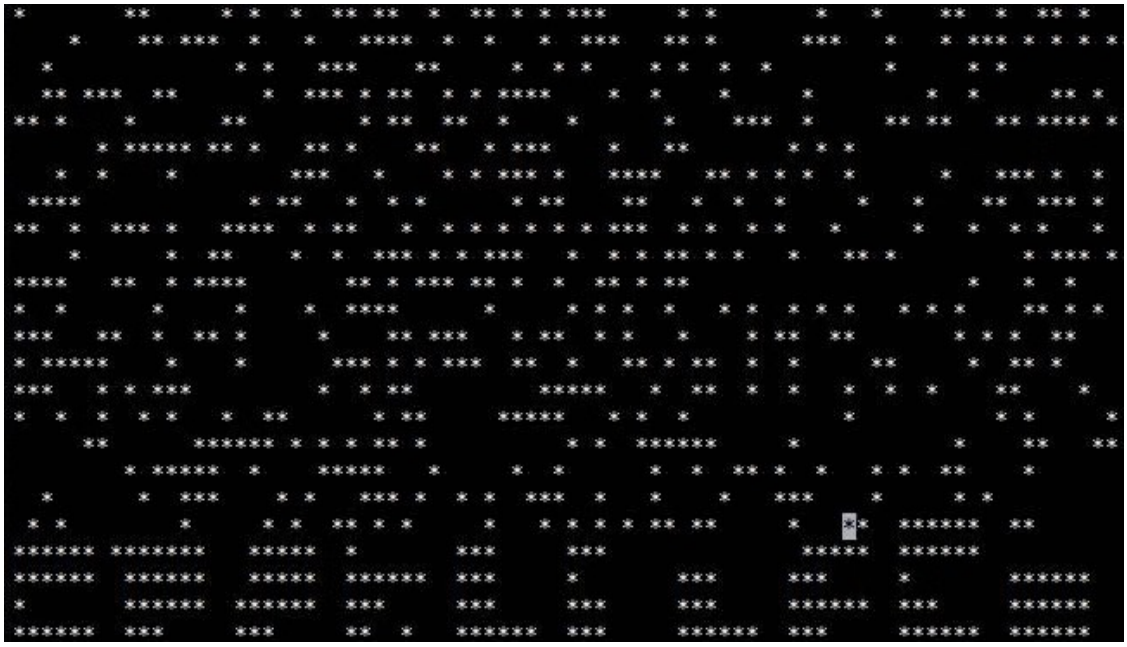
GAME-OF-LIFE

Παράλληλα Και Διανεμημένα Συστήματα
Εργασία 2 (Χρήση της διεπαφής MPI)



Μπεκιάρης Θεοφάνης ΑΕΜ:8200

Περιγραφή Παιχνιδιού

Το παιχνίδι προσομοιώνει ένα κυψελικό αυτόματο του οποίου η εξέλιξη εξαρτάται μόνο από τις αρχικές συνθήκες.

Οι βασικοί κανόνες που διέπουν την εξέλιξη του παιχνιδιού είναι:

1. Το παιχνίδι παίζεται σε ένα ορθογώνιο πλέγμα, όπου το κάθε κελί μπορεί να έχει δύο καταστάσεις, ζωντανό και μη ζωντανό. Οι καταστάσεις των κελιών μεταβάλλονται σε κάθε επανάληψη του παιχνιδιού (γενιά).

2. Stasis: Κάθε κελί με ακριβώς δύο ζωντανούς γείτονες παραμένει στην ίδια κατάσταση στην επόμενη γενιά. Αν ήταν ζωντανό, παραμένει ζωντανό και αντίστροφα.

3. Growth: Κάθε κελί με ακριβώς τρεις ζωντανούς γείτονες θα είναι ζωντανό στην επόμενη γενιά, ανεξάρτητα από την τωρινή του κατάσταση.

