Level-k Thinking Strategies for *Pacman* Players - Final Report

Hemanth Sarabu, Venkata Ramana Makkapati, Vinodhini Comandur

May 2^{nd}, 2018

1 Introduction

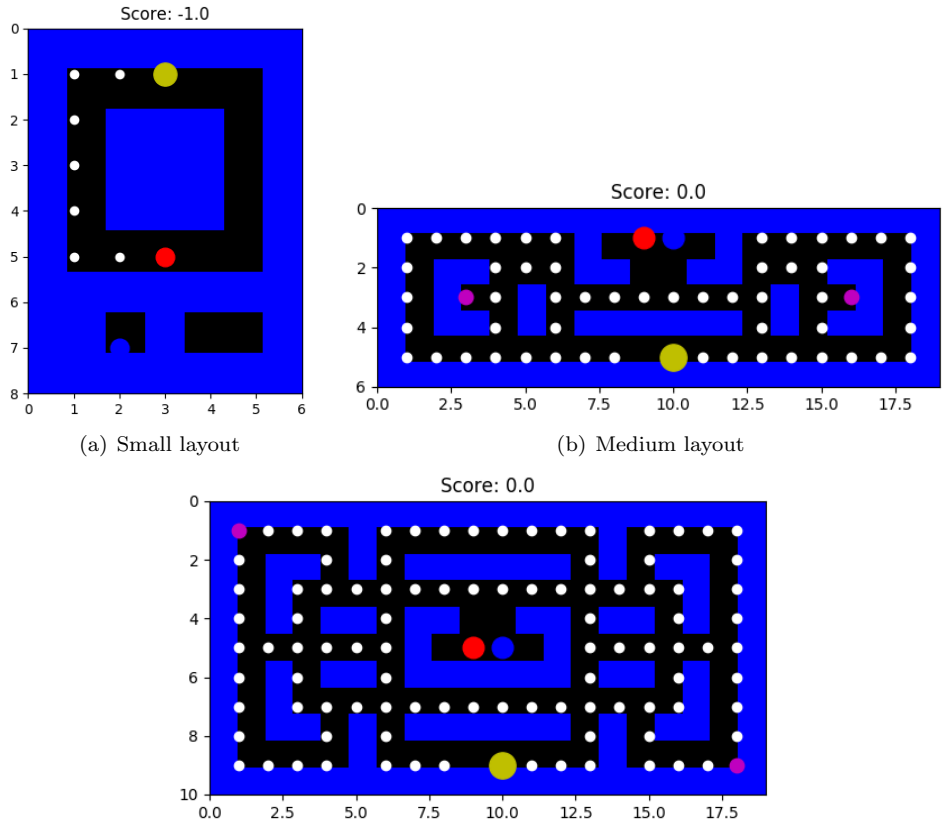
Level-k thinking (also called cognitive hierarchy theory) is a behavioral model used to describe human thought processes and outcomes in strategic games. It was developed by two researchers, Stahl and Wilson, after analyzing human behavior through a series a experiments [1]. It was understood that when a group of players have to perform under an adversarial situation, at first, each player tends to form a prior about the behavior of other players, and then chooses the best response given the prior. Players can differ in their priors and in their abilities to identify best responses. The simplest case is one in which a player selects a strategy at random without any prior belief, a level-0 player. A more sophisticated player of level-1 operates under a prior that his opponents are all level-0 players, and finds a best response. Generalizing this notion, a level-k player best-respond assuming that other players are distributed among level-$(k-1)$ and lower. Some more experimental data and a more mature version of this theory can be found in Refs. [2], [3].

Pacman is a popular arcade game and it is an ideal testbed to analyze the cognitive hierarchy theory (CHT) in depth leading to interesting results. In a general Pacman game, the ghosts' movements vary from random to semi-random, where each ghost has a specified task, including one of them pursuing Pacman (the agent). From the CHT point of view, ghosts can be categorized as level-0 players, and under that assumption, the best response for Pacman can be found, positioning it at level-1. There exists prior work [4–8], concerned about finding best responses for Pacman. The next step, finding the best response for the ghosts that assume Pacman to be a level-1 player, has been partially achieved in Refs. [9–11], where the ghosts are made to evolve using genetic algorithms. It is important to observe that the past studies have analyzed the best strategy for either Pacman or ghosts alone, without any evolution or training of the adversary. Additionally, the focus of research in earlier works was primarily on the technical aspects of the schemes that were employed for finding what can be called "higher level responses for the players", but it was not in the direction of CHT.

