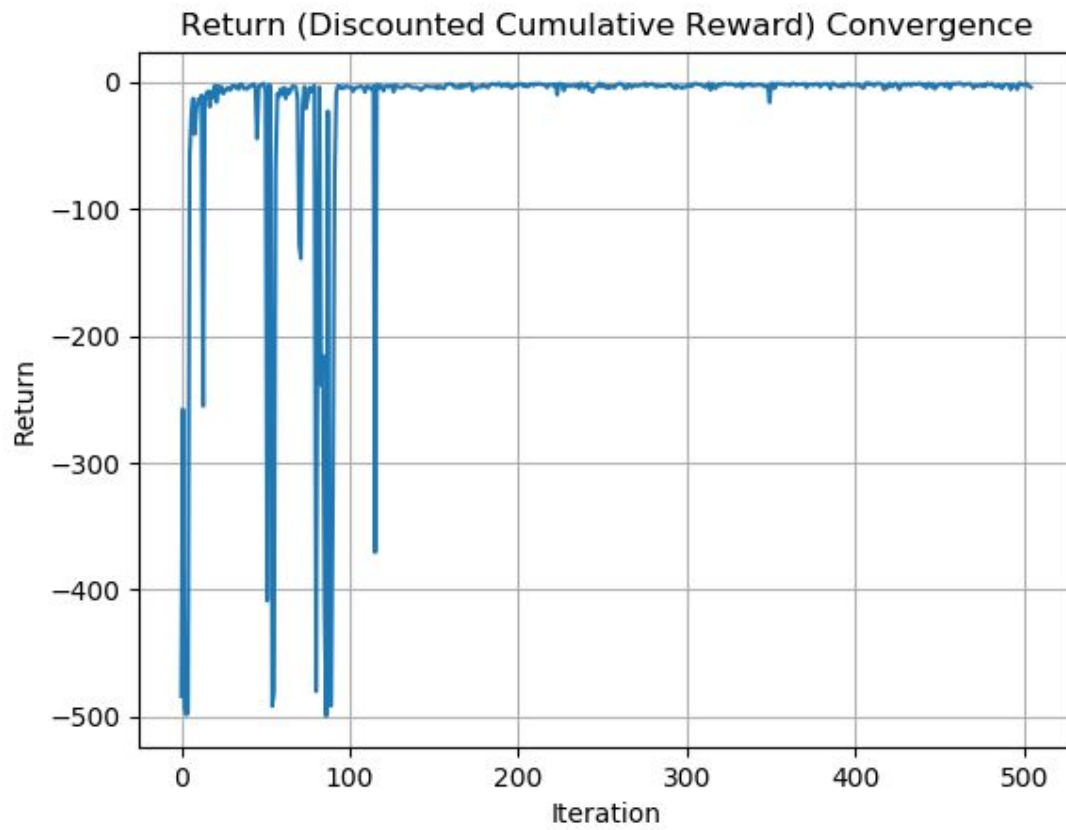


Relatório Lab 11 - CT213

Código: A implementação do código foi bem direta, os métodos `epsilon_greedy_action` e `learn` (dos dois algoritmos) foram o foco da parte não implementada. O primeiro método foi obtido utilizando a função `choice` da biblioteca `random` do `numpy`, onde passamos como argumento as escolhas possíveis e a probabilidade de cada uma delas. Ambos métodos `learn` foram baseados nas equações de Bellman (expectativa no Sarsa e otimalidade no Q-Learning).