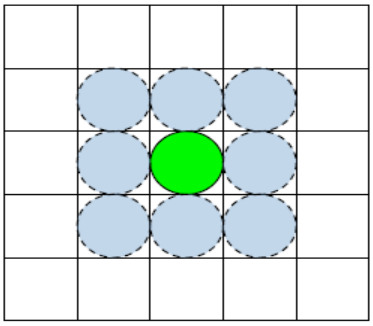
4. Death:Σε οποιαδήποτε άλλη περίπτωση το κελί θα είναι νεκρό στην επόμενη γενιά.

Απαιτήσεις Εργασίας

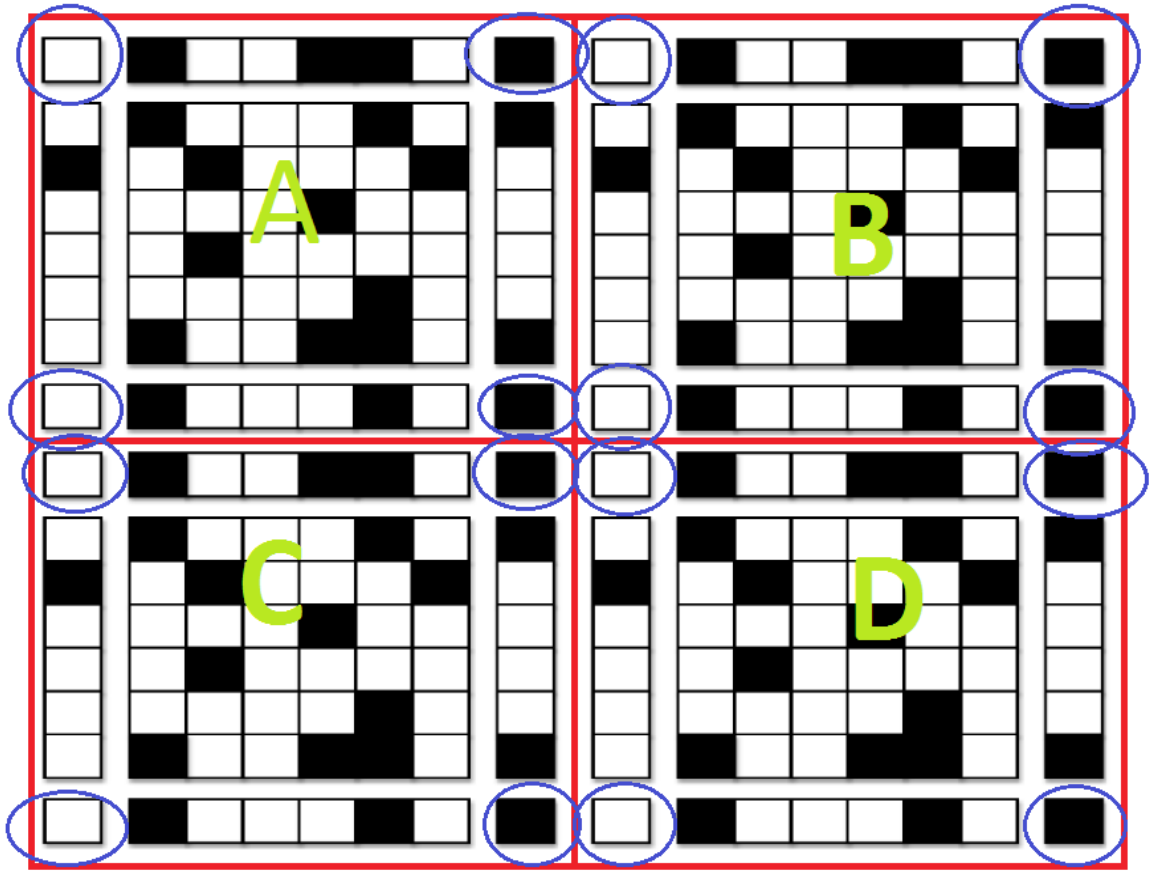
Η εργασία απαιτεί υλοποίηση κώδικα σε MPI, ώστε να χωρίζεται ο αρχικός πίνακας σε πλέγμα (ανάλογα με το πλήθος των διεργασιών) και να ανατίθεται κάθε τμήμα του πλέγματος σε διαφορετικές διεργασίες, χρησιμοποιώντας σε κάθε υποτμήμα υλοποίηση σε OpenMP.

