού των διαθέσιμων διαύλων για αναδιανομή.

chnum = 5

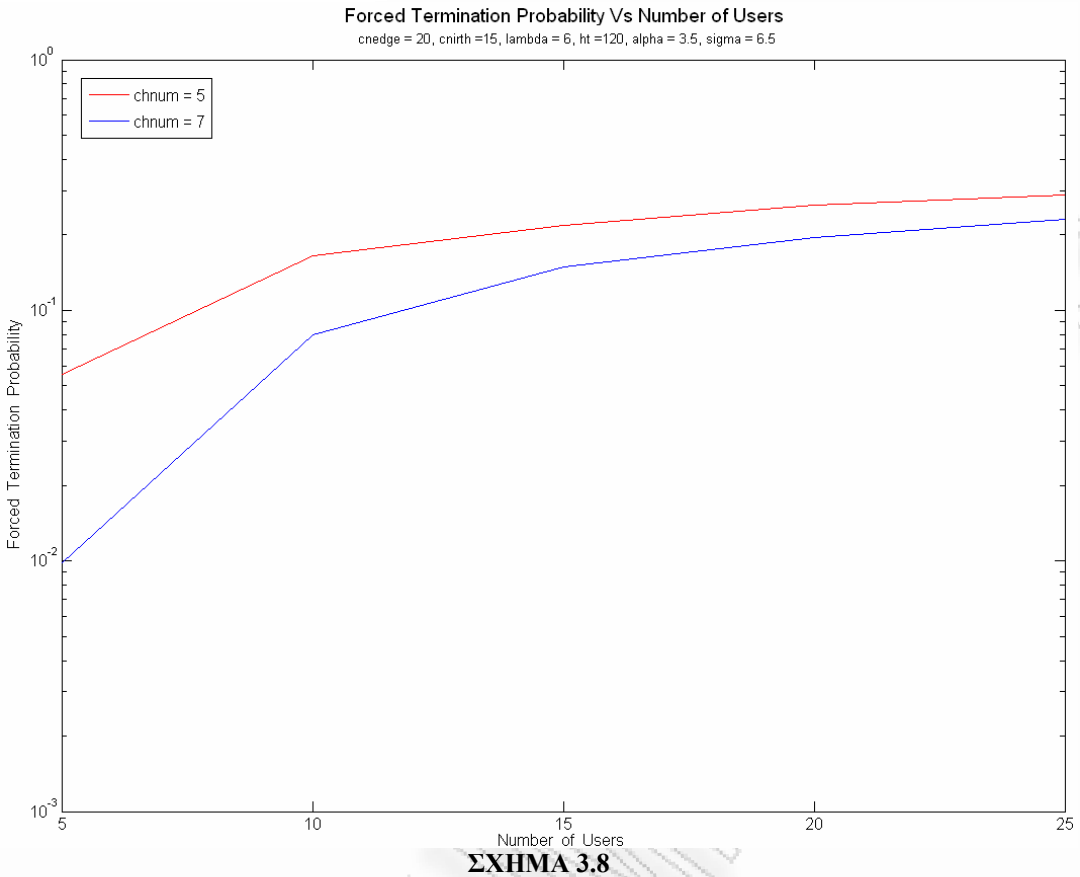
UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber-BlockNumber	:	594	1078	1459	1756	2033
ForcedNumber	:	33	178	318	463	585

chnum = 7

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber-BlockNumber	:	611	1204	1636	2065	2349
ForcedNumber	:	6	96	244	403	544



ΣΧΗΜΑ 3.7
Blocking probability Vs number of channels (chnum 5 & 7)



ΣΧΗΜΑ 3.8

Forced termination probability Vs number of channels (chnum 5 & 7)

Blocking & forced termination probability for chnum = 5 & chnum = 7

UserNumber	5	10	15	20	25
BlockingProb.	0.0861538 0.0497667	0.237084 0.126904	0.322656 0.219466	0.396771 0.27391	0.447254 0.3475
ForcedProb.	0.0555556 0.00981997	0.165121 0.0797342	0.217958 0.149144	0.263667 0.195157	0.287752 0.231588

3.3 DCA – επίδραση της ραδιοζεύξης

Εδώ θα μελετήσουμε πώς επηρεάζεται η επίδοση του συστήματος αν μεταβάλλουμε κάποιο από τα χαρακτηριστικά της ραδιοζεύξης όπως,

- η μέγιστη τιμή της εκπεμπόμενης ισχύος από τον κινητό σταθμό (καθορίζεται από τη τιμή του λόγου C/N στα όρια της κυψέλης)
- η τιμή κατωφλίου του λόγου CNIR.

Στα σχήματα 3.9 και 3.10 βλέπουμε τις γραφικές παραστάσεις της blocking και forced termination probability Vs number of users για τιμές του λόγου CNR στα όρια της κυψέλης cnedge = 15 και cnedge = 20 dB.

Παρατηρούμε ότι αυξανόμενης της τιμής του παραπάνω λόγου μειώνεται η τιμή της blocking probability κάτι που εξηγείται ως εξής:
Η αύξηση στην τιμή του λόγου αυτού, η οποία αντιστοιχεί σε αύξηση τόσο της εκπεμπόμενης ισχύος από τον κινητό σταθμό, όσο και της συνολικής ισχύος των παρεμβολών οδηγεί στην αύξηση της τιμής του λόγου CNIR και κατ' επέκταση της διαφοράς CNIR - $CNIR_{threshold}$. Το αποτέλεσμα είναι η μείωση του αριθμού των μπλοκαρισμένων κλήσεων και επομένως της τιμής της πιθανότητας απόρριψης κλήσεων. Η παραπάνω αύξηση στη τιμή του λόγου CNIR οφείλεται στο ότι η αύξηση της εκπεμπόμενης ισχύος σε σχέση με αυτήν των παρεμβολών είναι μεγαλύτερη, δηλαδή ισχύει ΔC > ΔI.

cnedge = 15 dB

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber	:	659	1422	2158	2918	3677
BlockNumber	:	136	431	816	1244	1749

cnedge = 20 dB

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber	:	650	1413	2154	2911	3678
BlockNumber	:	56	335	695	1155	1645

\

Το αντίθετο συμβαίνει με την forced termination probability της οποίας η τιμή αυξάνεται σε αντίστοιχη αύξηση του λόγου CNR. Πράγματι η αύξηση της τιμής του λόγου αυτού οδηγεί σε αύξηση του αριθμού των βίαια τερματισμένων κλήσεων όπως φαίνεται και στα παρακάτω αποτελέσματα. Ο λόγος είναι ότι ο αριθμός των διαθέσιμων καναλιών για αναδιανομή έχει μειωθεί εξαιτίας του μικρότερρου αριθμού των απορριπτόμενων κλήσεων και επομένως και η πιθανότητα να βρεθεί ένα τέτοιο που να εξασφαλίζει την απαιτούμενη τιμή για το λόγο CNIR

cnedge = 15 dB

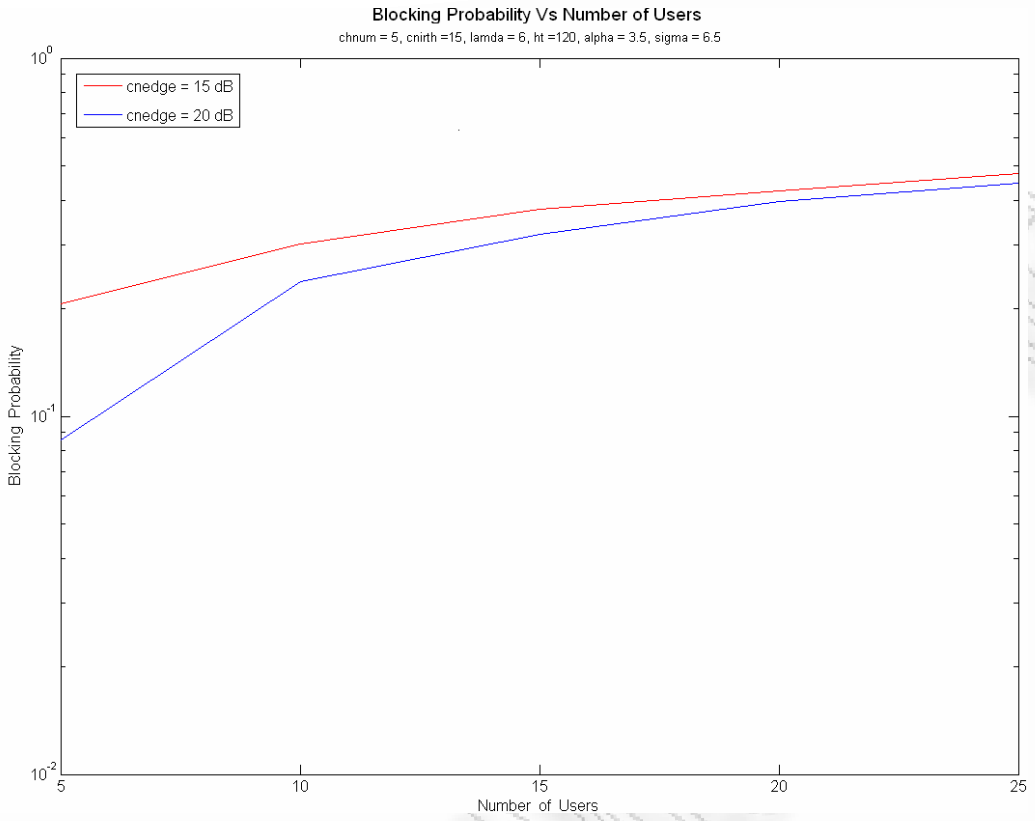
UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber-BlockNumber	:	523	991	1342	1674	1928
ForceNumber	:	17	132	257	382	512

cnedge = 20 dB

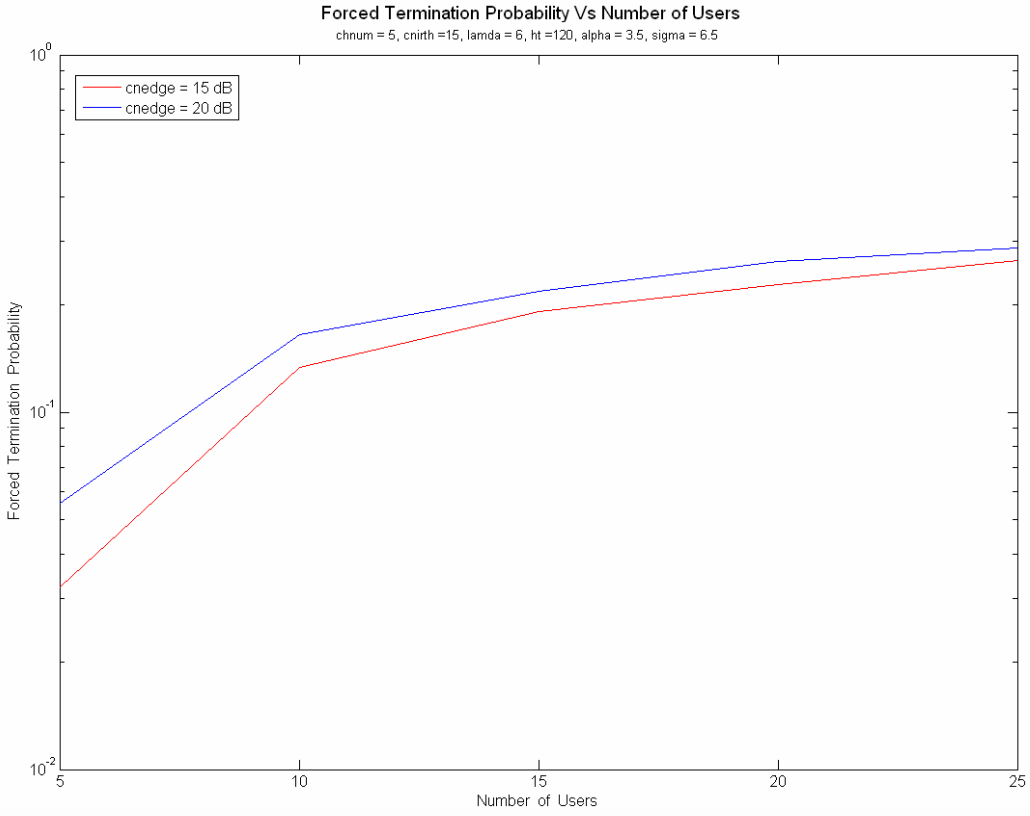
UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber-BlockNumber	:	594	1078	1459	1756	2033
ForceNumber	:	33	178	318	463	585

Blocking & forced termination probability for cnedge = 15 dB & cnedge = 20 dB

UserNumber	**5**	**10**	**15**	**20**	**25**
BlockingProb.	0.206373 0.0861538	0.303094 0.237084	0.378128 0.322656	0.426319 0.396771	0.47566 0.447254
ForcedProb.	0.0325048 0.0555556	0.133199 0.165121	0.191505 0.217958	0.228196 0.263667	0.26556 0.287752



ΣΧΗΜΑ 3.9
Blocking probability Vs number of users (cnedge 15 & 20)



ΣΧΗΜΑ 3.10
Forced termination probability Vs number of users (cnedge 15 & 20)

Στα σχήματα 3.11 και 3.12 βλέπουμε τις γραφικές παραστάσεις της blocking και forced termination probability Vs number of users για τιμές της τιμής κατωφλίου cnirth = 10 και cnirth = 15 dB.

Από αυτές παρατηρούμε ότι αυξανομένης της τιμής κατωφλίου αυξάνεται και η τιμή της πιθανότητας απόρριψης κλήσεων όπως φαίνεται και στα παρακάτω αποτελέσματα της προσομοίωσης.. Ο λόγος είναι ότι η αύξηση του $CNIR_{thresold}$ έχει σαν συνέπεια τη μείωση της διαφοράς CNIR - $CNIR_{threshold}$ και κατ' επέκταση την αύξηση του αριθμού των μπλοκαρισμένων κλήσεων.

cnirth = 10 dB

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber	:	644	1392	2101	2865	3606
BlockNumber	:	24	183	479	900	1275

cnirth = 15 dB

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber	:	650	1413	2154	2911	3678
BlockNumber	:	56	335	695	1155	1645

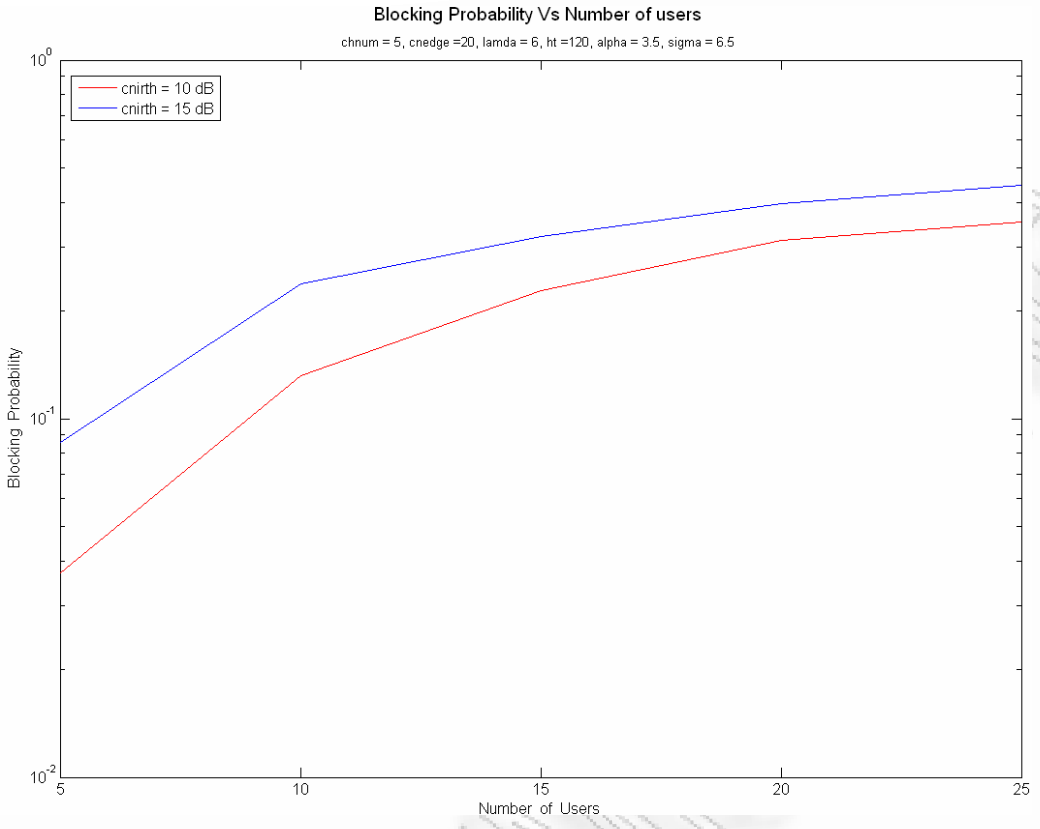
Ομοίως το ίδιο συμβαίνει και με την πιθανότητα βίαιου τερματισμού κλήσεων. Και εδώ η αιτία είναι η μείωση της διαφοράς CNIR - $CNIR_{threshold,}$ η οποία οδηγεί στην αύξηση του αριθμού των βίαια τερματισμένων κλήσεων και τελικώς στη τιμή της παραπάνω πιθανότητας.

cnirth = 10 dB

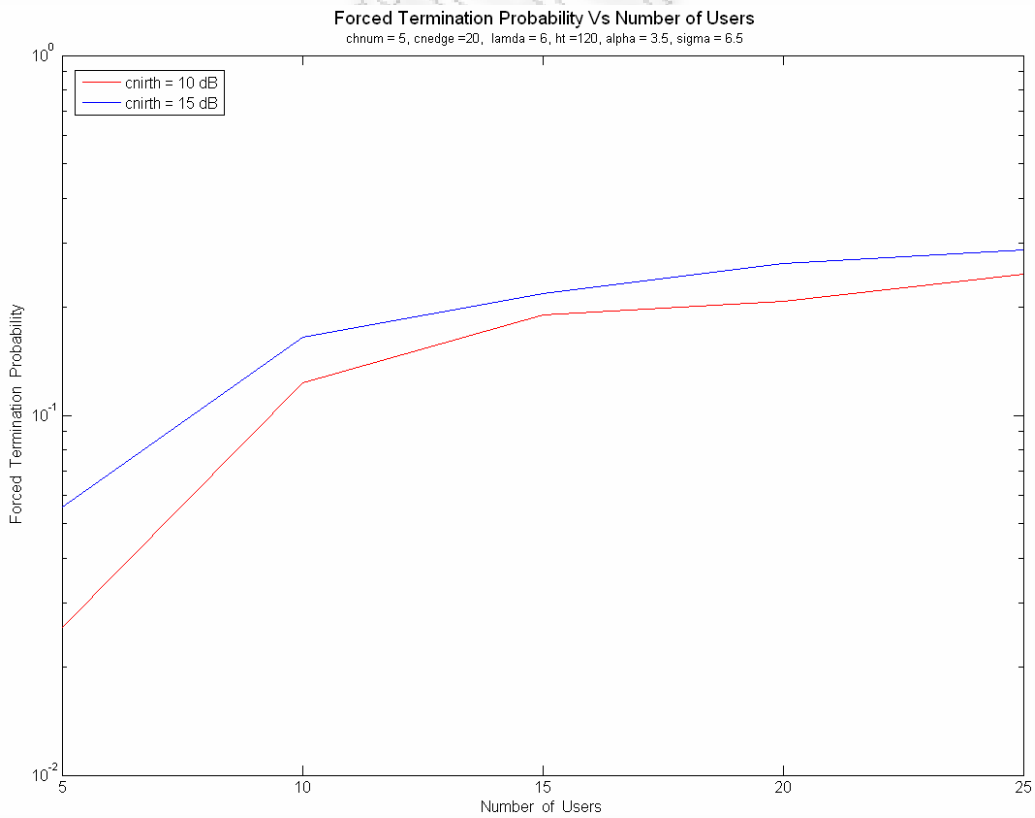
UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber-BlockNumber	:	620	1209	1622	1965	2331
ForcedNumber	:	16	149	308	407	576

cnirth = 15 dB

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber-BlockNumber	:	594	1078	1459	1756	2033
ForceNumber	:	33	178	318	463	585



ΣΧΗΜΑ 3.11
Blocking probability Vs number of users (cnirth 10 & 15)



ΣΧΗΜΑ 3.12

Forced termination probability Vs number of users (cnirth 10 & 15)

Blocking & forced termination probability for cnirth = 10 dB & cinrth = 15 dB

UserNumber	5	10	15	20	25
BlockingProb.	0.0372671 0.0861538	0.131466 0.237084	0.227987 0.322656	0.314136 0.396771	0.353577 0.447254
ForcedProb.	0.0258065 0.0555556	0.123242 0.165121	0.189889 0.217958	0.207125 0.263667	0.247104 0.287752

3.4 DCA – επίδραση της τηλεπικοινωνιακής κίνησης

Εδώ θα μελετήσουμε το πώς επηρεάζεται η επίδοση του συστήματος αν μεταβάλλουμε κάποιο από τα χαρακτηριστικά της τηλεπικοινωνιακής κίνησης όπως

- η μέση τιμή του ρυθμού άφιξης κλήσεων
- μέσος χρόνος διάρκειας μιας κλήσης.

Στα σχήματα 3.13 και 3.14 βλέπουμε τις γραφικές παραστάσεις της blocking και forced termination probability Vs number of users για τιμές του μέσου ρυθμού άφιξης κλήσεων λ = 6 και λ =10 κλήσεις ανά ώρα.

