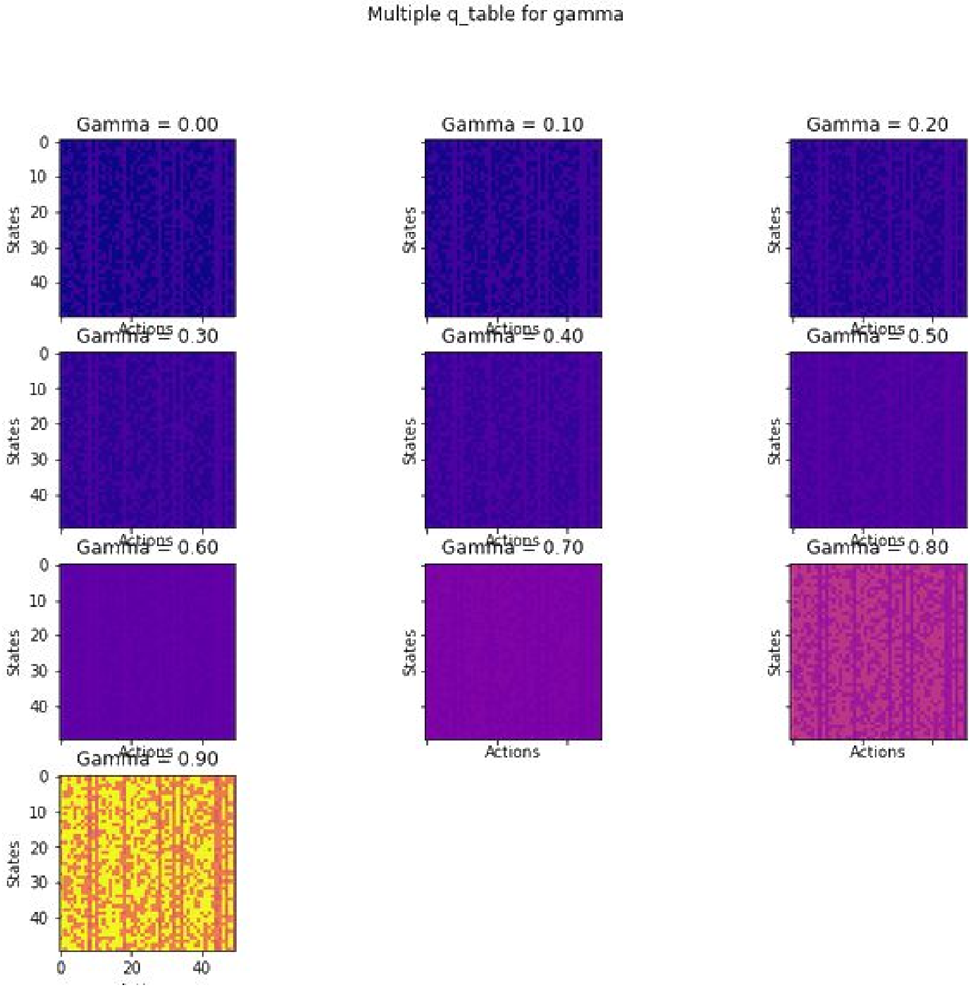
Report 28/04 : Reinforcement learning for Cache-Friendly Recommendations

Jade Bonnet and Clément Bernard

1) Normalize the q_table for different values of gamma

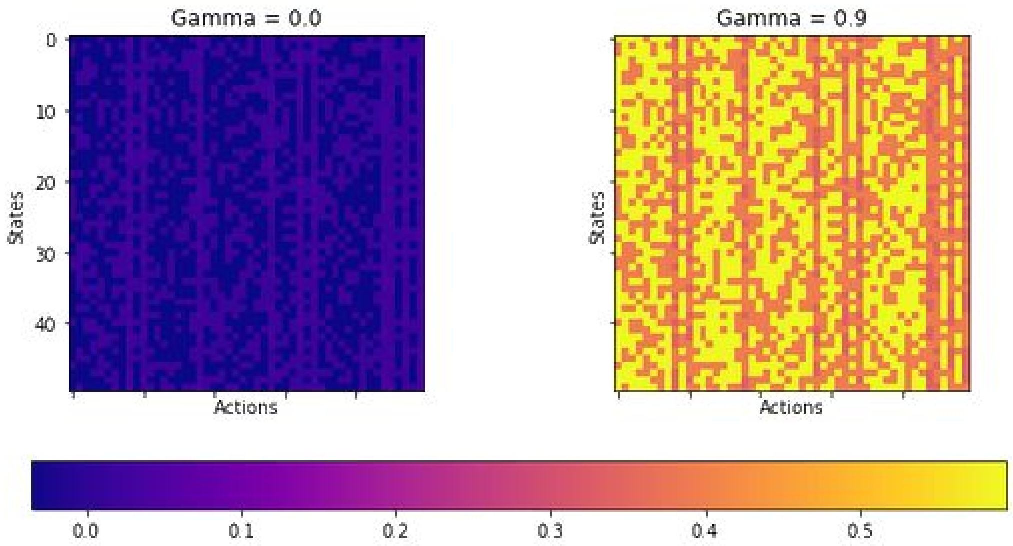
We normalized the q_tables to visualize them in a better way.



