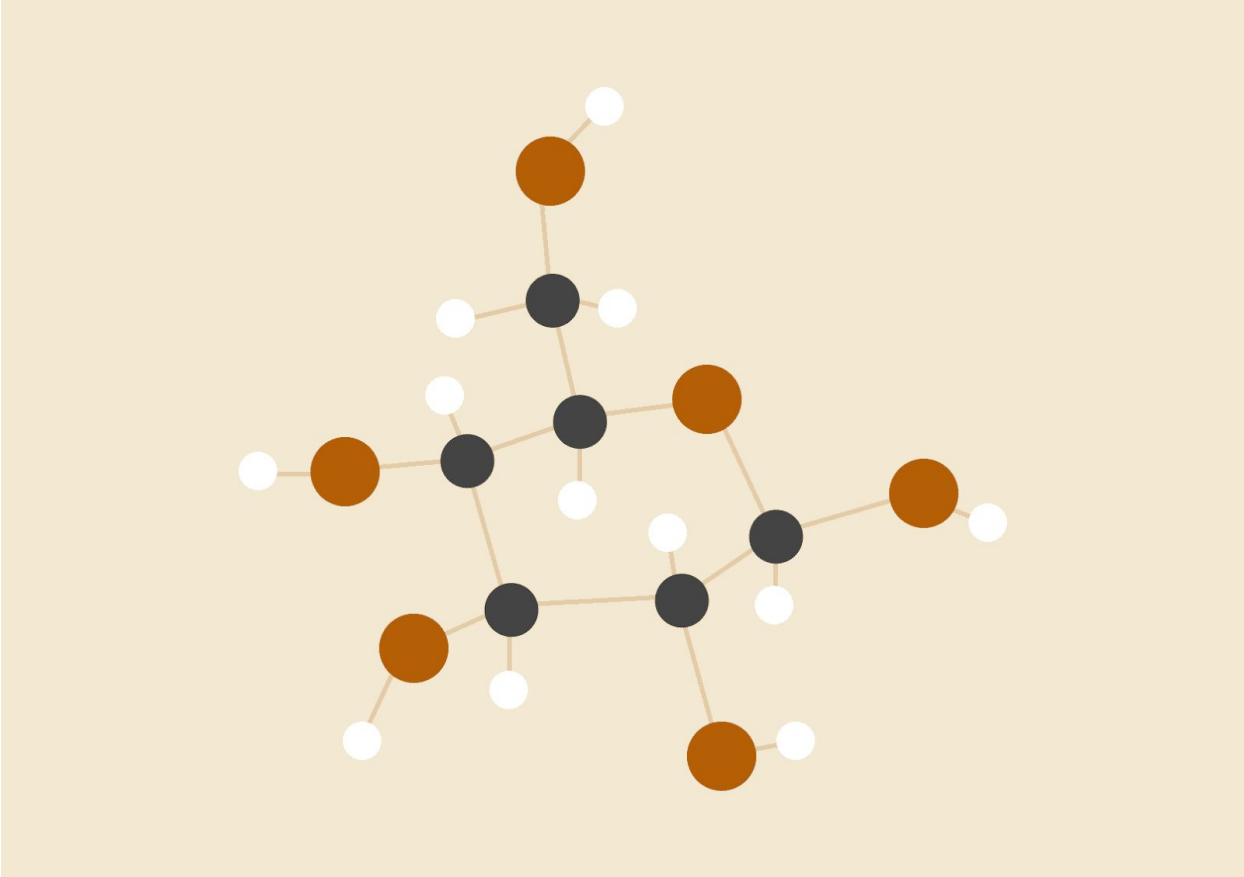
PacPack Diary

CS 188 - Phase I & II



Wayne Li

3032103452
lishixuan001@berkeley.edu

July 19, 2018

Start Time: 10:00 PM

End Time: +1 5:00 AM

Time Spent: 7 Hours

Attempts:

a) First tried to play with features and weights. The initial purpose was just to let the pacman get out from the start corner. But I adjusted the features and weights for a while using food distance (I also created an algorithm using BFS distance instead of the provided algorithm API). But things didn't work. Thus I stopped and turned to pure search algorithm
b) I first attempted BFS algorithm, greedily searching for the closest food and go for it. Then I found that things did not go well. I may need to add some rewards and punishments while searching (the cost). Thus I turned into AStar.
c) Firstly I used UCS, setting every step the pacman goes to cost = 1. Then start searching for the closest food. This is not same comparing with BFS since it does not allow attempts like going back home to happen, while those can happen in BFS since there's not cost. (in BFS, when I print out the path BFS turned out for searching for closest food, it asks the pacman to go back [the pacman start at (1, 2), the first step is always (1, 1) which goes right opposite to the closest path])
d) The UCS runs well (usually pass 6/10). But I think I should consider my teammate's position and design my better route as well. Thus, the new algorithm split the screen into Left and Right. When pacman is on the left side and the left side still has food (amount > 2), it should focus on eating all the food on the left side before going to the right. Then I tried to consider the location of my teammate, but considering his location did not help since that guy is so stupid and even if he is around a food, he sometimes spend too much time eating it. So the final algorithm forgets that.
e) The result turned out that I usually get 8/10