Παρατηρούμε ότι αυξανομένης της τιμής του λ αυξάνεται και η τιμή της blocking probability. Αυτό είναι λογικό εάν αναλογιστούμε ότι όσο μεγαλύτερη είναι η τιμή αυτή, τόσο μεγαλύτερη είναι και η τηλεπικοινωνιακή κίνηση του συστήματος, όπως φαίνεται και από τη σχέση A = ht * λ. Αυτό έχει σαν αποτέλεσμα την αύξηση του αριθμού των κατειλημμένων διαύλων που εξυπηρετούν τις κλήσεις του συστήματος (και επομένως τη μείωση του αριθμού των ελεύθερων), γεγονός που οδηγεί στην αύξηση του αριθμού των μπλοκαρισμένων κλήσεων.

λ = 6 κλήσεις / ώρα

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber	:	650	1413	2154	2911	3678
BlockNumber	:	56	335	695	1155	1645

λ = 10 κλήσεις / ώρα

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber	:	1001	2216	3411	4647	5922
BlockNumber	:	150	789	1447	2306	3267

Ομοίως το ίδιο συμβαίνει και για την forced termination probability της οποίας η τιμή επίσης αυξάνεται σε αντίστοιχη αύξηση του μέσου ρυθμού άφιξης κλήσεων. Η αύξηση του λ αντιστοιχεί και εδώ στη μείωση του αριθμού των διαθέσιμων διαύλων για αναδιανομή και επομένως στην αύξηση του αριθμού των βίαια τερματισμένων κλήσεων.

λ = 6 κλήσεις / ώρα

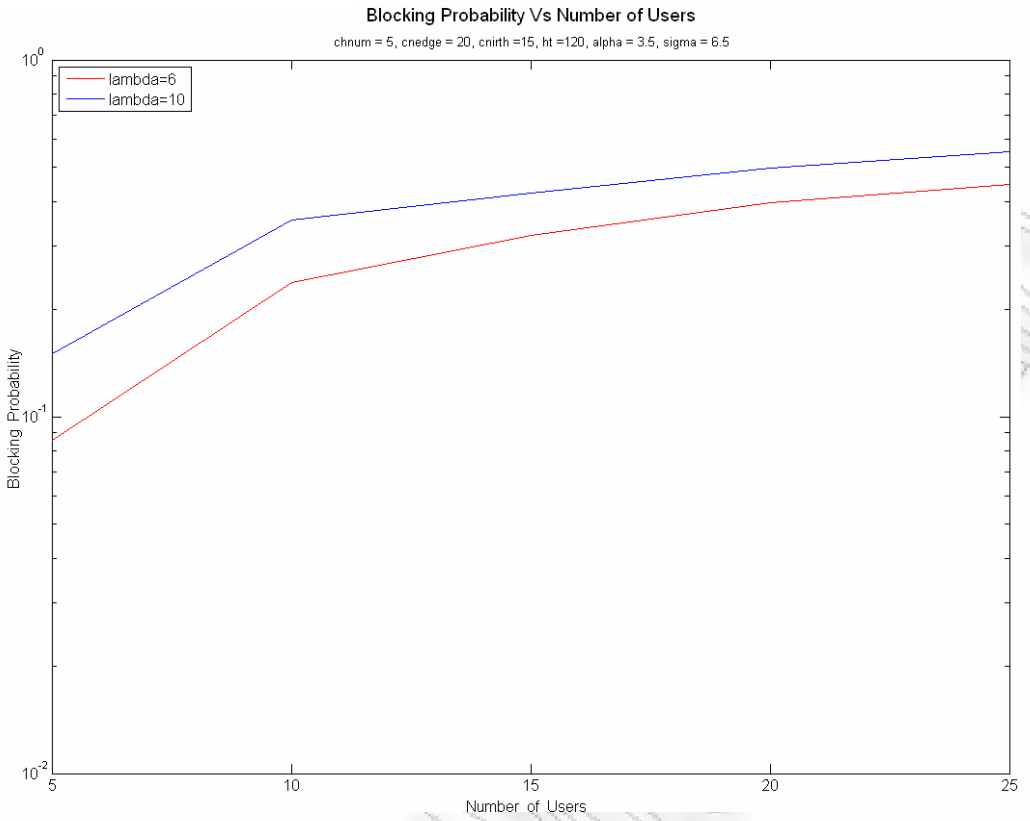
UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber-BlockNumber	:	594	1078	1459	1756	2033
ForceNumber	:	33	178	318	463	585

λ = 10 κλήσεις / ώρα

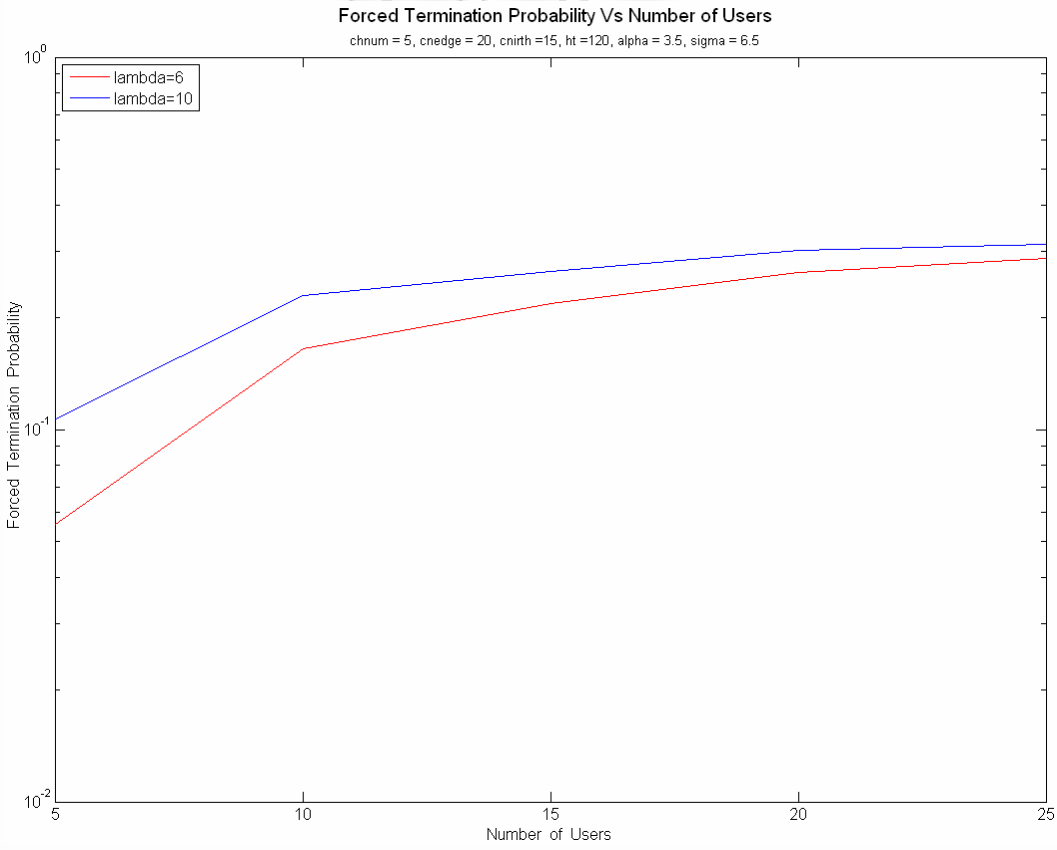
UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber-BlockNumber	:	851	1427	1964	2341	2655
Forcenum	:	91	327	523	708	835

Blocking & forced termination probability for λ= 6 calls/hour & λ = 10 calls/hour

UserNumber	**5**	**10**	**15**	**20**	**25**
BlockingProb.	0.0861538 0.14985	0.237084 0.356047	0.322656 0.424216	0.396771 0.496234	0.447254 0.551672
ForcedProb.	0.0555556 0.106933	0.165121 0.229152	0.217958 0.266293	0.263667 0.302435	0.287752 0.314501



ΣΧΗΜΑ 3.13
Blocking probability Vs number of users (lambda 6 & 10)



ΣΧΗΜΑ 3.14
Forced termination probability Vs number of users (lambda 6 & 10)

Στα σχήματα 3.15 και 3.16 βλέπουμε τις γραφικές παραστάσεις της blocking και forced termination probability Vs number of users για τιμές του μέσου χρόνου διάρκειας κλήσεων ht = 120 και ht =180 sec.

Όπως και στην περίπτωση του μέσου ρυθμού άφιξης κλήσεων, έτσι και εδώ η αύξηση του μέσου χρόνου διάρκειας κλήσεων αντιστοιχεί σε αύξηση των blocking και forced termination probability. Η αύξηση της τιμής του παραπάνω χρόνου αντιστοιχεί και εδώ στην αύξηση της τηλεπικοινωνιακής κίνησης (A = ht * λ). Αυτό έχει σαν συνέπεια την αύξηση του αριθμού των μπλοκαρισμένων κλήσεων λόγω της μείωσης των διαθέσιμων καναλιών από τη μία και την αύξηση του αριθμού των βίαια τερματισμένων κλήσεων λόγω της μείωσης των ελεύθερων διαύλων για αναδιανομή από την άλλη. Τα παραπάνω συμπεράσματα αποτυπώνονται στα ακόλουθα αποτελέσματα της προσομοίωσης.

ht = 60 sec

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber	:	707	1466	2215	2966	3748
BlockNumber	:	44	196	418	781	1164

ht = 120 sec

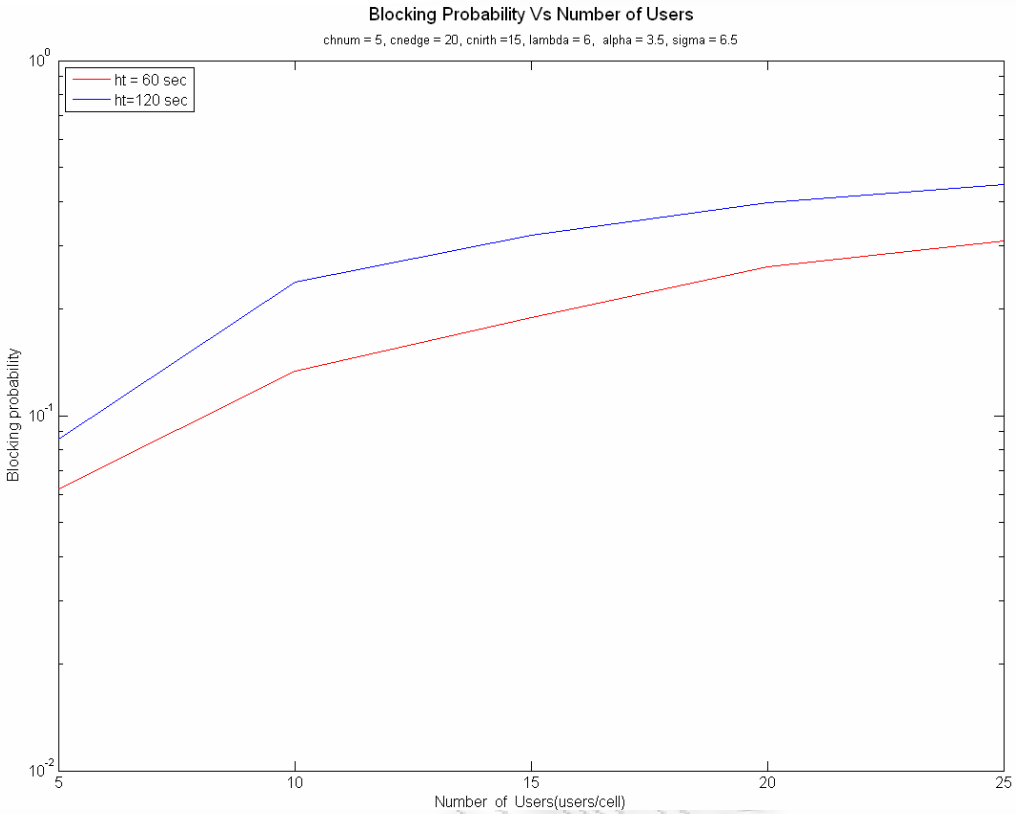
UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber	:	650	1413	2154	2911	3678
BlockNumber	:	56	335	695	1155	1645

ht = 60 sec

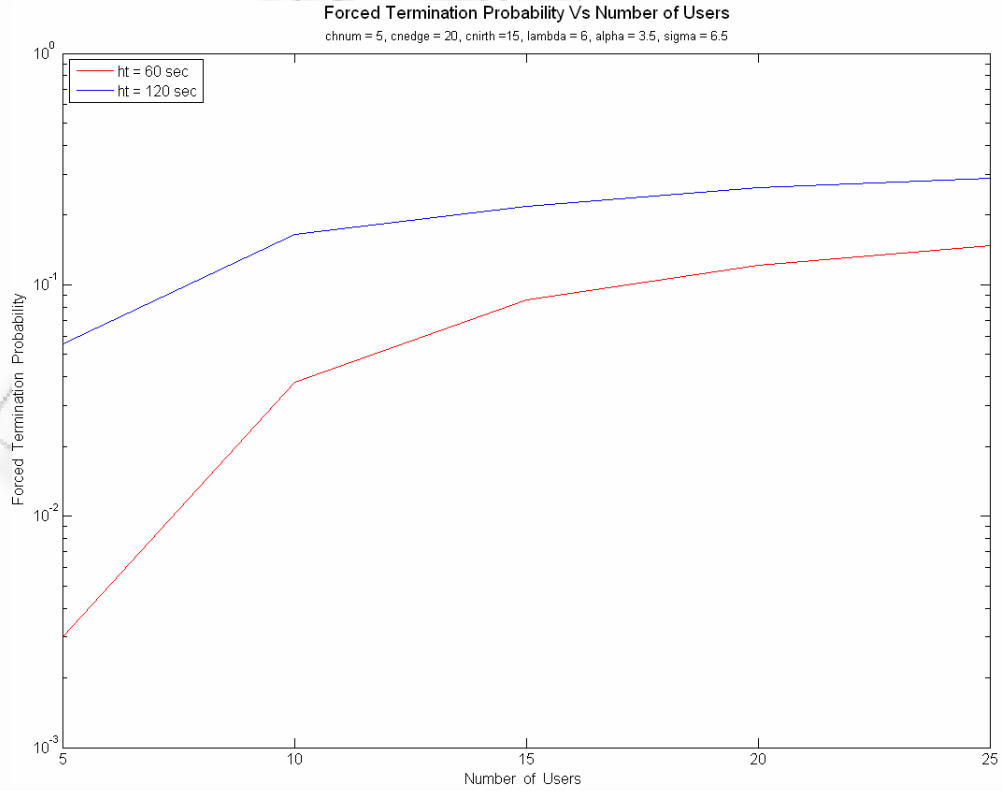
UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber-BlockNumber	:	663	1270	1797	2185	2584
ForcedNumber	:	2	48	155	266	381

ht = 120 sec

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber-BlockNumber	:	594	1078	1459	1756	2033
ForcedNumber	:	33	178	318	463	585



ΣΧΗΜΑ 3.15
Blocking probability Vs number of users (ht 120 & 180)



ΣΧΗΜΑ 3.16
Forced termination probability Vs number of users (ht 120 & 180)

Blocking & forced termination probability for ht= 60 sec & ht = 120 sec

UserNumber	5	10	15	20	25
BlockingProb.	0.0622348 0.0861538	0.133697 0.237084	0.188713 0.322656	0.263318 0.396771	0.310566 0.447254
ForcedProb.	0.00301659 0.0555556	0.0377953 0.165121	0.086259 0.217958	0.121739 0.263667	0.147446 0.287752

3.5 DCA – επίδραση του περιβάλλοντος διάδοσης

Εδώ θα μελετήσουμε πώς επηρεάζεται η απόδοση του συστήματος αν μεταβάλλουμε κάποιο από τα χαρακτηριστικά του περιβάλλοντος διάδοσης όπως

- η τιμή του συντελεστή απωλειών διάδοσης α (path loss factor)
- η τιμή της τυπικής απόκλισης του συντελεστή σκέδασης σ.

Στα σχήματα 3.17 και 3.18 βλέπουμε τις γραφικές παραστάσεις της blocking και forced termination probability Vs number of users για τιμές του συντελεστή απωλειών διάδοσης α = 2 και α = 3. 5.

Μελετώντας τις γραφικές παραστάσεις βλέπουμε ότι αυξανομένου της τιμή του α μειώνεται η τιμή της blocking probability. Η αιτία είναι ότι η αύξηση στην τιμή του συντελεστή αυτού, η οποία χαρακτηρίζει ένα περιβάλλον με μεγαλύτερες απώλειες (η τιμή α = 2 αντιστοιχεί στις απώλειες ελεύθερου χώρου (free space loss)), συνεπάγεται μεγαλύτερη εξασθένιση για την ισχύ τόσο του εκπεμπόμενου σήματος, όσο και των παρεμβολών. Το αποτέλεσμα αυτού είναι η αύξηση της τιμής του λόγου CNIR που φτάνει στο σταθμό βάσης, μιας και η μείωση της συνολικής ισχύος των παρεμβολών σε σχέση με την ισχύ του σήματος είναι μεγαλύτερη ($\Delta C < \Delta I$). Η αύξηση αυτή στην τιμή του παραπάνω λόγου, αντιστοιχεί σε αύξηση της διαφορά CNIR - $CNIR_{thesold}$, σε μείωση του αριθμού των μπλοκαρισμένων κλήσεων και εν τέλει σε μείωση της πιθανότητας απόρριψης κλήσεων.

α = 2

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber	:	660	1445	2198	2976	3756
BlockNumber	:	127	501	1021	1544	2136

α = 3.5

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber	:	650	1413	2154	2911	3678
BlockNumber	:	56	335	695	1155	1645

Ομοίως με την blocking probability, η αύξηση του συντελεστή α οδηγεί και σε μείωση της forced termination probability. Όπως και πριν, έτσι και εδώ η αύξηση του παραπάνω συντελεστή έχει σαν συνέπεια τη μείωση του αριθμού των βίαια τερματισμένων κλήσεων (αύξηση του CNIR) και επομένως της forced termination probability. Αυτό μπορούμε να το διαπιστώνουμε και από παρακάτω αποτελέσματα της προσομοίωσης.

α = 2

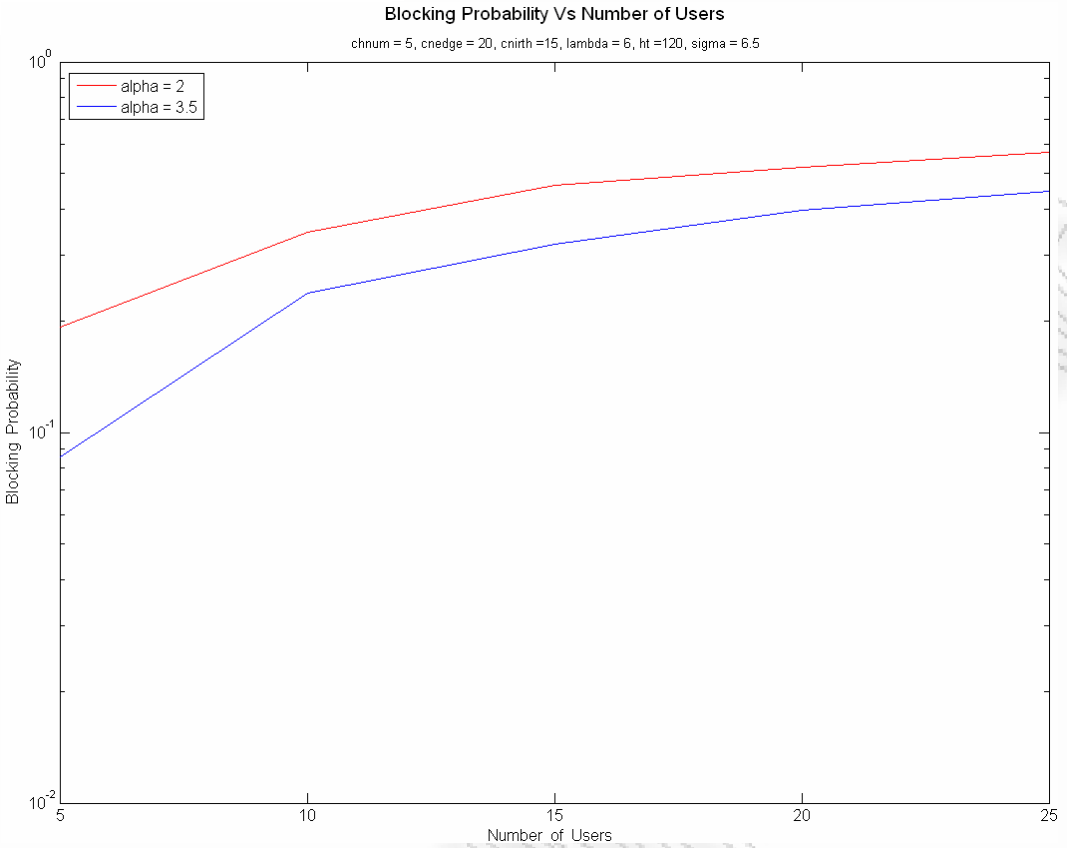
UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber-BlockNumber	:	533	944	1177	1432	1620
Forcenum	:	33	192	341	476	578

α = 3.5

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber-BlockNumber	:	594	1078	1459	1756	2033
ForcedNumber	:	33	178	318	463	585

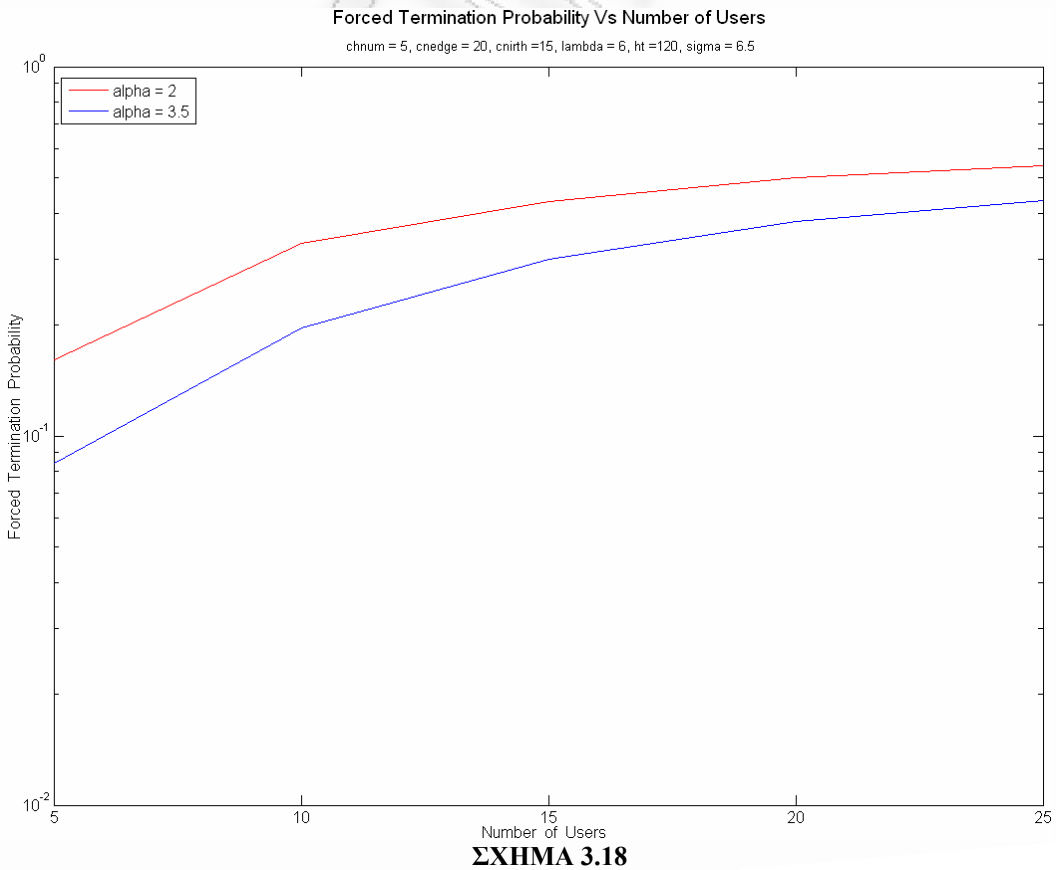
Blocking & forced termination probability for α = 2 & α = 3.5

UserNumber	**5**	**10**	**15**	**20**	**25**
BlockingProb.	0.192424 0.0861538	0.346713 0.237084	0.464513 0.322656	0.518817 0.396771	0.56869 0.447254
ForcedProb.	0.0619137 0.0555556	0.20339 0.165121	0.28972 0.217958	0.332402 0.263667	0.35679 0.287752



ΣΧΗΜΑ 3.17

Blocking probability Vs number of users (alpha 2 & 3.5)



ΣΧΗΜΑ 3.18

Forced termination probability Vs number of users (alpha 2 & 3.5)

Στα σχήματα 3.19 και 3.20 βλέπουμε τις γραφικές παραστάσεις της blocking και forced termination probability Vs number of users για τιμές του συντελεστή τυπικής απόκλισης σ = 6.5 και σ = 10.

