Πολυτεχνείο Κρήτης

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ

Αναφορά εξαμηνιαίας προγραμματιστικής

Ονομ/μο:

Ζήσκας Χρήστος **Α.Μ.** 2014030191

Εντριτ Μεται **Α.Μ.** 2014030005

Χατζηιωάννου Αναστάσης **Α.Μ.** 2013030026

Παληαλέξης Κωνσταντίνος **Α.Μ.** 2013030072

Σκοπός

Η προγραμματιστική άσκηση αφορά την μελέτη των αποτελεσμάτων της επίλυσης του προβλήματος των Ν-Βασιλισσών σύμφωνα με τον αλγόριθμο αναρρίχησης λόφων (hill climbing) και ικανοποίησης περιορισμών (constraint satisfaction problem με πρώιμο έλεγχο).

Περιγραφή του προβλήματος

Το πρόβλημα των Ν-Βασιλισσών ερμηνεύεται ως μια σκακιέρα που περιέχει Ν αριθμό βασιλισσών σε ταμπλό διαστάσεων. Η πλήρωση της σκακιέρας πραγματοποιείται με τυχαία τοποθέτηση των βασιλισσών στο ταμπλό. Η δομή του προβλήματος επεξηγείται από το συνολικό αριθμό των υπο-απειλή βασιλισσών. Η επίλυση του προβλήματος αναπαριστάται από το στάδιο της σκακιέρας στην οποία δεν υπάρχει βασίλισσα που να δέχεται απειλή ή διαφορετικά ο συνολικός αριθμός των υπο-απειλή βασιλισσών να είναι μηδενικός. Ο αλγόριθμος λειτουργεί χρησιμοποιώντας αντικειμενική συνάρτηση που υπολογίζει μετά από κάθε στάδιο της σκακιέρας, τον αριθμό των συγκρούσεων ενώ η τοποθέτηση του πιονιού γίνεται αναλογίζοντας την θέση που μπορεί να λάβει το πιόνι σε οποιαδήποτε κατεύθυνση, με τον μικρότερο αριθμό συγκρούσεων. Η λύση προκύπτει έπειτα από ανάλογες μετακινήσεις βασιλισσών με κατάληξη στην επιθυμητή κατάσταση της σκακιέρας.

Μετρα Απόδοσης

Η Λειτουργικότητα του πράκτορα στοχεύει στα ακόλουθα κριτήρια:

- Μέσος όρος χρόνου επίλυσης για Ν αριθμό επαναλήψεων ανά αριθμό βασιλισσών]
- Μέσος όρος του αριθμού initialization του grid για την καθιέρωση της λύσης
- Αξιοπιστία της λύσης.
- Ποσοστό επιτυχίας του αριθμού των collision . (Για κάθε περιβάλλον, κατά την τελική παρουσίαση της σκακιέρας να απομονώνεται ο αριθμός των συγκρούσεων στο ελάχιστο δυνατό σύνολο.)

ΠΕΡΙΓΡΑΦΗ ΕΠΙΛΥΣΗΣ ΣΚΑΚΙΕΡΑΣ ΜΕ ΑΛΓΟΡΙΘΜΟ ΤΟΠΙΚΗΣ ΑΝΑΖΗΤΗΣΗΣ

Ανάλυση των μεταβλητών

Η εξήγηση της σκακιέρας αποδίδεται σε δισδιάστατο πίνακα μεγέθους $N \times N$ όπου Ν ο αριθμός των βασιλισσών που εντοπίζονται στο πεδίο. Ο πίνακας αναπαρίσταται απο τιμές ακεραίων όπου μηδενική τιμή αποδίδεται σε σημείο που απουσιάζει βασίλισσα .

