Ms Pac-Man vs. Ghosts

Jiacheng Geng, jg4754

October 25, 2016

1 Depth-first search, breadth-first search and iterative deepening

I just followed the example given on the note. I limited a certain depth, 12,then used depth-first search to get all the possible game state after 11 movements, discarding those game state where Pac-man was eaten by ghosts. Then I compared all the path, chose the path that leading to the highest score. In the end, the first movement of that path would be my choice for this step.

This strategy usually have more than 1000 score and near 1500. It will go away from a ghost when the ghost is approaching it in order to survive. Meanwhile it may jam in some jeopardy if there are two paths leading to a same score(usually in no-pill-around case).

Breadth-first search is just like depth-first search, I just replaced the stack with queue with the same depth limitation, 12.

This strategy also performed almost the same as depth-first search.

Iterative deepening is just a growing-limit depth-first search. I applied depth from 1 to 10 and still chose the highest-score path and used its first movement. A slight technical difference is that Pac-man may get eaten for a higher score if he has no choice but to die in the limited depth. This strategy also performed almost the same as depth-first search.

2 A*

In my opinion, this is actually a bad algorithm if it is used in its pure form. The A* we talked about on the class is aiming at finding a shortest path to a certain goal. Meanwhile, in Pac-Man, the goal may be the state where all pills are eaten, so the pills you can eat is always a constant, and there is no "shortest path" or any "least cost"in any simple definition I can figure out. So, I can only choose my heuristic function by pills left to eat multiplied by 1 and cost function as if Pac-Man is eaten 10000, otherwise 0.Under this, I can choose the smallest node to expand, and the depth limit is 10.

The result is a little better than the three algorithms above, around 2000 maybe because it really wants to eat a ghost when it can. But it jammed in some jeopardy happened for more times.

3 Hill-climber and anealing

For hill-climber, we are actually evaluate the movement sequence. So I generate a random movement sequence with 10 step movement as the initial movement. Then I defined the movement sequence with only one step movement different from the original movement sequence is called the neighbor of the original sequence. So all the neighbor of my movement sequence is all those movement sequence have exactly one different movement at exactly one step(could be 1 to 10) from my original sequence. I measure the score after applying the sequence to the game state and use it to judge the goodness of the sequence. If no neighbor is better, I choose the first movement of current sequence as my choice. Otherwise, I choose the best neighbor to "climb" until the round limitation is reached.

The result is very bad. The pac-man was always going forth and back while moving slowly on some certain path full of pills and it never avoided ghosts. The score it got is usually only near 800.

All other elements are the same as hill-climber. But I only choose one neighbor to measure this time. I set T as 1000, which is a constant. And repeat until the round limitation is reached.

The result is extremely bad. Instead of slowly moving on some certain path full of pills, which is done in hill-climber, it just moved forth and back at the start point and waited for ghosts to eat it. The score was only below 100.

4 Mutation and crossover

All other elements are the same as hill-climber. I generate the initial movement sequence population randomly with a size of 10 and each has a depth of 10. The fitness function is just the score after the sequence. The variation is just becoming one of the neighbor defined in hill-climber. I leave the best half of population as the survivors to mutate until round limitation is reached. Then I choose the first movement of the best sequence.

The result is just like hill-climber.

All other elements are the same as mutation. I just randomly choose two sequence A and B from the survivors to produce a child by the first 5 movements from A and the last 5 movements from B and keep the population size constant until round limitation is reached.

The result is bad. I don't even know what's the point of this method, so how could it possibly make sense? The score is always under 300. And the Pac-man sometimes just moved forth and back near some certain point even there were some pills near it to eat.