Λογική Υλοποίησης

Το παιχνίδι αποτελείται απο ένα πίνακα με κελιά.Όπως μπορούμε να παρατηρήσουμε κάθε κελί έχει 8 γείτονες απο τους οποιούς εξαρτάται η επόμενη κατάσταση του(ζωντανό ή νεκρό).

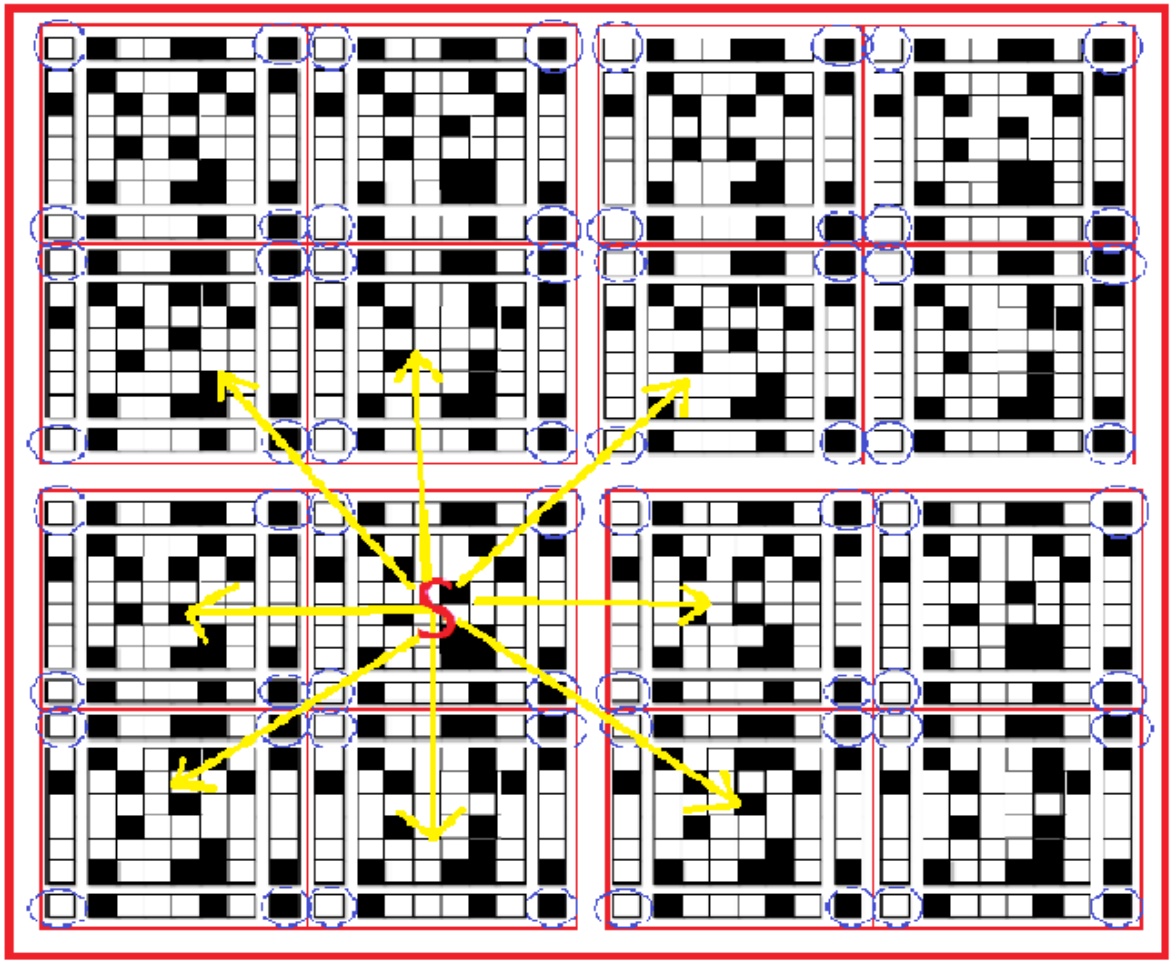


Κατα την διαδικασία παραλληλοποίησης ο αρχικός πίνακας θα πρέπει να χωριστεί και να ανατεθεί το αντίστοιχο κομμάτι του σε κάθε μία απο τις διεργασίες που θα κλιθούν.Το πρώτο πρόβλημα το οποίο συναντάμε είναι η απόφαση για τον πώς θα γίνει ο διαχωρισμός του πίνακα.Όπως αναφέρθηκε και προηγουμένως η κατάσταση ενός κελιού εξαρτάται απο τα γειτονικά του κελιά,με αποτέλεσμα οι διεργασίες θα πρέπει να ανταλάσουν μεταξύ τους δεδομένα για ενημερώνονται για την κατάσταση των κελιών που βρίσκονται στα όρια του διαχωρισμού του αρχικού πίνακα. Ο διαχωρισμός δεδομένου ότι το παιχνίδι χρησιμοποιεί κυκλικές οριακές συνθήκες,θα πρέπει να γίνει με τέτοιο τρόπο ώστε να ελαχιστοποιούνται οι επικοινωνίες μεταξύ των διεργασιών.