ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ
ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΑΚΑΔΗΜΑΙΚΟ ΕΤΟΣ : 2015-2016

Θέμα 1

Τεχνητή Νοημοσύνη

Βαβουλιώτης Γεώργιος
Α.Μ. : 03112083
7ο εξάμηνο

Καραμπλίας Φώτιος
Α.Μ. : 03112004
7ο εξάμηνο

Ερώτηση 1 : Πρόβλημα - Σκιαγράφιση Τρόπου Επίλυσης

Πρόβλημα : Στο πρόβλημα αυτό μας ζητείται να επιλύσουμε ουσιατικά ένα πρόβλημα αναζήτησης, στο οποίο έχουμε 2 ρομπότ (Robot1, Robot2) τα οποία έχουν τη δυνατότητα να κινηθούν μια θέση(κουτάκι) ανα κίνηση. Θα πρέπει να τονιστεί ότι στον επίπεδο χώρο στον οποίο γίνεται η κίνηση υπάρχουν εμπόδια και καθένα απο τα 2 ρομπότ καλείται να περάσει απο προκαθορισμένα σημεία χωρίς όμως να συγκρουστούν μεταξύ τους ή να χτυπήσουν στα εμπόδια. Η τελική συνάντηση των 2 ρομπότ θεωρείται επιτυχής όταν κάποιο απο τα 2 ρομπότ καταφέρει να φτάσει στο σημείο συνάντησης και το άλλο έχει καταφέρει να βρεθεί στην αμέσως γειτονική θέση του σημείου συνάντησης. Επίσης θα πρέπει να επισημάνουμε πως έχουμε θεωρήσει οτι κάθε ρομπότ γνωρίζει την θέση και το πλάνο του άλλου και οργανώνει την αναζήτηση του με βάση αυτά. Για την αναζήτηση της πορεία την οποία πρέπει να επιλέξει κάθε ρομπότ απο την τρέχουσα θέση του σε μια νέα θα κάνουμε χρήση του αλγορίθμου A* με τη χρήση υπο-εκτιμητή(admissible heuristic) και με τη χρήση υπερ-εκτιμήτή(non-admissible heuristic). Εν τέλει θα συγκρίνουμε τα αποτελέσματα που προέκυψαν απο τις δυο διαφορετικές εκδοχές. Συγκεκριμένα, για την επίλυση του προβλήματος χρησιμοποιήθηκε ο αλγόριθμος A* ως μια παραλλαγή του Djiikstra με την προσθήκη της Manhattan για τον υπολογισμό των ευριστικών αποστάσεων. Η εκτίμηση αποστάσεων είναι χρήσιμη ώστε με λιγότερες αναζητήσεις να οδηγηθούμε σε βέλτιστη λύση δηλαδή ώστε να γίνει η ελάχιστη η διάσχιση του χώρου καταστάσεων. Αφού το ένα ρομπότ γνωρίζει για την κίνηση του άλλου,το πρόβλημα λύνεται τρέχοντας τον A* ανεξάρτητα μία φορά για κάθε ρομπότ δίνοντας ως παράμετρο στην συνάρτση του A* τις θέσεις από τις οποίες πρέπει να περάσει το καθένα πτιν φτάσουν στον τελικό στόχο. Έπειτα μπορούμε να αποφασίσουμε να βάλουμε το ένα από τα δύο να περιμένει (π.χ. πάντα το Robot1) δηλαδή για μία χρονική στιγμή να μείνει ακίνητο αν προκύψει να βρίσκονται μια συγκεκριμένη χρονική στιγμή στην ίδια θέση δηλαδή έχουμε μία σύγκρουση.Συνεχίζοντας το ρομπότ που θα φτάσει δεύτερο στο στόχο μπορεί να μείνει σε κουτάκι που είναι γειτονικό του στόχου, αφού αυτό ορίζεται από την εκφώνηση αλλά σίγουρα αυτή η θέση θα είναι μέρος της διαδρομής προς τον τελικό στόχο.

