CSCE475/875 Multiagent Systems

Handout 12: Game Day 1 Learning Day Analysis

February 19, 2020

State Transition Map and Rewards

There were six states (S1-S6) and six actions (A1-A6). Each team started with S1. Each team was capable of performing all six actions. Table 1 shows the rewards for transitioning into each state and, its average and standard deviation, based on a Gaussian distribution.

	Average	Std. Dev.
S1	$0	$10
S2	$100	$10
S3	$1500	$10
S4	$500	$10
S5	$5000	$10
S6	$1000	$10

Table 1. Rewards, average and standard deviation values, Gaussian distribution.

Table 2 shows the probabilistic transition map for each state-action pair. Looking at both Tables, if one aimed to obtain the highest reward for a state (i.e., S5 @ $5000), then starting for S1, one would probably have to go with A4 to transition into S4 (with a high probability @ .7), and then go with A5 to transition into S5 (with a high probability @ .9). And then to get back to S1, one could perform an action of A6, if so desired. This sequence of A4-A5-A6, when repeated, should allow an agent to reach S5 with a relatively high probability (= .7 x .9 x .5 = .315), and a relatively high reward (= $500 + $5000 + $0 = $5500). With enough exploration, an agent should be able to discover this sequence.

	A1	A2	A3	A4	A5	A6
S1	→ S1 (.50) → S2 (.30) → S3 (.15) → S4 (.05)	→ S2 (.80) → S3 (.20)	→ S2 (.10) → S3 (.90)	→ S1 (.05) → S2 (.25) → S4 (.70)	NA	NA
S2	→ S1 (.70) → S2 (.15) → S4 (.15)	→ S1 (.55) → S3 (.35) → S4 (.10)	NA	NA	NA	→ S5 (.50) → S6 (.50)
S3	NA	NA	→ S1 (.70) → S2 (.20) → S4 (.10)	→ S1 (.60) → S3 (.30) → S4 (.10)	NA	NA
S4	→ S1 (.60) → S2 (.20) → S3 (.20)	→ S1 (.65) → S2 (.14) → S3 (.20) → S4 (.01)	→ S1 (.98) → S2 (.02)	NA	→ S5 (.90) → S6 (.10)	NA
S5	NA	NA	NA	NA	NA	→ S1 (.50) → S2 (.30) → S3 (.15) → S4 (.05)
S6	→ S1 (1.0)	→ S2 (1.0)	→ S3 (1.0)	→ S4 (1.0)	NA	NA

Table 2. State transitions by actions. NA for a state-action cell means the action is not applicable for the state.

Because of the limited time on Game Day, we did not expect teams to obtain *accurate* Q-values. However, teams should be able to obtain *fairly accurate ordering* of their Q-values. The ordering of the best state-action pairs (Q(s,a)) is as follows:

Group 1: (S5,A6)
Group 2: (S4,A5); (S5,A1); (S5,A2); (S5,A3); (S5,A4); (S5,A5)
Group 3: (S2,A6); (S6,A4)
Group 4: (S3,A3); (S3,A4); (S4,A4); (S6,A2); (S6,A3)
Group 5: (S1,A4); (S4,A1); (S4,A2); (S4,A6); (S6,A1); (S6,A5); (S6,A6)

For the above, we also define a function called Group_true(s,a) that returns the group ID of a state-action pair. So, for example, Group_true(S5,A6) is 1; Group_true(S4,A6) is 2; Group_true(S5,A1) is 2; and so on.

Team Statistics

Tables 3 and 4 show the ordering of the teams after Round 1 and Round 2, respectively.

To compute the accuracy of a Q-table, we use the grouping shown earlier. We consider only the top 18 state/action pairs in each team's Q-table (where 18 is half of the 36 possible values). (Important Note: the last group actually only has 4 elements (not 7) when we limit ourselves to only looking at the top 18 for each group. We however put 7 state/action pairs in Group 5 to be fair to teams since they are all pretty equivalent in that group, and using only 4 would mean teams wouldn't get credit if they had the other 3 (equivalent) pairs, instead.)

First, we sort each team's Q-values.

And second, for each state-action pair on the sorted list, we assign Group_found(s,a) using the 1-6-2-5-7 grouping strategy. So, take GZ's Round 1 ordering: Group_found(S4,A5) is 1; Group_found(S6,A3) is 2; Group_found(S2,A6) is 2; Group_found(S6,A4) is 2; and so forth. (Please see the color-coding in Tables 3 and 4).

