Δομές Δεδομένων 2018
Εργασία Γ
«Φιδάκι»

Υπατία Δάμη
8606
ypatiadm@gmail.com

6977960438

Δημήτρης Κουτζιαμπάσης
8989
dkoutzia@ece.auth.gr
6979023896

Περιγραφή προβλήματος

Στη 3η εργασία καλούμαστε να υλοποιήσουμε την κλάση MinMaxPlayer καθώς και να τροποποιήσουμε την συνάρτηση Game, έτσι ώστε να παίζουν αντίπαλοι ένας παίκτης Player και ένας MinMaxPlayer.
Η κλάση MinMaxPlayer περιέχει τις συναρτήσεις της κλάσης HeuristicPlayer,με ανανεωμένες τις συναρτήσεις evaluate() και getNextMove() .Για αυτή την κλάση καλούμαστε να υλοποιήσουμε επίσης τις συναρτήσεις createMySubTree(),creatOpponentSubTree() και chooseMinMaxMove().Το δέντρο που θα κατασκευάσουμε θα έχει βάθος 2 κινήσεων, μιας δικής μας και μιας του αντιπάλου. Η συνάρτηση createMySubTree() κατασκευάζει το πρώτο επίπεδο του δέντρου που αντιπροσωπέυει τις πιθανές μας κινήσεις στο τρέχον ταμπλό του παιχνιδιού , ενώ η createOpponentSubTree() , που καλείται μέσα στην createMySubTree() κατασκευάζει το δεύτερο επίπεδο του δέντρου που αντιπροσωπεύει τις πιθανές κινήσεις του αντιπάλου για κάθε μια από τις δικές μας προηγούμενες κινήσεις.Τέλος με την chooseMinMaxMove() ,μέσα από το δέντρο που φτιάξαμε επιλέγουμε την καλύτερη ζαριά ,έχοντας καταφέρει να προβλέψουμε σε βάθος 2 κινήσεων την καλύτερη κίνηση για μας, σύμφωνα και με τις πιθανές επιλογές του αντιπάλου.

Περιγραφή Αλγορίθμου

Η συνάρτηση createMySubTree() δημιουργεί το πρώτο επίπεδο του δέντρου μας,δηλαδή τους κόμβους που αντιπροσωπεύουν τις κινήσεις του MinMaxPlayer και στο τέλος καλεί την createOpponentSubTree(), η οποία δημιουργεί το δέυτερο επίπεδο του δέντρου που αφορά τις κινήσεις του αντιπάλου. Σε κάθε επανάληψη του βρόχου for, που έχει τόσες επαναλήψεις όσες και οι πιθανές μας ζαριές , δημιουργείται η μεταλβητή simBoard όπου αρχικά έχει τιμη το ταμπλό του κόμβου parent,δηλαδή αυτό του τρέχοντος παιχνιδιού.Δημιουργούμε έναν simPlayer (simulation) , με όρισμα το simBoard και τον κινούμε σύμφωνα με τη ζαριά “i” για να ανανεωθεί η τιμή του simBoard σύμφωνα με τη συγκεκριμένη κίνηση .Δημιουργούμε έναν κόμβο child που αντιπροσωπέυει την κίνηση “i”,με όρισμα την ανανεωμένη τιμή του simBoard και στη συνέχεια προσθέτουμε τον κόμβο –κίνηση στον κόμβο parent και για τον κάθε έναν από τους κόμβους child που δημιουργήσαμε δημιουργόυμε το δεύτερο επίπεδο του δέντρου, που περιέχει 6 κινήσεις για κάθε κόμβο child

Η συνάρτηση createOpponentSubtree() δημιουργέι το δεύτερο επίπεδο του δέντρου . Για κάθε κίνηση μας(κομβο child) δέχεται ως παιδιά τις πιθανές κινήσεις του αντιπάλου. Δημιουργείται ένα αντικείμενο Board με όνομα simBoard με την τιμή του simBoard του κόμβου που αφορά την κίνηση μας και έναν παίκτη simPlayer. Μέσα σε βρόχο με 6 επαναλήψεις δημιουργούμε τον κόμβο child που αντιπροσωπεύει την κίνηση του αντιπάλου για την συγκεκριμένη ζαριά για την οποία κλήθηκε η createOpponentSubtree() από την createMySubtree().Καλείται η evaluate() από τον simPlayer για τον κόμβο αυτόν , με όρισμα τη θέση του αντιπάλου και το ανανεωμένο simBoard (σε σχέση με το Board του κανονικού παιχνιδιού , το simBoard έχει υποστεί μια δική μας κίνηση “i” και ο αντίπαλος καλεί την evaluate με όρισμα το συγκεκριμένο ταμπλό , το οποίο ενδέχεται να διαφέρει στο προηγούμενο σε σκάλες ή στα μήλα.)Παίρνουμε την αρνητική τιμή της evaluate επειδή εκτιμούμε τις κινήσεις του αντιπάλου και θέλουμε την μικρότερη τιμή της . Τέλος , προσθέτουμε τον κόμβο –κίνησή του αντιπάλου ως παιδί στον κόμβο-κινισή μας .Κάθε κόμβος που αντιπροσωπεύει μια από τις πιθανές μας κινήσεις αποκτά έως και 6 κόμβους –κινήσεις του αντιπάλου.