Through the course project, we initially wanted to analyze the best responses for the two sets of players (ghosts and Pacman) in the Pacman game, looking at the problem from a CHT perspective, and explore some directions listed below.

1. What is the best response for Pacman (ghosts) assuming that the ghosts are of level-0/2/4 (Pacman is of level-1/3/5)?
2. Is Pacman (ghosts) at a given level objectively superior to ghosts (Pacman) of all the lower levels or is it superior for just one level below?
3. What is a way to estimate the level of the opponent player online by observing the game?

All the above questions are new to the existing literature in the context of Pacman, and finding their solutions is a first step in extending them to a more general class of pursuit-evasion games. During the course of this project, a lot of challenges have shown up and the findings are discussed in the following sections.



(a) Small layout

(b) Medium layout

(c) large layout

Figure 1: Pacman layouts considered for the project

2 Reinforcement learning and Challenges

To address the first question, it was necessary to train Pacman against level-0 ghosts to obtain a level-1 Pacman and repeat this procedure by training ghosts and Pacman alternately against the lower level counterpart. UC Berkeley offers a course on the Introduction to Artificial Intelligence, with the lectures and source codes for some Pacman projects available in the open-source domain[1]. One of the projects focuses on reinforcement learning of Pacman to win the game against two ghosts. Brief details of the project as relevant to the current project are summarized below.

1. A traditional environment has been set up with multiple layouts as grids or classic Pacman (small and medium sizes) to choose from and the number of ghosts is fixed to two.
2. The full game state consists of the food, capsules, agent (Pacman and ghosts) configurations and score changes, which can be used by Pacman to reason about the game. Some basic rules of how Pacman and the ghosts interact with the environment have been specified, including the score calculation.
3. The objective is to use the environment created to implement Q-learning and approximate Q-learning for Pacman against the ghosts which can move randomly (level-0) or directional. Directional ghosts prefer to rush Pacman or flee when scared.

Challenge 2.1. *While the code from UCB is complete in terms of Pacman environment, it is inaccessible to make any changes that would help in implementing RL algorithms for level-k thinking.*

[1] http://ai.berkeley.edu/project_overview.html

Table 1: Reward Structure

Attributes	Pacman	Ghost
Time penalty	-1	-1
Food	2	-2
Ghost capture	100	-100
Pacman death	-500	500
Pacman win	500	-500

To overcome this challenge, an environment was built from scratch in python along with modules for Q-learning and approximate Q-learning. The Pacman layouts considered for this project are shown in Figure 1.

2.1 Rewards

The rewards for the players are documented in Table 1. The cumulation of rewards at every time instant results in the game score. Depending on whether Pacman or the ghosts are trained and tested, the corresponding reward scheme is invoked. It is important to note that both players are penalized at every move for the increasing time. While Pacman is rewarded whenever it consumes food or a scared ghost, or when it completes all the food (which is when Pacman wins the game), it has a major penalty if it dies, i.e. consumed by a ghost. On the other hand, the reward structure for a ghost depends on Pacman's penalties. Hence, the ghost's reward structure comprises more of penalties corresponding to the rewards Pacman receives. The only positive reward the ghost receives is when it eats Pacman.

Challenge 2.2. *The reward structure favors Pacman more and is sparse for the ghost. This could make it more challenging for a ghost to capture Pacman.*

This challenge is perhaps best overcome by allowing cooperation between the ghosts in some manner. The current format does not provide any such information and it could be an additional incentive in capturing Pacman.