Third, we compute two subvalues: matching score, and non-matching score. For matching score, if Group_found(s,a) == Group_true(s,a), then we will multiply it with a weight and add it to the score: weights = 1, 0.5, 0.25, 0.125, and 0.1 for the five groups, respectively. This scheme rewards teams that have high accuracy for the top state-action pairs. For the non-matching score, if Group_found(s,a) – Group_true(s,a) == 1 OR Group_true(s,a) – Group_found(s,a) == 1, then we will multiply it with the lower group weight of Group_found(s,a), Group_true(s,a) and add to the score. This is to compensate state-action pairs that miss their true grouping just by one group.

Then we add up the matching and non-matching scores.

Rank	GZ	Matrix	Null Pointer*	Optimal Alligators	Simulated Ground Truth
1	S4-A5	S2-A6		S2-A6	**S5-A6**
2	S6-A3	S2-A2		S4-A5	**S4-A5**
3	S2-A6	S1-A3		S1-A3	**S5-A1**
4	S6-A4	S2-A1		S5-A6	**S5-A2**
5	S1-A4	S1-A4		S1-A2	**S5-A3**
6	S1-A2	S3-A3		S3-A3	**S5-A4**
7	S4-A1	S1-A2		S3-A2	**S5-A5**
8	S1-A3	S5-A6		S6-A6	**S2-A6**
9	S6-A2	S1-A1		S6-A1	**S6-A4**
10				S5-A5	**S3-A3**
11				S5-A4	**S3-A4**
12				S5-A1	**S4-A4**
13				S2-A1	**S6-A2**
14				S1-A1	**S6-A3**
15				S2-A5	**S1-A4, S4-A1, S4-A2, S4-A6, S6-A1, S6-A5, S6-A6**
16				S5-A3	
17				S2-A4	
18				S3-A3	

Table 3. The ordering of state-action pairs from each team after Round 1. (Only the top 18 state-action pairs are listed) Colors show grouping. * Team did not submit the correct Q-matrix.

Rank	GZ	Matrix	Null Pointer *	Optimal Alligators	Simulated Ground Truth
1	S3-A5	S2-A6		S4-A5	**S5-A6**
2	S4-A5	S5-A6		S2-A6	**S4-A5**
3	S1-A6	S2-A6		S1-A3	**S5-A1**
4	S6-A3	S2-A3		S4-A1	**S5-A2**
5	S2-A6	S3-A3		S1-A4	**S5-A3**
6	S5-A4	S2-A1		S1-A2	**S5-A4**
7	S6-A4	S6-A2		S5-A6	**S5-A5**
8	S1-A4	S1-A2		S3-A3	**S2-A6**
9	S1-A2	S1-A4		S1-A1	**S6-A4**
10	S4-A1	S2-A1		S2-A1	**S3-A3**
11	S1-A3	S5-A6		S1-A6	**S3-A4**
12	S6-A2	S2-A3		S3-A5	**S4-A4**
13		S2-A4		S5-A5	**S6-A2**
14		S2-A5		S1-A5	**S6-A3**
15		S5-A5		S5-A1	**S1-A4, S4-A1, S4-A2, S4-A6, S6-A1, S6-A5, S6-A6**
16		S3-A5		S5-A4	
17		S2-A3		S3-A1	
18				S4-A6	

Table 4. The ordering of state-action pairs from each team after Round 2. (Only the top 18 state-action pairs are listed) Colors show grouping. * Team did not submit the correct Q-matrix.

Now, we present the more detailed team statistics in Tables 5-7. The number of actions and rewards were tallied based on the log that our program captured during the Game Day. As shown in Table 5, after Round 1, Optimal Alligators performed the most actions (95) and earned the largest reward and with the highest efficiency. On the other hand, Matrix performed only 24 actions, earning the least reward and with the lowest efficiency. Furthermore, their Q-matrix did not register Q values for high-rewarding <s,a> pairs well, resulting in a 0 score.

Team Name	#actions	Rewards	Efficiency	Normalized	Order Accuracy	Normalized	Total
GZ	69	$43,798.66	$634.76	0.559	**1.125**	**1.000**	1.559
Matrix	24	$9,582.00	$399.26	0.122	0.000	0.000	0.122
Null Pointer*	88	$37,592.90	$427.19	0.480	NA	NA	NA
Optimal Alligators	**95**	**$78,350.16**	**$824.74**	**1.000**	1.000	0.889	**1.889**
Average	**69.00**	**$64,893.42**	**$571.49**				

Table 5. *Statistics of Round 1*. Optimal Alligators had the best total score, balancing between rewards and order accuracy, for Round 1. GZ scored the highest order accuracy with 1.125, while Optimal Alligators obtained the largest amount of rewards with $78,350.16. Bold red texts = high value *Team did not submit the correct Q-matrix.