Κάθε βασίλισσα αντιπροσωπεύεται από ένα σειριακό αυξανόμενο αριθμό που εκπροσωπεί την ταυτότητα της βασίλισσας(int id) με τιμές στο σύνολο $[1, N+1]$

Η κίνηση που υφίσταται η κάθε βασίλισσσα αναπαρίσταται απο μεταβλητές που θεσπίζουν το εύρος των διαθέσιμων ακέραιων τιμών που μπορει να κινηθεί μια βασίλισσα. Ως διαθέσιμη κίνηση θεωρείται κάποια θεση στην οποία δεν παρεμβάλεται αλλη βασίλισσα ή ότι δεν ξεφεύγει απο τις διαστάσης της σκακιέρας.Το διάστημα της κίνησης μια βασίλισσα βρίσκεται στο δίαστημα

$[0, N-1]$ όπου N η θέση στην οποία παρεμποδίζει την κίνηση κάποια άλλη βασίλισσα σε εκείνη την κατεύθυνση. Για τις διαγώνιες θέσεις φυλάσσεται μια μεταβλητή που απεικονίζει τα βήματα(int southeast,southwest κλπ) που απαιτούν οι δείκτες γραμμης,στήλης ώστε να φτάσουν στην επιθυμητή θέση. Ταυτόχρονα συντηρείται πίνακας που περιέχει τους υποψήφιους εχθρούς. Ο πίνακα είναι διαστάσεων $N \times N$ καθώς μια βασίλισσα μπορεί να απειλεί όλες τις υπόλοιπες. Ο πίνακας αρχικά διαθέτει όλους τους υποψήφιους εχθρούς για κάθε βασίλλισα και έπειτα υφίσταται έλεγχος που διαγράφει την μια σχέση απο τον πίνακα. Ο αριθμός των βασιλισσών που θα απομείνουν ως υποψηφιοι εχθροί αποτυπώνεται στον αριθμό των συνολικών συγκρούσεων. Η επιλογή της βασίλισσας που θα κινηθεί για την βελτίωση του αριθμού των συγκούσεων προκύπτει απο γεννήτρια τυχαίων αριθμών στο διάστημα $[1, N+1]$

Περιγραφή μεθόδων τοπικής αναζήτησης

generateQueens: Αρχικοποιείται η ταυτότητα της βασίλισσας στο 1 και σύμφωνα με τη γεννήτρια τυχαίων αριθμών επιλέγεται το πεδίο στο οποίο θα εκχωρηθεί η βασίλισσα. Καλείται από τον constructor.Επιλέγεται η γραμμή και η στήλη. Αν υπάρχει βασίλισσα στο αντίστοιχο πεδίο επαναλαμβάνεται η ανάθεση τιμών στην γραμμή και τη στήλη στην οποία πρόκειται να τοποθετηθεί η βασίλισσα μέχρι να βρεθεί διαθέσιμο πεδίο. Τοποθετείται το πιόνι και η ταυτότητα αυξάνεται ώστε να εκπροσωπεί την επόμενη βασίλισσα προς καταχώρηση.

printGrid: Συνάρτηση που τυπώνει τα πεδία μιας σκακιέρας καθώς και τα σημεία στα οποία εντοπίζονται βασίλισσες. Τα ζητούμενα πεδία τυπώνονται με το γράμμα Q που λογίζεται σε Queen

```
This is the chess board right now
-------------------------------------

| - | - | - | Q | - |
| - | - | - | - | - |
| - | - | Q | Q | - |
| - | Q | Q | - | - |
| - | - | - | - | - |
```

printCollisionPairs: Συνάρτηση που τυπώνει τον πίνακα των συσχετίσεων μεταξύ των διαφόρων βασιλισσών. Στη πρώτη στήλη κάθε γραμμής τοποθετείται η κάθε βασίλισσα και στα υπόλοιπα κελία κρατούνται οι υποψήφιοι εχθροί. Οι σχέσεις είναι μόνοδρομες(Ο 1 χτυπάει τον 4 . Αρα ο 4 κρατιέται στη στήλη που λογίζονται τα θύματα του 1 .Ο 1 δεν υπάρχει στον 4).

Παρακάτω το δοσμένο σενάριο:

```
| 1 | 4 | 0 | 0 | 0 |
| 2 | 4 | 3 | 5 | 0 |
| 3 | 5 | 0 | 4 | 0 |
| 4 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 |
Total collision:6
```

findCollisionPairs:Συνάρτηση που δέχεται ως ορίσματα τις θέσεις καθώς και το ταμπλό στο οποίο βρίσκεται κάποια βασίλισσα. Αποθηκεύεται η βασίλισσα.