Επιλογή Τρόπου Υλοποίησης : Επιλέχθηκε η γλώσσα ***C***++ λόγω του ότι προσφέρει χρήσιμες έτοιμες βιβλιοθήκες δομών δεδομένων που βοηθάνε στην επίλυση του προβλήματος. Χρησιμοποιήθηκαν οι δομές set, vector, list για να είναι ευκολότερη η ταξινόμηση και διατήρηση καταστάσεων και το πρόγραμμα χωρίστηκε σε 3 αρχεία: robot.cpp, astar.cpp, visualizer.cpp (και τα header files(.h) για το καθένα) .

- Το robot.cpp διαβάζει τα δεδομένα και υλοποιεί τις απαραίτητες κλήσεις συναρτήσεων
- Η astar.cpp υλοποιεί την αναζήτηση με χρήση του αλγορίθμου A*
- Το visualizer.cpp αναλαμβάνει την οπτικοποίηση του αποτελέσματος σε εικόνα μορφής PNG.

Επιπλέον Χρήσιμες Πληροφορίες :

1)Η οπτικοποίηση έγινε χρησιμοποιώντας τη C βιβλιοθήκη γραφικών cairo. Η απεικόνιση γίνεται στην εικόνα ntua-ai-robots.png η οποία φαίνεται σε ένα από τα παρακάτω ζητούμενα για δεδομένο testcase.

2)Εκτός της οπτικής αναπαράσταση το πρόγραμμα επίσης εμφανίζει βοηθητικά μηνύματα σχετικά με το τι προσθέτουμε στην ουρά κάθε φορά ,με τον αν βρίσκουμε εμπόδιο και με τον προκύπτει σύγκρουση ανάμεσα στα 2 ρομπότ.

Ερώτηση 2 : Επιλογή δομής δεδομένων

Χρησιμοποιήσαμε την εξής δομή δεδομένων στην C++ για την επίλυση του προβλήματος(η κλάση αυτή αντιπροσωπεύει τη μετάβαση από μία θέση σε μία άλλη):

```
struct Robot_Node {

    public:

        int parent;

        Point from;

        Point to;

          // approximation of how far the "to" end of Edge is from the target

        int heuristic;

          // actual distance between the "to" end of the Edge and the source

        int distance;

};
```

Περιεχόμενα Κλάσης :

- θέση προέλευσης(from)

- θέση προορισμού(to)
- ευριστική εκτίμηση μέχρι το στόχο(heuristic)
- πραγματική απόσταση που έχουμε διανύσει από την πηγή(distance).

Ερώτηση 3 : Συναρτήσεις-Τελεστές Για Υλοποίηση A*

Οι συναρτήσεις που χρησιμοποιούνται για την υλοποίηση του A* είναι οι ακόλουθες :

- **aStar** : η βασική συνάρτηση υλοποίησης του αλγορίθμου καθώς ξεκινά από αυτήν βάζοντας σε μιά ουρά τις συνδέσεις στο χάρτη τις οποίες μπορεί να χρησιμοποιήσει.
- **enqueue :** Η συνάρτηση enqueue χρησιμοποιείται βοηθητικά εντός της συνάρτησης aStar για να προσθέσει μία νέα πιθανή επιλογή στην ουρά προτεραιότητας.

Γενικό Σχόλιο : Όλες οι συναρτήσεις είναι κατασκευασμένες να μην χρησιμοποιούν global μεταβλητές και χωρίς παρενέργειες(side effects) ώστε να είναι επαναχρησιμοποιήσιμες.

Επεξήγηση : Αρχικά καλείται η aStar και μπαίνουν σε μια ουρά οι συνδέσεις στο χάρτη οο οποίες μπορεί να χρησιμοποιηθούν.Στη συνέχεια ο αλγόριθμος χρησιμοποιεί την ουρά σαν ουρά προτεραιότητας ταξινομώντας τις υποψήφιες ακμές με βάση μια συνάρτηση σύμφωνα με την εκτιμώμενη απόσταση από τον στόχο.