Ας υποθέσουμε ότι ο πίνακας θα χωριστεί σε τετράγωνα και ανατεθεί κάθε κομμάτι σε μία διεργασία(A,B,C,D) όπως φαίνεται παρακάτω.



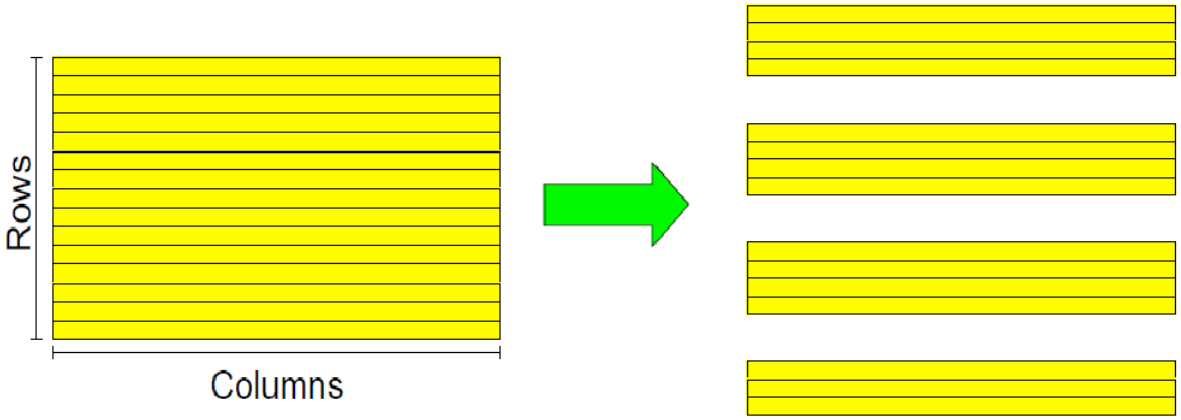
Ας χρησιμοποιήσουμε ως παράδειγμα την περίπτωση της διεργασίας A. Η διεργασία A για να ξέρει την επόμενη κατάσταση των κελιών που βρίσκονται στα όρια του πίνακα που της αντιστοιχεί,θα πρέπει να ξέρει και την κατάσταση των οριακών κελιών των διεργασιών B και C όπως ειναι φανερό,επιπλέον όμως λόγο των κελιών που βρίσκονται στις γωνίες(κυκλωμένα),θα πρέπει να ανταλάξει δεδομένα και με την

διεργασία D.Βλέπουμε δηλαδή ότι σε αυτή την περίπτωση θα πρέπει όλες οι διεργασίες να ανταλάξουν μεταξύ τους δεδομένα και σε μία πιο γενική περίπτωση με περισσότερες διεργασίες και με περισσότερες τομές του αρχικού πίνακα θα έπρεπε η κάθε διεργασία να ανταλάση δεδομένα τουλάχιστον με άλλες 8 διεργασίες όπως απεικονίζεται και στο παρακάτω σχήμα για την διεργασία S.