Ακολούθως ελέγχονται οι εχθροί που βρίσκονται στην εμβέλεια της βασίλισσα σε γραμμή και σε στήλη. Αυτό σημαίνει ότι στην εμβέλεια της βασίλισσα πρέπει να υπάρχουν κελία με τιμή διαφορετική από την μηδενική και η θέση τους να μην αποτελεί την τοποθεσία της βασίλισσα για την οποία αναζητούνται οι εχθροί. Αν υπάρχει κάποιος εχθρός αποθηκεύεται το id του στον πίνακα στη γραμμή που βρίσκεται η βασίλισσα που δόθηκε ως όρισμα. Η αναζήτηση εχθρών πραγματοποιείται προς κάθε κατεύθυνση που μπορεί να κινηθεί η βασίλισσα

totalCollisionsPaired:Συνάρτηση που υπολογίζει το συνολικό αριθμό των συγκρούσεων για ένα δοσμένο πλέγμα. Ο πίνακας συγκρούσεων επαναρχικοποιείται ώστε να μην υπάρχει σύγχυση με τις τιμές προηγούμενων πλεγμάτων. Καλεί την συνάρτηση ***findCollisionPairs.*** Εξετάζονται για όλες τις βασίλισσες τα πιόνια που απειλούν. Η κάθε βασίλισσα τοποθετείται στην πρώτη στήλη κάθε γραμμής. Απο κει και πέρα στην κάθε γραμμή μπαίνει το id κάθε βασίλισσα που είναι θύμα της σε οποιαδήποτε κατεύθυνση. Η σχέση είναι αμφίδρομης κατεύθυνσης .Παρακάτω παρουσιάζονται τα αποτελέσματα των ζευγαριών των συγκρούσεων για δεδομένο πλέγμα

findAvailableMoves:Συνάρτηση που είναι υπεύθυνη για την αναζήτηση των διαθέσιμων κινήσεων που μπορεί να εκτελέσει μια συγκεκριμένη βασίλισσα προς οποιαδήποτε κατεύθυνση . Ως όρισμα δέχεται το id της βασίλισσα και σύμφωνα με το id γίνεται ο εντοπισμός της στο πλέγμα. Έπειτα συμπληρώνονται στις αντίστοιχες μεταβλητές η μέγιστη και ελάχιστη τιμή που μπορεί να λάβει η αντίστοιχη βασίλισσα στην γραμμή της , στην στήλης της , καθώς και στις διαγώνιους. Για τις διαγώνιους χρησιμοποιείται μια μεταβλητή ώς βήμα που κατευθύνει τους δείκτες σε ελάχιστα και μέγιστα στις δυο διαγώνιους. Ουσιαστικά δηλαδή τέσσερα βήματα για κάθε προσανατολισμό στις διαγώνιους (northeast,northwest,southeast,southwest).Ως επιστροφή οδηγείται πίνακας που διαθέτει τις κατάλληλες τιμές για κάθε διαθέσιμη θέση στη στήλη , στη γραμμή καθώς και στα διαγώνια spots.

successorFunc: Συνάρτηση που λαμβάνει τα εξαγόμενα της ***findAvailableMoves*** και έχει ως όρισμα το id μιας βασίλισσας. Έχει ως αρμοδιότητα να αντικαθιστά τη βασίλισσα σε ορισμένη θέση για την οποία ο αριθμός των συγκρούσεων ελαττώνεται βέλτιστα για την συγκεκριμένη βασίλισσα. Για να επιτευχθεί αυτό πραγματοποιείται μεταφορά της βασίλισσα σε όλες τις διαθέσιμες θέσεις σε κάθε κατεύθυνση και εκ νέου εξέταση του αριθμού των συγκρούσεων. Καλείται η ***totalCollisionsPaired***.Κάθε φορά που βρίσκεται μικρότερος αριθμός collision τότε αποθηκεύεται η θέση σε στήλη και γραμμή καθώς και το αριθμός αυτός. Το πλέγμα τροποποιείται και στην νέα θέση μετακινείται η βασίλισσα.