2.2 Q-learning and its approximate variant

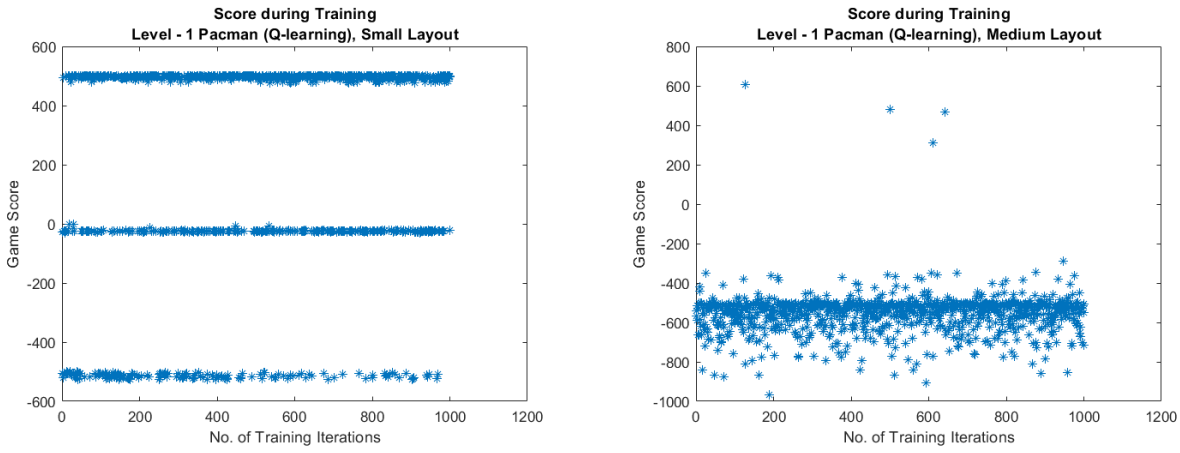
Q-learning algorithm is implemented and tested, where Pacman learns by trial and error from interactions with the environment through its update of the state, action and reward. The Q-values are computed using the Bellman equations to maximize rewards (refer Eq.1). It is important to note that the reward values are already encoded, with points gained as food is eaten and points constantly decreasing with time, which drives Pacman to complete the game soon. Brief tests were carried out with different values to study their effects on the game.

$$Q(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\left(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} \right)}^{\text{learned value}} \tag{1}$$

2.2.1 Training Pacman to achieve Level-1

Q-learning was successfully implemented and tested for attaining level-1 Pacman against level-0 ghosts for the small layout, which can be seen in the attached video, *qlearning_small.gif*. During the training phase, the number of games resulting in high negative rewards (which is indicative of Pacman losing) began decreasing

as the number of training iterations grew (refer Fig.2a). Furthermore, Pacman won around 75% of the games in testing phase with a high positive score, as seen in Table 2. However, Q-learning for level-1 Pacman did not work as well when trained and tested on larger layouts (refer the attached video, *qlearning_medium.gif*), where Pacman's average training rewards remained negative throughout training, despite a large number of training sessions, which is depicted in Fig.2b. At test time, he played badly, losing almost all of his test games (only 1% of wins as indicated in Table 2). Additionally, the training also took a long time, despite its ineffectiveness. This is a major caveat of Q-learning because each layout configuration is a separate state with separate Q-values. Pacman has no way to generalize that running into a ghost is bad for all positions. Obviously, this approach will not scale.



(a) (b)

Figure 2: Plots showing of evolution of final score during the training phase of Q-learning

Challenge 2.3. *Q-learning approach does not scale well with board configurations and the number of training sessions for larger layouts cannot be increased beyond a point due to RAM limitations. This would result in inefficient agents that underperform.*

This challenge is overcome by resorting to approximate Q-learning for training the agents. It uses a derived low dimensional form of state representation with Q-values approximated as a linear combination of certain features, where Pacman learns weights for these features. The mathematical representation of approximate Q-learning is summarized in Eqs. 2 through 4.

$$Q(s_t, a_t) = \sum_{i=1}^{n} f_i(s_t, a_t) w_i \tag{2}$$

$$w_i \leftarrow w_i + \alpha f_i(s_t, a_t) d \tag{3}$$

$$d = \Big(r_t + \gamma \max_a Q(s_{t+1}, a_t)\Big) - Q(s_t, a_t) \tag{4}$$

A feature extractor script is already provided in the source code files, where a simple version returns the following for a basic reflex Pacman -

1. Whether a ghost collision is imminent
2. Whether food will be eaten immediately
3. Whether a pill eating is imminent
4. How far away the next food is
5. What is the distance to the nearest pill

The approximate Q-learning function was successfully implemented and tested for level-1 Pacman against level-0 ghosts. This agent won almost every time with these simple features for even larger and complex layouts, as seen in the attached videos, *approxq_medium_L1.gif* and *approxq_large_L1.gif*. Figure 3 shows the game score trends during the training phase on the medium layout. Unlike the Q-learning training (refer Fig.2), approximate Q-learning shows a large number of games with high positive scores throughout the training. Furthermore, the learning is much quicker with steep convergence as seen in the first few iterations, where the number of games with large negative scores immediately reduces. Table 2 shows that Pacman wins a majority of the games in testing phase, with 85% in the medium layout and 94% in the large layout. Although training for level-1 Pacman was done on the medium layout, it is interesting to see that level-1 Pacman performs better when tested on the large layout. This is because the state space has grown with more pills and food, while the number of ghosts is maintained at two. This gives Pacman more state-space to explore and higher rewards to win the game.

