2η Εργασία Τεχνητής Νοημοσύνης Ι

READ_ME για το Project 2: Multi-Agent Search:

Ονοματεπώνυμο: Απόστολος Καρβέλας

A.M.:1115201800312

Διαχειριζόμαστε το αρχείο multiAgents.py

QUESTION 1: Reflex Agent (ReflexAgent)

Το πρόγραμμα υλοποιεί την συνάρτηση evaluationFunction της κλάσης ReflexAgent η οποία επιστρέφει ένα νούμερο ανάλογα το πόσο καλή είναι μια κίνηση του πακμαν για να κερδίσει το παιχνίδι. Συγκεκριμένα, το πρόγραμμα θα διαλέγει την κίνηση με την μικρότερη τιμή που θα επιστρέφει η συνάρτηση. Για τον υπολογισμό της τιμής αυτής πρέπει να πάρει υπόψιν του τις τοποθεσίες του φαγητών όσο και των φαντασμάτων, οπότε η συνάρτηση βρίσκει τις αποστάσεις από όλα τα φαγητά και τα φαντάσματα και κρατάει εκείνα που βρίσκονται πιο μακριά και κοντά σε αυτόν για τις αποστάσεις των φαγητών ενώ κρατάει μόνο εκείνο το φάντασμα που είναι κοντά του. Στην συνέχεια επιστρέφει το getScore() προσθέτοντας την απόσταση του πιο κοντινού φαντάσματος και το μειώνουμε με τις 2 αποστάσεις των φαγητών εκτός αν είναι το τελευταίο οπότε το μειώνουμε μόνο με μια.

QUESTION 2: MINIMAX (MinimaxAgent)

Για την 2η ερώτηση της εργασίας πρέπει να υλοποιηθεί ένας αλγόριθμος minimax για να καθορίσει ποια κατεύθυνση θα πάρει το πακμαν. Οπότε δημιουργούμε 2 νέες συναρτήσεις την minvalue και την maxvalue για να βοηθήσουν στον υπολογισμό της καλύτερης διαδρομής. Η minvalue επιστρέφει την τιμή της κίνησης αυτής μέσω της συνάρτησης scoreEvaluationFunction αν βρίσκεται στην κατάσταση στόχου του ή αν έχει μηδενικό βάθος. Στην συνέχεια, υπολογίζει την μικρότερη τιμή για κάθε κίνηση που παίρνουν όλα τα φαντάσματα σε αυτό το επίπεδο καλώντας αναδρομικά την συνάρτηση minvalue και για την τελευταία περίπτωση, δηλαδή για το τελευταίο φάντασμα, καλεί την maxvalue στο επόμενο βάθος. Η maxvalue είναι παρόμοια με την minvalue αλλά βρίσκει την μέγιστη τιμή για κάθε κίνηση του πακμαν, τρέχοντας την minvalue σε αυτό το επίπεδο, αρχίζοντας από το πρώτο φάντασμα. Τέλος, στην συνάρτηση getAction υπολογίζει τους min κόμβους μέσω της minvalue για κάθε κίνηση του πακμαν και βρίσκει την μέγιστη τιμή επιστρέφοντας το action με την τιμή αυτήν.

QUESTION 3: Alpha-Beta Pruning (AlphaBetaAgent)

Για την δημιουργία του αλγόριθμου alpha-beta pruning θα χρησιμοποιήσουμε τον minimax από το q2 προσθέτοντας 2 μεταβλητές alpha και beta που θα περνιούνται σαν ορίσματα στις συναρτήσεις. Η μεταβλητή beta εκπροσωπεύει την καλύτερη περίπτωση για τα φαντάσματα ενώ η alpha την καλύτερη περίπτωση για τον παίκτη. Οπότε, για το σωστό κλάδεμα του δέντρου η minvalue, στην επανάληψη που υπολογίζει την ελάχιστη τιμή για κάθε κίνηση προσθέτουμε ένα if statement και όταν η τιμή είναι μικρότερη του alpha δεν έχει σημασία να συνεχίσει το ψάξιμο οπότε επιστρέφει την τιμή που έχει βρει και στο τέλος υπολογίζει το νέο beta ως την ελάχιστη τιμή μέχρι στιγμής. Το ίδιο γίνεται και για την maxvalue αλλά βλέπει αν η τιμή είναι μεγαλύτερη του beta και στο τέλος υπολογίζει το alpha ως την μεγαλύτερη τιμή. Τέλος, στην getAction όπως και πριν υπολογίζει την τιμή για τους κόμβους min αλλά ταυτόχρονα υπολογίζει και το νέο alpha και λειτουργεί ως ρίζα κόμβος του δέντρου.

QUESTION 4: Expectimax (ExpectimaxAgent)

Η υλοποίηση του αλγορίθμου expectimax είναι παρόμοια με το minimax του q2 με διαφορά την minvalue που γίνεται expvalue. Για την εύρεση του expected value αντί να βρίσκει την ελάχιστη τιμή για κάθε κίνηση των φαντασμάτων, προσθέτει όλες τις τιμές των παιδιών για να υπολογίσει τον μέσο όρο, δηλαδή την τιμή που αναμένουμε.

QUESTION 5: Evaluation Function (betterEvaluationFunction)

Η betterEvaluationFunction είναι ίδια με την evaluationFunction της q1 με την διαφορά ότι παίρνει υπόψιν του και τα power-ups. Συγκεκριμένα, αν το πακμαν έχει πάρει power-up και πάνω από 16 κινήσεις για να φάει τα φαντάσματα τότε επιστρέφει το getScore() προσθέτοντας μόνο την απόσταση από το πιο κοντινό φάντασμα.