Pacman: Project 2

Multi-Agent Search

Κωνσταντίνος Χούσος

Περίληψη

Το συγκεκριμένο report αναλύει τις λύσεις μου στο project 2 των Pac-Man Projects του Berkeley [1], στα πλαίσια του προπτυχιακού μαθήματος "Τεχνητή Νοημοσύνη". Στο πρώτο ερώτημα προτιμούμε τις καταστάσεις που μας φέρνουν πιο κοντά σε τελείες και πιο μακριά από φαντάσματα, αποφεύγοντας πάντα την ήττα. Τα ερωτήματα 2, 3 και 4 υλοποιούν τους αντίστοιχους αλγορίθμους με βάση την θεωρία, ελέγχοντας κάθε φορά ποιον agent αφορά. Το ερώτημα 5 αποτελεί μια βελτίωση του κώδικα του ερωτήματος 1, με κύρια διαφορά τον καινούργιο έλεγχο για capsules.

1: REFLEX AGENT

Το συγκεκριμένο evaluation function ακολουθεί την εξής λογική:

1. Πρώτον, αν το successorGameState επιφέρει ήττα, επιστρέφουμε μια minimum τιμή (-99) κάνοντάς την χείριστη. Αλλιώς, αν επιφέρει νίκη, επιστρέφουμε μια maximum τιμή (99) κάνοντάς την βέλτιστη.

2. Αν τίποτα από τα παραπάνω δεν ισχύει, πρώτα ασχολούμαστε με τα φαντάσματα. Βρίσκουμε τις αποστάσεις για κάθε φάντασμα που δεν είναι σε κατάσταση scared, λαμβάνουμε υπόψιν το κοντινότερο κι άρα πιο απειλητικό και αφαιρούμε από το score το "βάρος" του—όπου το "βάρος" υπολογίζεται ως ο αντίστροφος της απόστασής του. Το συγκεκριμένο βήμα δεν είναι απαραίτητο, αλλά με αυτό η υλοποίηση είναι ικανή να νικάει και directional ghosts.

3. Ακολουθείται μια αντίστοιχη διαδικασία με αυτήν του βήματος 2, αλλά αυτήν τη φορά για τις τελείες. Η μόνη διαφορά είναι ότι σε αυτήν την περίπτωση το βάρος προστίθεται στο score, καθώς θέλουμε να έρθουμε κοντά σε όλες τις τελείες.

2: MINIMAX

```
function MINIMAX-SEARCH(game, state) returns an action
  player ← game.TO-MOVE(state)
  value, move ← MAX-VALUE(game, state)
  return move

function MAX-VALUE(game, state) returns a (utility, move) pair
  if game.IS-TERMINAL(state) then return game.UTILITY(state, player), null
  v, move ← −∞
  for each a in game.ACTIONS(state) do
    v2, a2 ← MIN-VALUE(game, game.RESULT(state, a))
    if v2 > v then
      v, move ← v2, a
  return v, move

function MIN-VALUE(game, state) returns a (utility, move) pair
  if game.IS-TERMINAL(state) then return game.UTILITY(state, player), null
  v, move ← +∞
  for each a in game.ACTIONS(state) do
    v2, a2 ← MAX-VALUE(game, game.RESULT(state, a))
    if v2 < v then
      v, move ← v2, a
  return v, move
```

Σχήμα 1: Ψευδοκώδικας **minimax**

Η υλοποίηση στο συγκεκριμένο ερώτημα ακολουθεί την λογική του ψευδοκώδικα του βιβλίου [2] που υπάρχει στο σχήμα 1.

Πρώτον, υπάρχει μια βοηθητική συνάρτηση **terminal**, η οποία επιστρέφει **True** αν έχουμε φτάσει σε κατάσταση νίκης, ήττας ή είμαστε στο μέγιστο βάθος, αλλιώς επιστρέφει **False**.

