pacman readMe sdi1900085

Pacman Project 2: Multi-Agent Search

Q1: Reflex Agent
Στον reflex agent η get action ψάχνει την επιτρεπτή κίνηση με το μεγαλύτερο σκορ οπότε στην evaluation επιστρέφω το κόστος της κίνησης αυτής με τα παρακάτω κριτήρια: Κρατάω μια λίστα με όλες τις θέσεις που υπάρχει φαγητό για να ελέγξω εαν ο pacman πέφτει πάνω σε φαγητό με την επόμενη κίνηση, αν συμβαίνει αυτό το σκορ της κίνησης αυξάνεται κατά 100. Πριν από αυτό βρίσκω το κοντινότερο φάντασμα που δεν ειναι scared και σιγουρεύω ότι δεν θα είναι πολύ κοντά, εάν είναι μειώνω το σκορ της κίνησης κατα 100. Τέλος αν δεν συμβαίνει τίποτα από τα παραπάνω το σκορ είναι 100 - την απόσταση του pacman απ την κοντινότερη τροφή.

Q2: Minimax
Για την υλοποίηση της minimax αρχικά δίνω ως παραμέτρους το gamestate που χρησιμοποιώ για την λύψη διαφορων πληροφοριών για την καθε κατάσταση, το βάθος, τον αριθμό του παίκτη και μια κίνηση που χρειάζεται για να κρατηθεί στο τέλος η κίνηση που θα επιστραφεί απ την συνάρτηση. Αρχικά ελένχω εάν έχουμε φτάσει στο πιο χαμηλό βάθος ή εάν το παιχνίδι έχει τελεώσει και σ αυτή την περίτωση επιστρέφω ένα tuple με το σκορ της κίνησης που χρειάζεται για συγκρίσεις εντώς της συνάρτησης και την κίνηση που χρειάζεται σε περίπτωση που αυτή είνα η τελευταία επιστροφή απ τη συνάρτηση. Έπειτα ανανεώνω τις ενδείξεις για το βάθος και τον παίκτη και ανάλογα άμα παίζει ο πακμαν η φαντασματάκι, για τον πακμαν (παικτης 0) ψάχνω την δυνατή κίνηση που θα επιφέρει το καλύτερο σκορ καλόντας την συνάρτηση αναδρομικά και πέρνοντας μόνο το πρώτο στοιχείο του tuple που επιστρέφει, το σκορ. Εάν η κίνηση είναι να σταματήσει προχωράω χωρίς έλενχο καθώς δεν θα αλλάξει κάτι για τον πακμαν. Τελικά επιστρέφω την καλύτερη δυνατή κίνηση και το σκορ της. Εάν παίζει φαντασματάκι (παίκτης > 0) ψάχνω την καλύτερη δυνατή κίνηση που ελαχιστοποιεί το σκορ καλόντας την συνάρτηση αναδρομικά και τελικά πάλι επιστρέφω την κίνηση και το σκορ της. Την συνάρτηση καλώ μέσα στην getAction με παραμέτρους το τωρινό gamestate, για βάθος 0, για τον παίκτη 0 (πακμαν) και δεν δίνω καμία κίνηση, επιστρέφω μόνο το δεύτερο στοιχείο που επιστρέφει, την κίνηση.

Q3: Alpha-Beta Pruning
Στην υλοποίηση του apha-beta πήρα την υλοποίηση του minimax και πρόσθεσα στην αναζήτηση της καλύτερης κίνησης από τον maximizer και minimizer agent αντίστοιχα παραπάνω ελένχους για τις a, b μεταβλητές τις οποίες πρόσθεσα και στις παραμέτρους της συνάρτησης για να κρατάω την τιμή τους. Για τον maximizer (πακμαν) ελένχω αν το max score ειναι <= του β κ αν δεν είναι επιστρέφω αμέσως το σκορ και την κίνηση κλαδεύοντας τον έλενχο για τις υπόλοιπες κινήσεις, αν ειναι ανανεώνω το α εάν το max score είναι μεγαλύτερο του. Στον minimizer αντίστοιχα ελένχω ότι το min score είναι >= του α αλλιως κλαδεύονται οι υπόλοιποι έλενχοι κινήσεων του φαντάσματος, αν είναι ανανεώνω το β εάν το min score είναι μικρότερό του. Η συνάρτηση καλείται μέσα στην getAction με τις ίδιες παραμέτρους όπως ο minimax και -άπειρο για τον α και άπειρο για τον β.