chooseQueen: Συνάρτηση που οδηγεί στη λύση του προβλήματος. Δεν δέχεται ορίσματα. Πραγματοποιεί τυχαία επιλογή βασίλισσας και υπολογίζει το συνολικό collision του αρχικού πλέγματος. Θεωρείται λύση του προβλήματος όταν τα collisions ισούται με μηδενική τιμή. Αναθέτεται όριο για το οποίο όσο τα collision είναι μεγαλύτερα απο το όριο , η εκλογή της βασίλισσα εκπληρείται τυχαία . Αλλίως επιθεωρούνται σειριακά έτσι ώστε η τυχαία επιλογή να μην έχει αποκλίσει βασίλισσα που θα παρείχε λύση. Σε κάθε περίπτωση καλείται η ***successorFunc***. Εφόσον προέκυψε μικρότερο collision το πλέγμα οδηγείται σε καλύτερη λύση. Αν η σειριακή προσπέλαση παραχωρεί σε κάθε αλλαγή ίδιο αριθμό collision τότε καθίσταται αδύνατη η λύση για μηδενικό αριθμό collision και το πλέγμα αρχικοποιείται ξανά

ΠΕΡΙΓΡΑΦΗ ΕΠΙΛΥΣΗΣ ΣΚΑΚΙΕΡΑΣ ΜΕ ΑΛΓΟΡΙΘΜΟ ΙΚΑΝΟΠΟΙΗΣΗΣ ΠΕΡΙΟΡΙΣΜΩΝ

Η φόρμα της σκακιέρας λαμβάνει την παρακάτω μορφή:

```
This is the chess board right now
-------------------------------------

| Q | + | + | + |
| - | * | - | - |
| - | - | * | - |
| - | - | - | * |
```

Υπόμνημα

Σύμβολα	Περιγραφή
Q	Βασίλισσα που δεν απειλείται
*	Οι θέσεις που απειλούνται από κάποια βασίλισσα
+	Βασίλισσα που απειλείται
-	Κενός χώρος

Παρακάτω εκθέτονται οι αιτίες για τους παραπάνω χαρακτηρισμούς

<u>Ακολουθία βημάτων επίλυσης</u>

Βασικό κομμάτι της επίλυσης είναι η προσωρινή εγκατάσταση της κάθε βασίλισσας στο πρώτη γραμμή της στήλης

Έστω το σενάριο με αφετηρία τυχαίες καταστάσης βασιλισσών πλήθους $N = 4$ μαζί με την προσωρινή εγκατάσταση παρακάτω:

Γίνεται iterate στο grid από αριστερά προς τα δεξιά και από πάνω προς τα κάτω. Όταν εντοπισθεί βασίλισσα, μετακινείται προσωρινά στη στήλη 0, γραμμή 0. Ακολούθως, το επόμενο πιόνι απομακρύνεται στη στήλη 1, γραμμή 0, κ.ο.κ. Οι κινήσεις που κάνουν οι βασίλισσες είναι τύπου Γ (δεξιά και πάνω) και είναι πάντα νόμιμες σύμφωνα και τους κανόνες του σκακιού. Αν μία βασίλισσα παρεμποδίζεται, τότε δε θα μετακινηθεί. Η διαδικασία ολοκληρώνεται με την διαδοχική τοποθέτηση των εναπομενόντων βασιλισσών που δεν κινήθηκαν καθώς δεσμεύονταν λόγω εμποδίων

Η λειτουργία αυτή είναι πολύ σημαντική για το execution του αλγορίθμου, καθώς εξασφαλίζεται η παροχή κάθε στήλης από μια βασίλισσα και η ανυπαρξία vertical collisions. Η πολυπλοκότητά βρίσκεται σε χαμηλά επίπεδα το κόστος αναπτύσσεται στις τάξεις του 1% του συνολικού χρόνου για μεγάλα N.

Επιπρόσθετα, βασικό κομμάτι είναι τα constraints. Το πρόβλημα των βασιλισσών απο την οπτική του constraint satisfaction, χρειάζεται ένας τρόπος για απεικόνιση του προβλήματος με τα ελάχιστο όριο constraints για ταυτόχρονη καλή απόδοση και χαμηλή χωρική πολυπλοκότητα.