Για την blocking probability παρατηρούμε ότι η τιμή της διαφοροποιείται ανάλογα με τον αριθμό των χρηστών του συστήματος. Πιο συγκεκριμένα, για σχετικά μικρό αριθμό χρηστών (5, 10 και 15 χρήστες) η τιμή της αυξάνεται σε αντίστοιχη αύξηση του συντελεστή τυπική απόκλισης σ. Ο λόγος είναι ότι η αύξηση του σ, η οποία αντιστοιχεί σε ένα πιο τραχύ και δύσκολο περιβάλλον διάδοσης (με μεγαλύτερη σκέδαση), συνεπάγεται μεγαλύτερη εξασθένιση του εκπεμπόμενου σήματος και των παρεμβολών. Το αποτέλεσμα είναι η μείωση του λόγου CNIR, εν συνεχεία της διαφοράς CNIR – $CNIR_{thresold}$ και εν τέλει η αύξηση της blocking probability. Ο λόγος που οδηγεί στην παραπάνω μείωση του CNIR είναι ότι για μικρό αριθμό χρηστών, ο οποίος συνεπάγεται και μικρότερη τιμή παρεμβολής η εξασθένιση της εκπεμπόμενης ισχύος λόγω της σκέδασης είναι μεγαλύτερη από αυτήν που υφίσταται η συνολική παρεμβολή. (ΔC > ΔI). Αντιθέτως για μεγαλύτερο αριθμό χρηστών (20 και 25 χρήστες), όπου η εξασθένιση της συνολικής παρεμβολής είναι μεγαλύτερη από την αντίστοιχη της εκπεμπόμενη ισχύος, η τιμή του CNIR αυξάνεται, γεγονός που οδηγεί σε μείωση στην τιμή της παραπάνω πιθανότητας.

σ = 6.5

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber	:	650	1413	2154	2911	3678
BlockNumber	:	56	335	695	1155	1645

σ = 9

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber	:	653	1417	2151	2909	3675
BlockNumber	:	94	357	714	1150	1635

Για την forced termination probability συμβαίνει ακριβώς το αντίθετο. Πιο συγκεκριμένα, για μικρότερο αριθμό χρηστών (5, 10 και 15 χρήστες) η τιμή της μειώνεται σε αντίστοιχη αύξηση του συντελεστή τυπική απόκλισης σ, ενώ για μεγαλύτερο αριθμό (20 και 25 χρήστες) η τιμή της αυξάνεται.

Ο λόγος για την πρώτη περίπτωση είναι ότι ο αριθμός των διαθέσιμων καναλιών για αναδιανομή έχει αυξηθεί εξαιτίας του μεγάλου αριθμού των απορριπτόμενων κλήσεων, οπότε και η πιθανότητα να βρεθεί ένας που να εξασφαλίζει την απαιτούμενη τιμή του λόγου CNIR, Αντίθετα για την δεύτερη περίπτωση είναι ότι ο αριθμός των διαθέσιμων καναλιών για αναδιανομή έχει μειωθεί

$\sigma = 6.5$

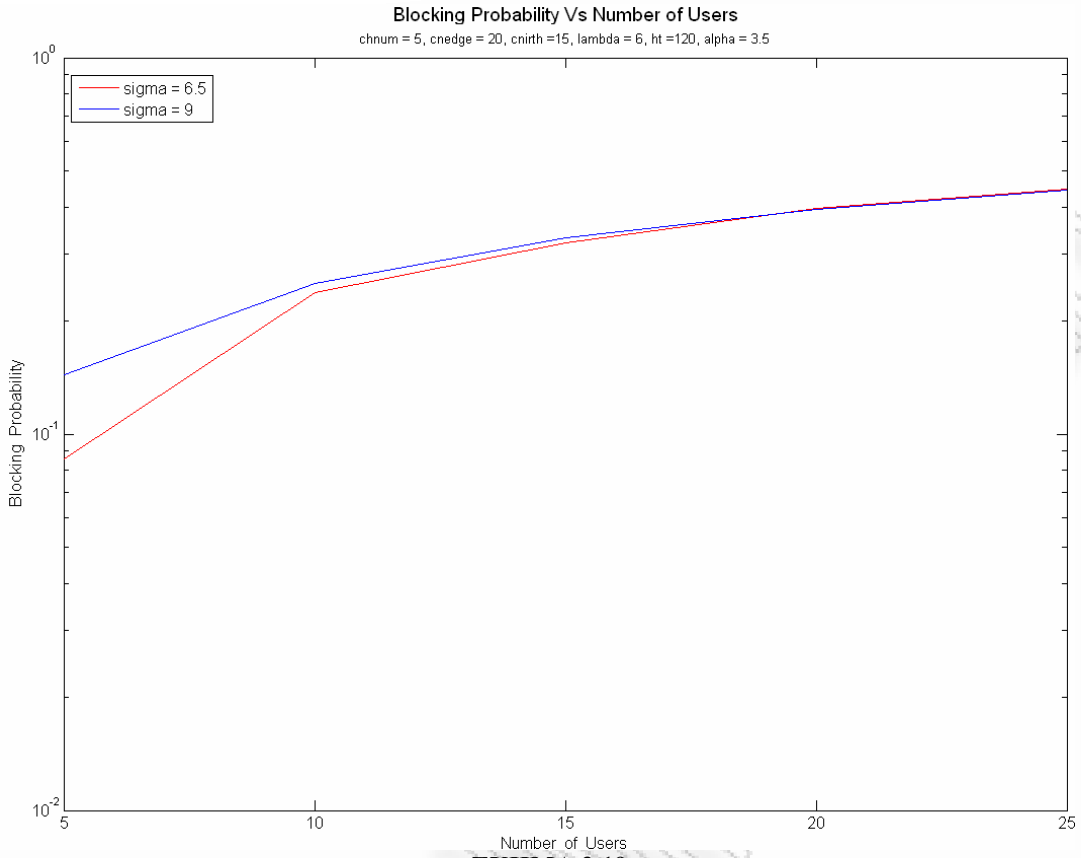
UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber-BlockNumber	:	594	1078	1459	1756	2033
ForcedNumber	:	33	178	318	463	585

$\sigma = 9$

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber-BlockNumber	:	559	1060	1437	1759	2040
Forcenum	:	23	162	297	467	591

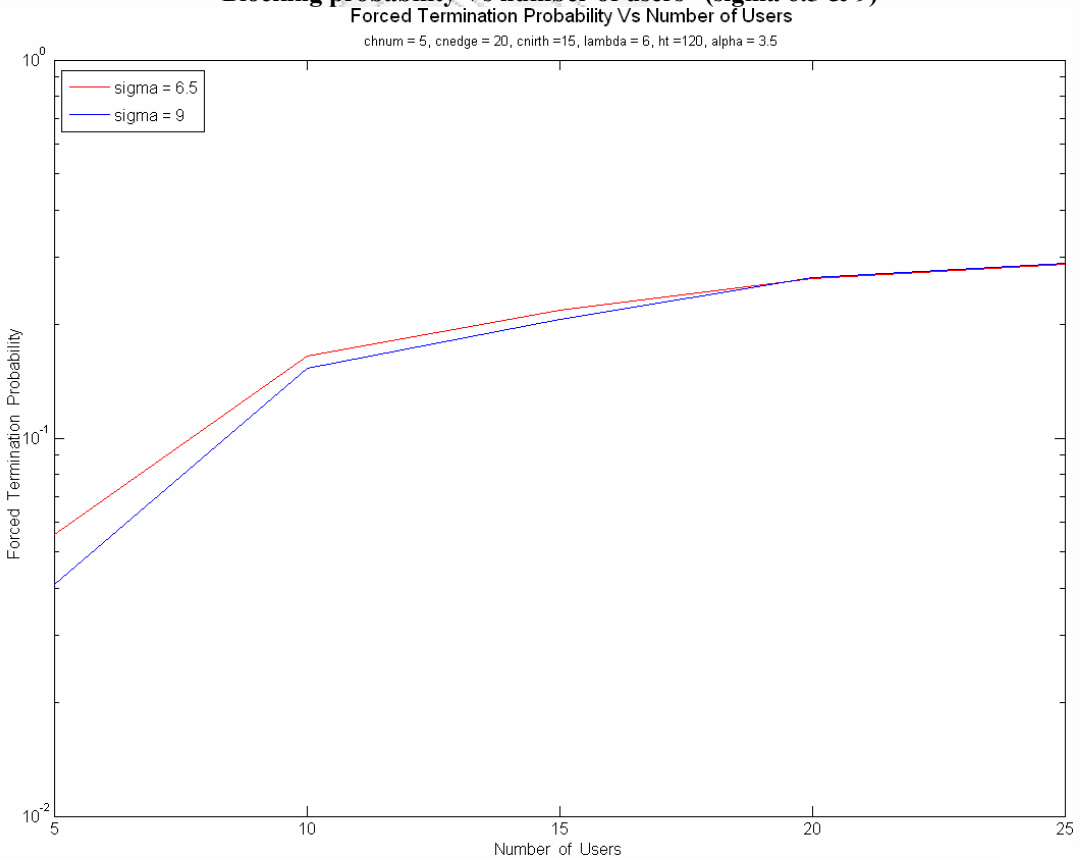
Blocking & forced termination probability for $\sigma = 6.5$ & $\sigma = 9$

UserNumber	**5**	**10**	**15**	**20**	**25**
BlockingProb.	0.0861538 0.143951	0.237084 0.251941	0.322656 0.331939	0.396771 0.395325	0.447254 0.444898
ForcedProb.	0.0555556 0.0411449	0.165121 0.15283	0.217958 0.206681	0.263667 0.265492	0.287752 0.289706



ΣΧΗΜΑ 3.19

Blocking probability Vs number of users (sigma 6.5 & 9)



ΣΧΗΜΑ 3.20

Forced termination probability vs number of users (sigma = 6.5 & 9)

3.6 Επίδραση του Beamforming στο λόγο CNIR

Με τη βοήθεια της συνάρτησης BeamformingCNIR.m την οποία αναλύσαμε πιο πάνω υπολογίσαμε το μέσο όρο της διαφοράς για όλη τη διάρκεια της προσομοίωσης μεταξύ των δύο τιμών του λόγου CNIR με και χωρίς την τεχνική beamforming.

Το διάγραμμα που προκύπτει φαίνεται στο σχήμα 3.21

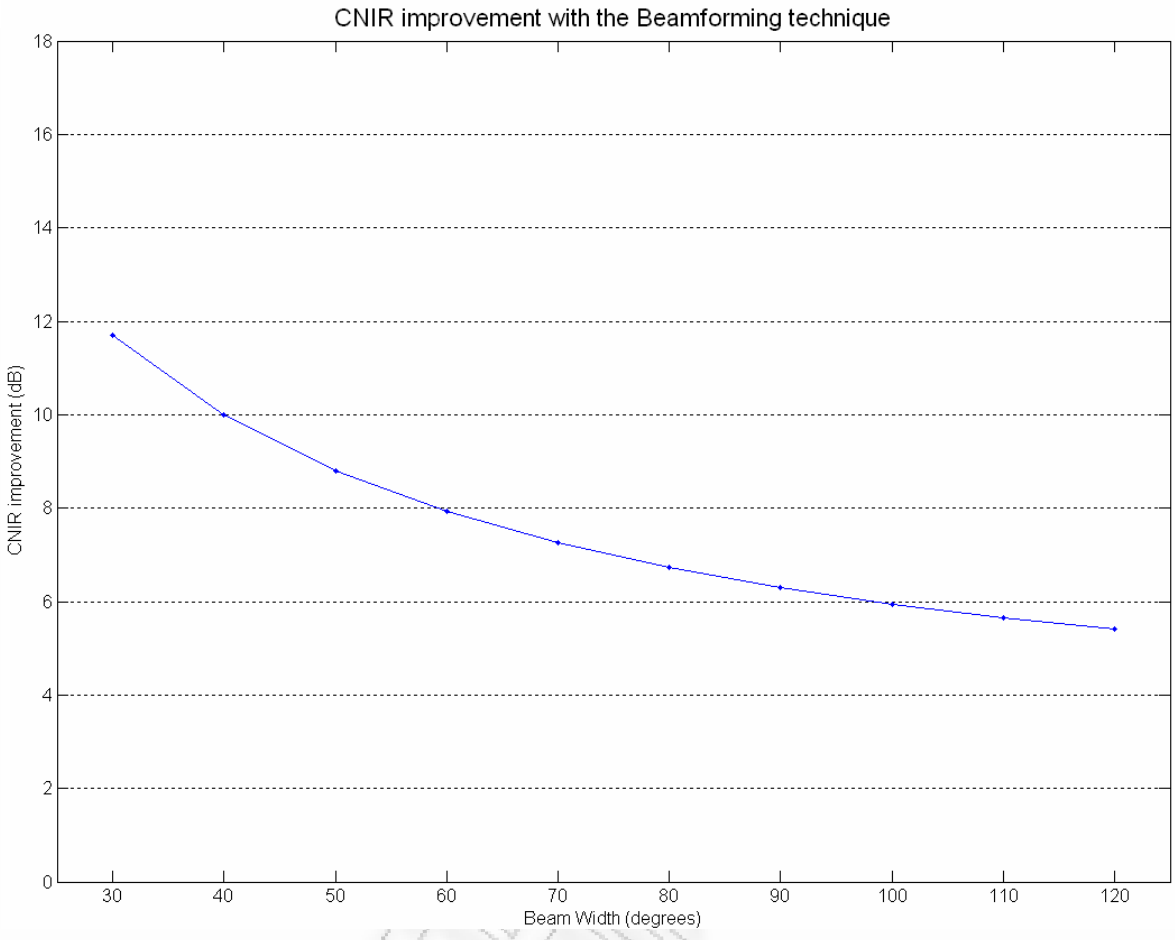
```
plot(30:10:120, output, '.-')
axis([25 125 0 18])
xlabel('Beam Width (degrees)');
ylabel('CNIR improvement (dB)');
grid on;
```

Όπου **output** ο πίνακας ο οποίος περιέχει την διαφορά ΔCNIR για κάθε τιμή του ανοίγματος δέσμης του κύριου λοβού της κεραίας

```
output(1, i) = (cnirdb_BF_all - cnirdb_all) / (timeend / timestep);
```

Παρατηρώντας το βλέπουμε πόσο μεγάλη είναι η βελτίωση στη τιμή του λόγου CNIR, ιδιαίτερα για μικρές τιμές του πλάτους της δέσμης του κύριου λοβού της κεραίας.

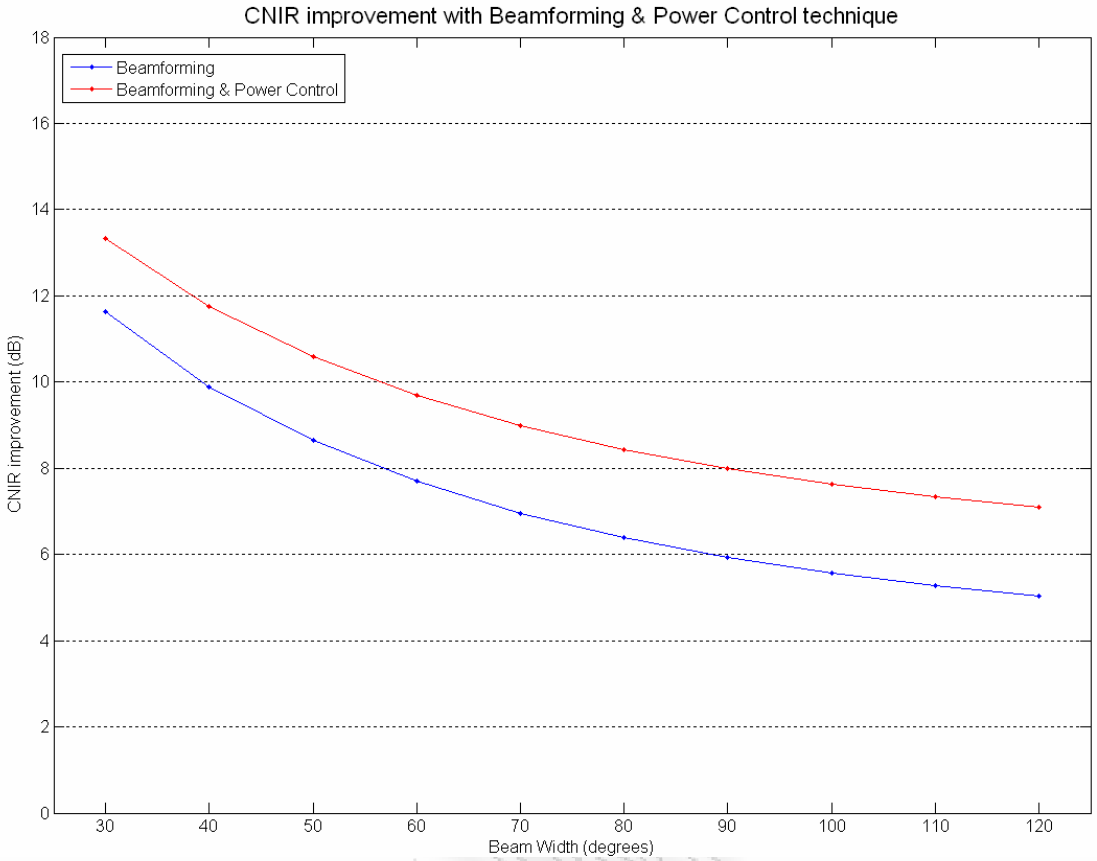
Πράγματι όσο μικρότερη είναι η τιμή του beamwidth, τόσο πιο κατευθυντική είναι η κεραία. Αυτό οδηγεί σε μεγαλύτερη μείωση των παρεμβολών και κατά συνέπεια σε αύξηση στην τιμή του λόγου CNIR.