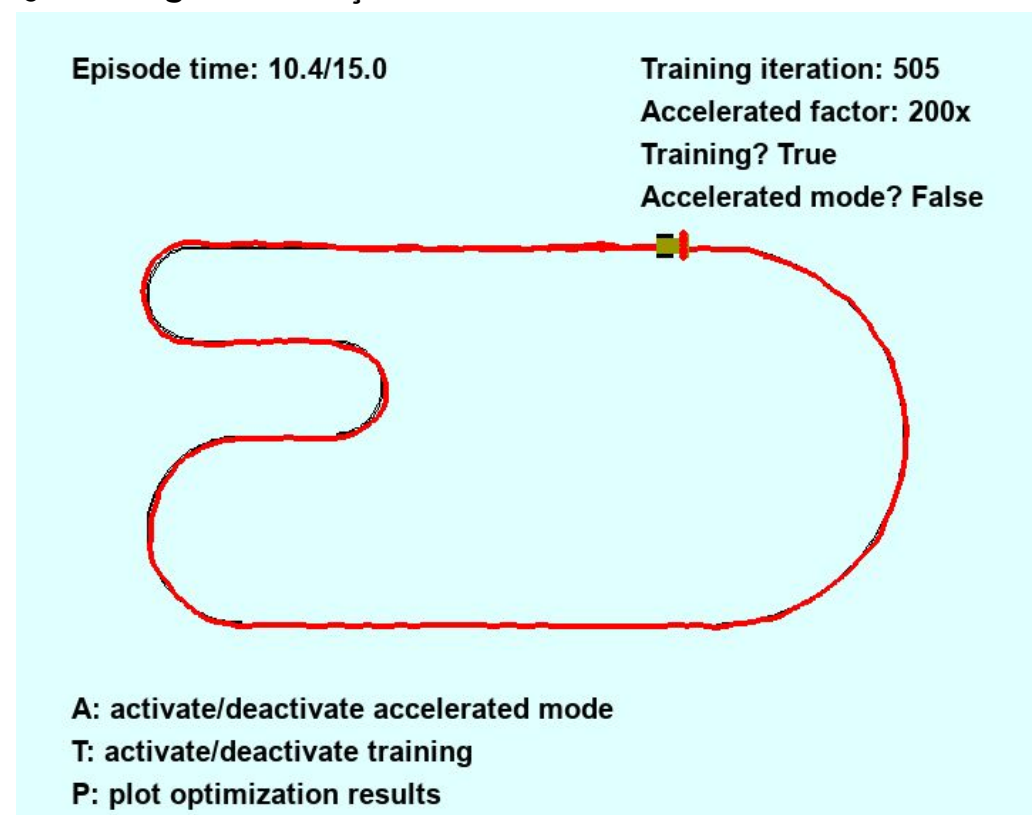
Resultados: Podemos observar que nas 100 primeiras interações ambos algoritmos exploram (o Sarsa explora fortemente apenas no começo). Ambos algoritmos convergem rapidamente para uma política capaz de concluir o trajeto (vide caminho percorrido com 20 interações)

Q-Learning: Return:



Q-Learning Q Table:

Q-Learning 505 interações:



Q-Learning 21 interações:

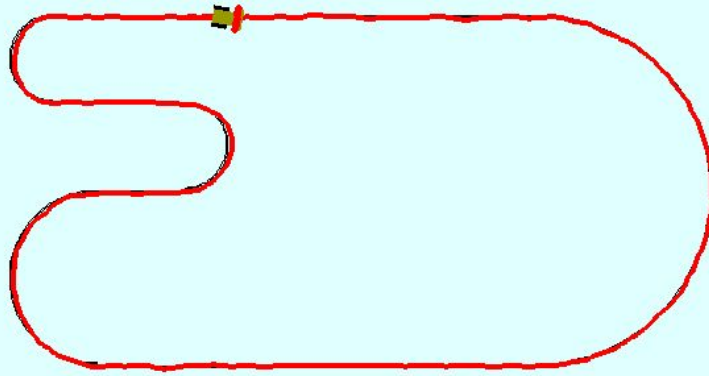
Episode time: 9.3/15.0

Training iteration: 21

Accelerated factor: 200x

Training? False

Accelerated mode? False

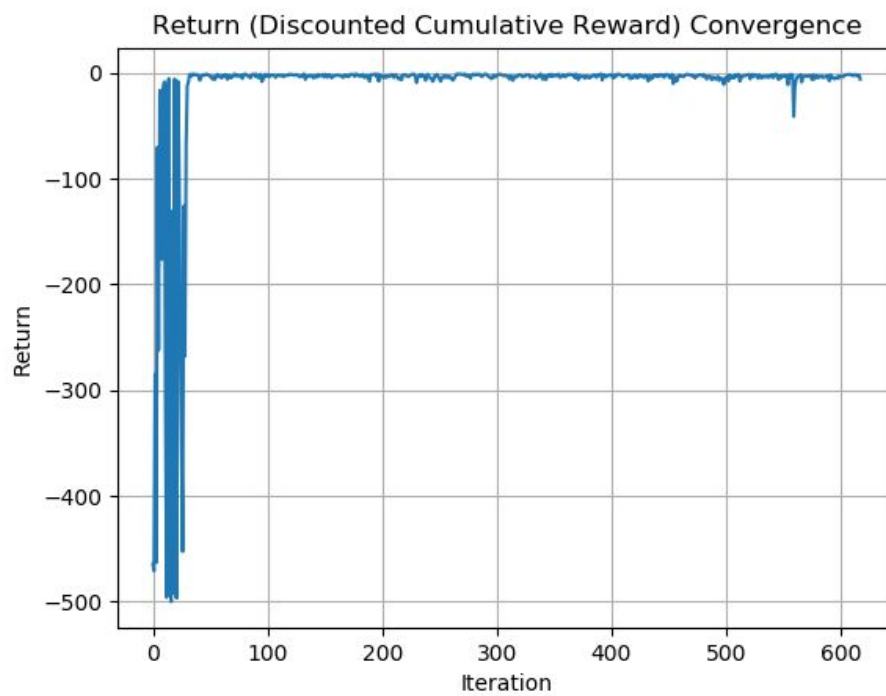


A: activate/deactivate accelerated mode

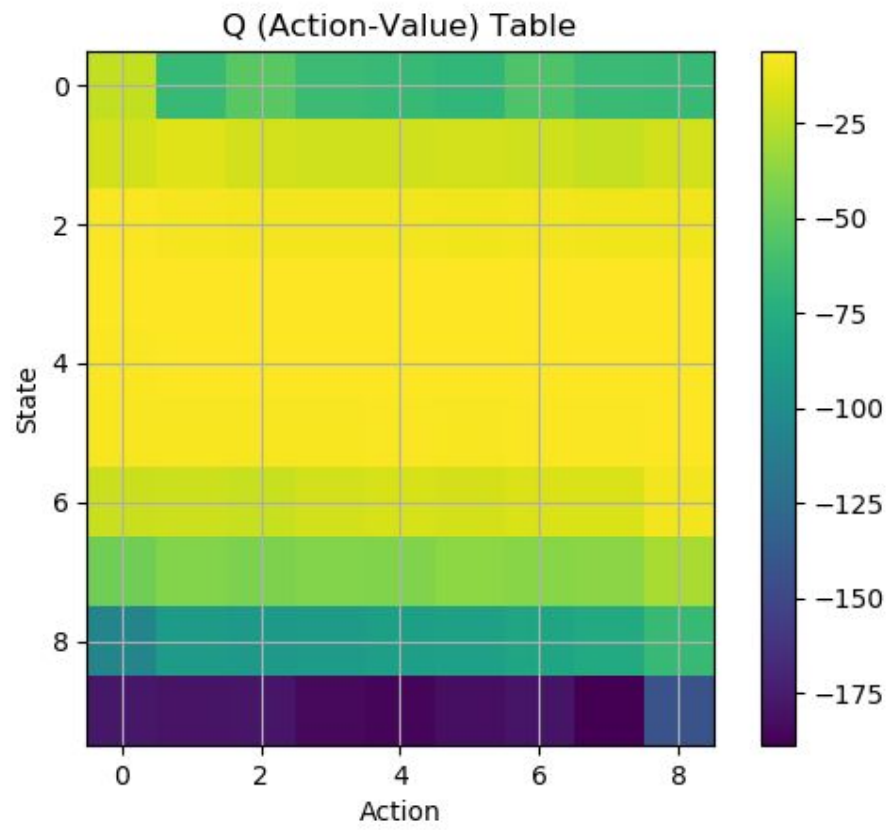
T: activate/deactivate training

P: plot optimization results

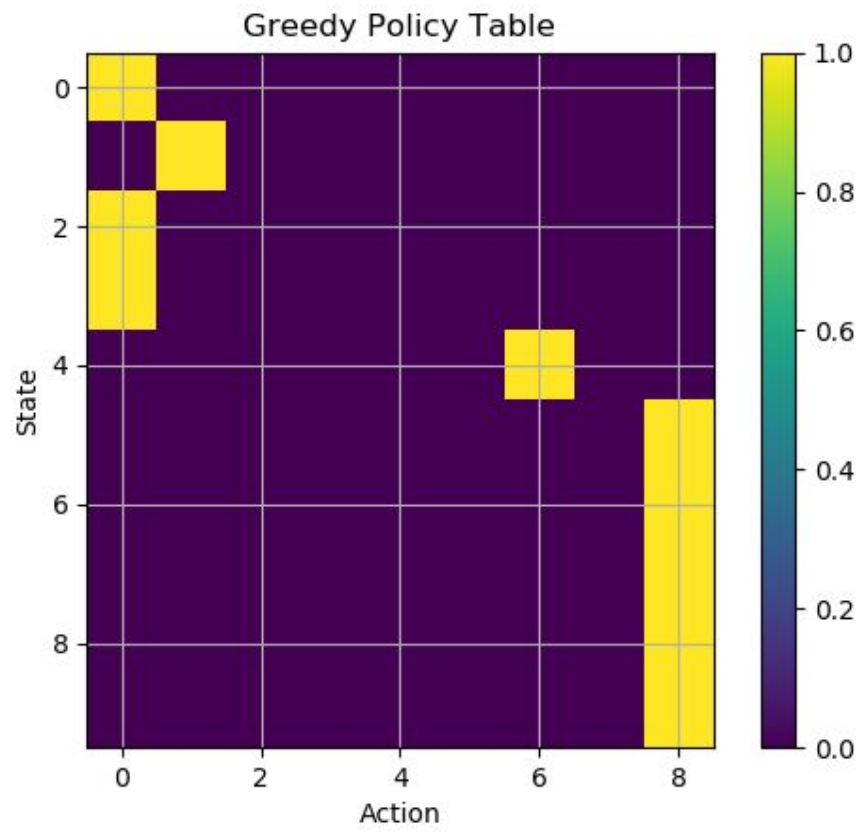
Sarsa - Return Convergence:



Sarsa - Q Table:



Sarsa - Greedy Policy Table



Sarsa 600 interações:

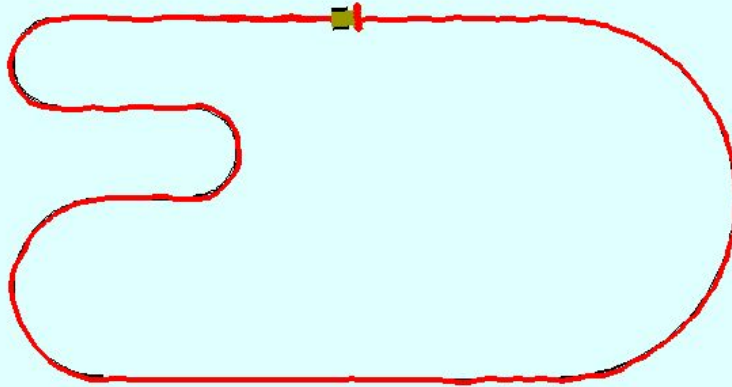