Η συνάρτηση chooseMinMaxMove() επιλέγει την καλύτερη κίνηση για τον MinMaxPlayer χρησιμοποιώντας την εκτίμηση της evaluate για την κίνηση του αντιπάλου στο δεύτερο επίπεδο του δέντρου που δημιουργήσαμε . Στον εξωτερικό βρόχο for , θέτουμε έναν μετρητή i με αριθμό έως τον αριθμό των διαθέσιμων κινήσεων μας για κάθε μία από τις δυνατές μας κινήσεις και στον εσωτερικό βρόχο for θέτουμε έναν μετρητή j με τιμή έως τον αριθμό των διαθέσιμων κινήσεων του αντιπάλου,δεδομένου ότι εμείς παίξαμε την κίνηση i.Στη μεταβλητή eval είναι αποθηκευμένη η τιμή που επέστρεψε η evaluate() για την κίνση του αντιπάλου και χρησιμοποώντας την μεταβλητή worstEval .Αποθηκεύουμε στο τέλος του εσωτερικού βρόχου for κάθε "ί" επανάληψης την χειρότερη κίνηση που θα κάνει ο αντίπαλος εις βάρος μας.(Δεδομένου ότι καλούμε την evaluate με - μπροστά επιλέγουμε την μικρότερη τιμή που επιστρέφει για τις πιθανές κινήσεις ως χειρότερη κίνηση. Καλέσαμε την evaluate έτσι ώστε το μείον άπειρο να αντιπροσωπεύει την νίκη του αντιπάλου. Ύπειτα στο τέλος της επανάληψης του εξωτερικού βρόχου ελέγχουμε με την μεταβλητή bestEval και αποθηκεύουμε την μεγαλύτερη τιμή από αυτές που αποκτά η worstEval , δηλαδή διαλέγουμε την καλύτερη για εμάς κίνηση .Ταυτόχρονα αποθηκεύουμε στην bestDice την τιμή της συγκεκριμένης ζαριάς και επιστρέφουμε στο τέλος την βέλτιστη ζαριά bestDice.

Η συνάρτηση getNextMove() αρχικά δημιουργεί τον κόμβο root με όρισμα το ταμπλό του τρέχοντος παιχνιδιού . Έπειτα δημιουργεί το δέντρο βάθους δύο κινήσεων και αποθηκεύει στην μεταβήτή bestDice την καλύτερη ζαριά της συυνάρτησης chooseMinMaxMove().Καλούμε την την συνάρτηση move() για τον παίκτη με όρισμα την bestDice και ενημερώνουμε τις μεταβλητές STEPS και GAINPOINTS . Δημιουργούμε τον πίνακα roundInfo[] για να προσθέσουμε τις πληροφορίες του γύρου. Στις θέσεις 5,6,7 αποθηκεύονται οι μεταβλητές STEPS,BestDice,GAINPOINTS που ενημερώθηκαν ποηγουμένως,ενώ στις πρώτες θέσεις οι τιμές του πίνακα που επέστρεψε η move().Προσθέτουμε τον πίνακα roundInfo στο path και επιστρέφουμε την καινούρια θέση του παίκτη (roundInfo[0]).Επειδή στο συγκεκριμένο παιχνίδι παίζει ένας MinMaxPlayer και ένας απλός Player, δεν επηρεάζεται συχνά η κίνηση του ενός από του άλλου λόγω του γρήγορου προβαδίσματος που αποκτά ο minMaxPlayer και του ότι υπάρχουν πολύ λιγες σκάλες και μήλα στο ταμπλό. Επομένως για την ορθή λειτουργία του προγράμματος σε περίπτωση που η chooseMinMaxMove επιστρέψει -1, δηλαδή οι εκτιμήσεις της evaluate() ήταν ίδιες για όλες τις ζαριές, τότε ο παίκτης καλεί την

evaluate και επιλέγει την καλύτερη κίνηση χωρίς να υπολογίζει την κίνηση του αντιπάλου.