Student Name: Κωνσταντίνος Φράγκος
Student ID: sdi2000207

Τεχνητή Νοημοσύνη
Εργασία 2

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ
ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
Εθνικόν και Καποδιστριακόν Πανεπιστήμιον Αθηνών
ΙΔΡΥΘΕΝ ΤΟ 1837

Πρόβλημα 1:

Κατασκευάζουμε μία evaluate function με την λογική να βαθμολογεί με τεράστιο αρνητικό αριθμό καταστάσεις που βρίσκοντε κοντά σε φαντάσματα. Για τις υπόλοιπες, βρίσκει την απόσταση με την κοντινότερη κουκίδα, και επιστρέφει το αντίστροφο αυτής της απόστασης (Καθώς θέλουμε μία μεγάλη τιμή όταν είμαστε κοντά στην κουκίδα), σιν το σκορ της κατάστασης που όσο πιο λίγες κουκίδες υπάρχουν στο game , όσο ο pacman δεν μένει στάσιμος, όσο ο pac man τρώει capsules αυτό είναι και μεγαλύτερο. Συνεπώς το σκορ είναι ένας πολύ καλός προσδιοριστής της evaluation.

Πρόβλημα 2:

Φτιάχνουμε 2 συναρτήσεις, μία Min και μία Max. Σαν ορίσματα περνάμε το depth και μία κατάσταση. Αρχικά καλούμε την Max η οποία επιστρέφει το βέλτιστο action και μία evaluate value η οποία υπολογίζεται αναδρομικά και δεν χρησιμοποιείται στην συνάρτηση get Action. Σαν depth περνάμε το self.depth*numofagents. Αυτό διότι κάθε φορά που θα καλούμε αναδρομικά μία επόμενη συνάρτηση θα το κατεβάζουμε κατά 1 ώστε η αναδρομή μας να σταματάει στο base case depth = 0. Δεδομένου ενός depth αν θέλουμε να πάρουμε τον agend του εκάστοτε depth υπολογίζουμε το $|depthmodnumofagends - numofagends|$, ενώ άν η διαίρεση χωρίς την αφαίρεση αυτή είναι ίση με 0 τότε μιλάμε για τον agend 0, δηλαδή τον pac man.
Όσον αφορά τη λειτουργία των συναρτήσεων, η max καλεί την min για όλα τα len(actions), και επιστρέφει αυτό με την μεγαλύτερη evaluate value και την τιμή αυτή. Η min πάλι καλεί len(actions) φορές την min αν ο επόμενως agend είναι min agend αλλιώς καλεί len(actions) φορές την max και επιστρέφει την τιμή και την action με το μικρότερο evaluate value.

Πρόβλημα 3:

Χρησιμοποιόντας την υλοποίηση του προηγούμενου ερωτήματος, περνάμε σαν ορίσματα στην min και τη max ένα a και ένα b αρχικιποιημένα με -άπειρο και +άπειρο αντίστοιχα, και απλά στην max κάνουμε έναν έλεγχο όταν βρίσκουμε ένα value αν αυτό είναι μεγαλύτερο από το εκάστοτε β μας. Αν είναι δε συνεχίζουμε την αναζήτηση για τα υπόλοιπα actions και επιστρέφουμε αυτό. Αντίστοιχα στην min κάνουμε το ίδιο αλλά με το α και η συνθήκη που ελέγχεται είναι το value να είναι μικρότερο του α.

Πρόβλημα 4:

Ακολουθούμε την ίδια λογική με τα προηγούμε ερωτήματα, Χρησιμοποιούμε την συνάρτηση max των προηγούμενων ερωτημάτων και αντί για την συνάρτηση min κατασκευάζουμε μία evaluate συνάρτηση όπου υπολογίζει το άθροισμα των evaluate τιμών των states των επόμενων actions πολλαπλασιασμένο με την πιθανότητα ο agend να πάρει αυτή την απόφαση για αυτό το action, όπου στην

περίπτωσή μας αυτή η πιθανότητα είναι $\frac{1}{actions}$, καθώς είναι τυχαία και ομοιόμορφα κατανεμιμένη η πιθανότητα ο ghost agend να επιλέξει μία κίνηση

Πρόβλημα 5:

Η λογική αυτής της καλύτερης evaluate funciton είναι για την κατασάστης νίκης και ήτας να επιστρέφει μία τεράστια και μία πολύ μικρή ποσότητα αντίστοιχα. Σε κάθε άλλη περίπτωση επιστρέφει έναν γραμμικό συνδιασμό του πόσες τελείες υπάρχουν στην πίστα, το αντίστροφο της απόσταστης από την κοντινότερη κουκίδα μέ έναν συντελεστή 10(τυχαία δοκιμή) και το σκορ που υπάρχει στην πίστα την εκάστοτε στιγμή.(Εξηγείτε στο 1ο ερώτημα τι περιέχει το σκορ)