Τεχνητή Νοημοσύνη 1 - Χειμερινό 2020-2021
Εργασια Δεύτερη

Ροβιθάκης Ιωάννης | sdi1800164

Σχολιασμός Project 2 Pacman:

Τα σχόλια σε αυτό το αρχείο αποτελούν μια γενική περίληψη τεχνικών λεπτομερειών, καθώς και του τρόπου εργασίας μου. Για αναλυτικές πληροφορίες και επεξηγήσεις πάνω στα παραπάνω μπορείτε να συμβουλευτείτε τα σχόλια που θα βρείτε μαζί με τον κώδικα ή να έρθετε σε επικοινωνία μαζί μου. Γράφω τα σχόλια ταυτόχρονα με τον κώδικα, οπότε το θεώρησα άσκοπο να τα αντιγράψω απλά εδώ. Τα ερωτήματα 2, 3 και 4 είναι απλή υλοποίηση ψευδοκώδικα και συνεπώς δεν έχω γράψει εκτεταμένα σχόλια αφού η λογική τους είναι απλή.
(Δεν έκανα καμία αλλαγή στο pacman.py οπότε παραδίδω μόνο το multiAgents.py)

1) Question 1 : Reflex Agent

Μια πολύ απλή συνάρτηση αξιολόγησης για μια δεδομένη κίνηση. Με βάση τις υποδείξεις της εκφώνησης καθώς και των σχολίων που υπήρχαν στον κώδικα, χρησιμοποίησα ως κριτήρια αξιολόγησης μιας κίνησης την απόσταση του πακμαν από το κοντινότερό του φαγητό, καθώς και το επιπλέον σκορ που του αποδίδει τελικά η κίνηση αυτή. Έδωσα μεγάλη έμφαση στο φαγητό και στο φάγωμα καψουλων και φαντασμάτων (έμμεσα, συμπεριλαμβάνοντας το σκορ στο οποίο μετρούνται τα παραπάνω) καθώς ήθελα να επιτυγχάνω κάθε φορά ένα αρκετά καλό σκορ για τον autograder. Όσον αφορά τα φαντάσματα, απλά ελέγχω αν κάποιο αρκετά κοντινό φάντασμα είναι ή όχι, τρομαγμένο και ανάλογα το αποφεύγω με κάθε κόστος, ή το κυνηγάω. Τέλος, πάλι μετά από υπόδειξη της εκφώνησης, χρησιμοποίησα αντίστροφους για την τιμή του φαγητού, καθώς τα κλάσματα μεγαλώνουν όσο μικραίνει ο παρανομαστής τους, ιδιότητα που μας ταιριάζει ακριβώς στις απαιτήσεις της άσκησης αυτής.

2) Question 2 : Minimax

Υλοποίησα τον minimax αναδρομικά, αποτελούμενο από μία συνάρτηση maxValue(), μία minValue() καθώς και ένα κομμάτι κώδικα το οποίο είναι στην πράξη μια παραλλαγή της maxValue() και χρησιμοποιείται για να αρχίσει η αναδρομή, καθώς και για επιστροφή αποτελεσμάτων. Η λογική που ακολουθεί ο minimax μου σαν αλγόριθμος είναι αυτή που διδαχτήκαμε στη θεωρία, με μόνη διαφορά ότι στην περίπτωσή μας έχουμε "πολλούς MIN" = φαντάσματα. Λόγω της ιδιαιτερότητας αυτής, στον κώδικα μου, για κάθε κίνηση-επίπεδο του MAX-pacman, ακολουθεί ένα σύνολο από κινήσεις MIN-φαντασμάτων, και το "βάθος" του minimax σε λογικό επίπεδο, αυξάνεται πάντα από τον MAX μόνο. (1 κίνηση MAX + x κινήσεις φαντασμάτων = 1 επίπεδο). Τέλος, ο έλεγχος τέλους αναδρομής λόγω βάθους αναδρομής, γίνεται μόνο στον MAX, καθώς μόνο σε αυτόν μεγαλώνει το βαθος όπως είδαμε.
Κατά τα άλλα, η υλοποίηση του minimax είναι μία μικρή παραλλαγή του ψευδοκώδικα στις διαφάνειες 19-20 του anazitisi_me_antipalotita.pdf του μαθήματος μας.

3) Question 3 : Alpha-Beta Pruning

Για τον α-β, ισχύουν τα ίδια σχόλια που ανέφερα παραπάνω για τον minimax όσον αφορά την δομή του, καθώς και τα πολλαπλά επίπεδα MIN. Στην εικόνα μπάινουν επιπλέον οι παράμετροι α και β, με βάση τις οποίες πραγματοποιείται "κλάδεμα" υποδέντρων (υπό τις κατάλληλες συνθήκες), όπως έχουμε δεί και στο μάθημα, επιτυγχάνοντας διόλου ασήμαντη βελτίωση στην απόδοση, γεγονός που σε κανονικές συνθήκες μας επιτρέπει να αναζητούμε σε μεγαλύτερο βάθος. Η υλοποίηση μου βασίζεται στον κώδικα που είχα γράψει για το προηγούμενο ερώτημα, με μικρές αλλαγές για να λειτουργήσει σαν τον ψευδοκώδικα που δίνεται στην ιστοσελίδα-εκφώνηση της εργασίας Project 2.

