<u>Τεχνητή Νοημοσύνη</u>
<u>(Project1)</u>

<u>Ονοματεπώνυμο:</u> Δημήτριος Σιταράς
<u>Αριθμός μητρώου:</u> 1115201800178
<u>Εξάμηνο:</u> 5ο

<u>Πρόβλημα 2</u>

<u>Δεδομένα:</u> Π δέντρο αναζήτησης βάθους d, με παράγοντα διακλάδωσης b, και ο κόμβος με το μικρότερο βάθος που αντιστοιχεί σε κατάσταση στόχου βρίσκεται σε βάθος g $\leq$ d.

<u>Ο μικρότερος αριθμός</u> των κόμβων που μπορούν να δημιουργηθούν από τον αλγόριθμο πρώτα σε βάθος με επαναληπτική εκβάθυνση είναι:

$$g + (g-1)b \ + (g-2)b^2 + ... + 2b^{g-2} + b^{g-1} + gb + 1 = \sum_{i=0}^{g-1} (g-i)b^i \ + (gb \ + 1)$$

<u>Αιτιολόγηση:</u> Το άθροισμα αυτό προκύπτει γιατί στην καλύτερη περίπτωση της επαναληπτικής αναζήτησης εκβάθυνσης ο πρώτος κομβος (ο πιο αριστερος) στο επίπεδο g αποτελεί κατάσταση στόχου,άρα για να φτάσω σε αυτον το κόμβο παράγω gb+1 κομβους μαζί με την ρίζα,επομένως οι κόμβοι στο επίπεδο g-1 παράγονται 1 φορα, οι κόμβοι στο g-2 επίπεδο παραγονται 2 φορες και ούτω καθεξής μέχρι τη ρίζα του δέντρου αναζήτησης η οποία παράγεται g φορές.

<u>Ο μεγαλύτερος αριθμός</u> των κόμβων που μπορούν να δημιουργηθούν από τον αλγόριθμο πρώτα σε βάθος με επαναληπτική εκβάθυνση είναι:

$$(g+1) + gb + (g-1)b^2 + ... + 3b^{g-2} + 2b^{g-1} + b^g = \sum_{i=0}^{g} (g+1-i)b^i$$

<u>Αιτιολόγηση:</u> Το άθροισμα αυτό προκύπτει γιατί στην χειρότερη περίπτωση της επαναληπτικής αναζήτησης εμβάθυνσης παραγεται μέχρι και ο τελευταίος κομβος στο επίπεδο g (ο πιο δεξιός), ο οποίος θα αποτελεί κατάσταση στόχου, συνεπώς οι κομβοι στο επίπεδο g παράγονται όλοι 1 φορά, οι κόμβοι στο επίπεδο g-1 παραγονται 2 φορες , στο g-2 επίπεδο παραγονται 3 φορες και ούτω καθεξής μέχρι τη ρίζα του δέντρου αναζήτησης η οποία παράγεται g+1 φορές.

Πρόβλημα 3

α) Μια ευρετική συνάρτηση λέγεται ότι είναι συνεπής εάν η εκτίμησή της είναι πάντα μικρότερη ή ίση με την εκτιμώμενη απόσταση από οποιαδήποτε γειτονική κορυφή προς τον στόχο, συν το κόστος επίτευξης για να φτασουμε σε αυτόν τον γείτονα. Συγκεκριμένα, για όλους του κόμβους του γράφου πρέπει να ισχύει:

$h(N) \leq c(N,P) + h(P)$ και $h(G) = 0$

οπου N ένας κόμβος του γραφήματος,
P ο απόγονος του N κομβου,
G ο κόμβος στόχου,
c(N,P) το κόστος του μονοπατιου για να φτάσουμε απο το τον κομβο N στον P.

