Ryan Monroe
rmonroe@ucsc.edu
CMPS 140
March 17, 2017

Sante

When I first began to plan my agent for the tournament, originally I was stumped. I had no direction we had learned so many agent types and algorithms how was I supposed to decide the perfect one to do well in the tournament. I needed something that would be quick and efficient to be effective within the time and move limits.

I began at square one, the baseline agents and the fundamental problem of the tournament, how to weigh defense vs offense. Looking over how the baseline agents performed I began to wonder if dedicated offensive and defensive agents were actually necessary. I noticed that it was actually more effective for both of my agents to be on offense and consume food faster than the one dedicated agent. Another conclusion I came to was that simpler might actually be better. The baseline agents performed very respectably by choosing random moves using such a simple heuristic, so I thought by improving the heuristic I could easily and effectively make a very successful agent.

So going off of what I had learned from the baseline agents I figured there were three cases I needed my agent to handle. First, was the food and score of the next move. This I took from the baseline agents. I lowered the success value a bit just to even thing out a bit. So when an agent is comparing a spot they value the higher scoring spot and one with the food over one that does not. Second, came the ghost and avoidance. There were three cases within this problem: the ghost in my next move, the ghost being scared, and me being in the neighbors of a ghost. I wanted to handle these in a way that would although the agent to avoid the ghost effectively yet avoid going to the extreme. Dialing in this heuristic was a bit of a challenge because of the neighboring moves heuristic. I wanted my agent to avoid ghost as much as possible, because I didn't have a real "escape" path. So early on I had it to high and my pacman were to scared to stay on the enemy side for long and too little and it made no difference. Along with this heuristic I added one that valued the power pellet a bit, so that I could move freely and not care about the ghost when I had it. I would have like to be able to have something like the winning team where I could have related this heuristic to whether my teammate was being chased or not. The final heuristic was the little bit of defense my agents do. If they are a ghost on there way to the other side of the board they will see if there is a pacman nearby and if they are in my next move I will eat them, but if they were just within my nearby moves I would go a bit further and eat them. I found this to be effective, because often it was just one of my agents going after a pac man and after would resume offense. This allowed for my team to buy time as the opposing offensive agent was reset. Once again this heuristic was also difficult to dial in, I kept getting my ghost and the opposing pacman to either becoming stuck or thrash, because it was just too good for them to leave. Eventually I got the heuristics to a point where I could achieve roughly an 80% win rate against the baseline team.

I decided on the reflex agent with a simple q-value system built on the baseline offensive agent, simply because of the confines of the tournament the move and time limit. I

thought I wanted to get as much food as possible and as quickly as possible. I assumed it would be more beneficial for me to just act in the moment rather than to plan a path or store information because both of these would violate the rules and my agents would crash. After hearing some of the other times I realize now that there were ways, but they didn't actually add much of an advantage. So looking over the baseline team this is exactly what the offensive agent did, he would navigate to the closest food at random. I knew if I changed this a bit I could easily start producing good results. This is exactly what I did I applied the heuristics discussed above and it turned out alright. Originally I left the food heuristic untouched, but then quickly discovered my agents were making the same moves and I was essentially playing with just one agent. This is when I decided to split the board in half, but rather than give the agent defined boundaries I simply divided the food list in half, one for each agent. This was also better than giving defined boundaries, because I updated as the food list changed. So as the food was eating the food list was recalculated and the agents could move on either side, but still would not be competing with the other teammate. This worked extremely well and increased my win rate against the baseline team drastically.

The computational strategy used to solve each problem is simply to multiply the feature of that action by the assigned weights. Each feature had a heuristic that was triggered to either increase the value or trigger a value high enough to always tell the agent to make that choice. Then by doing this for all of the future positions of the agent we can then choose the best option based on these values I left it as a random selection from this list of best possible moves due to not being able to find a way to see the true next best without a learning agent. But along with this came some of the challenges as well, such as creating the values to assign each feature. I needed to find the balance of a weight that would make the agent do what I want, but not high enough that the agent would then value it's current position higher than continuing with the game. I would have like to do more calculations and find more efficient paths through the maze, but could not find a way to do efficiently do it during the time limit. If i could have I would have like to do something similar to the winner and try to avoid, or decide to eat a pellet and fun into the ghost. I think that strategy would have been very effective, because my agents just ran around while avoiding the ghost randomly.

I believe my agent is very well done simply by the simplicity of it. I could have made a learning agent or used a different algorithm to choose the best path, but I didn't have to. I merely changed how the agent thinks about its next position and created a successful agent with an ~80% win rate against the baseline agents. Yes, there is more I would have liked to added, but that will always be the case there is a constant drive to improve. The main lesson I learned from this project is that simplicity is some time the best solution. I also was able to use a bit of history when designing a strategy to win against the other agents looking into military strategies. I looked into Blitzkrieg or "Lightning war" which is designed to quickly overwhelm then flank an opponent. Really this is what my strategy is doing I assumed most teams would uses an offensive and defensive agent pair so I thought I could overwhelm the single defensive agent with two offensive agent. Plus once one of the agents was being chased the other could move around freely and consume as much food as possible, similar to the flank. Going into this assignment I had no clue where to begin, so I began with the example agents. By simply changing the way the agent chose its moves I was able to create

a top 5 finisher. I really enjoyed this assignment and plan on continuing work on it to implement some other strategies I did not fully pursue.