Η **maxValue**—η οποία αφορά τον pacman—ελέγχει πρώτα αν είμαστε σε terminal state, δηλαδή αν είμαστε σε φύλλο του δένδρου

παιχνιδιού, όπου τότε επιστρέφει το utility του κόμβου. Αν δεν είμαστε σε φύλλο, επιστρέφει την μέγιστη τιμή από τις πιθανές τιμές της **minValue** για κάθε δυνατή κίνηση.

Η **minValue**—η οποία αφορά όλα τα φαντάσματα—ακολουθεί μια παρόμοια σκέψη. Η διαφορά βρίσκεται στο ότι κάθε φορά ελέγχεται αν ο επόμενος agent είναι άλλο φάντασμα ή ο pacman. Στην πρώτη περίπτωση, επιστρέφει την ελάχιστη τιμή από τις πιθανές τιμές της **minValue** για κάθε δυνατή κίνηση του επόμενου φαντάσματος. Στην δεύτερη, επιστρέφει την ελάχιστη τιμή από τις πιθανές τιμές της **maxValue** για κάθε δυνατή κίνηση του pacman, αυξάνοντας παράλληλα το βάθος κατά 1.

Η συνάρτηση **getAction** επιστρέφει την κίνηση με τη μέγιστη minimax τιμή.

3: ALPHA-BETA PRUNING

Ο κώδικας του συγκεκριμένου ερωτήματος είναι σχεδόν ίδιος με εκείνον του ερωτήματος 2, προσαρμοσμένος πλέον να λαμβάνει υπόψιν τις α, β τιμές. Η μόνη διαφορά με τον ψευδοκώδικα του βιβλίου [2] βρίσκεται στα σημεία σύγκρισης του v με τα α, β. Σε αντίθεση με αυτόν, ο ψευδοκώδικας που δίνεται ως παράδειγμα από την εκφώνηση του project [1] ορίζει σε εκείνα τα σημεία **αυστηρή** ανισότητα. Λόγω αυτού και καθώς η υλοποίηση με μη αυστηρή ανισότητα δεν περνάει τα tests, χρησιμοποιήθηκε αυστηρή ανισότητα.

4: EXPECTIMAX

Ομοίως με το ερώτημα 3, ο κώδικας αποτελεί μια παραλλαγή του ψευδοκώδικα του **minimax** (σχήμα 1). Η διαφορά έγκειται στο ότι αντί για την **minValue** έχουμε πλέον μια παραλλαγή της: την **expValue**. Στην δεύτερη, αντί να βρίσκουμε κάποια ελάχιστη τιμή βρίσκουμε το άθροισμα των πιθανοτήτων κάθε δυνατής κίνησης των φαντασμάτων και του pacman.

5: EVALUATION FUNCTION

Ο κώδικας είναι εμπνευσμένος από αυτόν του ερωτήματος 1. Εν συντομία, αποφεύγουμε τα φαντάσματα που δεν είναι φοβισμένα και πηγαίνουμε προς το κοντινότερο φαγητό. Το καινούργιο στοιχείο της συγκεκριμένης συνάρτησης είναι ότι πέρα από τα προηγούμενα ελέγχουμε και κατά πόσο υπάρχει κοντά κάποια *κάψουλα*. Με τον ίδιο τρόπο που ελέγχουμε το φαγητό ελέγχουμε κι αυτό, δηλαδή προσθέτοντας τον αντίστροφο της απόστασης της κοντινότερης κάψουλας.

Αναφορές

[1] John DeNero, Dan Klein και Pieter Abbeel. *Projects - CS 188: Introduction to Artificial Intelligence, Spring 2022*. 2022. URL: https://inst.eecs.berkeley.edu/~cs188/sp22/projects/.

[2] Stuart J. Russell και Peter Norvig. *Artificial Intelligence: A Modern Approach*. Συνεργασία από Ming-wei Chang κ.ά. Fourth edition, global edition. Pearson Series in Artificial Intelligence. Harlow: Pearson, 2022. ISBN: 978-1-292-40113-3.