ΣΧΗΜΑ 3.21
CNIR improvement with beamforming technique

3.7 Επίδραση του Beamforming & Power control στο λόγο CNIR

Από τη γραφική παράσταση του σχήματος 3.22 παρατηρούμε ότι εφαρμόζοντας επιπλέον και τον έλεγχο ισχύος (συνάρτηση PowerControlCNIR.m) η τιμή του λόγου CNIR βελτιώνεται ακόμη πιο πολύ. Το γεγονός αυτό οφείλεται στη μείωση της τιμής της εκπεμπόμενης από τους κινητούς σταθμούς ισχύος και κατά συνέπεια στην μείωση της παρεμβολής στην κεραία των σταθμών βάσης



ΣΧΗΜΑ 3.22
CNIR improvement with beamforming and power control technique

3.8 DCA - επίδραση Beamforming

Από τις γραφικές παραστάσεις των blocking και forced termination probability Vs number of users που σχεδιάσαμε (σχήματα 3.23 και 3.24) εφαρμόζοντας την τεχνική DCA και στη συνέχεια την τεχνική DCA μαζί με την Beamforming παρατηρούμε τη σημαντική βελτίωση στις τιμές αυτών των δύο πιθανοτήτων.

semilogy(usernum, output(4, :), 'b', usernum, output_BF(4, :), 'r')

Η αιτία είναι η αύξηση της τιμής του λόγου CNIR εξαιτίας της χρήσης κατευθυντικών κεραιών στους σταθμούς βάσης των κυψελών, γεγονός που οδηγεί στη μείωση του αριθμού τόσο των μπλοκαρισμένων κλήσεων όσο και των βίαια τερματισμένων κλήσεων

DCA + Beamforming

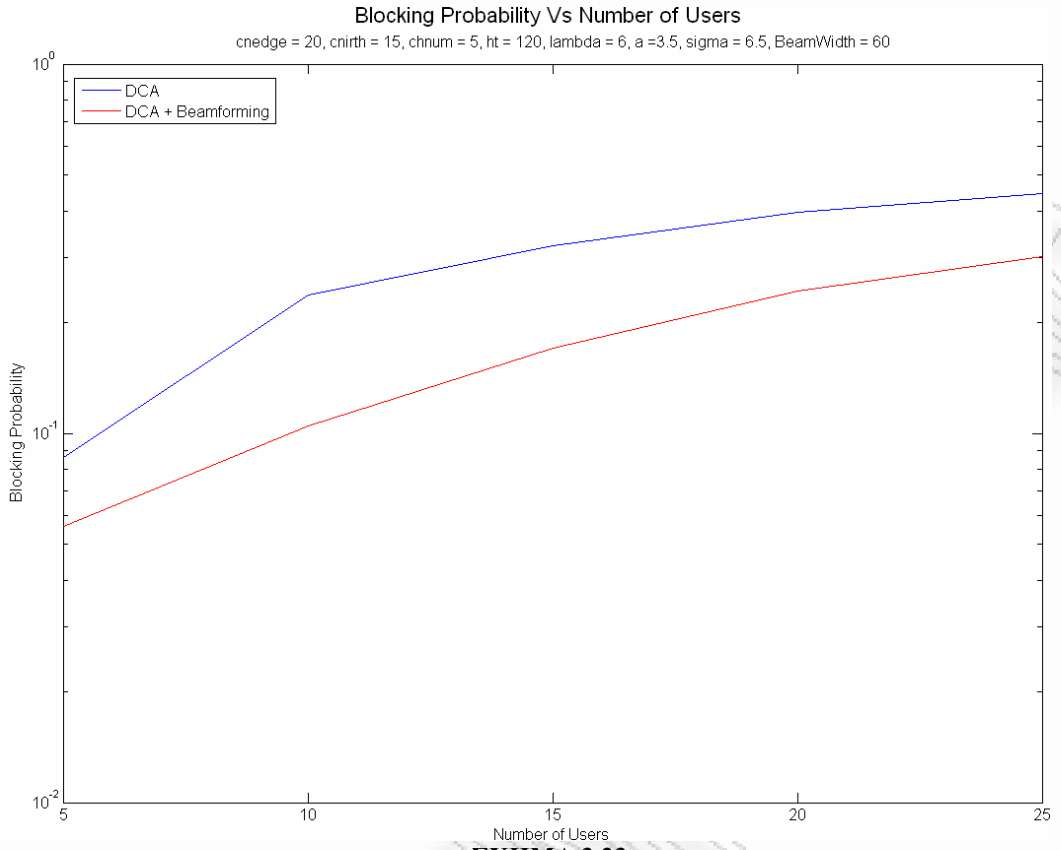
UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber	:	642	1368	2066	2797	3526
BlockNumber	:	36	143	351	679	1060
CallNumber-BlockNumber	:	606	1225	1715	2118	2496
ForcedNumber	:	2	51	145	267	313

DCA

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber	:	650	1413	2154	2911	3678
BlockNumber	:	56	335	695	1155	1645
CallNumber-BlockNumber	:	594	1078	1459	1756	2033
ForcedNumber	:	33	178	318	463	585

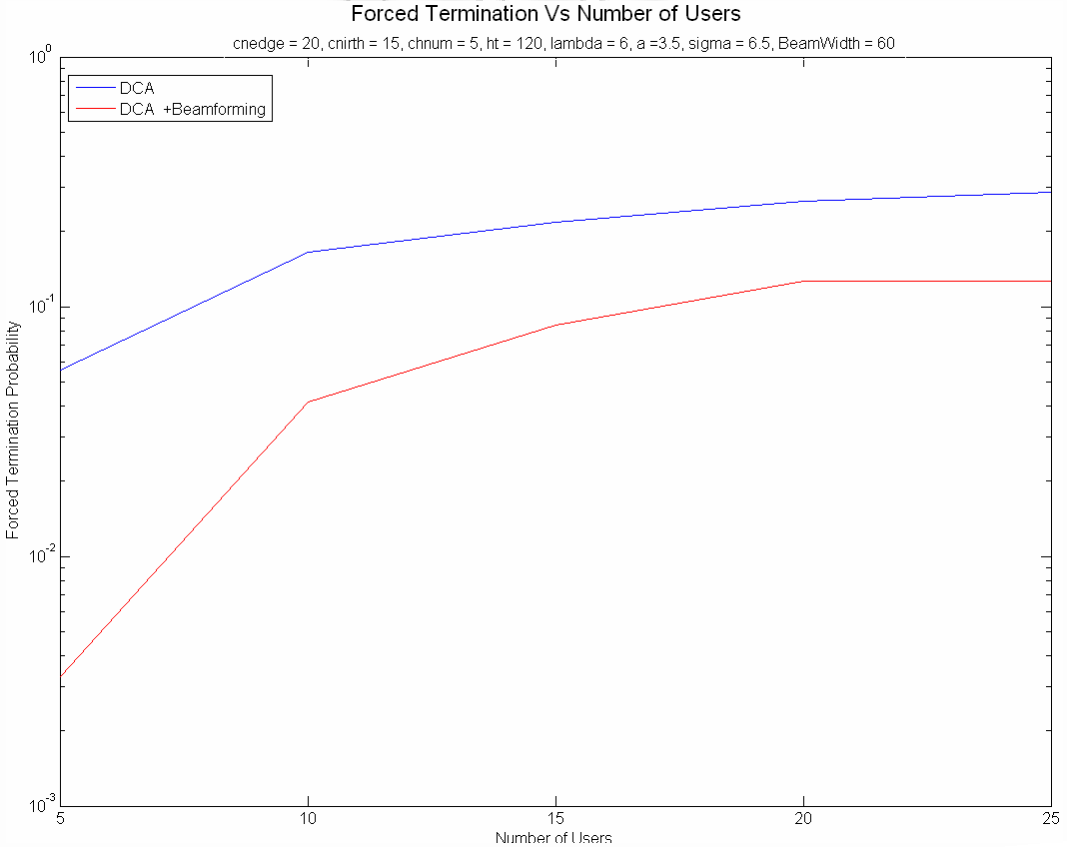
Blocking & forced termination probability for DCA + Beamforming & DCA

UserNumber	**5**	**10**	**15**	**20**	**25**
BlockingProb.	0.0560748 0.0861538	0.104532 0.237084	0.169894 0.322656	0.24276 0.396771	0.300624 0.447254
ForcedProb.	0.00330033 0.0555556	0.0416327 0.165121	0.084548 0.217958	0.126062 0.263667	0.126926 0.287752



ΣΧΗΜΑ 3.23

Blocking probability Vs number of users (DCA & DCA + beamforming)



ΣΧΗΜΑ 3.24

Blocking probability Vs number of users (DCA & DCA + beamforming)

Συγκρίνοντας επιπλέον τις τιμές αυτών των δύο πιθανοτήτων για τιμές του ανοίγματος δέσμης του κύριου λοβού της κεραίας 60 και 30 μοίρες (σχήμα 3.25 και 3.26) παρατηρούμε ότι μειώνοντας την παράμετρο αυτή, εξασφαλίζουμε ακόμη μεγαλύτερη μείωση στις τιμές τους. Ο λόγος είναι ότι όσο μικρότερο είναι το πλάτος δέσμης του κύριου λοβού της κεραίας τόσο μεγαλύτερη είναι και η κατευθυντικότητα της κεραίας, γεγονός που οδηγεί σε μεγαλύτερη καταστολή των παρεμβολών και επομένως σε αύξηση της τιμής του λόγου CNIR.

w_HBS = 30

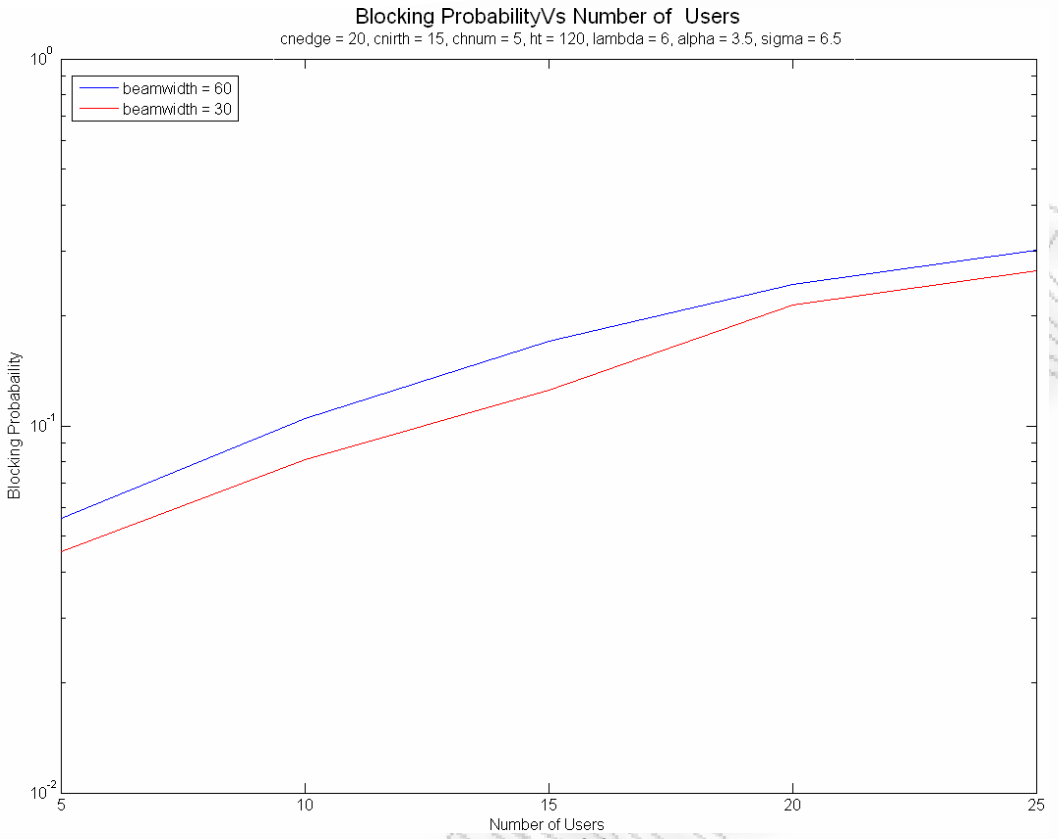
UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber	:	642	1368	2066	2797	3526
BlockNumber	:	36	143	351	679	1060
CallNumber-BlockNumber	:	606	1225	1715	2118	2496
ForcedNumber	:	2	51	145	267	313

w_HBS = 60

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber	:	650	1413	2154	2911	3678
BlockNumber	:	56	335	695	1155	1645
CallNumber-BlockNumber	:	594	1078	1459	1756	2033
ForcedNumber	:	33	178	318	463	585

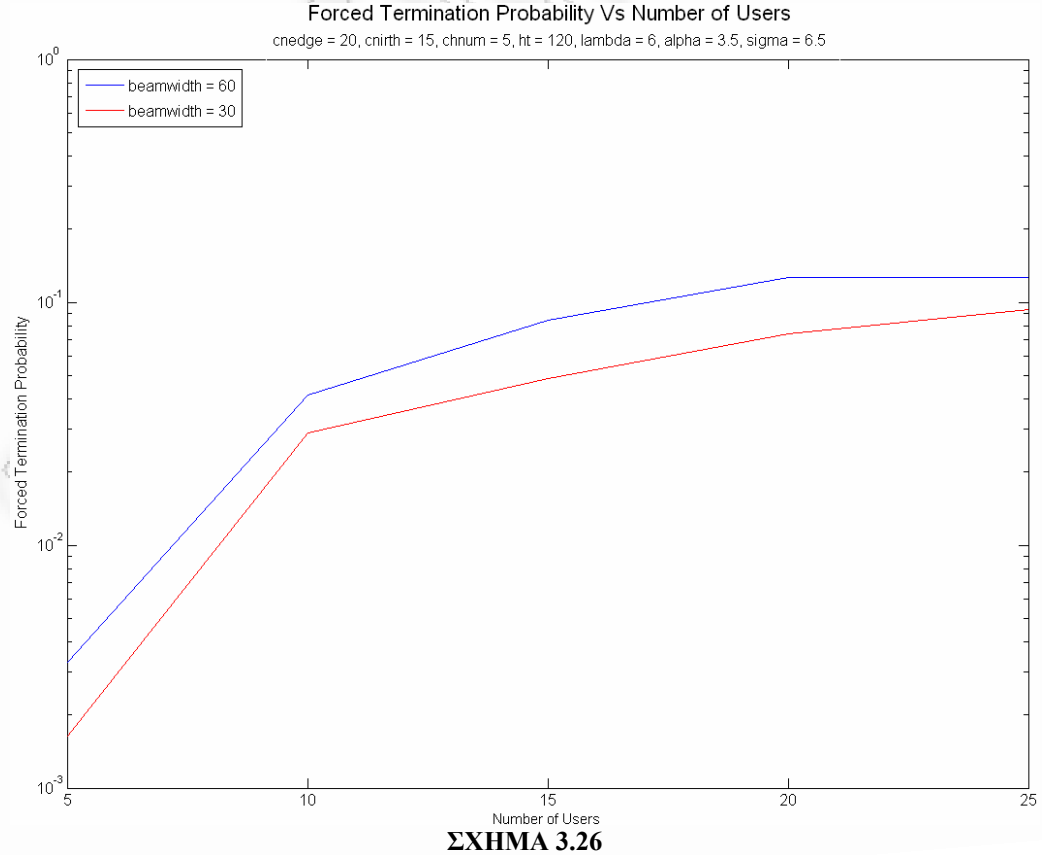
Blocking & forced termination probability for w_HBS = 30° and w_HBS = 60°

UserNumber	**5**	**10**	**15**	**20**	**25**
BlockingProb.	0.0454545 0.0560748	0.0810612 0.104532	0.124817 0.169894	0.213691 0.24276	0.264318 0.300624
ForcedProb.	0.00164204 0.00330033	0.0288693 0.0416327	0.048468 0.084548	0.074159 0.126062	0.093032 0.126926



ΣΧΗΜΑ 3.25

Blocking probability Vs number of users (beamwidth 60 & 30)



ΣΧΗΜΑ 3.26

Forced termination probability Vs number of users (beamwidth 60 & 30)

3.9 DCA – επίδραση Beamforming & Power Control

Από τις γραφικές παραστάσεις της blocking probability Vs Number of users για τις τρεις τεχνικές που εφαρμόσαμε (σχήμα 3.27), παρατηρούμε ότι για μικρό αριθμό χρηστών (έως 15 χρήστες) η τιμή της πιθανότητας αυτής εφαρμόζοντας και τον έλεγχο ισχύος είναι μεγαλύτερη από την αντίστοιχη χωρίς τον έλεγχο αυτό (DCA & Beamforming). Μάλιστα για πολύ μικρό αριθμό χρηστών (5 χρήστες) η πιθανότητα απόρριψης κλήσεων είναι μεγαλύτερη και από αυτή που υπολογίσαμε εφαρμόζοντας μόνο την τεχνική DCA. Τα παραπάνω δικαιολογούνται από το γεγονός ότι για μικρό αριθμό χρηστών, η μείωση που υφίσταται η τιμή του λόγου C/N είναι μεγαλύτερη από αυτήν που υφίσταται η συνολική παρεμβολή, η οποία για μικρό αριθμό χρηστών είναι μικρή. Το αποτέλεσμα όλων αυτών είναι η μείωση της τιμής του λόγου CNIR που λαμβάνεται από την κεραία του κάθε σταθμού βάσης και κατά συνέπεια η αύξηση του αριθμού των μπλοκαρισμένων κλήσεων και επομένως της blocking probability. Αντίθετα για μεγαλύτερο αριθμό χρηστών (από 15 έως και 25 χρήστες) η συνολική εξασθένιση πλέον είναι α μεγαλύτερη και επομένως και η μείωση στην τιμή της σε σχέση με τον λόγο C/N. Κατά συνέπεια ο αριθμός των μπλοκαρισμένων κλήσεων είναι μικρότερος και επομένως και η τιμή της παραπάνω πιθανότητας.

Από την άλλη, από τις γραφικές παραστάσεις της forced termination probability Vs Number of users (σχήμα 3.28), παρατηρούμε ότι με την εφαρμογή του ελέγχου ισχύος, η τιμή της παραπάνω πιθανότητας μειώνεται για όλο το εύρος του αριθμού των χρηστών

DCA + Beamforming + Power Control

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber	:	647	1370	2059	2775	3507
BlockNumber	:	61	151	333	630	1008
CallNumber-BlockNumber	:	586	1219	1726	2145	2499
ForcedNumber	:	2	45	113	198	277

DCA + Beamforming

UserNumber	:	**5**	**10**	**15**	**20**	**25**
CallNumber	:	642	1368	2066	2797	3526
BlockNumber	:	36	143	351	679	1060
CallNumber-BlockNumber	:	606	1225	1715	2118	2496
ForcedNumber	:	2	51	145	267	313

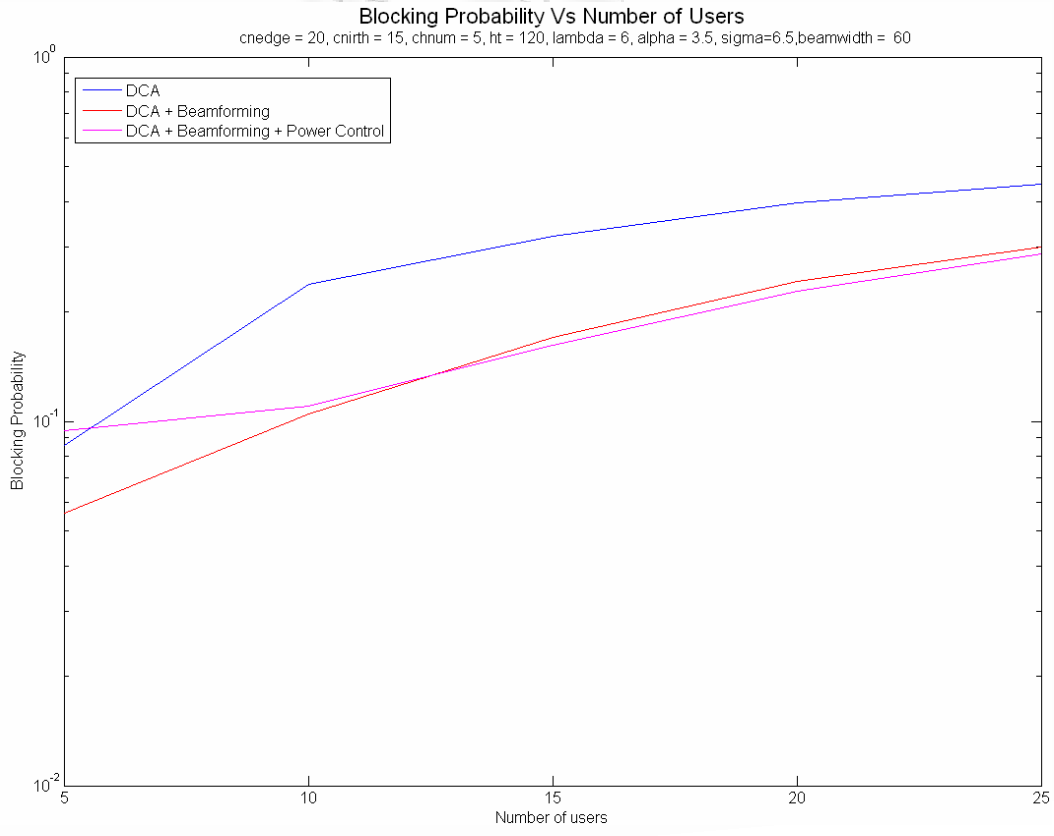
DCA

UserNumber	:	5	10	15	20	25
CallNumber	:	650	1413	2154	2911	3678
BlockNumber	:	56	335	695	1155	1645
CallNumber-BlockNumber	:	594	1078	1459	1756	2033
ForcedNumber	:	33	178	318	463	585

Blocking & forced termination probability for

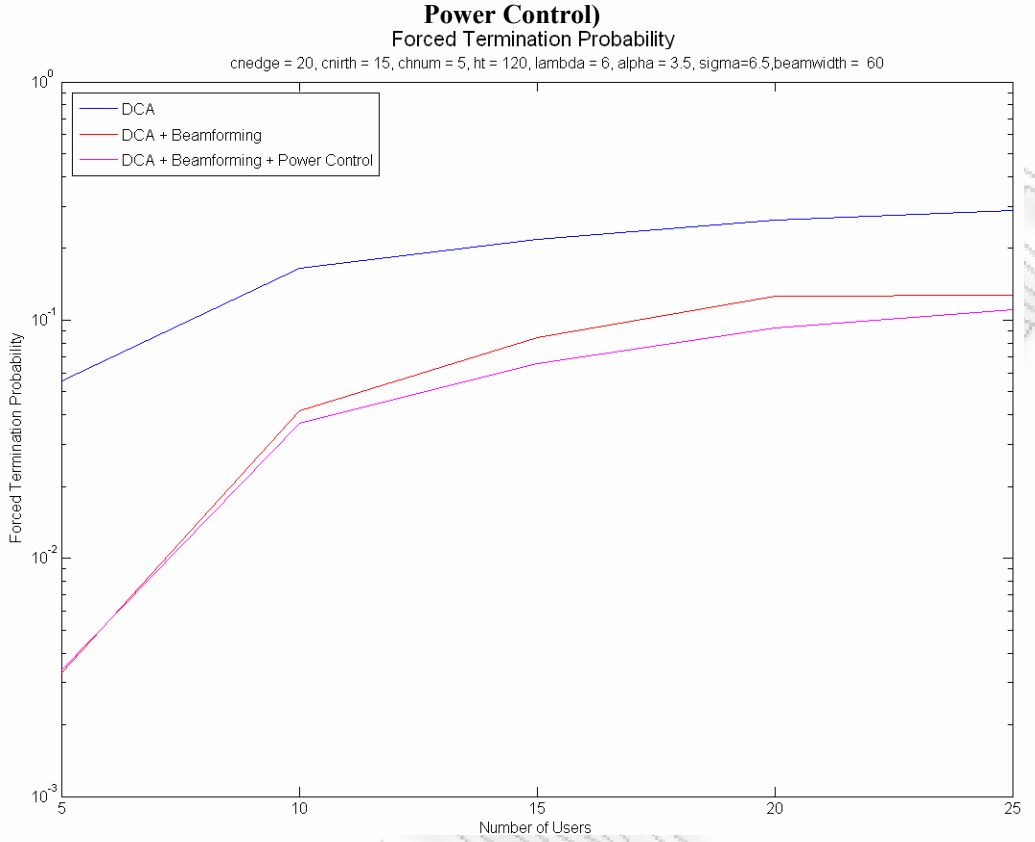
DCA + Beamforming + Power Control & DCA + Beamforming & DCA

UserNumber	5	10	15	20	25
BlockingProb.	0.0942813 0.0560748 0.0861538	0.110219 0.104532 0.237084	0.161729 0.169894 0.322656	0.227027 0.24276 0.396771	0.287425 0.300624 0.447254
ForcedProb.	0.00341297 0.00330033 0.0555556	0.0369155 0.0416327 0.165121	0.065469 0.084548 0.217958	0.092307 0.126062 0.263667	0.110844 0.126926 0.287752



ΣΧΗΜΑ 3.27

Blocking probability Vs number of users (DCA, DCA + Beamforming, DCA + Beamforming + Power Control)



ΣΧΗΜΑ 3.28
Forced termination probability Vs number of users (DCA, DCA + Beamforming, DCA + Power Control

ΚΕΦΑΛΑΙΟ 4
ΣΥΜΠΕΡΑΣΜΑΤΑ ΠΡΟΣΟΜΟΙΩΣΗΣ

Στην εργασία αυτή μελετήσαμε την επίδοση του DCA LOLIA αλγόριθμου σε ένα κυψελοειδές WiMAX σύστημα το οποίο προσομοιώσαμε μέσω του προγράμματος προσομοίωσης MATLAB.

