Τεχνητή Νοημοσύνη και Έμπειρα Συστήματα

Προαιρετική Εργασία

Αθανάσιος Παραβάντης

Π16112

thanosparavantis@gmail.com

Περιεχόμενα

Εκφώνηση

2019 – Θέμα προαιρετικής εργασίας για το μάθημα «Τεχνητή Νοημοσύνη και Έμπειρα Συστήματα». Η εργασία είναι ατομική. Ημερομηνία παράδοσης είναι η 31η Μαίου 2019.

Αναπτύξτε πρόγραμμα επίλυσης του προβλήματος των ιεραποστόλων και κανιβάλων με χρήση ενός τυφλού και ενός ευρετικού αλγορίθμου της επιλογής σας και σε γλώσσα προγραμματισμού της επιλογής σας.

Παραδοτέα της εργασίας είναι μία σύντομη αναφορά που να περιλαμβάνει τον τρόπο δράσης του υπολογιστή σύμφωνα με τον αλγόριθμο επίλυσης και παραδείγματα εκτέλεσης του προγράμματος που αναπτύξατε. Ημερομηνία παράδοσης είναι η 31η Μαίου 2019.

Περιγραφή

Το πρόβλημα των ιεραπόστολων και των κανίβαλων είναι ένα από τα κλασσικά ζητήματα που απασχολούν τον τομέα της τεχνητής νοημοσύνης. Αρχίζουμε με τρεις ιεραπόστολους και τρεις κανίβαλους που θέλουν να περάσουν στην αντίπερα όχθη ενός ποταμιού με τη βοήθεια μιας βάρκας. Οι κανόνες του προβλήματος έχουν ως εξής: (α) οι ιεραπόστολοι πρέπει να είναι περισσότεροι από τους κανίβαλους και στις δυο όχθες, (β) η βάρκα δεν μπορεί να κινηθεί από μόνη και (γ) η βάρκα μπορεί να μεταφέρει από ένα έως δυο άτομα χωρίς να μας ενδιαφέρει αν είναι ιεραπόστολοι ή κανίβαλοι.

Για την επίλυση του προβλήματος χρησιμοποιούμε τη γλώσσα προγραμματισμού Java ώστε να καθορίσουμε αυστηρές δομές και αλγορίθμους που μοντελοποιούν το πρόβλημα αποτελεσματικά. Η προσέγγιση του θέματος προγραμματιστικά προϋποθέτει τον ορισμό της τετράδας: $P = (I, G, T, S)$ αρχική κατάσταση, τελικές καταστάσεις, τελεστές μετάβασεις και χώρος καταστάσεων. Ο τυφλός αλγόριθμος **Breadth First Search (BFS)** επιτυγχάνει την εύρεση τελικής κατάστασης άρα και λύσης ενώ ο ευριστικός αλγόριθμος **Hill Climbing Search** αποδεικνύεται πως δεν είναι αποτελεσματικός.

Εκτέλεση

Αλγόριθμος Best First Search

```
Missionaries and Cannibals
Thanos Paravantis - P16112
thanosparavantis@gmail.com

Applying: Breadth Firsrt Search Algorithm

Visiting:
ooo xxx <===>           ___ ___
Visiting:
ooo xx_          <===> ___ X__
Visiting:
oo_ xx_          <===> O__ X__
Visiting:
ooo x__          <===> ___ XX_
```

(αρκετές ενδιάμεσες καταστάσεις...)

```
Visiting:
ooo ___ <===>           ___ XXX
Visiting:
o__ x__ <===>           OO_ XX_

End State:
___ ___           <===> OOO XXX
```

Αλγόριθμος Hill Climbing

```
Applying: Hill Climbing Search Algorithm

Current:
ooo xxx <===>           ___ ___
Current score: 0
Next:
oo_ xx_          <===> O__ X__
Next score: 2
Moving into the next best state...
Current:
oo_ xx_          <===> O__ X__
Current score: 2
Next:
ooo xx_ <===>           ___ X__
Next score: 1
Current score greater than next, stopping...

End State:
oo_ xx_          <===> O__ X__
```

Για να εκτελέσουμε το πρόγραμμα εκτελούμε την εντολή:

```
java -jar missionaries-cannibals.jar
```

Χρησιμοποιήθηκε η Java 11 για τη δημιουργία του εκτελέσιμου αρχείου, που βρίσκεται στον φάκελο της εργασίας.