Επομένως, βλεπω αν ισχύει η ανισωση για όλους τους κόμβους του γράφου ώστε να βγαλω συμπέρασμα αν είναι συνεπής ή όχι. Άρα έχω ότι:

1. $h(o103) \leq c(o103,ts)+h(ts)$
   $21 \leq 8+23=31$

2. $h(ts) \leq c(ts,mail)+h(mail)$
   $23 \leq 6+26=32$

3. $h(o103) \leq c(o103,b3)+h(b3)$
   $21 \leq 4+17=21$

4. $h(b3) \leq c(b3,b1)+h(b1)$
   $17 \leq 4+13=17$

5. $h(b1) \leq c(b1,c2)+h(c2)$
   $13 \leq 3+10=13$

6. $h(c2) \leq c(c2,c1)+h(c1)$
   $10 \leq 4+6=10$

7. $h(c1) \leq c(c1,c3)+h(c3)$
   $6 \leq 8+12=20$

8. $h(c2) \leq c(c2,c3)+h(c3)$
   $10 \leq 6+12=18$

9. $h(b3) \leq c(b3,b4)+h(b4)$
   $17 \leq 7+18=25$

10. $h(b1) \leq c(b1,b2)+h(b2)$
    $13 \leq 6+15=21$

11. $h(b2) \leq c(b2,b4)+h(b4)$
    $15 \leq 3+18=21$

12. $h(b4) \leq c(b4,o109)+h(o109)$
    $18 \leq 7+24=31$

13. $h(o103) \leq c(o103,o109)+h(o109)$
    $21 \leq 12+24=36$

14. $h(o109) \leq c(o109,o111)+h(o111)$
    $24 \leq 4+27=31$

15. $h(o109) \leq c(o103,o119)+h(o119)$
    $24 \leq 16+11=27$

16. $h(o119) \leq c(o119,storage)+h(storage)$
    $11 \leq 7+12=19$

17. $h(o119) \leq c(o119,o123)+h(o123)$
    $11 \leq 9+4=13$

18. $h(o123) \leq c(o123,o125)+h(o125)$
    $4 \leq 4+6=10$

19. $h(o123) \leq c(o123,r123)+h(r123)$
    $4 \leq 4+0=4$

και για τον κομβο στόχου ισχύει οτι h(r123)=0.

Οπότε, η ευρετική συνάρτηση h είναι <u>συνεπής</u>, επομενως ειναι και <u>παραδεκτή</u>.

β)
Συμφωνα με τους αλγοριθμους στις διαφάνειες, οταν βγάζω τον κόμβο από το σύνορο ελέγχω αν ειναι κατάσταση στόχου.
Επίσης,εισάγω τους κόμβους στο σύνορο με αλφαβητική σειρά.

Αναζήτηση πρώτα σε πλάτος(συνορο με Queue):
o103,b3,o109,ts,b1,b4,o111,o119,mail,b2,c2,o123,storage,c1,c3,o125,r123

Αναζήτηση πρώτα σε βάθος(συνορο με Stack):
o103,ts,mail,o109,o119,storage,o123,r123

Αναζήτηση πρώτα σε βάθος με επαναληπτική εκβάθυνση(συνορο με Stack, δεν εχει εξερευνημενο σύνολο):
Έστω i τα επίπεδα εμβάθυνσης της κάθε επανάληψης, επομένως:

Για i=0
o103

Για i=1
o103,ts,o109,b3

Για i=2
o103,ts,mail,o109,o119,o111,b3,b4,b1

Για i=3
o103,ts,mail,o109,o119,storage,o123,o111,b3,b4,o109,b1,c2,b2

Για i=4
o103,ts,mail,o109,o119,storage,o123,r123

Άπληστη αναζήτηση πρώτα στον καλύτερο με ευρετική συνάρτηση την h (συνορο με Priority Queue):
o103,b3,b1,c2,c1,c3,b2,b4,ts,o109,o119,o123,r123

A με ευρετική συνάρτηση την h (συνορο με Priority Queue):*
o103,b3,b1,c2,c1,b2,b4,c3,ts,o109,o119,mail,o123,r123

Πρόβλημα 4

α)

Αρχική κατάσταση του προβλήματος:

Το ρομποτ βρίσκεται στην θεση εκκινησης mail, δεν κουβαλάει κανενα πακέτο (boolean μεταβλητή) και έχω μια “λίστα” που περιέχει τα ζεύγη (ζεύγη συντεταγμένων) των αποστολέων και των παραληπτών αντιστοιχα.
αρχική κατασταση = [(συντεταγμενες του mail),μεταφέρει_πακετο =false,[(αποστολείς,παραληπτες)]]