Μέσω των τιμών της blocking και forced termination probability, οι οποίες αποτελούν δύο παραμέτρους οι οποίες σχετίζονται άμεσα με την επίδοση του συστήματός μας, καταλήξαμε σε συμπεράσματα για το πώς επηρεάζεται η επίδοση του συστήματος σε μεταβολές κάποιων παραμέτρων του, όπως ο αριθμός των καναλιών και των χρηστών, το περιβάλλον διάδοσης, το μέγεθος της τηλεπικοινωνιακής κίνησης, κ.ο.κ.

Πιο συγκεκριμένα, διαπιστώσαμε ότι

- η αύξηση τόσο της τηλεπικοινωνικής κίνησης όσο και της τραχύτητας του περιβάλλοντος διάδοσης συνεισφέρουν αρνητικά στην επίδοση του συστήματός μας.
- η αύξηση του αριθμού των εξυπηρετούμενων χρηστών από κάθε κυψέλη συνεπάγεται τη μείωση της επίδοσης του συστήματός μας, η οποία από την άλλη βελτιώνεται σημαντικά με την αύξηση του αριθμού των διαθέσιμων διαύλων.
- η μεταβολή των χαρακτηριστικών της ραδιοζεύξης επηρεάζει και αυτή σημαντικά την επίδοση του συστήματός μας. Η αύξηση της τιμής της μέγιστης ισχύος η οποία μπορεί να εκπεμθεί από τον κάθε κινητό σταθμό οδηγεί σε βελτίωση του λόγου CNIR και κατ΄επέκταση της επίδοσης του συστήματός μας , ενώ από την άλλη η αύξηση της τιμής κατωφλίου του λόγου CNIR οδηγεί σε υποβάθμιση της επίδοσης του συστήματός μας.

Στη συνέχεια αναλύσαμε την τεχνική Beamforming και διαπιστώσαμε το πόσο συνεισφέρει στην βελτίωση της τιμής του λόγου CNIR. Πιο συγκεκριμένα υπολογίσαμε την τιμή του παραπάνω λόγου στην κεραία του σταθμού βάσης για μια από τις 19 κυψέλες του συστήματος (παράλληλα θεωρήσαμε και έναν χρήστη για κάθε κυψέλη) με και χωρίς την παραπάνω τεχνική (με και χωρίς κατευθυντική κεραία) για ένα μεγάλο εύρος τιμών του ανοίγματος δέσμης του κύριου λοβού της

κεραίας (beamwidth). Από τη γραφική παράσταση της διαφοράς ΔCNIR (με και χωρίς την τεχνική beamforming) συναρτήσει της παραμέτρου beamwidth παρατηρήσαμε την εκθετική αύξηση της ΔCNIR σε σχέση με τη μείωση της beamwidth. Το αποτέλεσμα αυτό είναι αναμενόμενο αν λάβουμε υπόψη ότι όσο πιο κατευθυντική είναι η κεραία (μικρότερο άνοιγμα δέσμης) τόσο μεγαλύτερη είναι η καταστολή των παρεμβολών.

Στη συνέχεια προσθέσαμε την τεχνική beamforming στο βασικό πρόγραμμα προσομοίωσης μας και υπολογίσαμε τη συνεισφορά της στη μείωση των τιμών της πιθανότητας απόρριψης εισερχομένων κλήσεων και του εξαναγκασμένου τερματισμού των εν εξελίξει κλήσεων, ιδιαίτερα για μικρές τιμές του ανοίγματος δέσμης των κεραιών.

Τέλος, αναλύσαμε και την τεχνική ελέγχου ισχύος, για την οποία είδαμε ότι και αυτή συνεισφέρει ακόμα περισσότερο στην αύξηση της τιμής του λόγου CNIR (γραφική παράσταση ΔCNIR vs beamwidth) εξαιτίας της περαιτέρω μείωσης των παρεμβολών (μέσω της μείωσης της τιμής της εκπεμπόμενης ισχύος). Κατόπιν και η τεχνική αυτή προστέθηκε στο βασικό πρόγραμμα προσομοίωσης (σε συνδυασμό με τη τεχνική beamforming). Από τα αποτελέσματα της προσομοίωσης παρατηρήσαμε ότι συμβάλλει επιπλέον στη βελίωση της επίδοσης του συστήματός μας για σχετικά όμως μεγάλο αριθμό χρηστών (δηλαδή για μεγαλύτερες τιμές παρεμβολής). Για μικρό όμως αριθμό χρηστών (π.χ. 5 χρήστες/κυψέλη) όπου η παρεμβολή είναι σχετικά μικρή, είδαμε ότι η τεχνική αυτή δίνει αντίθετα αποτελέσματα με την αύξηση της τιμής της blocking probability.

ΠΑΡΑΡΤΗΜΑ

ΚΩΔΙΚΑΣ ΠΡΟΓΡΑΜΜΑΤΩΝ

5.1 DCAmain.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%% SIMULATION PROGRAM FOR DCA ALGORITHM %%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%%% STEP 1: VARIABLES DECLARATION & INITIALIZATION %%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


cnedge = 25.0;
% Carrier to Noise ratio on cell edge

cnirth = 15.0;
% CNIR threshold

lambda = 6.0;
% average call arrival rate (times/hour)

ht = 120.0;
% average call holding time (sec)

timestep = 10;
% timestep of the simulation loop (seconds)
timeend = 5000;
% maximum simulation time

chnum = 5;
% number of available channels per each base station

alpha = 3.5;
% path loss factor

sigma = 6.5;
% standard deviation of shadowing

usernum = [5, 10, 15, 20, 25];
% number of users per cell

output = zeros(5, 5);
% lines:    1: number of generated calls  2: number of blocking calls
%           3: blocking probability       4: forced termination probability
% columns: 1-5: results for usernum 5, 10, 15, 20 and 25 respectively

check = zeros(5, floor(timeend / timestep));
% stores the current value of blocking probability in every time period
% lines 1-5:    number of users: 5, 10, 15, 20 and 25 respectively
```

```
check2 = zeros(5, floor(timeend / timestep));
% stores the current value of forced termination propabilty
% in every time period
% lines 1-5:    number of users: 5, 10, 15, 20 and 25 respectively

for parameter = 1:5
% set the number of users per cell

    rand('state', 5);
    % initialize the internal state of the generator to 5

    randn('state', 1);
    % initialize the internal state of the generator to 1

    user = usernum(parameter);
    % number of users per cell

    baseinfo = zeros(19, 2);
    % initialization of the baseinfo matrix
    % baseinfo(cell #,1) x coordinates
    % baseinfo(cell #,2) y coordinates

    userinfo = zeros(19, user, 6);
    % initialization of the userinfo matrix
    % userinfo(cell #, user #, information)
    % 1: x axis  2: y axis  % 3:attenuation
    % 4:usage 0->non connected 1->connected
    % 5: call termination time  6: allocated channel

    [baseinfo] = basest;
    % set the position for each of the base stations
    [wrapinfo] = wrap;
    % determination of the neigbouring cells for every cell

    [meshnum, meshposition] = cellmesh;
    % set the position of the users in every cell

    timenow = 0;
    % set the initial value of the time simulation loop

    blocknum = 0;
    % set the initial value for the number of blocked calls

    forcenum = 0;
    % set the initial value for the number of forced terminated calls

    callnum = 0;
    % set the initial value for the number of generated calls

    users = 0;
    % set the initial value for the number of connected users

    while timenow < timeend

        callnumold = callnum;
        % the new value of generated calls in the current loop

        blocknumold = blocknum;
        % the new value of blocked calls in the current loop
```

```
        forcenumold = forcenum;
        % the new value of forced terminated calls in the current
        % loop


%%%%%%%%%%%%%%%%%%%%%%%% STEP 2: FINISHED CALLS %%%%%%%%%%%%%%%%%%%%


        for numcell = 1:19
        % for every cell

            for numuser = 1:user
            % for every user in the cell

                if userinfo(numcell, numuser, 4) == 1 &...
                        userinfo(numcell,numuser, 5) < timenow
                % userinfo(:,:,4)--> the usage: connected
                % userinfo(:,:,5)--> call termination time
                    userinfo(numcell, numuser, 4) = 0;
                    % the user is not connected any more

                    users = users - 1;
                    % the number of connected users decreased at 1
                end
            end
        end


%%%%%%%%%%%%%%%%%%%%%%%% STEP 3: REALLOCATION CHECK %%%%%%%%%%%%%%%%%


        for numcell = 1:19
        % for every cell

            for numuser = 1:user
            % for every user in the cell

                if userinfo(numcell, numuser, 4) == 1
                % if the user is connected

                    reallo = 0;
                    % flag: controls if reallocation is needed

                    cnirdb = 0.0;
                    % initialization of the user's CINR

                    dwave = userinfo(numcell, numuser, 3);
                    % pathloss  attenuation of the user

                    cn = power(10.0, cnedge / 10.0) * dwave;
                    % the received user's CNR

                    uwave = 0.0;
                    % initialization of the interference power

                    ch = userinfo (numcell, numuser, 6);
                    % stores the number of the channel which
                      allocated to the user
```

```
                for around = 2:7
                % for every neigbouring cell

                    othercell = wrapinfo(numcell, around);
                    % stores the number of the neigbouring cell

                    for other = 1:user
                    % for every user in the neigbouring cell

                        if userinfo(othercell, other, 4) == 1 ...
                            & userinfo(othercell, other, 6) == ch
                        % if the neigbouring user is connected
                        % and allocated with the same channel

                            userposi(1, 1:2) = ...
                                userinfo(othercell, other, 1:2);
                            % matrix with the coordinates of the
                            % neighbouring user

                            here = baseinfo(numcell, :);
                            % matrix with the coordinates of the
                            % user's BS

                            there = userposi - baseinfo...
                                (othercell, :) + baseinfo... (around, :)
                                + baseinfo... (numcell, :);
                            % the distance between the neighbouring user
                            % and the user's BS taking into account the
                            % cell wrapping effect

                            uwave = uwave + dist(here, there, alpha)...
                                * shadow(sigma);
                            % the total power of the interference

                        end
                    end
                end

                if uwave == 0
                % interference power = 0

                    cnirdb = 10.0 * log10(cn);

                else
                    cnirdb = 10.0 * log10(1 / (uwave / dwave + 1 /cn));
                    % the received CINR from the user

                end


%------------------------ reallocation process -------------------------%


                if cnirdb < cnirth
                % if the received CINR is lower than the threshold

                    reallo = 1;
                    % needs reallocation

                end
```

```
                    if reallo == 1
                        userinfo(numcell, numuser, 4) = 0;
                        % user's usage: not connected

                        users = users - 1;
                        % connected users decreased at 1

                        suceed = 0;
                         % flag: controls if the reallocation process
                         % succeeded

                        cnirdb = 0.0;
                        % set the user's CINR to 0


%----------------- control for non allocated channels --------------------%


                        for ch = 1:chnum
                        % for every channel

                            available = 1;
                            % flag: controls if a channel is allocated

                            for other = 1:user
                            % for all the other users in the cell

                                if userinfo(numcell, other, 4) == 1 &...
                                        userinfo(numcell, other, 6) == ch
                                % if any of the other users in the cell is
                                % allocated with the current channel

                                    available  = 0;
                                end
                            end


% ---------------calculation of the new CINR for the new channel----------%


                            if available == 1
                            % if there is a non allocated channel

                                uwave = 0.0;
                                 % initialization of the interference power

                                 for around = 2:7
                                 % calculation of the new CINR

                                    othercell = wrapinfo(numcell, around);
                                    for other = 1:user
                                        if userinfo(othercell, other, 4)...
                                                == 1 & userinfo(...
                                                othercell,other, 6) == ch
                                            userposi(1, 1:2) =...
                                                userinfo(othercell,...
                                                other, 1:2);
                                            here = baseinfo(numcell, :);
```

```
                                there = userposi - baseinfo...
                                      (othercell, :)+ baseinfo...
                                      (around, :) + baseinfo...
                                      (numcell, :);
                                  uwave = uwave + dist(here,...
                                      there, alpha) * shadow...
                                      (sigma);
                                end
                            end
                        end
                        if uwave == 0
                            cnirdb = 10.0 * log10(cn);
                        else
                            cnirdb = 10.0 *...
                                log10(1 / (uwave / dwave + 1 /cn));
                        end
                    else
                    % if there isn't a non allocated channel

                        cnirdb = 0.0;
                    end
                    % line 262

                    if cnirdb >= cnirth
                    % if CINR > threshold

                        suceed = 1;
                        % the flag is set to 1

                        users = users + 1;
                        % connected users increased at 1

                        userinfo(numcell, numuser, 4) = 1;
                        % usage: connected

                        userinfo(numcell, numuser, 6) = ch;
                        % the allocated channel = ch

                        break
                        % stop searching for other channels

                    end
                end
                % line 265

                if suceed == 0
                % if it there isn't a non allocated channel

                    forcenum = forcenum + 1;
                    % the number of forced terminated calls
                    % increased at 1

                end

            end
            % line 247

        end
        % line 162
```

```
            end
            % line 159

        end
        % line 156


%%%%%%%%%%%%%%%%%%%%% STEP 4: NEW CALL ARRIVAL %%%%%%%%%%%%%%%%%%%%%%%%%%


        for numcell = 1:19
            for numuser = 1:user
                if userinfo(numcell, numuser, 4) == 0 & rand <= lambda *...
                        (timestep / 3600)
                % usage --> not connected
                % lambda * (timestep / 3600): the user's traffic load in
                % the current loop

                    callnum = callnum + 1;
                    % the number of generated calls increased at 1

                    mesh = floor(meshnum .* rand) + 1;
                    % stores a random number between [1 meshnum+1]
                    % indicates the position of the connected user in
                    % the cell

                    while mesh > meshnum
                    % generates a new random number until mesh <= meshnum

                        mesh = floor(meshnum .* rand) + 1;
                    end

                    userinfo(numcell, numuser, 1:2) = baseinfo(numcell,...
                        :) + meshposition(mesh, :);
                    % userinfo:     the X and Y user coordinates in
                    %               relation from the beginning of axes
                    % baseinfo:     the X an Y BS coordinates  in
                    %               relation from the beginning of axes
                    % meshposition: the X an Y mesh coordinates in
                    %               relation from the BS's position

                    suceed = 0;
                    cnirdb = 0.0;
                    userposi(1, 1:2) = userinfo(numcell, numuser, 1:2);
                    here = baseinfo(numcell, :);
                    there = userposi;
                    % in this case it doesn't need the 3 compensating

                    dwave = dist(here, there, alpha) * shadow(sigma);
                    cn = power(10.0, cnedge / 10.0) * dwave;
                    % calculation of the user's CNR


%----------------- searching for a non allocated channel  ----------------%

                    for ch = 1:chnum
                        available = 1;
                        for other = 1:user
                            if userinfo(numcell, other, 4) == 1 &...
                                    userinfo(numcell, other, 6) == ch
```

```
                                available = 0;
                            end
                        end
%----------------------- calculation of the interference -----------------%


                        if available == 1
                            uwave = 0.0;
                            for around = 2:7
                                othercell = wrapinfo(numcell, around);
                                for other = 1:user
                                    if userinfo(othercell, other, 4) ==...
                                            1 & userinfo(othercell,...
                                            other, 6) == ch
                                        userposi(1, 1:2) = userinfo...
                                            (othercell, other, 1:2);
                                        here = baseinfo(numcell, :);
                                        there = userposi - baseinfo...
                                            (othercell, :) + baseinfo...
                                            (around, :) + baseinfo...
                                            (numcell, :);
                                        uwave = uwave + dist(here,...
                                            there, alpha) * shadow(sigma);
                                    end
                                end
                            end
                            if uwave == 0
                                cnirdb = 10.0 * log10(cn);
                            else
                                cnirdb = 10.0 * log10(1 / (uwave / dwave...
                                    + 1 / cn));
                            end
                        else
                            cnirdb = 0.0;
                        end


%-------------------- if CINR > CINR(threshold) ---------------------------%


                        if cnirdb >= cnirth
                            suceed = 1;
                            users = users + 1;
                            userinfo(numcell, numuser, 3) = dwave;
                            % the pathloss

                            userinfo(numcell, numuser, 4) = 1;
                            % usage: connected

                            userinfo(numcell, numuser, 5) = timenow +...
                                holdtime(ht);
                            % call termination time: the current simulation
                            % time + the average call holding time

                            userinfo(numcell, numuser, 6) = ch;
                            break
                            % stop searching for channel

                        end
                    end
                    % line 425
```

```
                    if suceed == 0
                    % if there isn't a non allocated channel