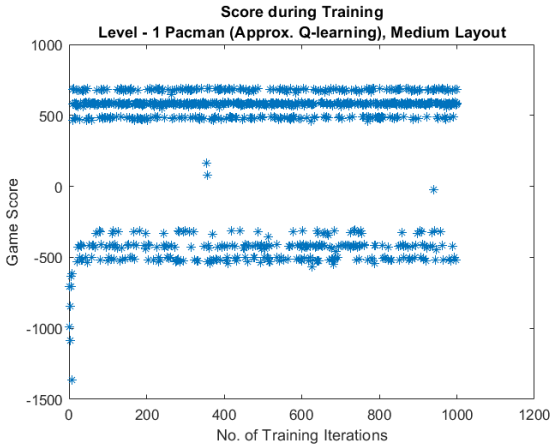


Figure 3: Evolution of final score during the training phase of approximate Q-learning for Level-1 Pacman

2.2.2 Training Ghost to achieve Level-2

With successful training of Pacman against level-0 ghosts, a level-1 Pacman was achieved and the corresponding weights were recorded. Now, by deploying level-1 Pacman in the medium layout, the objective was to train ghosts to beat level-1 Pacman in order to attain level-2 ghosts and determine the weights for features associated with ghosts. In contrast to the features identified for Pacman, the features for the ghost were simpler and direct -

1. What is the shortest distance between the ghost and Pacman
2. Is the ghost in 'afraid' mode, i.e. did Pacman eat a pill recently

The training was done using approximate Q-learning by considering the same rewards and features for both ghosts on the medium layout. As mentioned previously, no cooperation strategy was included. The trends in game scores with the number of training sessions can be seen in Fig.4. It is seen that the number of games with high positive scores are low, which implies a small number of wins for the ghosts. This is further corroborated by observing the percentage of wins in Table 2. The ghosts win only 12% in the medium layout and 47% in the large layout. The increased share of wins in the large layout is similar to the trend seen for Pacman, i.e. more state-space is available for exploring and capturing. The attached videos *approxq_medium_L2.gif* and *approxq_large_L2.gif* show the trained ghosts tested against level-1 Pacman in medium and large layouts. Although the ghosts have learned to actively pursue Pacman in order to capture it, the pursuit is not in an organized manner, thereby leading to multiple games being lost by ghosts (or won by Pacman). To summarize, the perfect "level-2" ghosts have not been achieved and this could be attributed to multiple reasons.

Challenge 2.4. *Level-2 ghosts could not be achieved with the current framework due to the sparse rewards for ghosts, lack of cooperation between the ghosts and insufficient features. Additionally, approximator function for Q-values is linear which limits its ability to include more detailed information about the game encapsulated in the features.*

This challenge could be overcome by improving the rewards structure and the quality of features. Providing path planning ability for the ghost or allowing some type of cooperation between the ghosts that would allow them to corner Pacman will be very useful. Another possible approach would be to consider a nonlinear approximator which could be obtained from deep Q-learning.

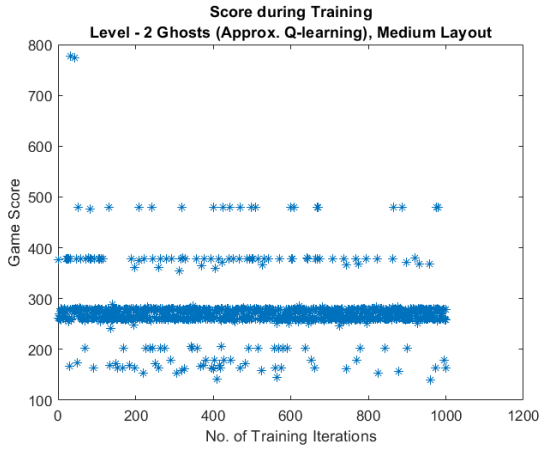


Figure 4: Evolution of final score during the training phase of approximate Q-learning for Level-2 ghosts

