Τεχνητή Νοημοσύνη
Εργασία 1 - Berkeley Pacman Project 1: Search
Χειμερινό εξάμηνο 2021-2022

Ευάγγελος Δημητριάδης 1115201700287

1 Depth First Search

Έγινε μια generic υλοποίηση του αλγόριθμου iterative DFS για να βρεθεί διαδρομή του Pacman σε ένα maze από την αρχική του θέση στην θέση που βρίσκεται ένα pellet. Το maze θεωρείται ως γράφος με κάθε τετράγωνο(που δεν είναι wall) να είναι ένας κόμβος με γειτονικούς κόμβους να είναι τα τετραγωνάκια στα οποία μπορεί να πάει ο Pacman με μία κίνηση.
Η συνάρτηση χρησιμοποιεί ένα stack για το fringe και ένα set για να κρατιούνται τα τετράγωνα που έχουν ήδη επισκεφθεί.
Για να βρεθούν οι γείτονες ενός τετραγώνου χρησιμοποιείται η συνάρτηση expand.
Στο stack μαζί με το state αποθηκεύεται και η διαδρομή για να φτάσουμε σε αυτό από το αρχικό state.Κάθε φορά που βρίσκουμε ένα γείτονα με το expand παίρνουμε και την κατεύθυνση προς αυτό την οποία προσθέτουμε στην μέχρι τώρα διαδρομή.Όταν βρεθεί με pop ο κόμβος στόχου τότε επιστρέφεται η διαδρομή από την αρχή προς αυτόν.

Η λύση που βρίσκει το DFS για ένα maze δεν είναι πάντα η βέλτιστη καθώς δεν επιλέγεται η καλύτερη διαδρομή αλλά η πρώτη που θα βρεθεί αναζητώντας με την σειρά σε βάθος.

2 Breadth First Search

Έγινε μια generic υλοποίηση του αλγόριθμου iterative BFS για να βρεθεί διαδρομή του Pacman σε ένα maze από την αρχική του θέση στην θέση που βρίσκεται ένα pellet. Η φιλοσοφία της υλοποίησης είναι ίδια με αυτή του DFS απλά με τον αλγόριθμο να αλλάζει. Χρησιμοποιείται Queue αντί για stack για το fringe αφού αλλάζει η σειρά επίσκεψης των κόμβων και τα τετράγωνα μαρκάρονται ως visited πριν γίνουν visited επειδή η αναζήτηση γίνεται κατά πλάτος και δεν μπορούμε από ένα κόμβο να επισκεφθούμε άλλον που βρίσκεται στο ίδιο πλάτος.

Η DFS βρίσκει το συντομότερο μονοπάτι αφού ο γράφος είναι undirected με ίδια weights σε κάθε γράφο όμως παίρνει περισσότερο χρόνο από την BFS καθώς εξετάζει περισσότερα πιθανά μονοπάτια.

3 A* search

Η A* έχει υλοποιηθεί με την ίδια φιλοσοφία με τους παραπάνω αλγορίθμους. Χρησιμοποιείται ένα PriorityQueue για το fringe και ένα dictonary για να κρατίεται το μικρότερο cost για κάθε state από το αρχικό state.
Τα states γίνονται push στο PQ με το priority να είναι η τιμή του heuristic τους.
Στο costs ως index χρησιμοποιούνται οι συντεταγμένες του state.Ένα state προσθέται στο fringe μόνο αν το τωρινό cost του είναι μεγαλύτερο η αν δεν έχει γίνει visited προηγουμένως.

Η A* βρίσκει το συντομότερο μονοπάτι κάντωντας expand λιγότερα nodes από την BFS.
Στο openMaze η bfs και η A* βρίσκουν και οι δύο το συντομότερο μονοπάτι ,με την A* να

το βρίσκει με λιγότερα expands. Η dfs κάνει expand τα περρισότερα nodes και βρίσκει το χειρότερο μονοπάτι ανάμεσα στους τρεις αλγορίθμους.