Q4: Expectimax
Για τον expectimax πήρα πάλι την υλοποίηση του minimax και άλλαξα μόνο τον έλενχο κινήσεων στον minimizer στον οποίο αντί για την επιλογή του καλύτερου σκορ απλά επιστρέφω τον μέσω όρο απ όλες τις πιθανές κινήσεις καθώς θεωρώ οτι ο minimizer είναι sub-optimal και θα επιλέξει κίνηση τυχαία.

Q5: Evaluation Function
Στην evaluation δημιούργησα ένα σύστημα σκορ και μετά απο διάφορες δοκιμές κατέληξα στις τιμές με τις οποίες ο πακμαν συμπεριφέρεται με αρκετά έξυπνο τρόπο. Για αρχή εάν η στην συηκεκριμένη κατάστσαση ο πακμαν χάνει επιστρέφω το χειρότερο σκορ -άπειρο ενώ άμα ο πακμαν κερδίζει επιστρέφω το καλύτερο δυνατό σκορ άπειρο. Έπειτα κρατάω κάποιες χρήσιμες τιμές όπως την θέση του πακμαν, τη λίστα με το φαγητό και την κατάσταση τον φαντασμάτων. Κρατάω επίσης τον αριθμό των φαγητών που απομένουν, των αριθμό κάψουλων καθώς θα είναι χρήσιμες παράμετροι για την εξέληξη του σκορ, θέτο επιπλέον τον αριθμό τον φοβισμένων και μη φαντασμάτων σε 0 και 0 αντίστοιχα. Ύπολογίζω την κοντινότερη απόσταση από φαγητό καθώς και από φαντασματάκι. Εάν υπάρχουν φοβισμένα φαντασματάκια άνεβάζω τον αντίστοιχο μετρητή και κρατάω την απόσταση απ το κοντινότερο, το ίδιο κάνω και για μη φοβισμένα φαντασματάκια για τα οποία άμα υπάρχει φαντασματάκι κοντινότερο απο απόσταση 2 επιστρέφω -άπειρο για να αποφευχθούν οι κινήσεις στις οποίες το πακμαν κινδυνεύει άμεσα. Αφού έχω κρατήσει όλα τα παραπάνω ξεκινάω να υπολογίζω το σκορ: προσθέτω όσα στοιχεία είναι του τύπου όσο μικρότερο τόσο καλύτερο μέσα στο κλάσμα 1/στοιχείο +1 το οποίο είναι πιο κοντά στο 1 όσο μικρότερο είναι το στοιχείο αλλίως όσο μεγαλύτερο είναι τόσο μικρότερος ο αριθμός απ το 1. Αυτά τα κλάσματα τα πολλαπλασιάζω με έναν συντελεστή για να μην περιορίζει το σκορ με ταβάνι το 1. Άμα υπάρχουν μη φοβισμένα φαντάσματα και είναι πιο κοντά απο απόσταση 4 τότε προσθέτω την απόσταση σκέτη καθώς όσο μικρότερη τόσο το χειρότερο και μετά πολλαπλασιάζω με - την απόσταση μέσα στο παραπάνω κλάσμα πολλαπλασιασμένο επί 100 έτσι ωστε το σκορ να μειώνεται εκθετικά ολοένα και πιο δραματικά όσο πιο κοντά βρίσκονται φαντασματάκια. Εάν υπάρχουν φοβισμένα φαντασματάκια τότε θέλουμε να πλησιάσει ο πακμαν για να τα φάει, το ίδιο και για την τροφή, επίσης όσο λιγότερη τροφή, κάψουλες και φοβισμένα φαντασματάκια υπάρχουν τόσο το καλύτερο για να παροτρίνουμε τον πακμαν να τα φάει. Οι συντελεστές που έχω πολλαπλασιάσει τις παραμέτρους που προσθέτω στο σκορ έχουν βγει μετά απο δοκιμές που έκανα να τρέξω το παιχνίδι πολλαπλές φορές και να καταλήξω σε αριθμούς που αποδίδουν μια καλή συμπεριφορά από τον πακμαν.