                        blocknum = blocknum + 1;
                        % the number of blocked calls increased at 1

                    end
                end
                % line 380

            end
            % line 379

        end
        % line 378


%%%%%%%%%%%%%%%%%%%% STEP 5: OUTPUT PART %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


        fprintf('%d\t%d\t%d\t%d\t%e\n', parameter, timenow, callnum -...
            callnumold, blocknum - blocknumold, blocknum / callnum);
        % in every simulation time loop the values of:
        %         1. the variable parameter    (in decimal notation (%d))
        %         2. the simulation time       (in decimal notation (%d))
        %         3. the new generated calls   (in decimal notation (%d))
        %         4. the new blocked calls     (in decimal notation (%d)))
        %         5. the blocking propability (in exponential notation (%e))

        check(parameter, timenow / timestep + 1) = blocknum / callnum;
        % the current value of blocking propability in every time period

        check2(parameter, timenow / timestep + 1) = forcenum / (callnum...
            - blocknum);
        % the current value of forced termination propability in every
        % time period

        timenow = timenow + timestep;
        % curent simulation time

    end
    % line 111

    output(1, parameter) = callnum;
    % number of generated calls

    output(2, parameter) = blocknum;
    % number of blocked calls

    output(3, parameter) = forcenum;
    % number of forced terminated calls

    output(4, parameter) = blocknum / callnum;
    % blocking propability

    output(5, parameter) = forcenum / (callnum - blocknum);
    % forced termination propability

end
% line 60
```

```
fid = fopen('data.txt', 'w');
% opens the file 'data.txt' for write

fprintf(fid, 'UserNumber\t');
% writes the string: 'UserNumber' adding one horizontal tab in the end

fprintf(fid, '%g\t%g\t%g\t%g\t%g\n', usernum(1,:));
% writes the values of the matrix usernum  in exponential notation

fprintf(fid, 'CallNumber\t        ');
fprintf(fid, '%g\t%g\t%g\t%g\t%g\n', output(1,:));
% writes the final number of generated calls in exponential notation for
% every number of users

fprintf(fid, 'BlockNumber\t        ');
fprintf(fid, '%g\t%g\t%g\t%g\t%g\n', output(2,:));
% writes the final number of blocked calls in exponential notation for
% every number of users

fprintf(fid, 'ForcedNumber\t        ');
fprintf(fid, '%g\t%g\t%g\t%g\t%g\n', output(3,:));
% writes the final number of forced terminated calls in exponential
% notation for every number of users

fprintf(fid, 'BlockingProb. \t      ');
fprintf(fid, '%g\t%g\t%g\t%g\t%g\n', output(4,:));
% writes the final number of blocking propability in exponential notation
% for every number of users

fprintf(fid, 'ForcedTerminationProb. \t');
fprintf(fid, '%g\t%g\t%g\t%g\t%g\n', output(5,:));
% writes the final number of forced termination propability in exponential
% notation for every number of users

fclose(fid);
```

5.2 basest.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%% FUNCTION WHICH SETS THE POSITIONS OF TH BASE STATIONS %%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%



function [out] = basest()

a=0.5*sqrt(3.0);
% the distance between the center of the cell and the means of its side

R=1.0;
% the radius of the cell (Km)
```

```
% calculation of the matrix baseinfo (19 x 2).
% The number of rows is the number of the base stations
% the first column contains the X coordinates of each base station
% the second column contains the Y coordinate of each base station

baseinfo(1,1)=0.0;
baseinfo(1,2)=0.0;
baseinfo(2,1)=-a;
baseinfo(2,2)=R+R/2;
baseinfo(3,1)=-2*a;
baseinfo(3,2)=0;
baseinfo(4,1)=-a;
baseinfo(4,2)=-(R+R/2);
baseinfo(5,1)=a;
baseinfo(5,2)=-(R+R/2);
baseinfo(6,1)=2*a;
baseinfo(6,2)=0.0;
baseinfo(7,1)=a;
baseinfo(7,2)=R+R/2;
baseinfo(8,1)=0.0;
baseinfo(8,2)=2*(R+R/2);
baseinfo(9,1)=-2*a;
baseinfo(9,2)=2*(R+R/2);
baseinfo(10,1)=-3*a;
baseinfo(10,2)=R+R/2;
baseinfo(11,1)=-4*a;
baseinfo(11,2)=0;
baseinfo(12,1)=-3*a;
baseinfo(12,2)=-(R+R/2);
baseinfo(13,1)=-2*a;
baseinfo(13,2)=-2*(R+R/2);
baseinfo(14,1)=0.0;
baseinfo(14,2)=-2*(R+R/2);
baseinfo(15,1)=2*a;
baseinfo(15,2)=-2*(R+R/2);
baseinfo(16,1)=3*a;
baseinfo(16,2)=-(R+R/2);
baseinfo(17,1)=4*a
baseinfo(17,2)=0.0;
baseinfo(18,1)=3*a;
baseinfo(18,2)=R+R/2;
baseinfo(19,1)=2*a;
baseinfo(19,2)=2*(R+R/2);
out=baseinfo;
```

5.3 cellmesh.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%% FUNCTION WHICH SETS MESHES IN A CELL %%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


function [meshnum, meshposition] = cellmesh()
% meshnum -> number of possible user's positions inside the cell area
% meshposition -> (meshnum x 2) matrix with the coordinates for every mesh


Fineness = 50;
% parameter for mesh fineness

k = 1;
% parameter that counts the right shifts of mesh in axis X

j = 0;
% parameter that counts the number of meshes

ds = sqrt(3.0) / Fineness;
% the distance between the meshes
% ds = (2 * a) / Fineness = (2 * (1/2 * sqrt(3.0))) / Fineness = 0.0577

xmesh = -0.5 * sqrt(3.0);
% initialize the mesh position in axis X
% the pic A of the triangle ABC
% xmesh_initial = -a = -0.8660


%-----calculation of the minimum and maximum positions of the meshes-------
%-------------------------------in the Y axis-------------------------------

while xmesh < 0.5 * sqrt(3.0)
% inside the diameter's region

    xmesh = (k-1) * ds - 0.5 * sqrt(3.0);
    % the position of the k+1 mesh in the X axis

    if xmesh < 0.0
    % in the left diameter's side

        ymin = -xmesh/sqrt(3.0) - ds - 1.0;
        % the minimum y position of k+1 mesh
        % the factor ds is added to suppress the factor ds
        % which is used in the equation of ymesh calculation

        ymax = xmesh/sqrt(3.0) + 1.0;
        % the maximum y position of k+1 mesh

    elseif xmesh == 0.0
            ymin = -1.0;
            ymax = 1.0;

    else
    %  in the right diameter's side
```

```
            ymin = xmesh/sqrt(3.0) - ds - 1.0;
            ymax = -xmesh/sqrt(3.0) + 1.0;

    end
            k = k + 1;
            % the next position of the mesh in X axis

            ymesh = ymin;
            % the y position of the mesh

            while ymesh < ymax
                ymesh = ymesh + ds;
                % the y position of the mesh increases at ds
                % the x position of the mesh remains constant

                if ymesh > ymax
                % the mesh position is right shifted at ds in the axis X

                    break
                end
                j = j + 1;
                % the next mesh

                posi(j,1) = xmesh;
                % the first column of the matrix meshposition contains
                % the coordinates x of the mesh

                posi(j,2) = ymesh;
                % the second column of the matrix meshposition contains
                % the coordinates y of the mesh

            end
        end

        meshnum = j;
        meshposition = posi;
        % (J x 2) matrix with  the coordinates of every mesh

        plot(meshposition(:,1),meshposition(:,2),'.')
```

5.4 dist.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%% FUNCTION WHICH GENERATES ATTENUATION DUE TO DISTANCE %%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%



function [x] = dist(a,b,alpha)
% a -> (1 x 2) matrix with the X and Y coordinates of the first point
% b -> (1 x 2) matrix with the X and Y coordinates of the second point
% alpha -> path loss factor
```

```
x = sqrt((a-b)*(a-b)');
x = power(x, -1.*alpha);
% x= d ^ (-a)
```

5.5 holdtime.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%% FUNCTION WHICH GENERATES THE HOLDING TIME %%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


function [x] = holdtime(ht)
% ht --> the mean value of average call holding time

para=rand;
% generates a random number between [0,1]

while para>=1
% avoid the propability the para be equal to 1
    para=rand;
    % generates a new value for the para

end

x=ht.*(-log(1 - para));
% the holding time follows exponential distribution with mean value ht
```

5.6 shadow.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%% FUNCTION WHICH GENERATES ATTENUATION OF SHADOWING %%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


function [x] = shadow(sigma)
% sigma -> standard deviation of shadowing

anoz=randn;
% creates pseudo-random values drawn from normal distribution
% with mean value 0 and standard deviation 1

db=sigma*anoz;
% σ in dB

x=power(10.0, 0.1*db);
% the shadowing follows a lognormal distribution
% with standard deviation sigma (σ)
```

5.7 wrap.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%% FUNCTION WHICH GIVES INFORMATION ABOUT CELL - WRAPPING %%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


function [out] = wrap(inputmat)
% inputmat (19 x 19).
% The number of rows is the number of the base stations
% each column contains the neighbouring base stations included
% and the basic base station.


inputmat(1,1:19) = 1:19;
inputmat(2,1:19) = [ 2,9,10,3,1,7,8,14,13,17,16,11,12,4,5,6,18,19,15 ];
inputmat(3,1:19) = [ 3,10,11,12,4,1,2,9,17,16,15,19,18,13,14,5,6,7,8 ];
inputmat(4,1:19) = [ 4,3,12,13,14,5,1,2,10,11,19,18,17,9,8,15,16,6,7 ];
inputmat(5,1:19) = [ 5,1,4,14,15,16,6,7,2,3,12,13,9,8,19,11,10,17,18 ];
inputmat(6,1:19) = [ 6,7,1,5,16,17,18,19,8,2,3,4,14,15,11,10,9,13,12 ];
inputmat(7,1:19) = [ 7,8,2,1,6,18,19,15,14,9,10,3,4,5,16,17,13,12,11 ];
inputmat(8,1:19) = [ 8,14,9,2,7,19,15,5,4,13,17,10,3,1,6,18,12,11,16 ];
inputmat(9,1:19) = [ 9,13,17,10,2,8,14,4,12,18,6,16,11,3,1,7,19,15,5 ];
inputmat(10,1:19) = [ 10,17,16,11,3,2,9,13,18,6,5,15,19,12,4,1,7,8,14 ];
inputmat(11,1:19) = [ 11,16,15,19,12,3,10,17,6,5,14,8,7,18,13,4,1,2,9 ];
inputmat(12,1:19) = [ 12,11,19,18,13,4,3,10,16,15,8,7,6,17,9,14,5,1,2 ];
inputmat(13,1:19) = [ 13,12,18,17,9,14,4,3,11,19,7,6,16,10,2,8,15,5,1 ];
inputmat(14,1:19) = [ 14,4,13,9,8,15,5,1,3,12,18,17,10,2,7,19,11,16,6 ];
inputmat(15,1:19) = [ 15,5,14,8,19,11,16,6,1,4,13,9,2,7,18,12,3,10,17 ];
inputmat(16,1:19) = [ 16,6,5,15,11,10,17,18,7,1,4,14,8,19,12,3,2,9,13 ];
inputmat(17,1:19) = [ 17,18,6,16,10,9,13,12,19,7,1,5,15,11,3,2,8,14,4 ];
inputmat(18,1:19) = [ 18,19,7,6,17,13,12,11,15,8,2,1,5,16,10,9,14,4,3 ];
inputmat(19,1:19) = [ 19,15,8,7,18,12,11,16,5,14,9,2,1,6,17,13,4,3,10 ];

out = inputmat;
```

5.8 DCA_theory.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%% SIMULATION PROGRAM FOR DCA ALGORITHM %%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%% BLOCKING PROBABILITY VS USERS NUMBER %%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%% THEORETICAL - SIMULATION CALCULATION %%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%



cnedge = 20.0;
cnirth = 15.0;
lambda = 6.0;
ht = 120.0;
timestep = 10;
timeend = 5000;
chnum = 5;
alpha = 3.5;
sigma = 6.5;
```

```
usernum = [10, 15, 20, 25];
output = zeros(5, 4);

for parameter = 1:4

    rand('state', 5);
    randn('state', 1);
    user = usernum(parameter);
    baseinfo = zeros(19, 2);
    userinfo = zeros(19, user, 15);
    [baseinfo] = basest;
    [wrapinfo] = wrap;
    [meshnum, meshposition] = cellmesh;
    timenow = 0;
    blocknum = 0;
    forcenum = 0;
    callnum = 0;
    users = 0;

    while timenow < timeend
        callnumold = callnum;
        blocknumold = blocknum;
        forcenumold = forcenum;

        for numcell = 1:19
            for numuser = 1:user
                if userinfo(numcell, numuser, 4) == 1 & userinfo(numcell...
                        , numuser, 5) < timenow
                    userinfo(numcell, numuser, 4) = 0;
                    users = users - 1;
                end
            end
        end

        for numcell = 1:19
            for numuser = 1:user
            if userinfo(numcell, numuser, 4) == 0 & rand <= lambda
                    *...(timestep / 3600)
                    callnum = callnum + 1;
                    mesh = floor(meshnum .* rand) + 1;
                    while mesh > meshnum
                        mesh = floor(meshnum .* rand) + 1;
                    end
                    userinfo(numcell ,numuser, 1:2) = baseinfo...
                        (numcell, :) + meshposition(mesh, :);
                    suceed = 0;
                    cnirdb = 0.0;
                    userposi(1, 1:2) = userinfo(numcell, numuser, 1:2);
                    here = baseinfo(numcell, :);
                    there = userposi;
                    dwave = dist(here, there, alpha) * shadow(sigma);
                    cn = power(10, cnedge / 10.0) * dwave;
                    cnirdb = 10.0 * log10(cn);
                    for ch = 1:chnum
                        available = 1;
                        for other = 1:user
                            if userinfo(numcell, other, 4) == 1 &...
                                    userinfo(numcell, other, 6) == ch
                                available = 0;
                            end
                        end
                        if available ==1
```

```
                                suceed = 1;
                                users = users + 1;
                                userinfo(numcell, numuser, 3) = dwave;
                                userinfo(numcell, numuser, 4) = 1;
                                userinfo(numcell, numuser, 5) = timenow +...
                                    holdtime(ht);
                                userinfo(numcell, numuser, 6) = ch;
                                break
                            end
                        end
                        if suceed == 0
                            blocknum = blocknum + 1;
                        end
                    end
                end
            end
            timenow = timenow + timestep;
        end

        output(1, parameter) = callnum;
        output(2, parameter) = blocknum;
        output(3, parameter) = blocknum / callnum;
        output(4, parameter) = forcenum / (callnum - blocknum);
        output(5, parameter) = block_prop(user, chnum, lambda, ht);
end

semilogy(usernum(1,:), output(5, :), 'r-', usernum(1,:), output(3, :),...
    'bo');
legend('theory', 'simulation results',2)
xlabel('Number of users(users/cell)');
ylabel('Blocking probability');
title('Graph for chnum=5, cnirth=15(dB) and cnedge=20(dB)');
```

5.9 block_prob.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FUNCTION WHICH CALCULATES THE THEORETICAL VALUE OF BLOCKING PROPABILITY %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


function [x] = block_prop (users, chan, l, h)
% users -> number of users
% chan - > number of channels
% l -> average call arrival rate (calls/hour)
% h -> average call holding time (sec)


l1=l/3600;
% transformation of average call arival rate from calls/hour to calls/sec

numerator = (factorial(users-1) / (factorial(users-1 - chan) *...
    factorial(chan))) * power(h*l1, chan);

denominator = 0;
```

```
i = 0;
while i <= chan
    result = (factorial(users-1) / (factorial(users-1 - i) *...
        factorial(i))) * power(h*l1, i);
    denominator = denominator + result;
    i = i+1;
end

x = numerator / denominator;
```

5.10 BeamformingCNIR.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% CALCULATION OF THE CNIR IMPROVEMENT FOR DIFFERENT VALUES OF BEAM WIDTH %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%%%%%%%%%%%%%%% STEP 1: VARIABLES DECLERATION %%%%%%%%%%%%%%%%%%%%%%%


output = zeros(1,((120 - 30) / 10) + 1);
% (1 x 10) matrix which stores the difference CNIR(with beamformer) -
% CNIR(without beamformer) for every value of beam width

i = 1;
% loop number indicator

for w_HBS = 30:10:120
% for every value of beam width
% [horizontal] w_HBS beam width at BS for the target direction (degree)

    rand('state', 5);
    randn('state', 1);

    cnedge = 20.0;
    timestep = 10;
    timeend = 5000;

    numcell = 1;
    % the simulation refers to CNIR improvement for one cell
    % (center cell (0, 0))

    alpha = 3.5;
    sigma = 6.5;

    user = 1;
    % number of users in the cell

    h = 0;
    % the height of the BS
    % the case of macrocell situation

    backg_BS = -100;
    % [horizontal] antenna gain at BS for the opposite direction (dB)
```

```
    w_VBS = 360;
    % [vertical] beam width at BS (degree)
    % case of Omni directional antenna

    cnirdb_all = 0;
    % initial value of CNIR (without beam forming)

    cnirdb_BF_all = 0;
    % initial value of CNIR (with beam forming)


%%%%%%%%%%%%%%%%%%%%%%%%% STEP 2: INITIALIZATION %%%%%%%%%%%%%%%%%%%%%%%%%


    baseinfo = zeros(19, 2);
    userinfo = zeros(19, user, 15);
    [baseinfo] = basest;
    baseinfo_new = baseinfo(: , 1) + baseinfo(: , 2) * j;
    % (19 x 1) matrix with the values of every BS's coordinates
    % in complex form

    [wrapinfo] = wrap;
    [meshnum, meshposition] = cellmesh;
    timenow = 0;

    g_HBS = antgain(w_HBS, backg_BS);
    % antenna gain calculation of BS in the horizontal direction

    g_VBS = antgain(w_VBS, 0);
    % antenna gain calculation of BS in the vertical direction


%%%%%%%% STEP 3: DECLARATION OF THE USER'S POSITION FOR EVERY CELL %%%%%%%%


    while timenow < timeend

        for cell = 1:19
            mesh = floor(meshnum .* rand) + 1;
            while mesh > meshnum
                mesh = floor(meshnum .* rand) + 1;
            end
            userinfo(cell, user, 1:2) = baseinfo(cell, :) +...
                meshposition(mesh, :);
            % the X and Y coordinates of every user in relation of
            % the beginning of axes
        end

        userposi = userinfo(: ,1:2);
        userposi_new = userposi(:, 1) + userposi(:, 2) * j;
        % the X and Y coordinates of every user in complex form

        meshposition_new = userposi_new - baseinfo_new;
        % the X and Y coordinates of every user in complex form
        % in relation of the cell's BS
```

```
z = [meshposition_new'; userposi_new'];
% (2 x 19) matrix
% row 1: the coordinates of every user in relation of the cell's BS
%      in complex form
% row 2: the coordinates of every user in relation of the beginning
%      of axes in complex form


%%%%%%%% STEP 4: CALCULATION OF THE ANTENNA GAIN AT THE CENTRAL BS %%%%%%%%


d(1, : ) = abs(z(1, :));
% the magnitude of the distance between every user and its BS
% in horizontal axis

d(2, : ) = abs(z(2, :));
% the magnitude of the distance between every user and
% the central BS in horizontal axis

d2 = sqrt((d .^ 2) + (h .^ 2));
% the distance taking into account the BS's height

phai(1, :) = angle(z(1, :));
% the angle difference between every user and its BS
% (horizontal plane)

phai(2, :) = angle(z(2, :));
% the magnitude of the distance between every user and the central
% BS (horizontal plane)

deg = phai * (180 / pi);
% conversion of radian to degree

deg_B = deg(2, 1) - deg(2, :);
% the angle difference between the main user and every neigbouring
% user from the central BS (horizontal plane)

degH = 90 * ones(1, 19);
% the angle difference between the main user and every neigbouring
% user from the central BS (vertical plane)

degVBS = mod(round(degH - degH(1)), 360);
% the angle difference between MSs and the central BS
% in vertical direction

degHBS = mod(round(deg_B), 360);
% adaptation of deg_B in the interval between 0 and 360

gain = g_HBS(degHBS(1:19) + 1) + g_VBS(degVBS(1:19) + 1);
% antenna gain calculation at the central BS (dB)

gain_W = 10 .^ ( gain ./ 10);
% conversion of dB -> W
```

```
%%%%%%%%%%%%% STEP 5: CALCULATION OF THE CINR AT THE CENTRAL BS %%%%%%%%%%

        for numuser = 1:user
            cnirdb = 0.0;
            % initial value of CNIR without the beamforming

            cnirdb_BF = 0.0;
            % initial value of CNIR with the beamforming

            here = baseinfo(numcell, :);
            there = userposi(1, :);
            dwave = dist(here, there, alpha) * shadow(sigma);
            cn = power(10, cnedge / 10.0) * dwave;
            uwave = 0;
            % initial value of interference without the beamforming

            uwave_BF = 0;
            % initial value of interference with the beamforming

            for around = 2:19
            % for every neigbouring cell

                othercell = wrapinfo(numcell, around);
                for other = 1:user
                    here = baseinfo(numcell, :);
                    there = userposi(around, :) - baseinfo(othercell, :)...
                        + baseinfo(around, :) + baseinfo(numcell, :);
                     shadowing = shadow(sigma);
                    % attenuation due to shadowing

                    uwave = uwave + dist(here, there, alpha) *...
                        shadowing;
                    % interference power without the beamforming

                    uwave_BF = uwave_BF + dist(here, there, alpha)...
                        * shadowing * gain_W(1, othercell);
                    % interference power with the beamforming
                end
            end
            if uwave == 0
                cnirdb = 10.0 * log10(cn);
            else
                cnirdb = 10.0 * log10(1 / ((uwave / dwave) + (1 / cn)));
            end

            if uwave_BF == 0
                cnirdb_BF = 10.0 * log10(cn);
            else
                cnirdb_BF = 10.0 * log10(1 /((uwave_BF /dwave) + (1 /cn)));
            end
        end
        cnirdb_all = cnirdb_all + cnirdb;
        % the total value of CNIR without beamforming in the current loop

        cnirdb_BF_all = cnirdb_BF_all + cnirdb_BF;
        % the total value of CNIR with beamforming in this current loop

        timenow = timenow + timestep;
    end
```

```
    output(1, i) = (cnirdb_BF_all - cnirdb_all) / (timeend / timestep);
    % the average value of the difference between cnirdb_BF and cnirdb

    i = i + 1;
   % next step of the beam width loop

end

plot(30:10:120, output, '.-')
axis([25 125 0 18])
xlabel('Beam Width (degrees)');
ylabel('CNIR improvement (dB)');
grid on;
```