Συνάρτηση Διαδοχής:

Επιστρέφει τα ζευγη απογονων και ενέργειας(πανω,κάτω,αριστερά,δεξιά), που απαιτειται καθε φορα για να φτασω σε έναν συγκεκριμένο απογονο απο την τρέχουσα κατάσταση.
Άρα επιστρεφει:
[(συντεταγμένες διαδόχων,ενέργειες)]

Χώρος Καταστάσεων:

Είναι ένα σύνολο από καταστάσεις (συντεταγμένες) στις οποίες μπορούμε να φτάσουμε από την τρέχων κατάσταση.
Χώρος καταστάσεων = [(τρεχων συντεταγμενες του ρομποτ),μεταφέρει_πακετο =true/false,[(αποστολείς,παραληπτες)]]

Κατάσταση στόχου:

Όταν αδειάσει η λίστα, δηλαδή παραδοθούν όλα τα πακέτα, και το ρομποτ έχει επιστρέψει στην αρχική του θέση mail τότε είμαστε σε κατάσταση στόχου (εννοείται πως δεν κουβαλάει κανένα πακέτο).
Κατασταση στοχου = [(συντεταγμενες του mail),μεταφέρει_πακετο=false,[ ]]

Συνάρτηση Κόστους:

Το ρομποτ μπορεί να κινηθεί κάτω,πάνω,δεξιά,αριστερά, αυτο ειναι το σύνολο των ενεργειών του. Για την κάθε ενέργεια του θεωρώ σταθερό κόστος ίσο με 1.
Επομένως, οταν μετακινείται το ρομποτ απο το state1 σε ενα state2 μεσω μιας ενέργειας (action) επιστρέφω:
cost(state1,action,state2)=1

β)<u>Ευρετική Συνάρτηση για Α*</u>
Το άθροισμα όλων των αποστάσεων παραλαβης-παραδοσης των πακέτων, συν την απόσταση από το πλησιέστερο δέμα.
Εξήγηση:

- το ρομποτ θα πρεπει να παραδώσει όλα τα τα πακέτα ενα-ενα, οπότε το άθροισμα των αποστάσεων παραλαβης-παραδοσης που θα διανυσει θα ειναι ενα κατω φράγμα (δεν γίνεται καμια υπερεκτίμηση γιατι ασχέτως με την σειρα που θα παραδώσει τα πακέτα το ρομποτ σιγουρα θα διανύσει τις αποστάσεις παραλαβής-παράδοσης).
- Εφόσον το ρομποτ ξεκινάει από το δωμάτιο mail, η απόσταση από το πλησιέστερο δέμα αποτελει ενα ακομα κάτω φράγμα (ανεξαρτητως σε ποιο πακέτο τελικά θα πάει το ρομποτ) της ευρετικης ετσι ωστε να εγγυαται οτι παντα θα ειναι αποδεκτή.

(Αγνοώντας τους τοίχους για τον υπολογισμό των αποστάσεων, χρησιμοποιώ απόσταση Manhattan για τους υπολογισμους)

<u>Πρόβλημα 5</u>

α)Με υπορουτινες την *αναζήτηση πρώτα σε πλάτος* και την *αναζήτηση περιορισμένου βάθους* ο αλγοριθμος της αμφιδρομης αναζήτησης είναι πλήρης υπο τις συνθήκες: ο παράγοντας διακλάδωσης b να είναι πεπερασμένος (για τον BFS) και να ισχυει $l \geq d$ οπου l ειναι το οριο βάθους και d είναι το βάθος της λυσης (για τον DLS). Δεν είναι βέλτιστος, γιατι στην *αναζήτηση περιορισμένου βάθους* αν το όριο βάθους είναι μεγαλύτερο από το βάθος του πιο ρηχού κόμβου στόχου (δηλαδή $l \geq d$) τότε επεκτείνεται μια πολύ μεγαλύτερη διαδρομή από την βέλτιστη.