Για το λόγο αυτό θεσπίζεται η εξής παραδοχή: Οι βασίλισσες απειλούν μόνο προς τα δεξιά οριζόντια και δεξιά διαγώνια. Τα constraints δεν γεμίζονται από αριστερά και κατακόρυφα, για τον απλό λόγο ότι οι απειλές είναι αμοιβαίες και κατακόρυφα δεν απειλείται καμία βασίλισσα. Με τον τρόπο αυτό συντηρείται πολύ πιο καθαρή εικόνα του τι συμβαίνει.

Ο αλγόριθμος, λοιπόν, τρέχει από αριστερά προς τα δεξιά, από τη στήλη 0 προς τη στήλη N-1 (μία κάθε φορά), και γεμίζει τα constraints από αριστερά προς τα δεξιά.

Για κάθε στήλη:

- Αν μία βασίλισσα δεν απειλείται από κάποια προηγούμενη, μένει εκεί που είναι.

- Αν απειλείται τότε θα ψάξει να βρει ένα row στο οποίο δεν ενοχλείται από τις προηγούμενες. Εάν δεν υπάρχει τέτοιο row, τότε ο αλγόριθμος θα επιστρέψει με τη θέση της βασίλισσας που έχει πρόβλημα.

Την επίλυση αυτού του προβλήματος αναλαμβάνει η main, στην οποία κατασκευάζεται ένα state (τύπου -save) της σκακιέρας πριν δημιουργηθεί το πρόβλημα και ένα state(τύπου "restore") μετά από το πρόβλημα. Αυτό που διαφοροποιείται σε κάθε iteration είναι η γνώση του τι ΔΕ ΕΙΝΑΙ ΛΕΙΤΟΥΡΓΙΚΟ. Αποθηκεύεται ένας πίνακα που διατηρεί αυτήν την πληροφορία, διατρέχονται προσπάθειες για αρκετές αρχικές θέσεις για τις βασίλισσες, αποκλείοντας κάθε λανθασμένη λύση. Όταν βρεθεί ικανοποιητική λύση, ο αλγόριθμος τίθεται προς ολοκλήρωση και εκτυπώνεται ο χρόνος διάρκειας της διεργασίας επίλυσης

```
This is the chess board right now
----------------------------------------

| - | - | - | * | * | Q | * | * |
| - | - | Q | * | * | * | * | * |
| - | - | * | * | Q | * | * | * |
| - | * | - | - | * | * | Q | * |
| Q | * | * | * | * | * | * | * |
| - | * | * | Q | * | * | * | * |
| - | Q | * | * | * | * | * | * |
| - | - | * | * | - | * | - | Q |
Win.
It took 5.76281E-4 seconds to finish.
```