Normalized q_tables for different values of gamma (200 000 epochs)

We can see that the values of the q_table for high values of gamma are higher than for low values of gamma (as expected).

Here is the plot for gamma = 0 and gamma = 0.9 :



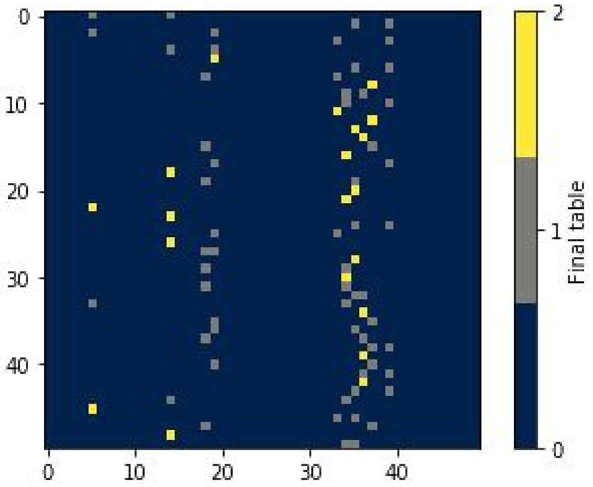
Normalized q_table for gamma = 0 and gamma = 0.9 (200 000 epochs)

2) Compare reward matrix row by row with q_table

We computed the final matrix which is basically the max of each row. We only took one of the maximum.

We then compared it for gamma = 0 and gamma = 0.9. We made +1 in the table if the action is the one that will be recommended.

In this case, the table will have values equal to +2 if it will be recommended by both the q_table with gamma = 0 and 0.9.

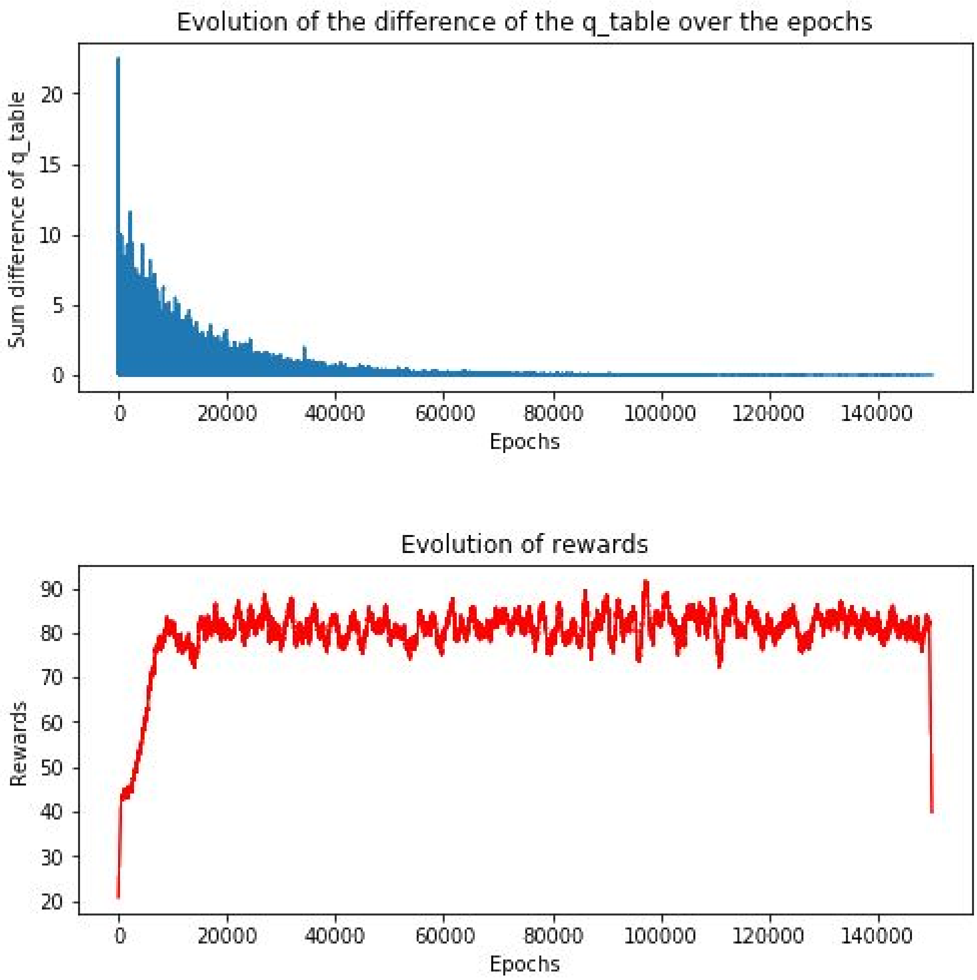


Comparison of final reward for gamma = 0 and gamma = 0.9

In this case, there is 25% of similitude.