β)Με υπορουτινες την *αναζήτηση με επαναληπτική εκβάθυνση* και την *αναζήτηση περιορισμένου βάθους* ο αλγοριθμος της αμφιδρομης αναζήτησης είναι πλήρης υπο τις συνθήκες: ο παράγοντας διακλάδωσης b να είναι πεπερασμένος (διοτι ουσιαστικα συμπεριφερεται σαν τον BFS) και να ισχυει $l \geq d$ οπου l ειναι το οριο βάθους και d είναι το βάθος της λυσης (για τον DLS). Όμοια με το α) ερώτημα δεν είναι βέλτιστος, γιατι στην *αναζήτηση περιορισμένου βάθους* αν το όριο βάθους είναι μεγαλύτερο από το βάθος του πιο ρηχού κόμβου στόχου (δηλαδή $l \geq d$) τότε επεκτείνεται μια πολύ μεγαλύτερη διαδρομή από την βέλτιστη.

γ)Με υπορουτινες τον *Α** και την *αναζήτηση περιορισμένου βάθους* ο αλγοριθμος της αμφιδρομης αναζήτησης είναι πλήρης υπο τις συνθήκες: ο γράφος να είναι πεπερασμένος, τα βάρη των ακμών να μην ειναι αρνητικά (για τον Α*) και να ισχυει $l \geq d$ οπου l ειναι το οριο βάθους και d είναι το βάθος της λυσης (για τον DLS). Όμοια με το α) ερώτημα δεν είναι βέλτιστος, γιατι στην *αναζήτηση περιορισμένου βάθους* αν το όριο βάθους είναι μεγαλύτερο από το βάθος του πιο ρηχού κόμβου στόχου (δηλαδή $l \geq d$) τότε επεκτείνεται μια πολύ μεγαλύτερη διαδρομή από την βέλτιστη.

δ)Και με τις δυο υπορουτινες να έχουν τον *A** ο αλγοριθμος της αμφιδρομης αναζήτησης είναι πλήρης υπο τις συνθήκες: ο γράφος να είναι πεπερασμένος, τα βάρη των ακμών να μην ειναι αρνητικά. Είναι βέλτιστος με την προϋπόθεση όμως οτι οι ευρετικές συναρτήσεις που χρησιμοποιούνται στους Α* να είναι συνεπής, γιατι τότε μόνο οι Α* εγγυώνται πως θα βρουν ένα βέλτιστο μονοπάτι.

α) Η αναζήτηση περιορισμένου βάθους δεν εχει εξερευνημενο σύνολο, οπότε για να κανω αποδοτικα τον έλεγχο για το αν οι δυο αναζητήσεις συναντιούνται ελέγχω κάθε φορά έναν-έναν τους απογόνους του τρέχον κομβου αν ανήκουν στο εξερευνημενο σύνολο της αναζήτησης πρώτα σε πλάτος.

β)Η αναζήτηση με επαναληπτική εκβάθυνση και η αναζήτηση περιορισμένου βάθους δεν έχουν εξερευνημένα σύνολα, οπότε για να κανω αποδοτικα τον έλεγχο για το αν οι δυο αναζητήσεις συναντιούνται πρεπει καθε φορα να κρατάω σε ενα προσωρινο set τους κομβους του τελευταίου επιπέδου από την αναζήτηση με επαναληπτική εκβάθυνση, το set θα ανανεώνεται σε κάθε νέα επανάληψη της αναζητησης με τους νεους κομβους χωρις να κρατάει τους προηγουμενους. Συνεπώς, τώρα μπορω να ελέγχω κάθε φορά έναν-έναν τους απογόνους του τρέχον κομβου της αναζητησης περιορισμένου βάθους αν ανήκουν στο παραπάνω set.

γ) Ομοια με το (α), η αναζήτηση περιορισμένου βάθους δεν εχει εξερευνημενο σύνολο, οπότε για να κανω αποδοτικα τον έλεγχο για το αν οι δυο αναζητήσεις συναντιούνται ελέγχω κάθε φορά έναν-έναν τους απογόνους του τρέχον κομβου αν ανήκουν στο εξερευνημενο σύνολο της Α*.

δ) Η Α* εχει εξερευνημενο σύνολο, οπότε για να κανω αποδοτικα τον έλεγχο για το αν οι δυο αναζητήσεις συναντιούνται ελέγχω αν υπάρχει τουλάχιστον ένας κοινός κομβος ανάμεσα στα εξερευνημένα σύνολα.