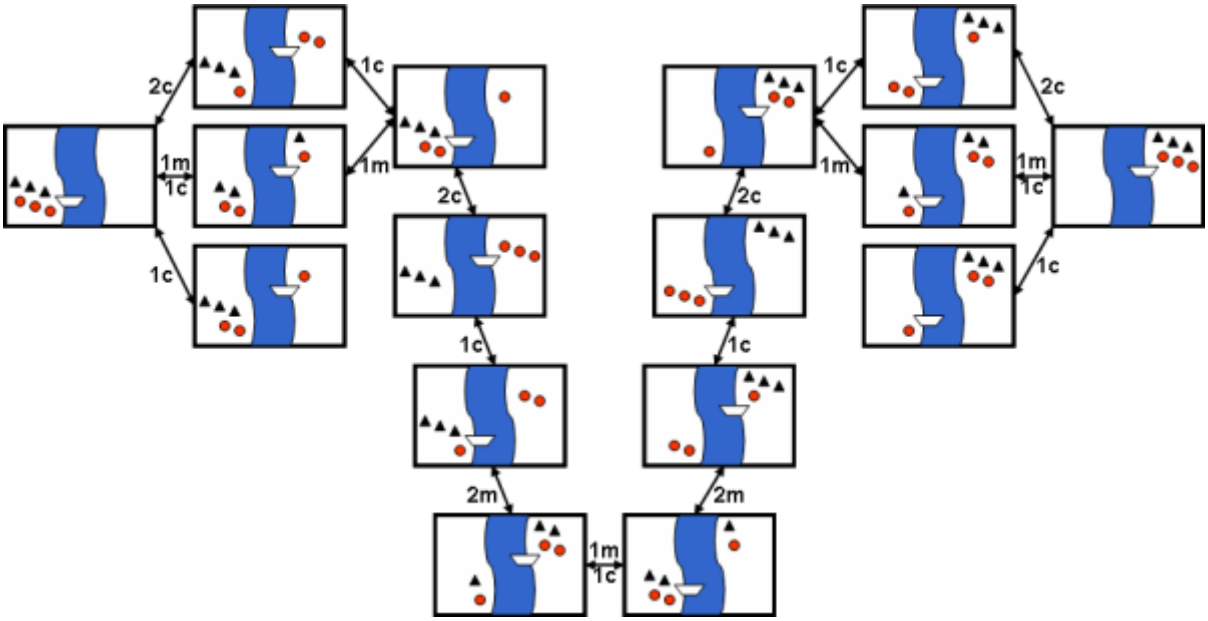
Παρατηρούμε ότι ο πρώτος αλγόριθμος μετά από την εφαρμογή των τελεστών μετάβασης καταλήγει σε τελική κατάσταση, στην οποία οι ιεραπόστολοι και οι κανίβαλοι είναι στην αντίπερα όχθη του ποταμιού.

Αντιθέτως, στον δεύτερο αλγόριθμο βλέπουμε ότι ενώ εφαρμόστηκαν μερικοί τελεστές μετάβασης, εν τέλει δεν κατέληξε σε επιθυμητή τελική κατάσταση. Αυτό συμβαίνει επειδή οι ευριστικοί αλγόριθμοι δεν μπορούν να εφαρμοστούν αποτελεσματικά για το πρόβλημά μας.

(τα αποτελέσματα της εκτέλεσης όλου του προγράμματος βρίσκονται στο αρχείο sample_run.txt)

Επεξήγηση

Η λύση του προβλήματος χρησιμοποιώντας τον κατάλληλο αλγόριθμο ανά περίπτωση χρησιμοποιεί το εξής σύνολο καταστάσεων:



Πηγή: http://www.aiai.ed.ac.uk/~gwickler/missionaries.html

Για να μοντελοποιήσουμε αυτές τις περιπτώσεις έχουμε ορίσει τη βασική κλάση **`State`** η οποία κρατάει τα δεδομένα που χαρακτηρίζουν κάθε κατάσταση. Έπειτα, για την εύρεση των μεταβάσεων η κλάση **`StateSpace`** καθορίζει το σύνολο των έγκυρων καταστάσεων παιδιών στις οποίες μπορούμε να μεταβούμε. Το σύνολο αυτό δημιουργείται με τη κλάση **`Operators`** που παρέχει βασικές μεθόδους-τελεστές που μεταβάλλουν τα στοιχεία μιας κατάστασης (πχ. πήγαινε έναν ιεραπόστολο από αριστερά στα δεξιά). Συνδυάζοντας όλα τα παραπάνω στις τάξεις **`BreadthFirstSearch`** και **`HillClimbingSearch`** μπορούμε να υλοποιήσουμε τους αλγορίθμους με ευκολία και αποτελεσματικότητα.