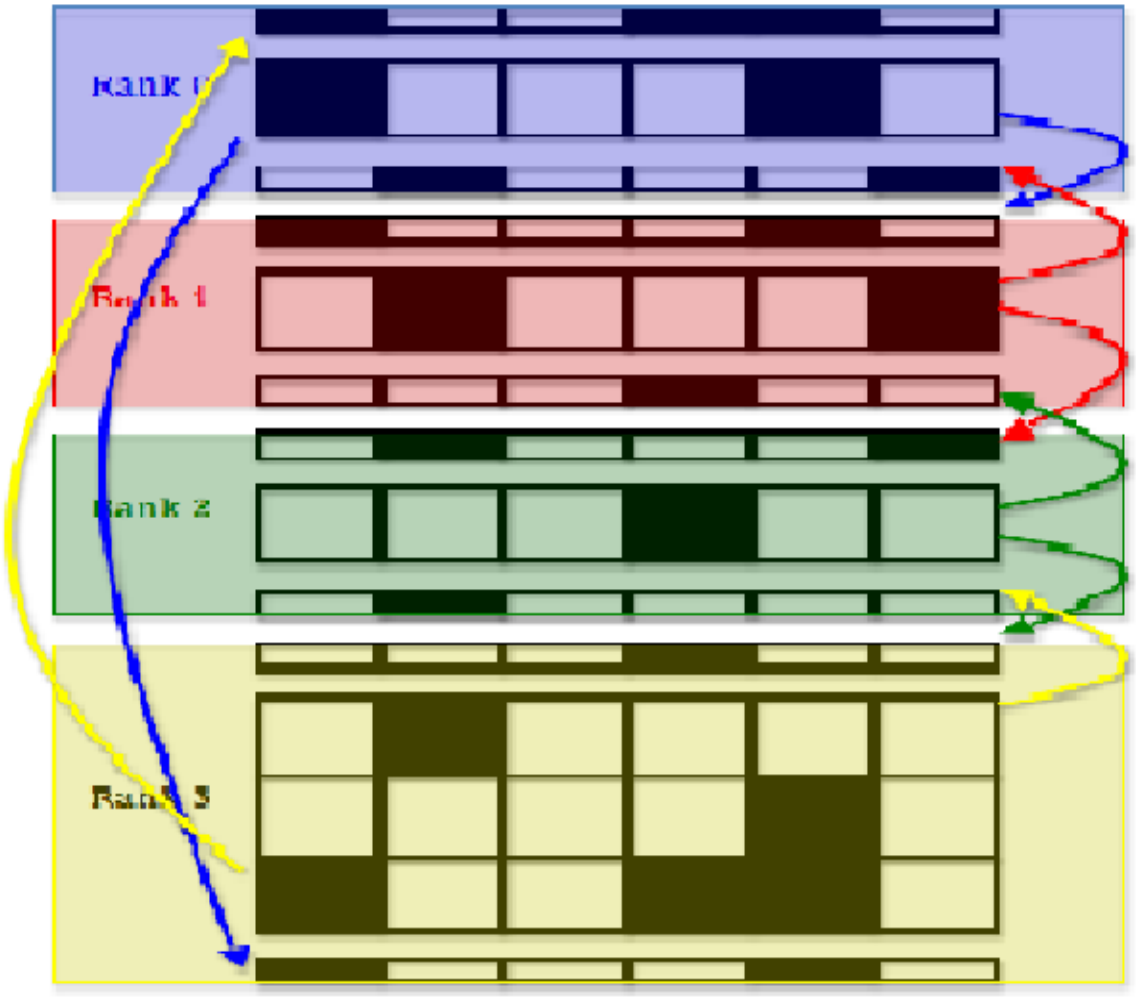


Δεδομένου των παραπάνω η επιλογή που έκανα για να διαχωρίσω τον αρχικό πίνακα είναι να τον χωρίσω σε ορθογώνιες λωρίδες καθώς με αυτόν τον τρόπο περιορίζουμε τις επικοινωνίες μεταξύ των 8 γειτώνων σε επικοινωνίες μεταξύ 2 γειτόνων.Σε αυτή την περίπτωση οι γειτονικές διεργασίες θα είναι υποχρεωμένες να ανταλλάξουν 2 ολόκληρες γραμμές μεταξύ τους επιπλέον όμως δεν είναι απαραίτητη η ανταλλαγή στηλών,δηλαδή περιορίζεται και ο συνολικός όγκος δεδομένων προς μεταφορά.