Table 6(a) shows only the statistics during Round 2, and *not* the total. Unexpectedly, there average number of actions taken was smaller than that in Round 1. Null Pointer took significantly fewer actions. In terms of Rewards, as expected, Round 2 yielded a higher average than Round 1 ($77,453.39 vs. $64,893.42). This is because all teams exploited better to gain rewards more efficiently. Note that GZ's efficiency increased the most from Round 1 to Round 2, meaning that the team exploited what they learned in Round 1 very well. The average order accuracy for Round 2 was higher than that for Round 1 as well, as expected due to teams carrying out more actions and gaining more "learning episodes." Note also that Optimal Alligators attempted to explore and gained more knowledge about the state-action space but ended up achieving the same order accuracy as GZ that attempted to exploit as much as possible. **There is a key insight here.** More learning episodes and exploration should lead more accurate ordering. Yet, Optimal Alligators did not achieve more accurate ordering. One likely reason is that Optimal Alligator in their attempt to explore attempted many different <s,a> combinations such that they require even more actions in order to achieve accurate ordering. Another possible but less likely reason was inaccurate computation of the Q-value in Table 4 by GZ: Round 1's Q value is almost 5 times greater than Round 2's Q value for for GZ.

Team Name	#actions	Rewards	Efficiency	Normalized	Order Accuracy	Normalized	Total
GZ	83	**$145,352.30**	**$1,751.23**	**1.000**	**1.475**	**1.000**	**2.000**
Matrix	22	$13,172.02	$598.73	0.091	0.750	0.509	0.600
Null Pointer*	59	$61,039.27	$1,034.56	0.420	NA	NA	NA
Optimal Alligators	**95**	$90,249.98	$960.11	0.621	1.475	1.000	1.621
Average	**64.75**	**$77,453.39**	**$1,086.16**				

Table 6(a). *Statistics of Round 2 (**not** including Round 1's rewards and # actions)*. GZ had the best total score, balancing between rewards and order accuracy, for Round 2. GZ and Optimal Alligators scored the highest order accuracy with 1.475 while GZ obtained the largest amount of rewards with $145,352.30. Bold red texts = high value *Team did not submit the correct Q-matrix.

Furthermore, though the grand total of the two rounds was not used in our scoring directly, we provide the grand total values for all teams here as a reference in Table 7. Optimal Alligators performed the most actions in each round. However, they did not exploit as well as GZ in Round 2.

Team Name	#actions 1	Rewards 1	#actions 2	Rewards 2	#actions Total	Rewards Total
GZ	69	$43,798.66	83	**$145,352.30**	**152**	**$189,650.90**
Matrix	24	$9,582.00	22	$13,172.02	46	$22,754.31
Null Pointer*	88	$37,592.90	59	$61,039.27	147	$98,632.18
Optimal Alligators	**95**	**$78,350.16**	**95**	$90,249.98	190	$168,600.10

Average	69.00	$64,893.42	64.75	$77,453.39	133.75	$119,909.37

Table 7. *Total rewards and total number of transactions after Round 2. GZ* had the highest rewards total with $189,650.90. Bold red texts = high value

To compute the final score for the Learning Day, we compute the following score for each round:

$$Score = OrderAccuracyNormalized + RewardsNormalized$$

And then we combine both rounds of scores to obtain the final score:

$$FinalScore = 0.5*Score(Round1) + 0.5*Score(Round2)$$

For *OrderAccuracyNormalized*, we normalize each team's order accuracy by the best order accuracy achieved by a team. So, the best team will have its *OrderAccuracyNormalized* = 1.0.

For *RewardsNormalized*, we normalize each team's total rewards (i.e., rewards earned from performing actions + revenue from selling Q-table – cost from purchasing Q-table) with the best rewards earned by a team. So, the best team will have its *RewardsNormalized* = 1.0.

Table 8 shows the result. Overall, GZ scored the highest overall total with 3.559. Optimal Alligators scored closely at second: 3.510, only 0.049 behind the winner of the Game Day. Matrix finished third. Null Pointer did not submit correct Q-matrices, and, as a result, did not register a score. They finished 4th.