Ερώτηση 4 : Ευριστικές Μεθόδοι

Οι ευριστικές μέθοδοι που θεωρήθηκαν είναι οι εξής :

- **Admissible Heuristic:** Χρησιμοποιήθηκε η συνάρτηση απόστασης Manhattan, διότι ένα Robot χρειάζετα τόσα βήματα όσα δείχνει η απόσταση Manhattan για να φτάσει στο στόχο(καλύτερη περίπτωση) αν δεν συναντήσει καθόλου εμπόδια. Προφανώς σε διαφορετική περίπτωση χρειάζεται περισσότερα βήματα.
- **Non-Admissible Heuristic:** Χρησιμοποιήθηκε η συνάρτηση $x^2 + y^2$ διότι υπερεκτιμά την απόσταση προς το στόχο, δηλαδή ενδέχεται ο υπερεκτιμητής να οδηγηθεί σε υποβέλτιστα αποτελέσματα κάνοντας λιγότερο πλήθος βημάτων(π.χ. αν δεν υπάρχουν εμπόδια ανάμεσα στη θέση του Robot και στο στόχο, τότε θα δώσει ως αποτέλεσμα μεγαλύτερο απ' ό,τι η πραγματική απόσταση(αποτέλεσμα της βέλτιστης λύσης)).

Ερώτηση 5 : Ψευδοκώδιακας

Τα βήματα του ζητούμενου ψευδοκώδικα φαίνονται παρακάτω (παρακάτω όταν συναντάς timeB εννοείται ο μετρητής χρόνου του B):

1.Εύρεση PathA

2.Αποθήκευση στοιχείων σε μια λίστα

3.Εύρεση PathB

4.Αν η κεφαλή της λίστας(head) ταυτίζεται με το σημείο του PathB τότε : Κάνουμε Stall το B

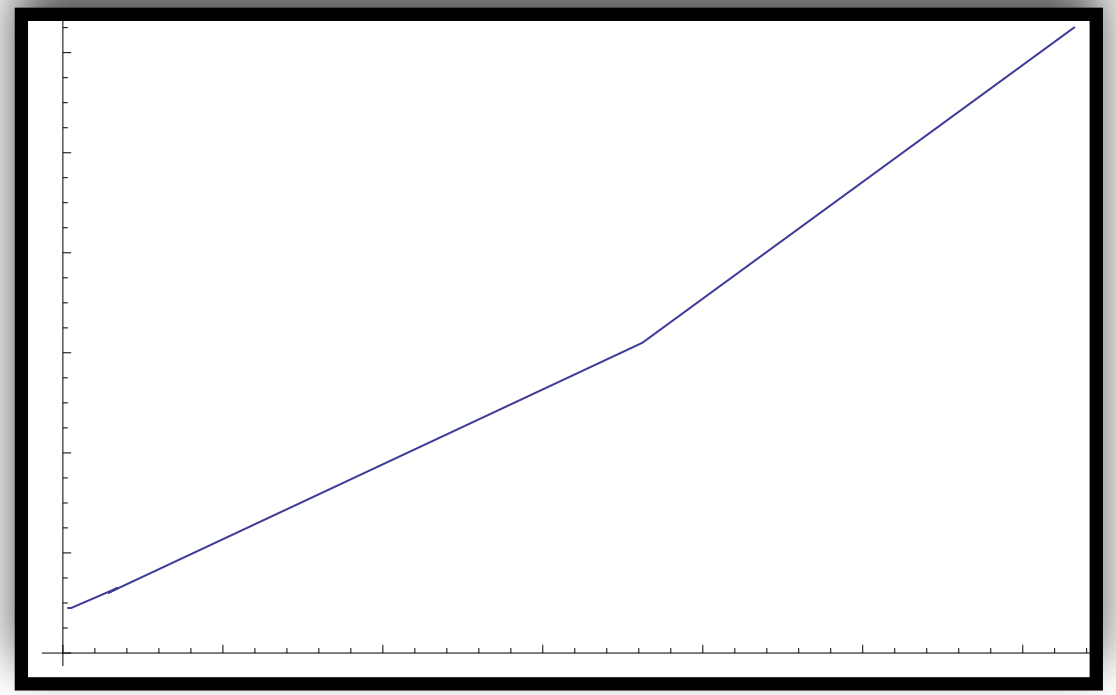
5.Αφαίρεση της κεφαλής(head), αύξηση του timeB και συνεχίζουμε κατά τον ίδιο τρόπο.