Χωρίζουμε τον αρχικό πίνακα σε ορθογώνιες λωρίδες



Βλέπουμε τελικά οτι οι επικοινωνίες περιορίζονται σε 2,δηλαδή κάθε διεργασία επικοινωνεί μονάχα με της 2 γειτονικές της και ανταλλάσει μόνο τις γραμμές των ορίων της.



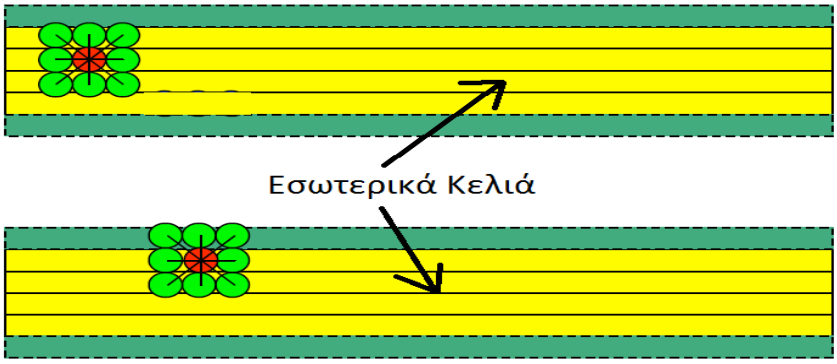
Η υλοποίηση επομένως έχει την εξής λογική:

1)Εκτελούμε το πρόγραμμα με τον επιθυμητό αριθμό διεργασιών με τα απαραίτητα δεδομένα(π.χ διαστάσεις αρχικού πίνακα,αριθμό γενεών κλπ).

2)Το πρόγραμμα(συγκεκριμένα η κάθε διεργασία)διαβάζει των αριθμό των διεργασιών και υπολογίζει τον αριθμό των γραμμών που αντιστοιχεί σε κάθε διεγρασία.Αν η διαίρεση δεν είναι ακριβής η MASTER διεγρασία θα προσαρμόσει κατάλληλα τον αριθμό των γραμμών της.

3)Η κάθε διεργασία δημιουργεί ένα δισδιάστατο πίνακα με αριθμό στηλών ίσο με τον αρχικό πίνακα και αριθμό γραμμών ίσο με αυτό που της αντιστοιχεί σύμφωνα με το προηγούμενο βήμα,και δίνει τυχαίες τιμές στα κελιά του πίνακα.

4)Αφού δημιουργηθούν οι πίνακες,οι διεργασίες σε κάθε γενιά θα πρέπει να ανταλλάζουν ανα 2 τις οριακές γραμμές.Ο στόχος είναι οι επικοινωνίες που θα γίνονται να μην καθυστερούν την εκτέλεση της κάθε διεργασίας, δηλαδή θα πρέπει να "κρύβουμε" τον χρόνο που απαιτείται για την ανταλλαγή δεδομένων βάζοντας τις διεργασίες να εκτελούν κάποια δουλεία μέχρι να γίνει ανταλλαγή των δεδομένων.Τα κελιά που βρίσκονται στο εσωτερικό των πινάκων,δηλαδή όλα τα κελιά εκτώς της πρώτης και τελευταίας γραμμής,μπορούν να υπολογίσουν την επόμενη κατάσταση τους χωρίς να χρειάζονται να περιμένουν τα δεδομένα απο τις άλλες διεργασίες.



