Οικονομικό Πανεπιστήμιο Αθηνών, Τμήμα Πληροφορικής

Μάθημα: Τεχνητή Νοημοσύνη

Ακαδημαϊκό έτος: 2020–21

1 η Προγραμματιστική εργασία

Μέλη:

Κοτσιφός Γεώργιος (ΑΜ:3190093)

Μάρκο Κωνσταντίνος (ΑΜ:3190112)

Γκατζής Βασίλειος (ΑΜ:3190043)

Περιγραφή Κώδικα Κλάσης Cannibals (Main)

Το πρόγραμμα ζητάει από τον χρήστη τρεις παραμέτρους. Ο πρώτος είναι ο αριθμός των κανιβάλων και των ιεραποστόλων(δηλαδή αν δώσει π.χ. το 5, ο αριθμός των κανιβάλων και των ιεραποστόλων συνολικά σε μία όχθη θα είναι 5+5=10), ο δεύτερος είναι ο αριθμός της χωρητικότητας της βάρκας και ο τρίτος ο μέγιστος αριθμός διασχίσεων. Οι δύο πρώτοι αριθμοί περνιούνται στην αρχική κατάσταση που δημιουργείται. Μόλις δημιουργείτε η αρχική κατάσταση καλείτε η Searcher ώστε να βρει το επιθυμητό μονοπάτι της λύσης. Εάν δεν υπάρχει τέτοιο μονοπάτι δηλαδή δεν υπάρχει λύση για τις δοσμένες παραμέτρους τότε η Searcher επιστρέφει την τιμή “null”, το πρόγραμμα τερματίζει και εμφανίζει το κατάλληλο μήνυμα. Διαφορετικά ελέγχει αν το μονοπάτι που βρήκε είναι μικρότερο ή ίσο με τον μέγιστο επιτρεπόμενο αριθμός διασχίσεων του ποταμού και σε αύτη την περίπτωση εκτυπώνει την λύση. Η βάρκα ξεκινάει πάντα από την αριστερή όχθη.

Περιγραφή Κώδικα Κλάσης State

Αναπαριστά τις καταστάσεις του προβλήματος

<u>Ορίσματα:</u>

private int cannibalLeft: ο αριθμός των κανιβάλων στην αριστερή όχθη

private int missionaryLeft: ο αριθμός των ιεραποστόλων στην αριστερή όχθη

private int cannibalRight: ο αριθμός των κανιβάλων στην δεξιά όχθη

private int missionaryRight: ο αριθμός των ιεραποστόλων στην δεξιά όχθη

private Position boat: η θέση της βάρκας στο ποτάμι(είτε δεξιά είτε αριστερά)

private int boatCapacity: το μέγεθος της βάρκας

private State father = null: Μεταβλητή που δείχνει στην προερχόμενη κατάσταση

private int score: το εκτιμώμενο σκορ της ευρετικής

private int g: το κόστος μετάβασης από την αρχική μετάβαση έως την τωρινή κατάσταση

Μέθοδοι:

- public State(int cannibalLeft, int missionaryLeft, Position boat, int cannibalRight, int missionaryRight, int boatCapacity, int g): κατασκευαστής με ορίσματα μια άλλης κατάστασης
- void print(): εκτυπώνει την κατάσταση(αριθμός κανιβάλων αριστερά, αριθμός ιεραποστόλων αριστερά, θέση βάρκας, αριθμός κανιβάλων δεξιά, αριθμός ιεραποστόλων δεξιά)
- public ArrayList<State> getChildren():Παράγει τις καταστάσεις για όλες τις δυνατές μεταβάσεις με μια διάσχιση. Ο τρόπος που το κάνει είναι αρχικά ελέγχοντας την θέση της βάρκας διότι ανάλογα η θέση της βάρκας τότε και ανάλογη θα είναι και η μεταφορά των κανιβάλων και τον ιεραποστόλων στην αντίπερα όχθη. Έπειτα με την χρήση εμφολευμένων for loop παράξαμε όλες τις πιθανές κινήσεις των κανιβάλων και τον ιεραποστόλων ανάλογα με το μέγεθος της βάρκας. Ο τρόπος που έγινε είναι με την χρήση της μιας for loop θα αναπαριστάτε ο αριθμός που θα διασχίσει κάθε φορά των ποταμό των κανίβαλων και με της άλλης for loop ο αριθμός των ιεραποστόλων. Δηλαδή κάθε φορά θα περνάνε 0 κανίβαλοι-1 ιεραπόστολος, 0 κανίβαλοι-2 ιεραπόστολοι… 1 κανίβαλος-0 ιεραπόστολοι και ούτω κάθε εξής. Χρησιμοποιήσαμε ένα if statement για να βγάλουμε ανεπιθύμητες τιμές π.χ. να διασχίσουν ταυτόχρονα 0 κανίβαλοι και 0 ιεραπόστολοι διότι θα είχαμε θέμα με το κλειστό σύνολο (!(i==0 && j==0)), π.χ αν η χωρητικότητα της βάρκας ήταν 3 τότε να μην είχαμε στα for loops i+j>3 (όπου i ο αριθμός των κανιβάλων και j ο αριθμός των ιεραποστόλων, (i+j<boatCapacity+1)) και τέλος κρατήσαμε και τον περιορισμό της εκφώνησης που λέει ότι στη βάρκα δεν μπορεί να έχουμε μεγαλύτερο αριθμό κανιβάλων από ότι ιεραποστόλων (προφανώς έκτος και αν οι ιεραπόστολοι είναι 0) ((j==0 | i<=j)).
- public boolean isValid(): ελέγχει εάν η κατάσταση υπακούει στους περιορισμούς της άσκησης
- private void testAndAdd(ArrayList<State> successors, State stateChildren): μέθοδος που δέχεται από την getChildren() λίστα και ένα παραγόμενο παιδί. Αρχικά ελέγχει αν το παιδί που παράχθηκε είναι valid και αποθηκεύει την παραγόμενη καταστάσεις κάνοντας την παλιά γονέα. Τέλος εδώ χρησιμοποιείτε η ευρετική συνάρτηση και αυξάνετε το βάθος κατά 1 καθώς προσθέσαμε ένα παιδί.
- Χρήση getter και setter.
- public boolean equals(Object obj): για να ελέγξουμε αν δύο καταστάσεις είναι ίδιες μεταξύ τους
- public int hashCode(): μέθοδος που χρησιμοποιείτε για το hashset στην state
- int identifier(): μοναδικός αριθμός που χρησιμοποιείτε στην hashet. Υπολογίζεται αθροίζοντας της δυνάμεις των αριθμών των κανιβάλων και των ιεραποστόλων που βρίσκονται κάθε φορά σε μια συγκεκριμένη όχθη.
- public int compareTo(State s): μέθοδος που χρειάζεται για τον A*
- private void heuristic(): ευρετική συνάρτηση. Χρησιμοποιήσαμε αυτή που κάναμε στο φροντιστήριο δηλαδή αφαιρέσαμε τον περιορισμό ότι οι ιεραπόστολοι κάθε φορά πρέπει να είναι μικρότεροι ή ίσοι με τους ιεραποστόλους. Είμαστε σίγουροι ότι υποεκτιμάτε ο αριθμός των διασχίσεων που χρειάζεται η βάρκα και στην χειρότερη είναι ίσος. Τέλος είναι και συνεπής.

Κλάση Searcher- Μέθοδοι Τεχνητής Νοημοσύνης

Παράγει την λύση του προβλήματος ,και περιέχει την Μέθοδο Τεχνητής Νοημοσύνης A*

Πήραμε τον αλγόριθμο BestFS που είδαμε στα φροντιστήρια και τον μετατρέψαμε σε A* χρησιμοποιώντας την απόσταση της αρχικής κατάστασης έως την τωρινή κατάσταση. Το συνολικό κόστος υπολογίζεται στην μέθοδο heuristic όπου προσθέτουμε την μεταβλητή “g” στην τιμή της ευρετικής όπου την βρίσκουμε με τον τρόπο που περιγράψαμε παραπάνω. Έτσι η ευρετική είναι το άθροισμα των δυο αυτών.

Αποτελέσματα διαφορετικών εισόδων:

Αριθμός Κανιβάλων/ Ιεραποστόλων	**Μέγιστη χωρητικότητα της βάρκας**	**Αριθμός Διασχίσεων**	**Αποτέλεσμα**	**Χρόνος**
3	2	15	Βρέθηκε Λύση	0.001 sec
3	2	10	Δεν βρέθηκε λύση	0.001 sec
35	4	100	Βρέθηκε Λύση	0.003 sec
67	20	20	Βρέθηκε Λύση	0.005 sec
3600	180	147	Βρέθηκε Λύση	3.734 sec
790	34	28	Δεν βρέθηκε λύση	0.061 sec
15000	9	100000	Βρέθηκε Λύση	12.519 sec
15000	47	1337	Βρέθηκε Λύση	22.987 sec
100	200	3	Βρέθηκε Λύση	0.003 sec

Παράδειγμα με 3 κανίβαλους και ιεραποστόλους, χωρητικότητα βάρκας 2 και αριθμός διασχίσεων 15:

```
C:\Users\deepblueibm\AppData\Local\Programs\AdoptOpenJDK\bin\java.exe
Dose ton arithmo ton kanivalon kai ton ierapostolon:
3
Dose thn megisth xoritikotita varkas:
2
Dose ton megisto epitrepomeno arithmo diasxiseon toy potamoy:
15
(3,3,L,0,0)
(2,2,R,1,1)
(2,3,L,1,0)
(0,3,R,3,0)
(1,3,L,2,0)
(1,1,R,2,2)
(2,2,L,1,1)
(2,0,R,1,3)
(3,0,L,0,3)
(1,0,R,2,3)
(1,1,L,2,2)
(0,0,R,3,3)

Solution Found!
Search time:0.0 sec.

Process finished with exit code 0
```