Ερώτηση 6 : Στατιστικά Αποτελέσματα

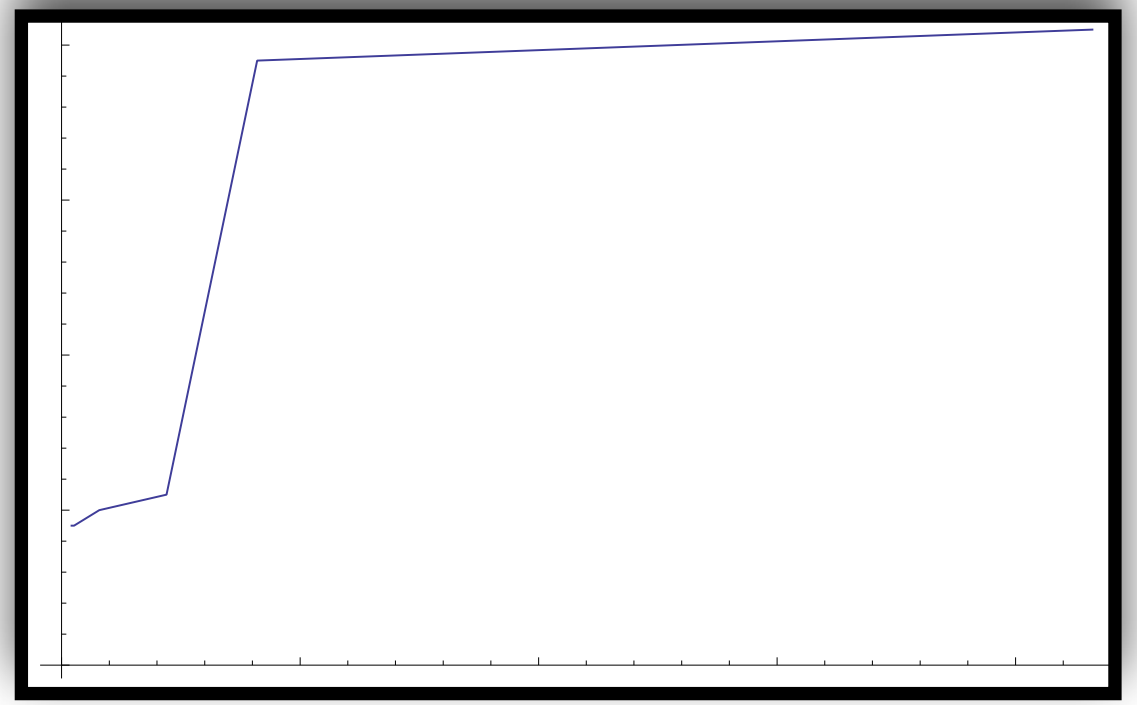
Γενικό Σχόλιο: Τα δύο διαγράμματα που φαίνονται παρακάτω είναι **<<*Χρόνος Εκτέλεσης*>>** συναρτήσει **<<*Αριθμός Κόμβων*>>** .