5.11 antgain.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%% FUNCTION WHICH CALCULATES THE ANTENNA GAIN OF THE BEAMFORMER %%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


function gain = antgain(width, back)
% width -> beam width of the beamformer at the target direction (degrees)
% back -> antenna gain for the opposite direction (dB)


theta = [0:359];
gain = zeros(1,360);

if width ~= 360
    % except of Omni-directional antenna

    n = log10(sqrt(1/2)) ./ log10(cos(width * (pi/360)));
    % the  n coefficient

    gain(1:89) = 10 * log10(cos(theta(1:89) * (pi/180)) .^ n);
    gain(272:360) = 10 * log10(cos(theta(272:360) * (pi/180)) .^ n);
    % antenna gain (dB)

    if back ~=0
        gain(90:271) = back;
        %definition of antenna gain for the opposite direction

    end
end
```

5.12 DCABeamforming.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%% SIMULATION PROGRAM FOR DCA ALGORITHM %%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%% APPLYING THE BEAMFORMING TECHNIQUE %%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%%%%%% STEP 1: VARIABLES DECLARATION & INITIALIZATION %%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


cnedge = 20.0;
% Carrier to Noise ratio on cell edge

cnirth = 15.0;
% CNIR threshold

lambda = 6.0;
% average call arrival rate (times/hour)

ht = 120.0;
% average call holding time (sec)

timestep = 10;
% timestep of the simulation loop (seconds)

timeend = 5000;
% maximum simulation time

chnum = 5;
% number of available channels per each base station

alpha = 3.5;
% path loss factor

sigma = 6.5;
% standard deviation of shadowing

w_HBS = 60
% antenna's beamwidth
 usernum = [5, 10, 15, 20, 25];
% number of users per cell

output_BF = zeros(5, 5);
% lines:    1: number of generated calls  2: number of blocking calls
%           3: blocking propability       4: forced termination propability
% columns: 1-5: results for usernum 5, 10, 15, 20 and 25 respectively

check_BF = zeros(5, floor(timeend / timestep));
% stores the current value of blocking propabilty in every time period
% lines 1-5:    number of users: 5, 10, 15, 20 and 25 respectively

check2_BF = zeros(5, floor(timeend / timestep));
% stores the current value of forced termination propabilty
% in every time period
% lines 1-5:    number of users: 5, 10, 15, 20 and 25 respectively
```

```
for parameter = 1:5
% set the number of users per cell

    rand('state', 5);
    % initialize the internal state of the generator to 5

    randn('state', 1);
    % initialize the internal state of the generator to 1

    user = usernum(parameter);
    % number of users per cell

    baseinfo = zeros(19, 2);
    % initialization of the baseinfo matrix
    % baseinfo(cell #,1) x coordinates
    % baseinfo(cell #,2) y coordinates

    userinfo = zeros(19, user, 6);
    % initialization of the userinfo matrix
    % userinfo(cell #, user #, information)
    % 1: x axis  2: y axis  % 3:attenuation
    % 4:usage 0->non connected 1->connected
    % 5: call termination time  6: allocated channel

    [baseinfo] = basest;
    % set the position for each of the base stations

    [wrapinfo] = wrap;
    % determination of the neigbouring cells for evey cell

    [meshnum, meshposition] = cellmesh;
    % set the position of the users in every cell

    timenow = 0;
    % set the initial value of the time simulation loop

    blocknum = 0;
    % set the initial value for the number of blocked calls
    forcenum = 0;
    % set the initial value for the number of forced terminated calls

    callnum = 0;
    % set the initial value for the number of generated calls

    users = 0;
    % set the initial value for the number of connected users

    while timenow < timeend

        callnumold = callnum;
        % the new value of generated calls in the current loop

        blocknumold = blocknum;
        % the new value of blocked calls in the current loop

        forcenumold = forcenum;
        % the new value of forced terminated calls in the current loop
```

```
%%%%%%%%%%%%%%%%%%%%%%%%% STEP 2: FINISHED CALLS %%%%%%%%%%%%%%%%%%%%%%%%%

        for numcell = 1:19
        % for every cell

            for numuser = 1:user
            % for every user in the cell

                if userinfo(numcell, numuser, 4) == 1 & userinfo(numcell...
                       ,numuser, 5) < timenow
                % userinfo(:,:,4)--> the usage: connected
                % userinfo(:,:,5)--> call termination time

                    userinfo(numcell, numuser, 4) = 0;
                    % the user is not connected any more

                    users = users - 1;
                    % the number of connected users decreased at 1

                end
            end
        end


%%%%%%%%%%%%%%%%%%%%%%%%% STEP 3: REALLOCATION CHECK %%%%%%%%%%%%%%%%%%%%%%%%%


        for numcell = 1:19
        % for every cell

            for numuser = 1:user
            % for every user in the cell

                if userinfo(numcell, numuser, 4) == 1
                % if the user is connected

                    reallo = 0;
                    % flag: controls if reallocation is needed

                    cnirdb = 0.0;
                    % initialization of the user's CINR

                    dwave = userinfo(numcell, numuser, 3);
                    % pathloss  attenuation of the user

                    cn = power(10.0, cnedge / 10.0) * dwave;
                    % the received user's CNR

                    uwave = 0.0;
                    % initialization of the interference power

                    ch = userinfo (numcell, numuser, 6);
                    % stores the number of the channel which allocated to
                    % the user

                    for around = 2:7
                    % for every neigbouring cell
```

```
                othercell = wrapinfo(numcell, around);
                % stores the number of the neigbouring cell

                for other = 1:user
                % for every user in the neigbouring cell

                    if userinfo(othercell, other, 4) == 1 ...
                        & userinfo(othercell, other, 6) == ch
                    % if the neigbouring user is connected and
                    % allocated with the same channel

                        userposi(1, 1:2) = ...
                            userinfo(othercell, other, 1:2);
                        % matrix with the coordinates of the
                        % neighbouring user

                        here = baseinfo(numcell, :);
                        % matrix with the coordinates of the user's
                        % BS

                        there = userposi - baseinfo(othercell,...
                            :) + baseinfo(around, :) +...
                            baseinfo(numcell, :);
                        % the distance between the neigbouring user
                        % and the user's BS taking into account the
                        % cell wrapping effect
                        ant_gain = antgain_interf (here(1, 1),...
                            here(1, 2), userinfo(numcell,...
                            numuser, 1), userinfo(numcell,...
                            numuser, 2), there(1, 1),...
                            there(1, 2), w_HBS);
                        % the antenna gain of the user's BS at the
                        % neigbouring users direction

                        uwave = uwave + dist(here, there, alpha)...
                            * shadow(sigma) * ant_gain;
                        % the total power of the interference
                        % taking into account the beamforming
                        % technique

                    end
                end
            end


            if uwave == 0
            % interference power = 0

                cnirdb = 10.0 * log10(cn);

            else
                cnirdb = 10.0 * log10(1 / (uwave / dwave + 1 /cn));
                % the received CINR from the user

            end
```

```
%------------------------ reallocation process -------------------------%

                    if cnirdb < cnirth
                    % if the received CINR is lower than the threshold

                        reallo = 1;
                        % needs reallocation

                    end

                    if reallo == 1
                        userinfo(numcell, numuser, 4) = 0;
                        % user's usage: not connected

                        users = users - 1;
                        % connected users decreased at 1

                        suceed = 0;
                         % flag: controls if the reallocation process
                         % succeeded

                        cnirdb = 0.0;
                        % set the user's CINR to 0


%----------------- control for non allocated channels --------------------%

                        for ch = 1:chnum
                        % for every channel

                            available = 1;
                            % flag: controls if a channel is allocated

                            for other = 1:user
                            % for all the other users in the cell

                                if userinfo(numcell, other, 4) == 1 &...
                                        userinfo(numcell, other, 6) == ch
                                % if any of the other users in the cell is
                                % allocated with the current channel

                                    available  = 0;
                                end
                            end


% ---------------calculation of the new CINR for the new channel----------%


                            if available == 1
                            % if there is a non allocated channel

                                uwave = 0.0;
                                 % initialization of the interference power

                                 for around = 2:7
                                 % calculation of the new CINR
```

```
                    othercell = wrapinfo(numcell, around);
                    for other = 1:user
                        if userinfo(othercell, other, 4)...
                                == 1 & userinfo(...
                                othercell,other, 6) == ch
                            userposi(1, 1:2) =...
                                userinfo(othercell,...
                                other, 1:2);
                            here = baseinfo(numcell, :);
                            there = userposi - baseinfo...
                                (othercell, :)+ baseinfo...
                                (around, :) + baseinfo...
                                (numcell, :);
                            ant_gain = antgain_interf...
                                (here(1, 1), here(1, 2),...
                                userinfo(numcell,...
                                numuser, 1), userinfo...
                                (numcell, numuser, 2),...
                                there(1, 1), there...
                                (1, 2), w_HBS);
                            uwave = uwave + dist(here,...
                                there, alpha) * shadow...
                                (sigma) * ant_gain;
                        end
                    end
                end
                if uwave == 0
                    cnirdb = 10.0 * log10(cn);
                else
                    cnirdb = 10.0 *...
                        log10(1 / (uwave / dwave + 1 /cn));
                end
            else
            % if there isn't a non allocated channel

                cnirdb = 0.0;
            end
            % line 265

            if cnirdb >= cnirth
            % if CINR > threshold

                suceed = 1;
                % the flag is set to 1

                users = users + 1;
                % connected users increased at 1

                userinfo(numcell, numuser, 4) = 1;
                % usage: connected

                userinfo(numcell, numuser, 6) = ch;
                % the allocated channel = ch

                break
                % stop searching for other channels

            end
        end
    end
    % line 265
```

```
                    if suceed == 0
                    % if it there isn't a non allocated channel

                        forcenum = forcenum + 1;
                        % the number of forced terminated calls
                        % increased at 1

                    end

                end
                % line 251

            end
            % line 165

        end
        % line 162

    end
    % line 159


%%%%%%%%%%%%%%%%%%%% STEP 4: NEW CALL ARRIVAL %%%%%%%%%%%%%%%%%%%%%%%%%%%%

    for numcell = 1:19
        for numuser = 1:user
            if userinfo(numcell, numuser, 4) == 0 & rand <= lambda *...
                    (timestep / 3600)
            % usage --> not connected
            % lambda * (timestep / 3600): the user's traffic load in
            % the current loop

                callnum = callnum + 1;
                % the number of generated calls increased at 1

                mesh = floor(meshnum .* rand) + 1;
                % stores a random number between [1 meshnum+1]
                % indicates the position of the connected user in
                % the cell

                while mesh > meshnum
                % generates a new random number until mesh <= meshnum

                    mesh = floor(meshnum .* rand) + 1;
                end

                userinfo(numcell, numuser, 1:2) = baseinfo(numcell,...
                    :) + meshposition(mesh, :);
                % userinfo:     the X and Y user coordinates in
                %               relation from the beginning of axes
                % baseinfo:     the X an Y BS coordinates  in
                %               relation from the beginning of axes
                % meshposition: the X an Y mesh coordinates in
                %               relation from the BS's position

                suceed = 0;
                cnirdb = 0.0;
                userposi(1, 1:2) = userinfo(numcell, numuser, 1:2);
                here = baseinfo(numcell, :);
```

```
                    there = userposi;
                    % in this case it doesn't need the 3 compensating

                    dwave = dist(here, there, alpha) * shadow(sigma);
                    cn = power(10.0, cnedge / 10.0) * dwave;
                    % calculation of the user's CNR


%----------------- searching for a non allocated channel  -----------------%


                    for ch = 1:chnum
                        available = 1;
                        for other = 1:user
                            if userinfo(numcell, other, 4) == 1 &...
                                    userinfo(numcell, other, 6) == ch
                                available = 0;
                            end
                        end
%----------------------- calculation of the interference -----------------%


                        if available == 1
                            uwave = 0.0;
                            for around = 2:7
                                othercell = wrapinfo(numcell, around);
                                for other = 1:user
                                    if userinfo(othercell, other, 4) ==...
                                            1 & userinfo(othercell,...
                                            other, 6) == ch
                                        userposi(1, 1:2) = userinfo...
                                            (othercell, other, 1:2);
                                        here = baseinfo(numcell, :);
                                        there = userposi - baseinfo...
                                            (othercell, :) + baseinfo...
                                            (around, :) + baseinfo...
                                            (numcell, :);
                                        ant_gain = antgain_interf...
                                            (here(1, 1), here(1, 2),...
                                            userinfo(numcell,...
                                            numuser, 1), userinfo...
                                            (numcell, numuser, 2),...
                                            there(1, 1), there(1, 2),...
                                            w_HBS);
                                        uwave = uwave + dist(here,...
                                            there, alpha) *...
                                            shadow(sigma) * ant_gain;
                                    end
                                end
                            end
                            if uwave == 0
                                cnirdb = 10.0 * log10(cn);
                            else
                                cnirdb = 10.0 * log10(1 / (uwave / dwave...
                                    + 1 / cn));
                            end
                        else
                            cnirdb = 0.0;
                        end
```

```
%-------------------- if CINR > CINR(thresold) ---------------------------%

                        if cnirdb >= cnirth
                            suceed = 1;
                            users = users + 1;
                            userinfo(numcell, numuser, 3) = dwave;
                            % the pathloss

                            userinfo(numcell, numuser, 4) = 1;
                            % usage: connected

                            userinfo(numcell, numuser, 5) = timenow +...
                                holdtime(ht);
                            % call termination time: the current simulation
                            % time + the average call holding time


                           userinfo(numcell, numuser, 6) = ch;
                            break
                            % stop searching for channel

                        end
                    end
                    % line 425

                    if suceed == 0
                    % if there isn't a non allocated channel

                        blocknum = blocknum + 1;
                        % the number of blocked calls increased at 1

                    end
                end
                % line 380

            end
            % line 379

        end
        % line 378

%%%%%%%%%%%%%%%%%% STEP 5: OUTPUT PART %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

        fprintf('%d\t%d\t%d\t%d\t%e\n', parameter, timenow, callnum -...
            callnumold, blocknum - blocknumold, blocknum / callnum);
        % in every simulation time loop the values of:
        %         1. the variable parameter   (in decimal notation (%d))
        %         2. the simulation time      (in decimal notation (%d))
        %         3. the new generated calls  (in decimal notation (%d))
        %         4. the new blocked calls    (in decimal notation (%d)))
        %         5. the blocking propability (in exponential notation (%e))


        check_BF(parameter, timenow / timestep + 1) = blocknum / callnum;
        % the current value of blocking propability in every time period
```

```
            check2_BF(parameter, timenow / timestep + 1) = forcenum /...
                (callnum - blocknum);
            % the current value of forced termination propability in every
            % time period

            timenow = timenow + timestep;
            % current simulation time

        end
        % line 114
        output_BF (1, parameter) = callnum;
        % number of generated calls

        output_BF(2, parameter) = blocknum;
        % number of blocked calls

        output_BF(3, parameter) = forcenum;
        % number of forced terminated calls

        output_BF(4, parameter) = blocknum / callnum;
        % blocking propability

        output_BF(5, parameter) = forcenum / (callnum - blocknum);
        % forced termination propability

    end
    % line 63

    fid = fopen('data_BF.txt', 'w');
    % opens the file 'data_BF.txt' for write

    fprintf(fid, 'UserNumber\t');
    % writes the string: 'UserNumber' adding one horizontal tab in the end

    fprintf(fid, '%g\t%g\t%g\t%g\t%g\n', usernum(1,:));
    % writes the values of the matrix usernum  in exponential notation

    fprintf(fid, 'CallNumber\t       ');
    fprintf(fid, '%g\t%g\t%g\t%g\t%g\n', output_BF(1,:));
    % writes the final number of generated calls in exponential notation for
    % evey number of users

    fprintf(fid, 'BlockNumber\t        ');
    fprintf(fid, '%g\t%g\t%g\t%g\t%g\n', output_BF(2,:));
    % writes the final number of blocked calls in exponential notation for
    % evey number of users

    fprintf(fid, 'ForcedNumber\t        ');
    fprintf(fid, '%g\t%g\t%g\t%g\t%g\n', output_BF(3,:));
    % writes the final number of forced terminated calls in exponential
    % notation for evey number of users

    fprintf(fid, 'BlockingProb. \t      ');
    fprintf(fid, '%g\t%g\t%g\t%g\t%g\n', output_BF(4,:));
    % writes the final number of blocking propability in exponential notation
    % for every number of users
```

```
fprintf(fid, 'ForcedTerminationProb. \t');
fprintf(fid, '%g\t%g\t%g\t%g\t%g\n', output_BF(5,:));
% writes the final number of forced termination propability in exponential
% notation for evey number of users

fclose(fid);
% close the file data_BF.txt
```

5.13 antgain_interf.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%% FUNCTION WHICH CALCULATES THE ANTENNA GAIN FOR %%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% THE INTERFERING USERS %%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%



function [out] =  antgain_interf (X_BS, Y_BS, X_user, Y_user,...
    X_otheruser, Y_oheruser, width)
% X_BS -> the X coordinates of the main BS
% Y_BS -> the Y coordinates of the main BS
% X_user -> the X coordinates of the main user
% Y_user -> the Y coordinates of the main user
% X_user -> the X coordinates of the neighbouring user taking into account
%           the cell-wrapping effect
% Y_user -> the Y coordinates of the neighbouring user taking into account
%           the cell-wrapping effect


 g_HBS = antgain(width, -100);
% the antenna gain for every direction

user_coord = X_user + Y_user * j;
% the main user's coordinates in complex form

otheruser_coord = X_otheruser + Y_oheruser * j;
% the neighbouring user's coordinates in complex form

BS_coord = X_BS + Y_BS * j;
% the BS's coordinates in complex form

phai1 = angle(user_coord - BS_coord);
% the angle between the main user and the main BS (radians)

deg1 = phai1 * (180 / pi);
% conversion of radian to degree

phai2 = angle(otheruser_coord);
% the angle between the neigbouring user and the main BS (radians)

deg2 = phai2 * (180 / pi);
% conversion of radian to degree
```

```
deg_B = deg1 - deg2;
% the angle difference between the main and the neigbouring user from the
% main BS

degHBS = mod(round(deg_B), 360);
% adaptation of the angle deg_B to the interval (0,360)

gain = g_HBS(degHBS + 1);
% the antenna gain of the main BS at the neigbouring user's direction

out = 10 .^ ( gain ./ 10);
% conversion of dB -> W
```

5.14 PowerControlCNIR.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CALCULATION OF THE CNIR IMPROVEMENT FOR DIFFERENT VALUES  OF BEAM WIDTH %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%%%%%%%%%%%%%%% STEP 1: VARIABLES DECLARATION %%%%%%%%%%%%%%%%%%%%%%%


output = zeros(2,((120 - 30) / 10) + 1);
% (2 x 10) matrix:
% 1st row stores the current value of CNIR(with beamformer) -
% CNIR(without beamformer) for every value of beam width
% 2nd row stores the current value of CNIR(with beamformer and
% power control) - CNIR(without beamformer and power control) for every
% value of beam width

i = 1;
% loop number indicator

for w_HBS = 30:10:120
% for every value of beam width
% [horizontal] w_HBS beam width at BS for the target direction (degree)

    rand('state', 5);
    randn('state', 1);

    cnedge = 20.0;
    timestep = 10;
    timeend = 5000;

    numcell = 1;
    % the simulation refers to CNIR improvement for one cell
    % (center cell (0, 0))

    alpha = 3.5;
    sigma = 6.5;

    user = 1;
    % number of users in the cell
```

```
h = 0;
% the height of the BS
% the case of macrocell situation

backg_BS = -100;
% [horizontal] antenna gain at BS for the opposite direction (dB)

w_VBS = 360;
% [vertical] beam width at BS (degree)
% case of Omni directional antenna

cnirdb_all = 0;
% initial value of CNIR (without beam forming)

cnirdb_BF_all = 0;
% initial value of CNIR (with beam forming)

cnirdb_PC_all = 0;
% initial value of CNIR (with beam forming and power control)


%%%%%%%%%%%%%%%%%%%%%%%%% STEP 2: INITIALIZATION %%%%%%%%%%%%%%%%%%%%%%%%%


baseinfo = zeros(19, 2);
userinfo = zeros(19, user, 15);
[baseinfo] = basest;
baseinfo_new = baseinfo(: , 1) + baseinfo(: , 2) * j;
% (19 x 1) matrix with the values of every BS's coordinates
% in complex form

[wrapinfo] = wrap;
[meshnum, meshposition] = cellmesh;
timenow = 0;

g_HBS = antgain(w_HBS, backg_BS);
% antenna gain calculation of BS in the horizontal direction

g_VBS = antgain(w_VBS, 0);
% antenna gain calculation of BS in the vertical direction


%%%%%%%% STEP 3: DECLARATION OF THE USER'S POSITION FOR EVERY CELL %%%%%%%%


while timenow < timeend

    for cell = 1:19
        mesh = floor(meshnum .* rand) + 1;
        while mesh > meshnum
            mesh = floor(meshnum .* rand) + 1;
        end
        userinfo(cell, user, 1:2) = baseinfo(cell, :) +...
            meshposition(mesh, :);
        % the X and Y coordinates of every user in relation of
        % the beginning of axes
    end

    userposi = userinfo(: ,1:2);
    userposi_new = userposi(:, 1) + userposi(:, 2) * j;
    % the X and Y coordinates of every user in complex form
```

```
meshposition_new = userposi_new - baseinfo_new;
% the X and Y coordinates of every user in complex form
% in relation of the cell's BS

z = [meshposition_new'; userposi_new'];
% (2 x 19) matrix
% row 1: the coordinates of every user in relation of the cell's BS
%      in complex form
% row 2: the coordinates of every user in relation of the beginning
%      of axes in complex form


%%%%%%%% STEP 4: CALCULATION OF THE ANTENNA GAIN AT THE CENTRAL BS %%%%%%%%


d(1, : ) = abs(z(1, :));
% the magnitude of the distance between every user and its BS
% in horizontal axis

d(2, : ) = abs(z(2, :));
% the magnitude of the distance between every user and
% the central BS in horizontal axis

d2 = sqrt((d .^ 2) + (h .^ 2));
% the distance taking into account the BS's height

phai(1, :) = angle(z(1, :));
% the angle difference between every user and its BS
% (horizontal plane)

phai(2, :) = angle(z(2, :));
% the magnitude of the distance between every user and the central
% BS (horizontal plane)

deg = phai * (180 / pi);
% conversion of radian to degree

deg_B = deg(2, 1) - deg(2, :);
% the angle difference between the main user and every neigbouring
% user from the central BS (horizontal plane)

degH = 90 * ones(1, 19);
% the angle difference between the main user and every neigbouring
% user from the central BS (vertical plane)

degVBS = mod(round(degH - degH(1)), 360);
% the angle difference between MSs and the central BS
% in vertical direction

degHBS = mod(round(deg_B), 360);
% adaptation of deg_B in the interval between 0 and 360

gain = g_HBS(degHBS(1:19) + 1) + g_VBS(degVBS(1:19) + 1);
% antenna gain calculation at the central BS (dB)

gain_W = 10 .^ ( gain ./ 10);
% conversion of dB -> W
```

```
%%%%%%%%%%%%% STEP 5: CALCULATION OF THE CINR AT THE CENTRAL BS %%%%%%%%%%

        for numuser = 1:user
            cnirdb = 0.0;
            % initial value of CNIR without the beamforming

            cnirdb_BF = 0.0;
            % initial value of CNIR with the beamforming

            cnirdb_PC = 0;
            % initial value of CNIR (with beam forming and power control)

            here = baseinfo(numcell, :);
            there = userposi(1, :);
            dwave = dist(here, there, alpha) * shadow(sigma);
            cn = power(10, cnedge / 10.0) * dwave;

            cn_PC = power(10, cnedge / 10.0) * dwave * d(1, 1);
            % user's CNR with power control
            % the distance between the main user and the central BS

            uwave = 0;
            % initial value of interference without the beamforming

            uwave_BF = 0;
            % initial value of interference with the beamforming

            uwave_PC = 0;
            % initial value of interference with the beam forming and power
            % control

            for around = 2:19
            % for every neigbouring cell

                othercell = wrapinfo(numcell, around);
                for other = 1:user
                    here = baseinfo(numcell, :);
                    there = userposi(around, :) - baseinfo(othercell, :)...
                        + baseinfo(around, :) + baseinfo(numcell, :);
                    shadowing = shadow(sigma);
                    % attenuation due to shadowing

                    uwave = uwave + dist(here, there, alpha) *...
                        shadowing;
                    % interference power without the beamforming

                    uwave_BF = uwave_BF + dist(here, there, alpha)...
                        * shadowing * gain_W(1, othercell);
                    % interference power with the beamforming

                    uwave_PC = uwave_PC + dist(here, there, alpha)...
                        * shadowing * gain_W(1, othercell) *...
                        d(1, around);
                    % interference power with the beamforming and power
                    % control
                end
            end
            if uwave == 0
                cnirdb = 10.0 * log10(cn);
            else
```

```
                cnirdb = 10.0 * log10(1 / ((uwave / dwave) + (1 / cn)));
            end

            if uwave_BF == 0
                cnirdb_BF = 10.0 * log10(cn);
            else
                cnirdb_BF = 10.0 * log10(1 /((uwave_BF /dwave) + (1 /cn)));
            end
            if uwave_PC == 0
                cnirdb_PC = 10.0 * log10(cn_PC);
            else
                cnirdb_PC = 10.0 * log10(1 / ((uwave_PC / dwave) +...
                    (1 / cn_PC)));
            end
        end
        cnirdb_all = cnirdb_all + cnirdb;
        % the total value of CNIR without beamforming in the current loop

        cnirdb_BF_all = cnirdb_BF_all + cnirdb_BF;
        % the total value of CNIR with beamforming in this current loop

        cnirdb_PC_all = cnirdb_PC_all + cnirdb_PC;
        % the total value of CNIR with beamforming and power control
        % in this current loop

        timenow = timenow + timestep;
    end

    output(1, i) = (cnirdb_BF_all - cnirdb_all) / (timeend / timestep);
    % the average value of the difference between cnirdb_BF and cnirdb

    output(2, i) = (cnirdb_PC_all - cnirdb_all) / (timeend / timestep);
    % the average value of the difference between cnirdb_BF and cnirdb

    i = i + 1;
   % next step of the beam width loop

end

plot(30:10:120, output(1, :), '.-b', 30:10:120, output(2, :), '.-r')
axis([25 125 0 18])
xlabel('Beam Width (degrees)');
ylabel('CNIR improvement (dB)');
grid on;
```