Team	Round 1 Score	Round 2 Score	Final Game Day Score
GZ	1.559	**2.000**	**3.559**
Matrix	0.122	0.600	0.722
Null Pointer*	NA	NA	NA
Optimal Alligators	**1.889**	1.621	3.510

Table 8. *Final Game Day scores*. Final Game Day Score = 0.5*Round 1 Score + 0.5*Round 2 Score. Bold text = high value. *Team did not submit the correct Q-matrix.

Individual Team Analysis

First, Table 9 shows the learning rate and discount factor used in Round 1 and Round 2 by each team. Null Pointer's alpha (learning rate) and beta (discount factor) were not submitted to the game site and thus not recorded.

Team Name	Round 1		Round 2	
	Alpha	**Beta**	**Alpha**	**Beta**
GZ	1	0	1	1
Matrix	0.1 → decreasing	0.85	0.05	0.90
Null Pointer	NA	NA	NA	NA
Optimal Alligators	0.7484 → 0.1 in 70 iterations	0.3	0.15	0.5

Table 9. Learning rates and discount factors used by each team for Round 1 and Round 2.

Before we start looking at teams individually, here is a general sense of the two rounds and the role of the intermission's information sharing.

In general, Round 1 is more for exploration, and Round 2 is for a bit more exploitation. That is, Round 1 should be used to explore different state-action pairs. And as a result, one should use a higher learning rate, to emphasize each current transaction and its reward more. If a team carried out a large number of actions in Round 1, then that team could use Round 2 more for exploitation since it would be rather confident that its Q-values had converged. In that scenario, using a lower learning rate and a bigger discount factor would help towards that.

There are also other factors. Note that for any learning approach to work, in particular for reinforcement learning to work, there must be sufficient learning episodes. In this Game Day, that means each team should secure a lot of transactions in order to better model the stochastic nature of the environment.

Conceptually, the learning rate should decrease from Round 1 to Round 2. However, we see that for two teams (i.e., Optimal Alligators and GZ), the learning rate was kept constant. For Matrix, they actually lowered the learning rate from Round 1 to Round 2. *But, inexplicably, they chose an extreme low learning rate for Round 1. A learning rate that low would not allow the Q-learning algorithm to learn anything meaningful. And this explained why their order accuracy was 0 after Round 1 even though they had 20+ actions.*

Teams did *not* take advantage of the intermission to do information gathering. For example, if a team realizes that they had not performed many actions, then it would be rational for that team to seek out other information and perhaps purchase a successful team's Q-matrix, such that they could exploit that to gain rewards in Round 2. Matrix had the motivation to do this in particular, but they did not choose to act on this opportunity.

Table 10 documents my comments on each team's worksheet and reports. My observations are contextualized on the discussions above. For "Post-Game", I selected some statements from each team's post-game analysis.

Team Name	**Comments**	
GZ	**Pre-Game**	Fairly detailed strategies. But planned to turn alpha and beta to both 0 and 0 in Round 2 was not rational, as that would mean no learning at all, assuming that the optimal solution could have been found in Round 1 alone. How could an agent be so certain of that?
	Round 1 Tracking	Not accurately updated
	Mid-Game	Made a significant strategic change: turned alpha and beta to both 1 and 1. That was not rational, as that would mean forgetting what have been learned in previous time ticks.
	Round 2 Tracking	Not accurately updated
	Post-Game	They did not correctly submit their Q-matrices.
	My Observation	This team did fairly well due to their speed in carrying out the actions (and computing Q(s,a) values due to alpha and beta both being 1s. Not clear how they selected their actions.
Matrix	**Pre-Game**	Lack of understanding of alpha (learning rate). It was set too low: an agent with that learning rate would not be able to learn well. No strategic contingency. Less prepared due to lack of automation.
	Round 1 Tracking	Correctly updated
	Mid-Game	Didn't change strategies.
	Round 2 Tracking	Correctly updated
	Post-Game	Didn't relate to multiagent system design
	My Observation	This team's choice of learning rate was not conducive to agent learning.
Null Pointer	**Pre-Game**	Fairly good strategies with contingency. However, there was a lack of understanding about discount factor: it does not matter in the exploration vs. exploitation tradeoff, at least not directly. The discount factor is more for looking ahead: if your best solution path requires several steps, including some "bad" or "low rewarding" steps, then a high beta will allow you find that path. In other words, a low beta would delay learning convergence, especially if the optimal state or state-action pairs are