Τα στατιστικά αποτελέσματα φαίνονται παρακάτω, με την βοήθεια γραφημάτων :

Admissible heuristic



Non-admissible heuristic



Σχολιασμός :

- Ο χρόνος εκτέλεσης είναι αμελητέος για τις μικρές δοκιμαστικές περιπτώσεις. Επίσης είναι πολύ μικρός για τις μεγαλύτερες. Αυτά ισχύουν και για τις 2 περιπτώσεις και θα πρέπει να αναφέρω πως για τη μέτρηση χρησιμοποιήθηκε η κλήση time του UNIX.

- Η χρήση της υπερεκτιμήτριας(ύψωση στο τετράγωνο) ανοίγει λιγότερες καταστάσεις συγκρητικά με την υποεκτιμήτρια όπως αυτό είναι εμφανές αλλά δίνει περισσότερο βάρος στο ευριστικό κριτήριο παρά στην απόσταση που έχει διανυθεί ως τη συγκεκριμένη χρονική στιγμή. Άρα, αν υποθέσουμε ότι το κριτήριο είναι αποδεκτό, η χρήση υποεκτιμήτριας ανοίγει λιγότερες καταστάσεις αφού τα βήματα που γίνονται είναι πιο σίγουρα.

Παρατήρηση : Τα testcases στα οποία βασιστήκαμε για να βγάλουμε τα παραπάνω αποτελέσματα έχουν επισυναφθεί στο zip αρχείο της άσκησης.

Ερώτηση 7 : Θέσεις Ελέγχου Του Ρομπότ

Η έξοδος για το testcase της εκφώνησης είναι η παρακάτω :

```
A* algorithm completed in 37 steps.
A* algorithm completed in 25 steps.
A* algorithm completed in 58 steps.
A* algorithm completed in 9 steps.
A* algorithm completed in 23 steps.
A* algorithm completed in 25 steps.
A* algorithm completed in 58 steps.
A* algorithm completed in 9 steps.
Robot A path:
(13, 8) at moment:1
(13, 7) at moment:2
(13, 6) at moment:3
(13, 5) at moment:4
(13, 4) at moment:5
(12, 4) at moment:6
(11, 4) at moment:7
(10, 4) at moment:8
(9, 4) at moment:9
(8, 4) at moment:10
(7, 4) at moment:11
(6, 4) at moment:12
(5, 4) at moment:13
(5, 5) at moment:14
(5, 6) at moment:15
(5, 7) at moment:16
(5, 8) at moment:17
(5, 9) at moment:18
(5, 10) at moment:19
(5, 11) at moment:20
(5, 12) at moment:21
(5, 13) at moment:22
(6, 13) at moment:23
(7, 13) at moment:24
(7, 12) at moment:25
(7, 11) at moment:26
(7, 10) at moment:27
(8, 10) at moment:28
(9, 10) at moment:29
(10, 10) at moment:30
(10, 9) at moment:31
(11, 9) at moment:32
(11, 8) at moment:33
(12, 8) at moment:34
(13, 8) at moment:35
(14, 8) at moment:36
(15, 8) at moment:37
(16, 8) at moment:38
(17, 8) at moment:39
(17, 9) at moment:40
(16, 9) at moment:41
(15, 9) at moment:42
(14, 9) at moment:43
(13, 9) at moment:44
A* algorithm completed in 23 steps.
Robot B path:
(2, 8) at moment:1
(2, 7) at moment:2
(2, 6) at moment:3
(2, 5) at moment:4
(2, 4) at moment:5
(3, 4) at moment:6
(4, 4) at moment:7
(5, 4) at moment:8
A* algorithm completed in 25 steps.
(5, 5) at moment:9
(5, 6) at moment:10
(5, 7) at moment:11
(5, 8) at moment:12
(5, 9) at moment:13
(5, 10) at moment:14
(5, 11) at moment:15
(5, 12) at moment:16
(5, 13) at moment:17
(6, 13) at moment:18
(7, 13) at moment:19
A* algorithm completed in 58 steps.
```

```
(7, 12) at moment:20
(7, 11) at moment:21
(7, 10) at moment:22
(8, 10) at moment:23
(9, 10) at moment:24
(10, 10) at moment:25
(10, 9) at moment:26
(11, 9) at moment:27
(11, 8) at moment:28
(12, 8) at moment:29
(13, 8) at moment:30
(14, 8) at moment:31
(15, 8) at moment:32
(16, 8) at moment:33
(16, 8) at moment:34
(16, 8) at moment:35
(16, 8) at moment:36
(16, 8) at moment:37
(16, 8) at moment:38
(16, 8) at moment:39
A* algorithm completed in 47 steps.
(16, 9) at moment:40
(16, 10) at moment:41
(15, 10) at moment:42
(14, 10) at moment:43
(13, 10) at moment:44
(12, 10) at moment:45
(11, 10) at moment:46
(10, 10) at moment:47
(9, 10) at moment:48
(8, 10) at moment:49
(8, 9) at moment:50
(7, 9) at moment:51
(6, 9) at moment:52
(5, 9) at moment:53
(4, 9) at moment:54
(3, 9) at moment:55
(2, 9) at moment:56
```