<u>Περιγραφή μεθόδων ικανοποίησης περιορισμών</u>

***<u>saveState()</u>*:** Αποθήκευση του grid.

***<u>loadState()</u>*:** Επαναφορά του grid (rollback) στην κατάσταση που σώσαμε στη saveState()

***<u>nQueensProblem()</u>*:** Είναι η successor function που περιέχει όλη την επίλυση που σχολιάσαμε εκτενώς παραπάνω.

***<u>printConstraints()</u>*:** Τυπώνει το grid με τους περιορισμούς όπως επεξηγήθηκε στο υπόμνημα παραπάνω.

***<u>singleColumnTransformation()</u>*:** Πηγαίνει κάθε βασίλισσα σε μία μοναδική στήλη (το πρώτο βήμα του αλγορίθμου)

***<u>isQueen(x,y)</u>*:** Επιστρέφει true αν το σημείο (x,y) περιέχει βασίλισσα.

<u>Ανάλυση των μεταβλητών</u>

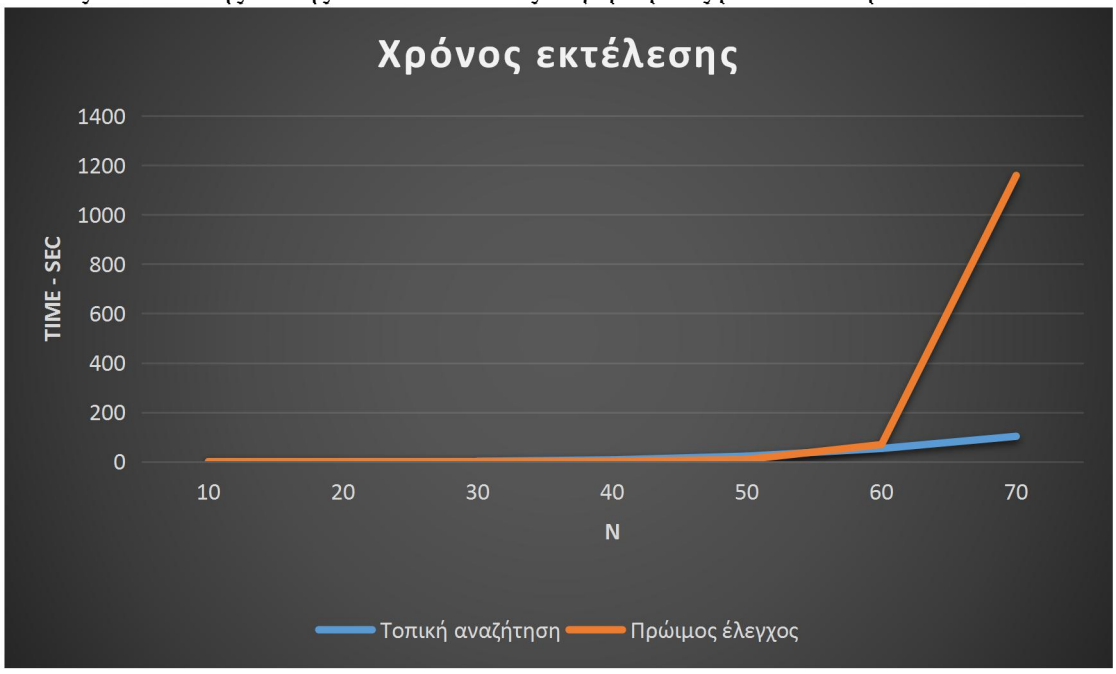
***<u>Grid</u>*:** N x N, integers, περιέχει το grid.

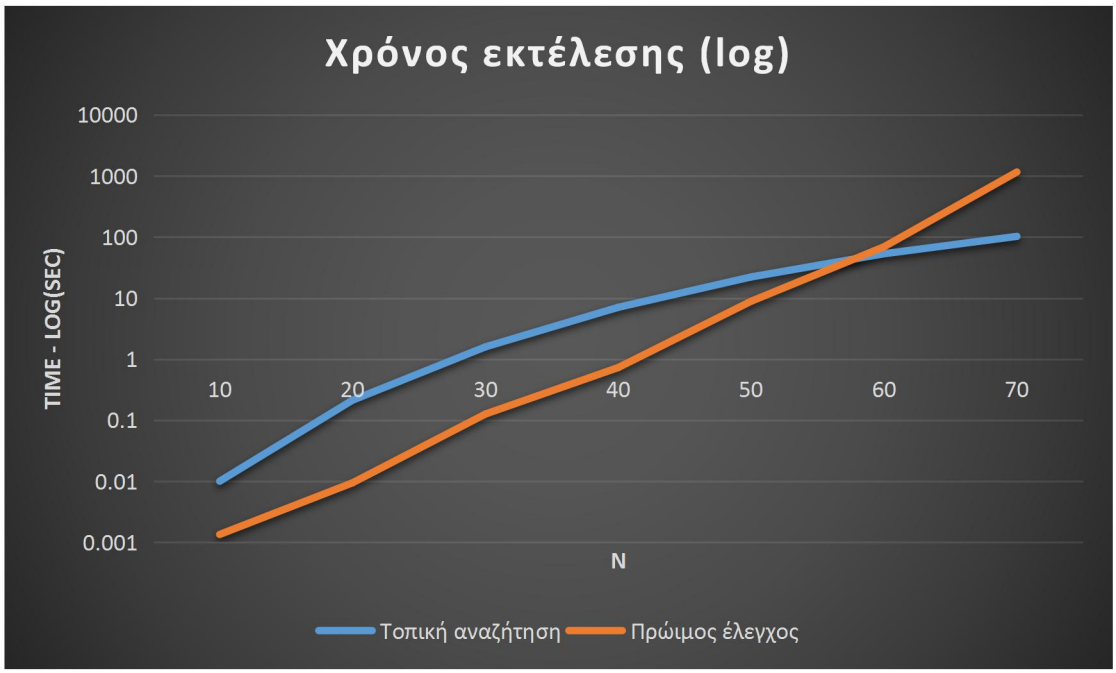
***InitialGrid*:** N x N, integers, περιέχει το αποθηκευμένο grid.