3) *Find convergence criteria*

As there is some noise in the sum of rewards through the epochs, using a batch could have been a solution. Nevertheless, we’ve found another metric to see when the algorithm has converged. Here is the evolution of the total difference between two q_table over the epochs :



Evolution of the difference of the q_table over the epochs

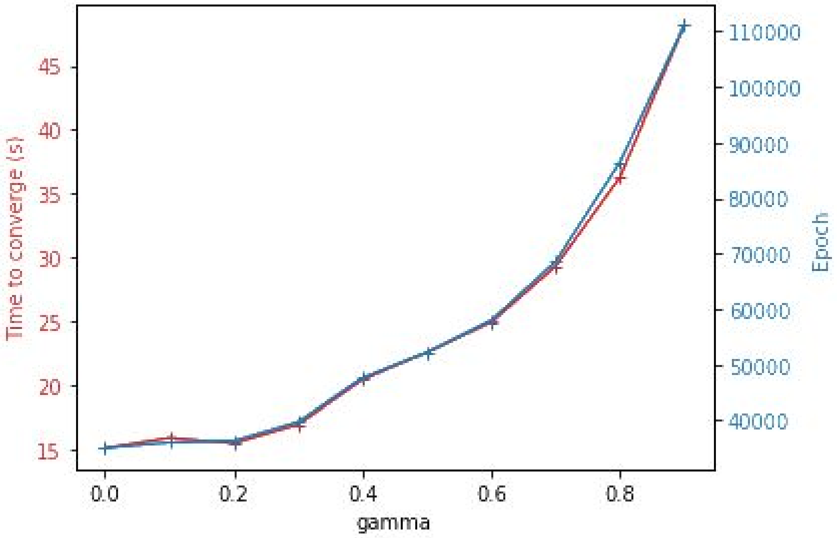
Evolution of rewards

Evolution of the sum difference of q_table over the epochs with rewards

Therefore the criteria to converge will be that the sum of difference of rewards over 100 epoch is less to 0.01.

4) Time of convergence for different rewards

We computed the time to converge and the number of epochs required for each value of gamma. Here is the result :



Time and epoch to converge for different values of gamma

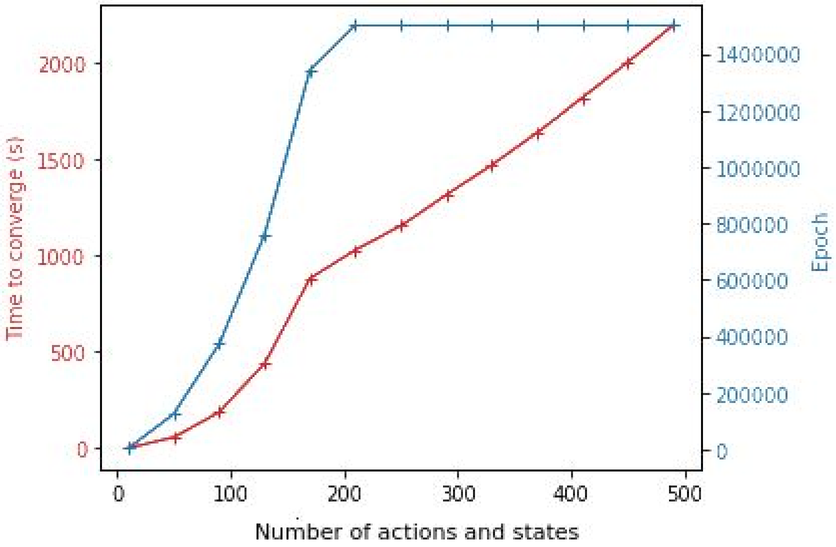
We observed that the final table obtained is the same after this epoch.

5) Check U matrix with the classic solution

We had issues to compare our results with Theodoros. Indeed, in his algorithm, he added some constraints in the recommendation with a value q. We didn't have these constraints because we decided to make constraints on the rewards.
In this case, to avoid the difference between our two algorithms, the q value should be equal to 1 and therefore it will require the content that maximises the U matrix.
We're still discussing to compare our results in a relevant way.

6) "Kill the computer"

We are currently running our algorithm to see how much we can compute.
Here is a recent results :



TIme and epoch to converge for different values of states/actions
(We set max_iter to 1 500 000)

We will try during the nights to run the algorithm to have better plots.

7) Deep q learning

We aim to look at this during the next week.