Αν υπάρξει σύγκρουση η έξοδος είναι η εξής :

```
A* algorithm completed in 17 steps.
A* algorithm completed in 8 steps.
A* algorithm completed in 22 steps.
A* algorithm completed in 11 steps.
A* algorithm completed in 8 steps.
A* algorithm completed in 14 steps.
Robot B path:
(4, 1) at moment:1
(3, 1) at moment:2
(3, 2) at moment:3
(3, 3) at moment:4
(3, 4) at moment:5
(3, 5) at moment:6
(3, 6) at moment:7
(3, 5) at moment:8
(4, 5) at moment:9
(5, 5) at moment:10
(6, 5) at moment:11
(7, 5) at moment:12
(6, 5) at moment:13
(5, 5) at moment:14
(4, 5) at moment:15
(3, 5) at moment:16
(3, 4) at moment:17
(3, 3) at moment:18
(3, 2) at moment:19
(3, 1) at moment:20
(4, 1) at moment:21
(5, 1) at moment:22
A* algorithm completed in 17 steps.
Robot A path:
(1, 2) at moment:1
(1, 3) at moment:2
(1, 4) at moment:3
(1, 5) at moment:4
(1, 6) at moment:5
(2, 6) at moment:6
```

Collision - Robot A stay at his position to avoid collision

```
(2, 6) at moment:7
(3, 6) at moment:8
A* algorithm completed in 8 steps.
(3, 5) at moment:9
(4, 5) at moment:10
(5, 5) at moment:11
(6, 5) at moment:12
A* algorithm completed in 20 steps.
(5, 5) at moment:13
(4, 5) at moment:14
(3, 5) at moment:15
(3, 4) at moment:16
(3, 3) at moment:17
(3, 2) at moment:18
(3, 1) at moment:19
(2, 1) at moment:20
(1, 1) at moment:21
```

Παρατήρησεις:

- Οι εντολές με τις οποίες έγινε η μεταγγλώτιση των πηγαιών αρχείων καθώς και η εκτέλεση του εκτελέσιμου αρχείου που δημιουργήθηκε έγιναν με τις παρακάτω εντολές:

```
g++ -c astar.cpp -o astar.o
g++ -c robot.cpp -o robot.o
g++ -o exec_robot robot.o astar.o
./exec_robot <input.txt
```

- Η κακή ποιότητα των εικόνων οφείλεται στο γεγονός οτι είναι screenshots.

Ερώτηση 8 : Βήματα Μονοπατιού

Παρακάτω φαίνονται τα βήματα του κάθε robot για ένα τυχαίο από τα input files σε μοφή εικόνας(το πως προέκυψε έχει εξηγηθεί παραπάνω):