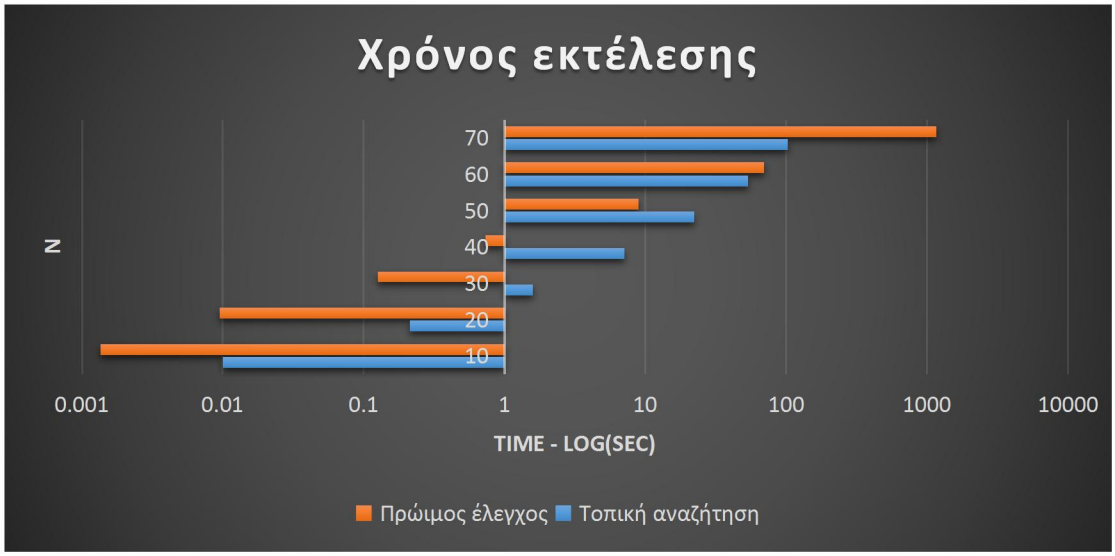
***Constraints*:** N x N, integers, περιέχει το grid με περιορισμούς.

ΣΥΜΠΕΡΑΣΜΑΤΑ

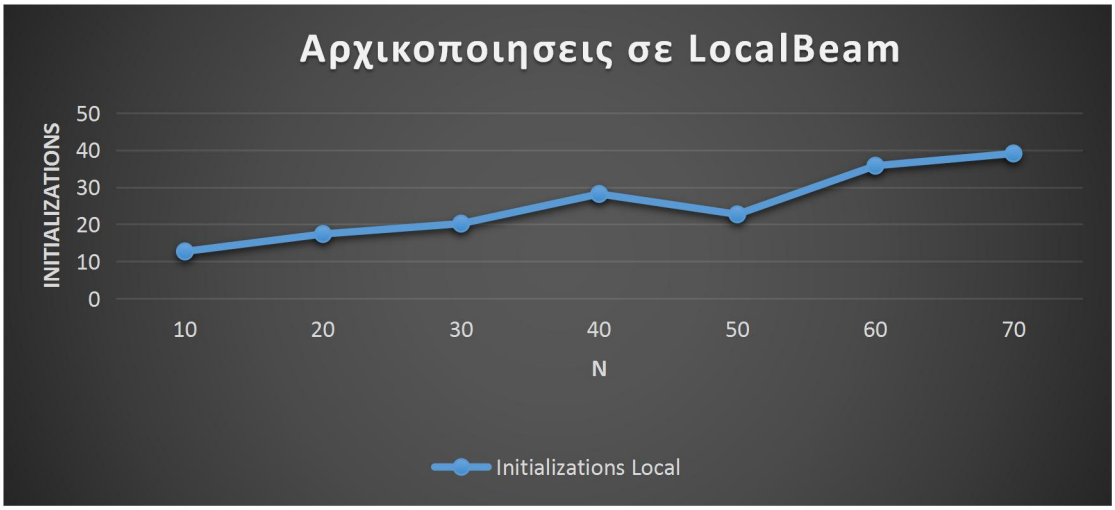
- Ποσοστό επιτυχίας στους αλγορίθμους με πιθανότητα στο 1
- Αξιοπιστία της λύσης -accurate στους αλγορίθμους με πιθανότητα στο 1