July 20, 2018

Start Time: 4:00 PM

End Time: 7:00 PM

Time Spent: 3 Hours

Attempts:

a) I moved to Phase II.
b) I first tried to move my algorithm straightly to Phase II. I founded it worked well actually. (usually 6/10)
c) Then I start to consider how to implement the location my teammate. I realized that my teammate gives me all the positions he will be at for the first 400 moves. So every time I extract his first 120 steps and deleted all the food he gonna eat from the foodGrid. Then do my algorithm again. It now reaches 7/10 or 8/10.
d) I tried to formalize my algorithm by creating a Food Map using a Food class which takes in “position, distance (to pacman), and path (from pacman)”. Then, I still do the left-right split thing. Almost the same as before. But creating a food map takes too long. Every step is close to 1 sec. Thus I give up. Then I added one more constraint that when left side has only 2 food left, if any of them has distanceToPacman <= 5, then it judges that pacman should go for it before going to the right.

July 20, 2018

Start Time: 11:00 PM

End Time: +1 3:00 AM

Time Spent: 4 Hours

Attempts:

a) I just realized that I have a good idea that can help me improve Phase I -- to deal with the situation when there are only few food left. I hope my pacman can find a way to maximize the food value (using lowest cost to finish the game). When I do

the left/right screen thing, I set that pacman need not to finish eating all the dots before going from left side to the right side as long as there are at most two food left on the left and they are all far from the pacman (when he is still at the left side). For each food we consider the number of food around it (-4, +4). If it's zero, we delete that from foodGrid until we delete 2 of them, then do the algorithm. Now it is steady at 8/10. I also added this to Phase I, but there's minor change.

b) I also tried to create a food density map. Each food has a high value and radically decrease its influence (values) as further from each food. But the overlapping areas are too complicated to do and the decrease portion is too hard to design. So after some attempts, I give up.
c) I am make Phase I 9/10 now, and Phase II 8/10. And the results are pretty steady. So I think I should go for Phase III.
d) I tried the food map thing for Phase III. I first run it by splitting the screen into several different pieces, and generate food map for (-20, +20) distance from the pacman. Then I found it didn't work since sometimes my teammate run to me and it waste much time. Thus I decided to use a risky way to do this. Since I firstly considered the ghost that I will not choose a step which will make me too close to the ghost, but this method contradicts my solution method since I find my way based on teammate and food but if I consider ghost there was just a mess. Well maybe things will go well next day.

July 21, 2018

Start Time: 10:00 AM

End Time: 1:00 PM

Time Spent: 3 Hours

Attempts:

a) This time I tried to do something for Phase III with my code in Phase I and II. I first tried to ignore the ghosts and go directly to the cooperation part. I found it pretty hard to change my logistics everytime I receive the broadcast from my teammate. I sometimes got caught up in a trap.
b) Temporarily what I consider the ghost is not to make my predicted steps for exploring food to be too close to the ghost, and others stays with previously

changed things.

c) I have a badminton competition today so I stopped here for now

July 21, 2018

Start Time: 9:00 PM

End Time: +1 1:00 AM

Time Spent: 3 Hours

Attempts:

a) I just realized that I need at least 8+ for Phase I and 9+ for Phase II for a guaranteed full credit, so I work on Phase I and II again. I rewrote the Phase I with a more formalized way. However, different from the last time i tried to formalize the code, I did not create a food map at the very beginning, but I do this after I judge if left side is empty, this can save me half of the time creating the map. When left side still have food, I delete all the food on the right side to force my pacman to go find food on the left side. Well, this does inspire me somehow, although it took me so long to debug. But I got 10/10 for Phase I

b) Oh! Phase II is a torture! I tried a new way to do this. I tried to use K-Means to split the food into several clusters. I use Elbow method to find the best K value (usually 3 or 4 according to my result). Then I go for each cluster to finish the cluster before exploring the next one. But there are always some bugs that make the pacman just do stupidly left-right-left-right…. I think maybe I should update my clusters values? Well, I tried but the problem is still there. I have to go to sleep since I'm too tired coming back from Sacramento for badminton. Final grade for Phase II is still 8/10. TAT

c) Final glance to Phase III. It is stupid but it can run at least. So good luck for me hhh.