5.15 DCAPowerControl.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%% SIMULATION PROGRAM FOR DCA ALGORITHM %%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%% APPLYING THE BEAMFORMING TECHNIQUE AND %%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% POWER CONTROL %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%%%%% STEP 1: VARIABLES DECLARATION & INITIALIZATION %%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


cnedge = 20.0;
% Carrier to Noise ratio on cell edge

cnirth = 15.0;
% CNIR threshold

lambda = 6.0;
% average call arrival rate (times/hour)

ht = 120.0;
% average call holding time (sec)

timestep = 10;
% timestep of the simulation loop (seconds)

timeend = 5000;
% maximum simulation time

chnum = 5;
% number of available channels per each base station

alpha = 3.5;
% path loss factor

sigma = 6.5;
% standard deviation of shadowing

w_HBS = 60;
% antenna's beamwidth

usernum = [5, 10, 15, 20, 25];
% number of users per cell

output_PC = zeros(5, 5);
% lines:    1: number of generated calls  2: number of blocking calls
%           3: blocking propability       4: forced termination propability
% columns: 1-5: results for usernum 5, 10, 15, 20 and 25 respectively

check_PC = zeros(5, floor(timeend / timestep));
% stores the current value of blocking propabilty in every time period
% lines 1-5:    number of users: 5, 10, 15, 20 and 25 respectively

check2_PC = zeros(5, floor(timeend / timestep));
% stores the current value of forced termination propabilty
% in every time period
```

```
% lines 1-5:    number of users: 5, 10, 15, 20 and 25 respectively

for parameter = 1:5
% set the number of users per cell

    rand('state', 5);
    % initialize the internal state of the generator to 5

    randn('state', 1);
    % initialize the internal state of the generator to 1

    user = usernum(parameter);
    % number of users per cell

    baseinfo = zeros(19, 2);
    % initialization of the baseinfo matrix
    % baseinfo(cell #,1) x coordinates
    % baseinfo(cell #,2) y coordinates

    userinfo = zeros(19, user, 6);
    % initialization of the userinfo matrix
    % userinfo(cell #, user #, information)
    % 1: x axis  2: y axis  % 3:attenuation
    % 4:usage 0->non connected 1->connected
    % 5: call termination time  6: allocated channel

    [baseinfo] = basest;
    % set the position for each of the base stations

    [wrapinfo] = wrap;
    % determination of the neigbouring cells for evey cell

    [meshnum, meshposition] = cellmesh;
    % set the position of the users in every cell

    timenow = 0;
    % set the initial value of the time simulation loop

    blocknum = 0;
    % set the initial value for the number of blocked calls

    forcenum = 0;
    % set the initial value for the number of forced terminated calls

    callnum = 0;
    % set the initial value for the number of generated calls

    users = 0;
    % set the initial value for the number of connected users

    while timenow < timeend

        callnumold = callnum;
        % the new value of generated calls in the current loop

        blocknumold = blocknum;
        % the new value of blocked calls in the current loop

        forcenumold = forcenum;
        % the new value of forced terminated calls in the current loop
```

```
%%%%%%%%%%%%%%%%%%%%%%%%% STEP 2: FINISHED CALLS %%%%%%%%%%%%%%%%%%%%%%%%%

        for numcell = 1:19
        % for every cell

            for numuser = 1:user
            % for every user in the cell

                if userinfo(numcell, numuser, 4) == 1 & userinfo(numcell...
                       ,numuser, 5) < timenow
                % userinfo(:,:,4)--> the usage: connected
                % userinfo(:,:,5)--> call termination time

                    userinfo(numcell, numuser, 4) = 0;
                    % the user is not connected any more

                    users = users - 1;
                    % the number of connected users decreased at 1
                end
            end
        end


%%%%%%%%%%%%%%%%%%%%%%%%% STEP 3: REALLOCATION CHECK %%%%%%%%%%%%%%%%%%%%%%%%%


        for numcell = 1:19
        % for every cell

            for numuser = 1:user
            % for every user in the cell

                if userinfo(numcell, numuser, 4) == 1
                % if the user is connected

                    reallo = 0;
                    % flag: controls if reallocation is needed

                    cnirdb = 0.0;
                    % initialization of the user's CINR

                    dwave = userinfo(numcell, numuser, 3);
                    % pathloss  attenuation of the user

                    distance = abs(userinfo(numcell, numuser, 1) -...
                        baseinfo(numcell, 1) + (userinfo(numcell,...
                        numuser, 2) - baseinfo(numcell, 2)) * j);
                    % the distance between the main user and the main BS
                    % the magnitude of the complex number userinfo(numcell,
                    % numuser, :) - baseinfo(numcell, :)

                    cn = power(10.0, cnedge / 10.0) * distance * dwave;
                    % the received user's CNR applying the power control

                    uwave = 0.0;
                    % initialization of the interference power
```

```
ch = userinfo (numcell, numuser, 6);
% stores the number of the channel which allocated to
% the user

for around = 2:7
% for every neigbouring cell

    othercell = wrapinfo(numcell, around);
    % stores the number of the neigbouring cell

    for other = 1:user
    % for every user in the neigbouring cell

        if userinfo(othercell, other, 4) == 1 ...
            & userinfo(othercell, other, 6) == ch
        % if the neigbouring user is connected and
        % allocated with the same channel

            userposi(1, 1:2) = ...
                userinfo(othercell, other, 1:2);
            % matrix with the coordinates of the
            % neighbouring user

            here = baseinfo(numcell, :);
            % matrix with the coordinates of the user's
            % BS

            there = userposi - baseinfo(othercell,...
                :) + baseinfo(around, :) +...
                baseinfo(numcell, :);
            % the distance between the neigbouring user
            % and the user's BS taking into account the
            % cell wrapping effect

            distance = abs(userposi(1) - baseinfo...
                (othercell, 1) + (userposi(2) -...
                baseinfo(othercell, 2)) * j);
            % the distance between the other user and
            % the other user's BS

            ant_gain = antgain_interf (here(1, 1),...
                here(1, 2), userinfo(numcell,...
                numuser, 1), userinfo(numcell,...
                numuser, 2), there(1, 1),...
                there(1, 2), w_HBS);
            % the antenna gain of the user's BS at the
            % neigbouring users direction

            uwave = uwave + dist(here, there, alpha)...
                * shadow(sigma) * distance * ant_gain;
            % the total power of the interference
            % taking into account the beamforming
            % technique and the power control

        end
    end
end

if uwave == 0
% interference power = 0
```

```
                    cnirdb = 10.0 * log10(cn);

                else
                    cnirdb = 10.0 * log10(1 / (uwave / dwave + 1 /cn));
                    % the received CINR from the user

                end


%------------------------- reallocation process -------------------------%


                if cnirdb < cnirth
                % if the received CINR is lower than the thresold

                    reallo = 1;
                    % needs reallocation

                end

                if reallo == 1
                    userinfo(numcell, numuser, 4) = 0;
                    % user's usage: not connected

                    users = users - 1;
                    % connected users decreased at 1

                    suceed = 0;
                     % flag: controls if the reallocation process
                     % succeeded

                    cnirdb = 0.0;
                    % set the user's CINR to 0


%------------------ control for non allocated channels --------------------%


                    for ch = 1:chnum
                    % for every channel

                        available = 1;
                        % flag: controls if a channel is allocated

                        for other = 1:user
                        % for all the other users in the cell

                            if userinfo(numcell, other, 4) == 1 &...
                                    userinfo(numcell, other, 6) == ch
                            % if any of the other users in the cell is
                            % allocated with the current channel

                                available  = 0;
                            end
                        end
```

```
% ---------------calculation of the new CINR for the new channel----------%


                        if available == 1
                        % if there is a non allocated channel

                            uwave = 0.0;
                             % initialization of the interference power

                             for around = 2:7
                             % calculation of the new CINR

                                othercell = wrapinfo(numcell, around);
                                for other = 1:user
                                    if userinfo(othercell, other, 4)...
                                            == 1 & userinfo(...
                                            othercell,other, 6) == ch
                                        userposi(1, 1:2) =...
                                            userinfo(othercell,...
                                            other, 1:2);
                                        here = baseinfo(numcell, :);
                                        there = userposi - baseinfo...
                                            (othercell, :)+ baseinfo...
                                            (around, :) + baseinfo...
                                            (numcell, :);
                                        distance = abs(userposi(1) -...
                                            baseinfo(othercell, 1) +...
                                            (userposi(2) - baseinfo...
                                            (othercell, 2)) * j);
                                        ant_gain = antgain_interf...
                                            (here(1, 1), here(1, 2),...
                                            userinfo(numcell,...
                                            numuser, 1), userinfo...
                                            (numcell, numuser, 2),...
                                            there(1, 1), there...
                                            (1, 2), w_HBS);
                                        uwave = uwave + dist(here,...
                                            there, alpha) * shadow...
                                            (sigma) * distance *...
                                            ant_gain;
                                    end
                                end
                            end
                            if uwave == 0
                                cnirdb = 10.0 * log10(cn);
                            else
                                cnirdb = 10.0 *...
                                    log10(1 / (uwave / dwave + 1 /cn));
                            end
                        else
                        % if there isn't a non allocated channel

                            cnirdb = 0.0;
                        end
                        % line 265

                        if cnirdb >= cnirth
                        % if CINR > threshold

                            suceed = 1;
                            % the flag is set to 1
```

```
                                users = users + 1;
                                % connected users increased at 1

                                userinfo(numcell, numuser, 4) = 1;
                                % usage: connected

                                userinfo(numcell, numuser, 6) = ch;
                                % the allocated channel = ch

                                break
                                % stop searching for other channels

                            end
                        end
                        % line 293

                        if suceed == 0
                        % if it there isn't a non allocated channel

                            forcenum = forcenum + 1;
                            % the number of forced terminated calls
                            % increased at 1

                        end

                    end
                   % line 275

                end
                % line 167

            end
            % line 164

        end
        % line 161


%%%%%%%%%%%%%%%%%%%% STEP 4: NEW CALL ARRIVAL %%%%%%%%%%%%%%%%%%%%%%%%%%%%


        for numcell = 1:19
            for numuser = 1:user
                if userinfo(numcell, numuser, 4) == 0 & rand <= lambda *...
                        (timestep / 3600)
                % usage --> not connected
                % lambda * (timestep / 3600): the user's traffic load in
                % the current loop

                    callnum = callnum + 1;
                    % the number of generated calls increased at 1

                    mesh = floor(meshnum .* rand) + 1;
                    % stores a random number between [1 meshnum+1]
                    % indicates the position of the connected user in
                    % the cell
```

```
                while mesh > meshnum
                % generates a new random number until mesh <= meshnum
                    mesh = floor(meshnum .* rand) + 1;
                end

                userinfo(numcell, numuser, 1:2) = baseinfo(numcell,...
                    :) + meshposition(mesh, :);
                % userinfo:     the X and Y user coordinates in
                %               relation from the beginning of axes
                % baseinfo:     the X an Y BS coordinates  in
                %               relation from the beginning of axes
                % meshposition: the X an Y mesh coordinates in
                %               relation from the BS's position

                suceed = 0;
                cnirdb = 0.0;
                userposi(1, 1:2) = userinfo(numcell, numuser, 1:2);
                here = baseinfo(numcell, :);
                there = userposi;
                % in this case it doesn't need the 3 compensating

                distance = abs(there(1) - here(1) + (there(2) -...
                    here(2)) * j);

                dwave = dist(here, there, alpha) * shadow(sigma);
                cn = power(10.0, cnedge / 10.0) * distance * dwave;
                % calculation of the user's CNR


%----------------- searching for a non allocated channel  ----------------%


                for ch = 1:chnum
                    available = 1;
                    for other = 1:user
                        if userinfo(numcell, other, 4) == 1 &...
                                userinfo(numcell, other, 6) == ch
                            available = 0;
                        end
                    end


%---------------------- calculation of the interference -----------------%


                    if available == 1
                        uwave = 0.0;
                        for around = 2:7
                            othercell = wrapinfo(numcell, around);
                            for other = 1:user
                                if userinfo(othercell, other, 4) ==...
                                        1 & userinfo(othercell,...
                                        other, 6) == ch
                                    userposi(1, 1:2) = userinfo...
                                        (othercell, other, 1:2);
                                    here = baseinfo(numcell, :);
                                    there = userposi - baseinfo...
                                        (othercell, :) + baseinfo...
                                        (around, :) + baseinfo...
                                        (numcell, :);
                                    distance = abs(userposi(1) -...
```

```
                                        baseinfo(othercell, 1) +...
                                        (userposi(2) - baseinfo...
                                        (othercell, 2)) * j);
                                    ant_gain = antgain_interf...
                                        (here(1, 1), here(1, 2),...
                                        userinfo(numcell,...
                                        numuser, 1), userinfo...
                                        (numcell, numuser, 2),...
                                        there(1, 1), there(1, 2),...
                                        w_HBS);
                                    uwave = uwave + dist(here,...
                                        there, alpha) *...
                                        shadow(sigma) * distance * ...
                                        ant_gain;
                                end
                            end
                        end
                        if uwave == 0
                            cnirdb = 10.0 * log10(cn);
                        else
                            cnirdb = 10.0 * log10(1 / (uwave / dwave...
                                + 1 / cn));
                        end
                    else
                        cnirdb = 0.0;
                    end

%-------------------- if CINR > CINR(threshold) --------------------------%

                    if cnirdb >= cnirth
                        suceed = 1;
                        users = users + 1;
                        userinfo(numcell, numuser, 3) = dwave;
                        % the pathloss

                        userinfo(numcell, numuser, 4) = 1;
                        % usage: connected

                        userinfo(numcell, numuser, 5) = timenow +...
                            holdtime(ht);
                        % call termination time: the current simulation
                        % time + the average call holding time

                        userinfo(numcell, numuser, 6) = ch;
                        break
                        % stop searching for channel

                    end
                end
                % line 468

                if suceed == 0
                % if there isn't a non allocated channel

                    blocknum = blocknum + 1;
                    % the number of blocked calls increased at 1

                end
            end
```

```
                % line 420

            end
            % line 419

        end
        % line 418


%%%%%%%%%%%%%%%%%%%% STEP 5: OUTPUT PART %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


        fprintf('%d\t%d\t%d\t%d\t%e\n', parameter, timenow, callnum -...
            callnumold, blocknum - blocknumold, blocknum / callnum);
        % in every simulation time loop the values of:
        %         1. the variable parameter    (in decimal notation (%d))
        %         2. the simulation time       (in decimal notation (%d))
        %         3. the new generated calls   (in decimal notation (%d))
        %         4. the new blocked calls     (in decimal notation (%d)))
        %         5. the blocking propability (in exponential notation (%e))

        check_BF(parameter, timenow / timestep + 1) = blocknum / callnum;
        % the current value of blocking propability in every time period

        check2_BF(parameter, timenow / timestep + 1) = forcenum /...
            (callnum - blocknum);
        % the current value of forced termination propability in every
        % time period

        timenow = timenow + timestep;
        % current simulation time

    end
    % line 116

    output_PC (1, parameter) = callnum;
    % number of generated calls

    output_PC(2, parameter) = blocknum;
    % number of blocked calls

    output_PC(3, parameter) = forcenum;
    % number of forced terminated calls

    output_PC(4, parameter) = blocknum / callnum;
    % blocking propability

    output_PC(5, parameter) = forcenum / (callnum - blocknum);
    % forced termination propability

end
% line 65

fid = fopen('data_PC.txt', 'w');
% opens the file 'data_PC.txt' for write

fprintf(fid, 'UserNumber\t');
% writes the string: 'UserNumber' adding one horizontal tab in the end
```

```
fprintf(fid, '%g\t%g\t%g\t%g\t%g\n', usernum(1,:));
% writes the values of the matrix usernum  in exponential notation

fprintf(fid, 'CallNumber\t        ');
fprintf(fid, '%g\t%g\t%g\t%g\t%g\n', output_PC(1,:));
% writes the final number of generated calls in exponential notation for
% evey number of users

fprintf(fid, 'BlockNumber\t        ');
fprintf(fid, '%g\t%g\t%g\t%g\t%g\n', output_PC(2,:));
% writes the final number of blocked calls in exponential notation for
% evey number of users

fprintf(fid, 'ForcedNumber\t        ');
fprintf(fid, '%g\t%g\t%g\t%g\t%g\n', output_PC(3,:));
% writes the final number of forced terminated calls in exponential
% notation for evey number of users

fprintf(fid, 'BlockingProb. \t      ');
fprintf(fid, '%g\t%g\t%g\t%g\t%g\n', output_PC(4,:));
% writes the final number of blocking propability in exponential notation
% for every number of users

fprintf(fid, 'ForcedTerminationProb. \t');
fprintf(fid, '%g\t%g\t%g\t%g\t%g\n', output_PC(5,:));
% writes the final number of forced termination propability in exponential
% notation for evey number of users

fclose(fid);
% close the file data_PC.txt
```

Βιβλιογραφία

1. Fundamentals of WiMax: Understanding Broadband Wireless Networking, by Jeffrey G.Andrews, Arunabha Grosh, Rias Muhamed, Prentice Hall PTR, 1st edition (March 9, 2007).
2. Digital Cellular Radio, by Calhoun, G., Artech House Inc., 1988.
3. Oeting, J., "Cellular Mobile Radio – An Emerging Technology," IEEE Communications Magazine, pp.10-15, November 1983.
4. MacDonald, V. H., "The Cellular Concept," The Bell Systems Technical Journal, Vol.58, No.1, pp. 15-43, January 1979.
5. Simulation and Software Radio for Mobile Communications, by Hiroshi Harada and Ramjee Prasad, Artech House Publishers; 1st edition (May 1, 2002).
6. http://en.wikipedia.org/wiki/Dynamic_Frequency_Selection
7. A dynamic channel assignment simulation system for large scale cellular Telecommunications: P.M.Papazoglou, D.A.Karras, R.C.Papademetriou, http://www.aueb.gr/pympe/hercma/proceedings2005/H05-FULL-PAPERS-1/PAPAZOGLOU-KARRAS-PAPADEMETRIOU-1.pdf
8. Dynamic Channel Allocation, by Richard Sutton, May 1997, http://www.cs.ualberta.ca/~sutton/book/ebook/node112.html
9. Reinforcement Learning for Dynamic Channel Allocation in Cellular Telephone Systems, by Satinder Singh and Dimitri Bertsekas, http://www.cis.upenn.edu/~mkearns/papers/barbados/sb-channel.pdf
10. Tekinay, S., and Jabbari, B., "Handover and Channel Assignment in Mobile Cellular Networks," IEEE Communications Magazine, pp. 42-46, November 1991.
11. http://en.wikipedia.org/wiki/Beamforming