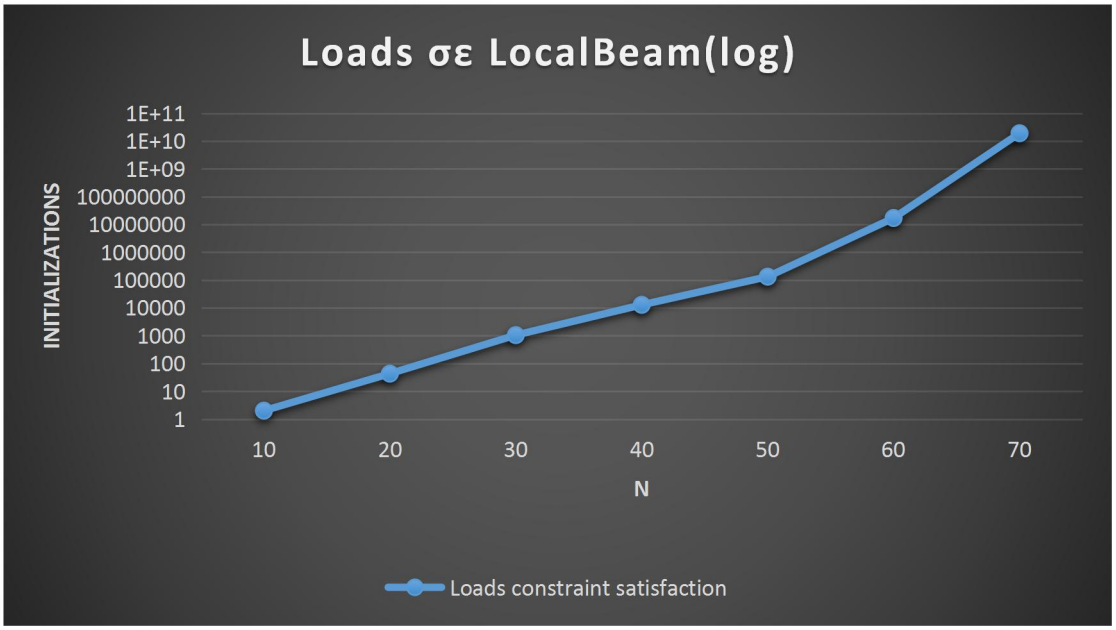






Εξέταση της απόδοσης του πρώιμου ελέγχου .Είναι πιο αποδοτικός για χαμηλά N, αλλά για τιμές μεγαλύτερες του 55, αρχίζει να πέφτει η απόδοση του εκθετικά. Ο αλγόριθμος τοπικής αναζήτησης είναι σταθερά αποδοτικός και μπορεί να χρησιμοποιηθεί για μεγάλα N. Ωστόσο, υπάρχει μεγάλη διακύμανση με καλό μέσο όρο στην τοπική αναζήτηση, και αυτό συμβαίνει λόγω στοχαστικότητας.





Ο αριθμός των αρχικοποιήσεων του ταμπλό είναι ασταθής και διαφοροποιείται για τον ιδιο αριθμό των πιονιών λόγω των διαφορετικών σημείων στα οποία τοποθετούνται οι βασίλισσες σε κάθε ταμπλό, ενώ ο αριθμός των load που πραγματοποιούνται στην ικανοποίηση περιορισμών δεν μεταβάλεται για τα διάφορα ύψη του N. Για μεγάλο αριθμό επαναλήψεων ο μέσος όρος θα προσωμοιώνονταν με μεγαλύτερη ακρίβεια.