July 23, 2018

Start Time: 4:00 PM

End Time: 7:00 PM

Time Spent: 3 Hours

Attempts:

a) I attempted to classify the food into clusters. The seperated food clusters may guide the pacman to decide which food to eat. The pacman will start from the very closest cluster and then explore all food in the cluster. When a cluster is fully explored, the pacman move to the next cluster.
b) I firstly do not consider my teammate and the ghost but just the route of the pacman.
c) I start to implement K-Means clustering to do the job. I use K-Means algorithms to do the clustering and use Elbow Analysis to find the best k value. Usually the food are splitted into 4 clusters. I tried to let the pacman to go for the food cluster by cluster. However, I realized that since there are many walls, even though the locations of food pairs are close (manhattan distance), it actually takes a long way to go from one to the other.

July 24, 2018

Start Time: 7:00 PM

End Time: 10:30 PM

Time Spent: 3.5 Hours

Attempts:

a) I tried to use some other clustering methods. I checked out bayes decision tree and gave it a try.
b) I want to build a tree which start from the pacman's location (the root), and then spread out according to the distance from the pacman. For each location on the map, the tree make a decision, and the decision (probability) of each node is calculated according a heuristic value calculation method. The method considers the distance from the closest ghost and the density of food (how many food around the location and what is the closest distance from the closet food to that location).

c) When I started building the algorithm, I realized that if I add too many features to each decision node, the calculation takes too long, except that I cut the depth of the tree. And it's actually really hard to determine the heuristic value, since there are always some conditions that the pacman do not go for the food but instead get over it. Or sometimes (indeed usually) the pacman just got stucked on a certain cycle and got trapped.

July 26, 2018

Start Time: 12:00 PM

End Time: 2:00 PM

Time Spent: 2 Hours

Attempts:

a) I decided to develop a clustering method which classify the food dots according to the distance between each other. I hope to divide them into many small clusters so that at a certain point, the pacman can make more flexible decisions (especially when the pacman meet the ghost and want to make some avoidance movements).
b) The algorithm considers the distance between the foods. Foods which has distance between which are <= 5 (the actual distance to go on the map) are classified into one cluster.
c) I do the design work for the algorithm, trying to make it has less complexity. Then I tried to mimic the situation how it can cooperate with the teammate. Then I may continue the work tonight.

July 26, 2018

Start Time: 7:00 PM

End Time: 11:30 PM

Time Spent: 4.5 Hours

Attempts:

a) The implementation process goes well. Now I can successfully split the food map into around 17 modules (classifications), each contain around 1~10 food.
b) Then I think, how can I make the pacman move? I can definitely let it find the closest food cluster or something like that, but I really do no want to increase the time cost on doing so. Thus, I decide to add some calculation on the init stage. For each location on the map, I calculate the closest food from it and label it with the module the food belongs to. Thus, wherever the pacman is at, it can know the module representation of his current location and make the decisions. If the module he is in has no more food available on the map, then he start to explore the closest food and the corresponding module. Otherwise, he continue exploring all the food in the module.
c) The algorithm works great! The pacman now work well, but just cannot cooperate with the teammate.
d) I design the cooperation method to be that I first calculate the distance between the teammate and its closest ghost, then, according to the future steps he broadcast, I delete the food he will explore in [the distance] steps, which I think will be a reasonable scale since the positions calculated according to the teammate's actions may be invalid if we consider too many steps.
e) The algorithm works well!
f) However, when I try the autograder, I found that it's different from the capture.py one since it says that I run out of time at the init stage. I may need to deal with it.

July 27, 2018

Start Time: 7:00 PM

End Time: 11:30 PM

Time Spent: 4.5 Hours

Attempts:

a) I gave up. [This took me 4 hours ヾ(ő∀ő๑)ﾉ]
b) I decide to go back and refine my previous ideas about "manually" help the bot to be smarter. Which means that I continue using the bot for Phase I and II but do some refinement.

c) I went back and start consider the situations for ghost avoiding. Honestly, I think my bot pretty well manage the situations for pure searching. The cooperation part is not hard as well. However, the ghost avoidance takes the most amount of time. I need to find a balance between eating dots as soon as possible and avoid the ghost. I tried some ways, but I may continue someday later. I have some personal stuff to do in the next two days. I saved my slip days for this moment and I have to use them for my PacPack and project 5.

July 28, 2018

Start Time: 10:00 PM

End Time: +1 2:00 AM

Time Spent: 4 Hours

Attempts:

a) I hoped I can save more time so that I do not need to use one more slip day for my PacPack …… [I am just 2 hours pass the EOD]
b) I updated the ghost avoidance. I noticed that when you go for the ghost, the ghost indeed run away, which is really interesting. I don't know if this is how the ghost was set, but I definitely want to use this feature. I firstly let the pacman to go for the ghost when they are too close.
c) Then I realize that when the ghost meets a corner (two walls or three walls around), it usually stop moving for some times. Then if I go for it, I die. Thus, I check the corner stages and let the pacman to try other ways when the ghost is at the corner and they are too close.
d) Then I let the pacman learn to "scare" the ghost, meaning when he is in a very bad situation, he may find a way to stay alive by fake run for the ghost and then try to find a way to get out of the trap.
e) This worked.