Επομένως αρχικά εκτελούμε αιτήματα για να γίνει αποστολή και παραλαβή δεδομένων και χωρίς να περιμένουμε προχωράμε και εκτελούμε τους απαραίτητους υπολογισμούς για τα εσωτερικά κελιά. Οταν τελειώσουμε αν δεν έχουν έρθει τα δεδομένα περιμένουμε να έρθουν (λογικά όμως περιμένουμε να έχουν σταλεί μέχρι τότε) και όταν τελικά τα παραλάβουμε κάνουμε τους αντίστοιχους υπολογισμούς για την πρώτη και την τελευταία γραμμή.

5)Τέλος επαναλαμβάνουμε το βήμα 4 για όλες τις γενιές.

Κώδικας

Στο κομμάτι της γραφής του κώδικα οι συναρτήσεις που έχουν τροποποιηθεί κυρίως ειναι η main και η play,στης υπόλοιπες έχουν γίνει απλώς οι απαραίτητες τροποποιήσεις για την συμβατότητα τους στο προγράμμα.Τέλος έχουν ορίστει και κάποιες επιπλέον απαραίτητες συναρτήσεις adjacent_toHelpFunction() και IamTheOnlyPlayer().

Main: Στην main η κάθε διεργασία ορίζει τον δικό της πίνακα board που αποτελεί στην ουσία κομμάτι του αρχικού ενιαίου πίνακα.Στην συνέχεια μέσα απο την generate_table() οι διεργασίες δίνουν τυχαίες τιμές στα κελιά του πίνακα και στην συνέχεια με μη-ανασταλτικές εντολές στέλνουν και ζητούν τα απαραίτητα δεδομένα και συνεχίζουν καλώντας την αντίστοιχη συνάρτηση play.

Play:Η συνάρτηση υπολογίζει την επόμενη κατάσταση των κελιών αρχικά για τα όλες τις γραμμές εκτώς τις οριακές και όταν τελειώσει υπολογίζει και για τις οριακές δεδομένου οτι τα δεδομένα έχουν έρθει,αλλιώς περιμένει να έρθουν.

Παραλληλοποίηση με OpenMp:

Παραλληλοποίηση με openMp έχει γίνει στην συνάρτηση generate_table() και στην συνάρτηση play().Να σημειωθεί οτι στην συνάρτηση generate_table() έχει αντικατασταθεί η συνάρτηση rand() που δεν ειναι thread safe και παράγει μεγάλες καθυστερίσεις στο πρόγραμμα σε περίπτωση παραλληλοποίησης,με την rand_r() όπου ειναι thread safe και μπορεί να χρησιμοποιηθεί σε παράλληλη περιοχή.

Παρατήρηση:Το πρόγραμμα έχει γραφτεί με τέτοιο τρόπο ώστε για οποιοδήποτε αριθμό διεργασιών ζητηθούν(ορισθούν) να χωρίζει κατάλληλα τον αριθμό των γραμμών του αρχικού πίνακα και να τον ανατεθεί σε κάθε μία απο τις διεργασίες,δηλαδή το πρόγραμμα τρέχει για οποιοδήποτε αριθμό διεργασιών.Η εκφώνηση όμως ζητάει συγκεκριμένο αριθμό διεργασιών με συγκερκιμένες διαστάσεις,ώστε να μπορεί να γίνει σύγκριση των αποτελεσμάτων για τον χρόνο εκτέλεσης μεταξύ των 3 περιπτώσεων που ζητούνται.Συγκεκριμένα όλες οι διεργασίες θα πρέπει να έχουν ίδιο αριθμό στοιχείων μεγέθους 40000x40000.Στην περίπτωση των 2 διεργασιών μέσα απο τον κώδικα γίνεται άμεσα η ανάθεση των ζητούμενων διαστάσεων στους πίνακες των διεργασιών αφού ο αρχικός πίνακας δεν έιναι τετραγωνικός και έχει διαστάσεις 80000x40000 και οι πίνακες των διεργασιών θα πρέπει να έχουν διαστάσεις 40000x40000.Στην

περίπτωση των 4 διεργασιών οι πίνακες θα έχουν δίασταση 80000x20000,δεν γίνεται όμως κάποια τροποποίηση σε αυτη την περίπτωση αφού και πάλι ο αριθμός των στοιχείων θα είναι ο ίδιος 80000x20000=40000x40000.