4 Corner Search

Για το συγκεκριμένο πρόβλημα ένα state θεωρείται ως goal όταν όλα τα pellets στις γωνίες έχουν φαγωθεί. Γι' αυτό το state αποθηκεύεται ως tuple από δύο tuples όπου το πρώτο tuple είναι οι συντεταγμένες του pacman για το τωρινό state και το δεύτερο μια λίστα με τις συντεταγμένες των γωνιών που δεν έχουν γίνει visited ακόμα στο τωρινό state.
Όταν καλείται η συνάρτηση getNextState γίνεται έλεγχος για το αν το επόμενο state είναι γωνία και αν ναι τότε αφαιρείται από την λίστα unvisited του state.
Το isGoalState επιστρέφει true αν το μέγεθος του unvisited list είναι ίσο με 0.

5 Corners Heuristic

Για το συγκεκριμένο πρόβλημα η καλύτερη διαδρομή είναι ο pacman να πάει πρώτα στην κοντινότερη σε αυτόν ο γωνία,μετά στην επόμενη κοντινότερη από την γωνία που έχει φτάσει κ.ο.κ.
Έτσι το heuristic υπολογίζει για ένα state το manhattan distance από τον pacman στην κοντινότερη γωνία,από την κοντινότερη γωνία στην κοντινότερη σε αυτήν γωνία κ.ο.κ.
Δηλαδή υπολογίζει την βέλτιστη διαδρομή αν ο pacman βρίσκεται σε maze χωρίς εσωτερικούς τοίχους.
Το heuristic είναι admissible αφού είναι η απόσταση στην καλύτερη περίπτωση (το manhattan distance δεν λαμβάνει υπόψιν τους τοίχους).
Είναι επίσης και consistent αφού υπολογίζεται η συνολική απόσταση που πρέπει να διανύσει ο pacman οπότε όσο προχωράει στην βέλτιστη διαδρομή το heuristic μειώνεται.

6 Food Heuristic

Στο συγκεκριμένο πρόβλημα η βέλτιστη διαδρομή είναι ο pacman να κατευθυνθεί προς την μία άκρη του maze και να προχωρήσει προς την άλλη άκρη του maze τρώγοντας τα pellets που βρίσκει στον δρόμο.
Το heuristic υπολογίζει τον μέσο όρο όλων των αποστάσεων των pellets από την τωρινή θέση του pacman.
Το heuristic είναι admissible αφού ο μέσος όρος όλων των αποστάσεων είναι πάντα μικρότερος από το μήκος της καλύτερης διαδρομής.
Το heuristic είναι επίσης consistent επειδή μειώνεται όταν ο pacman ακολουθεί την βέλτιστη διαδρομή.
Το heuristic μειώνεται όταν ακολουθείται η καλύτερη διαδρομή γιατί αν ο pacman ξεκινήσει από την μια πλευρά και κατευθυνθεί προς την άλλη ,δηλαδή ελαχιστοποιήσει τα τετράγωνα που πατάει πολλές φορές, ο μέσος όρος της απόστασης θα είναι μικρότερος από τις άλλες περιπτώσεις γιατί δεν μένουν ποτέ pellets στην μια πλευρά ενώ ο pacman βρίσκεται στην αντίθετη.

7 Closest Dot Heuristic

Για να βρεθεί η απόσταση προς το κοντινότερο pellet απλά καλείται η BFS για το gameState ως AnyFoodSearchProblem. Η AnyFoodSearchProblem ορίζει ένα state ώς goal αν έχει φαγητό και με την χρήση της BFS το πρώτο state που θα βρεθεί με φαγητό είναι και το πιο κοντινό.

Ένα παράδειγμα στο οποίο το heuristic δεν βρίσκει την βέλτιστη διαδρομή είναι το tricky-mazeSearch.Στο trickymazeSearch τα πιο κοντινά pellets βρίσκονται δεξιά του pacman,αφού τα φάει τα πιο κοντινά pellets βρίσκονται στην αριστερή άκρη του maze και μετά στο τέλος του maze.Ακολουθώντας αυτή την διαδρομή ο pacman καταλήγει να περάσει από κάποια τετραγωνάκια (τα κόκκινα στην παρακάτω εικόνα) του maze τρεις φορές επειδή πρέπει να γυρίσει πίσω για να φάει τα pellets που του ξέφυγαν. Αυτό μπορεί να αποφευχθεί με την χρήση ενός καλύτερου heuristic.