4) Question 4 : Expectimax
Για τον expectimax, επίσης ισχύουν τα σχόλια που ισχύουν για τον minimax. Μοναδική διαφορά είναι ότι πλέον ο MIN δεν είναι βέλτιστος, αλλά επιλέγει τυχαίες κινήσεις, ομοιόμορφα κατανεμημένες, άρα με ίσες πιθανότητες επιλογής. Συμφωνα με το παραπάνω, αλλάζει ο τρόπος που λειτουργεί μόνο ο MIN, και πλέον αντί για την ελάχιστη δυνατή αξία του υποδέντρου του, επιστρέφει τον μέσο όρο (αφού έχουμε ισοπίθανες κινήσεις) των πιθανών αξιων των διαφόρων κινήσεων του. Δηλαδή το άθροισμα όλων των αξιών των διαθέσιμων κινήσεων δια το πλήθος των κινήσεων αυτών. Για την υλοποίηση μου αυτή, βασίστηκα στον ψευδοκώδικα που βρήκα στο αντίστοιχο pdf του μαθήματος του Berkeley. (Διαφάνεια 2, https://inst.eecs.berkeley.edu/~cs188/sp20/assets/lecture/lec-9-6up.pdf)

5) Question 5 : Evaluation Function
Το ερώτημα αυτό χρειάστηκε αρκετό χρόνο και πειραματισμό για να επιτύχω ικανοποιητικό σκορ. Σημαντική παρατήρηση είναι ότι στο ερώτημα αυτό, αντίθετα με το ερώτημα 1, καλούμαστε να αξιολογήσουμε ένα state ως σύνολο, και όχι μια κίνηση.

Συνοπτικά, για τον υπολογισμό μιας τιμής αξιολόγησης έλαβα υπ όψην μου τα εξής:
i) Το **score** του παιχνιδιού στην δεδομένη κατάσταση (Συμπεριλαμβάνει το score που κερδίζεται από το φάγωμα φαγητού, κάψουλων και τρομαγμένων φαντασμάτων σημαντικές πληροφορίες για την αξιολόγηση κάθε κατάστασης)
ii) Το **πλήθος φαγητού** που υπάρχει διαθέσιμο (Όσο λιγότερο, τόσο καλύτερα, συνεπώς το αφαιρώ από το σύνολικο σκορ)
iii) Την **απόσταση του pacman από το κοντινότερο του φαγητό** στην κατάσταση αυτή. (Όσο πιο κοντά τόσο καλύτερο, αλλά δέν δίνω τόσο μεγάλο βάρος στο κοντινότερο, καθώς δεν είναι πάντα το καλύτερο)
iv) Την **μέση απόσταση του pacman από κάθε φαγητό** (Όσο πιο κόντά βρίσκεται σε όλα τα φαγητά, τόσο καλύτερο)
v) Την **απόσταση του pacman από την κοντινότερη του κάψουλα** (Όσο πιο κοντά, τόσο καλύτερα => ασφάλεια από φαντάσματα)
vi) Την **απόσταση του pacman από το κοντινότερο του φάντασμα** (Όσο πιο μακριά είναι τα φαντάσματα τόσο καλύτερο, εκτός εαν είναι τρομαγμένα, οπότε, οπότε τους δινουμε μεγάλο βάρος και τα κυνηγάμε για καλύτερο σκορ.)

Επιπλέον σχόλια για το ερώτημα 5:
Στα παραπάνω, έχω βρεί κατόπιν δοκιμών κάποια βάρη, τα οποία οδηγούν σε καλύτερη ισορροπία, και έμφαση στα πράγματα που έχουν μεγαλύτερη σημασία για επίτευξη όσο δυνατόν καλύτερου σκορ.

Τις περισσότερες από τις παραπάνω τιμες, τις έχω χρησιμοποιήσει στην "αντίστροφη" μορφή τους, βασιζόμενος στην λογική του πρώτου ερωτήματος και τις ιδιότητες των κλασμάτων, ώστε να επιτύχω καλύτερα τελικά αποτελέσματα.

Ιδανικά, θα είχα προυπολογίσει μία φορά όλες τις αποστάσεις για κάθε κατάσταση όπως έκανα στο food heuristic του Project 1 για καλύτερη απόδοση, αλλά δεν χρειάστηκε τελικά. (Με την τωρινή υλοποίηση υπολογίζουμε πολλές φορές τις ίδιες manhattan, για μεγάλα game-maps θα υπήρχε πρόβλημα)

Η mazeDistance() που είχαμε δει στο Project 1 θα μπορούσε να χρησιμοποιηθεί για μεγαλύτερη ακρίβεια επιλογών, αλλά και η manhattanDistance() λειτουργεί ικανοποιητικά και είναι πιο "οικονομική"

Για περισσότερες και αναλυτικές λεπτομέρειες διαβάστε τα σχόλια στον κώδικα.

Για οτιδήποτε προκύψει κατά την διόρθωση, επιπλέον διευκρινήσεις και επεξηγήσεις καθώς και προτάσεις για βελτίωση των υλοποιήσεων μου, μπορείτε να επικοινωνήσετε μαζί μου στο sdi1800164@di.uoa.gr