Αποτελέσματα

Παρακάτω παρατίθενται οι χρόνοι απο την εκτέλεση του προγράμματος στο Hellasgrid για τα αντίστοιχα δεδομένα που ζητούνται.Όλες οι μετρήσεις έγιναν με πιθανότητα 50% ζωντανών κελιών για την αρχικοποίηση των τιμών τους,για 3 γενιές,αριθμό νημάτων ίσο με 8 και οι διεργασίες εκτελέστηκαν σε διαφορετικά nodes.Οι χρόνοι αντιστοιχούν στον συνολικό χρόνος εκτέλεσης της for των γενεών.

"1 διεργασία και αρχικό πίνακα 40000x40000 nodes=1:pps=8"= 50.524885sec
"2 διεργασίες και αρχικό πίνακα 80000x40000 nodes=2:pps=8"= 50.381436sec
"4 διεργασίες και αρχικό πίνακα 80000x80000 nodes=4:pps=8"= 50.433429sec

Ο χρόνος της generate με την παραλληλοποίηση σε OpenMp είναι για κάθε μία απο τις παραπάνω περιπτώσεις περίπου ίσος με 2.4seconds

Οι χρόνοι προκύπτουν ως αποτέλεσμα του μέσου όρου απο 3 μετρήσεις/εκτελέσεις του προγράμματος για την κάθε περίπτωση.Να σημειωθεί ότι οι χρόνοι ήταν πολύ σταθερή για κάθε περίπτωση με ακρίβεια πρώτου δεκαδικού ψηφίου.Βλέπουμε ότι οι χρόνοι είναι ίση μεταξύ των διαφορετικών περιπτώσεων κάτι που περιμέναμε.Η κάθε διεργασία σε κάθε περίπτωση δημιουργεί ένα πίνακα μεγέθουν 40000x40000 και εκτελείται σε διαφορετικό υπολογιστικό σύστημα απο τις υπόλοιπες διεγρασίες,δηλαδή αν και αυξάνουμε το μέγεθος του αρχικού πίνακα σε πολύ μεγάλο βαθμό ο χρόνος εκτελέσεις δεν αλλάζει αφού αυξάνουμε ανάλογα και τον αριθμό των επεξεργαστών.Άρα σε όλες τις περιπτώσεις η κάθε διεγρασία εκτελεί την ίδια δουλεία με τους ίδιους υπολογιστικούς-υλικούς πόρους και επομένως δεδομένου ότι οι διεργασίες τρέχουν παράλληλα είναι απόλυτα λογικό που δεν μεταβάλλεται ο χρόνος εκτέλεσης του προγράμματος.Τέλος παρατηρείται οτι ο χρόνος για τις επικοινωνίες καλύπτεται αποτελεσματικά απο την δομή του προγράμματος αφού δεν προκύπτει καμία καθυστέρηση απο το πρόγραμμα,έχουμε ίδιο χρόνο για την περίπτωση της μίας διεργασίας(η οποία δεν έχει επικοινωνίες) με τις υπόλοιπες περιπτώσεις.

Ενδεικτικά παρατίθενται επιπλέον και οι χρόνοι εκτέλεσης για μεγαλύτερο αριθμό γενεών

Για 4 γενίες ο χρόνος εκτέλεσης(της play) ειναι περίπου ίσος με 67 sec
Για 5 γενίες ο χρόνος εκτέλεσης(της play) ειναι περίπου ίσος με 84 sec
Για 6 γενίες ο χρόνος εκτέλεσης(της play) ειναι περίπου ίσος με 100.5 sec
Για 7 γενίες ο χρόνος εκτέλεσης(της play) ειναι περίπου ίσος με 117.2 sec

Ο χρόνος της generate είναι σταθερός στα 2.4sec αφου δεν αλλάζει το μέγεθος του πίνακα (40000x40000).