Episode time: 9.8/15.0

Training iteration: 617

Accelerated factor: 200x

Training? True

Accelerated mode? False



A: activate/deactivate accelerated mode

T: activate/deactivate training

P: plot optimization results

Sarsa 20 Interações:

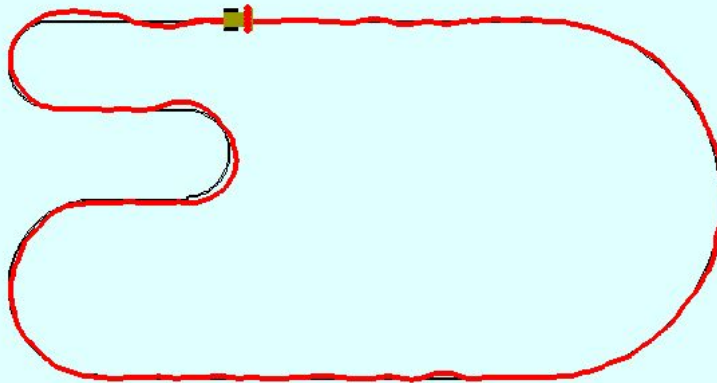
Episode time: 9.4/15.0

Training iteration: 20

Accelerated factor: 200x

Training? False

Accelerated mode? False



A: activate/deactivate accelerated mode

T: activate/deactivate training

P: plot optimization results