		surrounded by layers of bad state or state-action pairs.
	Round 1 Tracking	Correctly updated. But Q-matrix not correct.
	Mid-Game	Changed their strategies after making mistakes in tracking and learning from observing other teams.
	Round 2 Tracking	Correctly updated. But Q-matrix not correct.
	Post-Game	"An agent who acts faster than other agents gains a large reward in situations where speed is important."
	My Observation	This team was able to carry out many actions to gain fairly large amounts of rewards in both rounds. However, they didn't generate the correct Q-matrix in each round. Otherwise, they would have placed third.
Optimal Alligators	**Pre-Game**	Fairly well thought out pre-game strategy. But not enough contingency, and also seemed to look at 70 iterations as a sufficient number for learning.
	Round 1 Tracking	Correctly updated
	Mid-Game	Changed alpha and beta, with the correct reasoning. Good observations.
	Round 2 Tracking	Correctly updated
	Post-Game	They observed that in Round 2 the same state-action pair resulted in negative rewards consistently. No high-level insights or observations.
	My Observation	This team executed fairly well in balancing exploitation and exploration. They covered the most <state,action> pairs. Had the team used a higher beta (~0.85), they would have obtained a much higher order accuracy, and would have won the game day.

Table 10. My comments and observations of team strategies, worksheets, and reports.

Lessons Learned

Here are some overall lessons learned.

1. In general, more transactions led to better learning. Thus, acting quickly and efficiently was critical. Teams that were slow in submitting their actions received fewer transactions, leading to poorer performances.
2. Using a low learning rate in Round 1 usually did not fare well. Using a low discount factor also did not yield accurate Q-values.
3. Lowering the learning rate or keeping it the same appeared to work better than increasing the learning rate from Round 1 to Round 2 for this MAS environment. In general, ***increasing the learning rate as time progresses would tend to unlearn what has been learned.***
4. Using a high discount factor could have a clamping effect on the learning performance brought on by a high learning rate. This is because looking into the future term essentially incorporates other Q-values into the fray. At the same time, **using a high discount factor also allows an agent to find solution paths that start with low rewards but yield high rewards eventually.**
5. Several teams pointed out the nature of a tradeoff at play: ***trying to maximize rewards while trying to maximize the order accuracy***. These two objectives are in a tug-of-war. Maximizing rewards reduces exploration and increases exploitation, and vice versa with maximizing the order accuracy. Several teams had adopted an opportunistic balancing act: if they encountered a "rewarding" good state, they would keep acting on it until it transitioned out.
6. Teams that were better prepared—that came with the iterative valuation of the Q-learning algorithm and/or a program/application—performed better and thus were ranked higher. As an agent, each team should be observant, adaptive, responsive, and reflective. Not all teams were "responsive" in a timely manner.
7. **Note also that the Q-learning or reinforcement learning does *not* tell us which actions to take given a particular state. However, it does *inform* us that up to now, based on our**

experience, the Q-value of some state-action pairs and the value of a state. This information allows us to carry out our decision making: Should we explore some more? Should we exploit now?

Game Days League

Here are the League Standings.

Team Name	Learning Day	Voting Day	Auction Day	League Standings
GZ	1			1
Optimal Alligators	2			2
Matrix	3			3
Null Pointer	4			4

Addendum

We ran hundreds of thousands of iterations given Tables 1 and 2, with different alpha (learning rate) and beta (discount rate) values, to generate the Q-tables. Here we include a table for beta = 0.8 to give you a sense of the Q-value for each state-action pair.

S5	a6	10786.2249
S4	a5	8916.7527
S5	a1	8628.9786
S5	a2	8628.9786
S5	a3	8628.9786
S5	a4	8628.9786
S5	a5	8628.9786
S6	a4	8133.4008
S2	a6	7667.849
S3	a3	7247.5822
S3	a4	7216.2336
S6	a2	7134.2779
S4	a4	7133.4008
S4	a6	7133.4008
S1	a4	6798.9062
S6	a3	6798.0645
S6	a5	6506.7194
S6	a6	6506.7194
S6	a1	6439.1237
S4	a1	6149.9427
S2	a3	6134.2779
S2	a4	6134.2779
S2	a5	6134.2779
S4	a2	6125.1762
S1	a2	6067.0352
S4	a3	5953.0268
S2	a1	5897.5384
S2	a2	5834.1807
S1	a3	5831.6858
S3	a1	5798.0645
S3	a2	5798.0645

S3	a5	5798.0645
S3	a6	5798.0645
S1	a1	5786.2249
S1	a5	5439.1237
S1	a6	5439.1237