Table 2: Performance of the two learning methods for different layouts during the testing phase

Learning method	Layout and Level	Pacman Win %	Mean score	Standard deviation
Q-learning	Small layout with Level-1 Pacman	75 %	343	286
	Medium layout with Level-1 Pacman	1 %	-532	129
Approximate-Q	Medium layout with Level-1 Pacman	85 %	433	379
	Large layout with Level-1 Pacman	94 %	546	266
	Medium layout with Level-2 ghosts	88 %	277	43
	Large layout with Level-2 ghosts	53 %	126	53

3 What have we achieved and learned?

1. A level-1 Pacman has been successfully trained and tested, and the weights for the Q-value approximation have been obtained.
2. Key features which affect Pacman's ability to to learn and succeed in winning against level-0 ghosts have been identified.

3. A simple linear approximator function for Q-values, as opposed to actual Q-values, is capable of reducing number of training sessions and scaling for bigger layouts.
4. Achieving a good 'level-2' ghost is not as direct as obtaining level-1 Pacman using approximate Q-learning technique.

4 Some Future Directions

1. Further investigation to obtain level-2 ghost in the present setting is required by incorporating path planning, cooperation between the ghosts, improved features and rewards.
2. Achieve higher levels of Pacman and ghosts to ensure scaling and determining subsequent limitations.
3. Implementing more sophisticated techniques like deep q-learning with GPUs would help training multiple levels of agents better.
4. It would be interesting to determine if a higher level Pacman (say level-5) is objectively better than its lower level counterparts (level-0/2/4) or if it is unnecessarily 'over-thinking'.
5. Develop an observer for the agent to gauge the opponent's level from its movements and help the agent to adapt, so that it can play to win the game.

References

[1] D. O. Stahl and P. W. Wilson, "On players models of other players: Theory and experimental evidence," *Games Econ. Behav.*, vol. 10, no. 1, pp. 218–254, Jul. 1995.

[2] R. Nagel, "Unraveling in guessing games: An experimental study," *Am. Econ. Rev.*, vol. 85, no. 5, pp. 1313–1326, 1995.

[3] C. F. Camerer, T.-H. Ho, and J.-K. Chong, "A cognitive hierarchy model of games," *Q. J. Econ.*, vol. 119, no. 3, pp. 861–898, 2004.

[4] M. Gallagher and A. Ryan, "Learning to play Pac-Man: an evolutionary, rule-based approach," in *Congress on Evolutionary Computation, CEC*, vol. 4, Dec. 2003, pp. 2462–2469 Vol.4.

[5] I. Szita and A. Lõrincz, "Learning to play using Low-Complexity Rule-Based policies: Illustrations through ms. Pac-Man," *Journal of Articial Intelligence Research*, vol. 30, pp. 659–684, 2007.

[6] N. Tziortziotis, K. Tziortziotis, and K. Blekas, "Play Ms. Pac-Man using an advanced reinforcement learning agent," in *Lecture Notes in Computer Science*, 2014, pp. 71–83.

[7] L. Bom, R. Henken, and M. Wiering, "Reinforcement learning to train Ms. Pac-Man using higher-order action-relative inputs," in *IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, Apr. 2013, pp. 156–163.

[8] K. Ranjan, A. Christensen, and B. Ramos, "Recurrent deep Q-learning for Pac-man," 2016.

[9] F. Liberatore, A. M. Mora, P. A. Castillo, and J. J. Merelo, "Comparing heterogeneous and homogeneous flocking strategies for the ghost team in the game of Ms. Pac-Man," *IEEE Trans. Comput. Intell. AI Games*, vol. 8, no. 3, pp. 278–287, 2016.

[10] F. Liberatore, A. M. Mora, P. A. Castillo, and J. J. M. Guervós, "Evolving evil: Optimizing flocking strategies through genetic algorithms for the ghost team in the game of Ms. Pac-Man," in *Applications of Evolutionary Computation*. Springer Berlin Heidelberg, 2014, pp. 313–324.

[11] M. Gallagher and M. Ledwich, "Evolving Pac-Man players: Can we learn from raw input?" in *IEEE Symposium on Computational Intelligence and Games*, Apr. 2007, pp. 282–287.