General remarks

- Deadline: 11 December 2016
- Deliver the assignment to Roxana Rădulescu: rradules@vub.ac.be
- Provide an archive containing a PDF file with a self-contained report together with the code used to obtain all your results
- Put your name and affiliation (VUB/ULB) both inside the PDF document and in the archive name

1 N-Armed Bandit

Implement the N-Armed bandit problem using each of the following action selection methods:

- Random
- ϵ-Greedy with parameter ϵ
- Softmax with parameter τ

All Q_{a_i} are initialized to 0. The rewards are subject to noise according to a normal probability distribution with mean $Q^*_{a_i}$ and standard deviation σ_i, $i = 1, \ldots, N$. For each of the following exercises, run your simulation for 1000 time steps and plot the following graphs:

- One combined plot of the average reward for each algorithm (1 graph)
- One plot per arm showing the $Q^*_{a_i}$ of that action along with the *actual* Q_{a_i} estimate over time (All action selection methods should be depicted on the same plot) (One graph per arm).
- For each algorithm, plot a histogram showing the number of times each action is selected (One graph per action selection strategy).

See Section 2.2 of "Reinforcement Learning, An Introduction" by Sutton and Barto for a more comprehensive explanation of the problem[1].

1.1 Exercise 1

Let $N = 4$, $Q^*_{a_i}$ and σ_i as in Table 1.

Action	$Q^*_{a_i}$	σ_i
action #1	2.3	0.9
action #2	2.1	0.6
action #3	1.5	0.4
action #4	1.3	2

Table 1: $Q^*_{a_i}$ and σ_i for each action.

Run the following algorithms:

- Random
- ϵ-Greedy with parameter $\epsilon = 0$
- ϵ-Greedy with parameter $\epsilon = 0.1$
- ϵ-Greedy with parameter $\epsilon = 0.2$

[1] http://www.cs.ualberta.ca/%7Esutton/book/ebook/node16.html

- Softmax with parameter $\tau = 1$
- Softmax with parameter $\tau = 0.1$

Comment the results. Which algorithms arrive close to the best arm after 1000 iterations? Which one arrives there faster? Explain your findings?

1.2 Exercise 2

Re-run Exercise 1 doubling the standard deviations of each arm, and comment the results. Discuss the results. Is this problem harder or easier to learn? Does the performance of the algorithms change significantly? Which of the above performs best now?

1.3 Exercise 3

Re-run exercise 1 with two additional algorithms, and plot their results. The new algorithms have a time-varying parameter, which depends on the iteration number $t = 1, \ldots, 1000$ as follows:

- ϵ-Greedy with parameter $\epsilon(t) = 1/\sqrt{t}$
- Softmax with parameter $\tau = 4 * \frac{1000-t}{1000}$

Discuss the results. Are these algorithms better than the ones with a fixed parameter?

2 Stochastic Reward Game

	a_1	a_2	a_3
b_1	$\mathcal{N}(11, \sigma_0^2)$	$\mathcal{N}(-30, \sigma^2)$	$\mathcal{N}(0, \sigma^2)$
b_2	$\mathcal{N}(-30, \sigma^2)$	$\mathcal{N}(7, \sigma_1^2)$	$\mathcal{N}(6, \sigma^2)$
b_3	$\mathcal{N}(0, \sigma^2)$	$\mathcal{N}(0, \sigma^2)$	$\mathcal{N}(5, \sigma^2)$

Table 2: Stochastic climbing game

Table 2 shows the stochastic climbing game. It is a cooperative game, in which two agents must learn to maximize the score they receive from the game. Each has three available actions, and the rewards given to both agents depend on the actions of both agents, and are stochastic. For each combination of actions, the reward received by both agents is sampled from a normal distribution with the mean presented in the above table and a standard deviation of σ_0 (for the joint action $< a_1, b_1 >$), σ_1 (for the joint action $< a_2, b_2 >$) or σ (for all the other possible joint actions).

Implement two types of Joint-Action learners in the above problem. The first type with simple Boltzmann action selection with a fixed temperature of your choice. The second type with a more advanced multi-agent action selection heuristic of your choice (optimistic boltzmann, FMQ, ...). Recall that in Joint-Action learning, you learn about the quality of joint actions (9 in total in this setting), and you often have to maintain beliefs about the other agent's strategy. An explanation of these concepts can be found here: [1]. Further useful references are [3, 2, 6, 5, 4].

2.1 Plotting

Consider the following three cases:

(a) $\sigma_0 = \sigma_1 = \sigma = 0.2$

(b) $\sigma_0 = 4$ and $\sigma_1 = \sigma = 0.1$

(c) $\sigma_1 = 4$ and $\sigma_0 = \sigma = 0.1$

For each case, play two agents of the first type for 5000 trials and do the same for two agents of the second type. After learning, plot the following on the same figure, for each of the three cases:

- Total collected reward per episode versus the episode number for each of the two types.

You should have three plots (one for each case), each with two lines (one for each type). Using these plots, you can compare the two algorithms and decide which of the two types learns best in each setting. You can smooth the results using a moving average for readability. Discuss your results!

2.2 Discussion

Discuss the following:

1. How will the learning process change if we make the agents independent learners?
2. How will the learning process change if we make the agents always select the action that according to them will yield the highest reward (assuming the other agent plays the best response)?

NOTE: The choice of the programming language is free. Save yourself work by creating an abstraction for the number of actions and all parameters, so you can re-use your code. **Good luck !!!**

References

[1] C. Claus and C. Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems. *AAAI/IAAI*, (s 746):752, 1998.

[2] S. Kapetanakis and D. Kudenko. Reinforcement learning of coordination in heterogeneous cooperative multi-agent systems. In *Adaptive Agents and Multi-Agent Systems II*, pages 119–131. Springer, 2005.

[3] S. Kapetanakis, D. Kudenko, and M. Strens. Learning of coordination in cooperative multi-agent systems using commitment sequences. *Artificial Intelligence and the Simulation of Behavior*, 1(5), 2004.

[4] A. Nowé, P. Vrancx, Y. De Hauwere, M. Wiering, and M. van Otterlo. Reinforcement learning: state-of-the-art. *Game Theory and Multi-agent Reinforcement Learning*, pages 441–470, 2012.

[5] K. Verbeeck, A. Nowé, J. Parent, and K. Tuyls. Exploring selfish reinforcement learning in repeated games with stochastic rewards. *Autonomous Agents and Multi-Agent Systems*, 14(3):239–269, 2007.

[6] K. Verbeeck, A. Nowé, M. Peeters, and K. Tuyls. Multi-agent reinforcement learning in stochastic single and multi-stage games. In *Adaptive Agents and Multi-Agent Systems II*, pages 275–294. Springer, 2005.