



Reinforcement Learning Project

BLACKJACK

Hugo DENIS--MARTIN - Brice MABILLE

SOMMAIRE

1. Introduction
2. Définition de l'environnement
3. Méthodologie
4. Résultats et performances
5. Conclusion et perspectives
6. Références



1 - INTRODUCTION

Présentation du sujet : Blackjack comme environnement d'apprentissage.

Objectifs : Étudier les approches en RL et comparer leurs performances.

Originalité : Adaptation et expérimentation des algorithmes sur ce cas particulier.



2 - DÉFINITION DE L'ENVIRONNEMENT

But du Jeu : Obtenir une main dont la valeur est plus proche de 21 que celle du croupier, sans dépasser 21.

Actions Disponibles :

- Hit : Tirer une carte supplémentaire.
- Stand : Conserver sa main actuelle et passer le tour

LA VALEUR DES CARTES

LUCKY BONUS

A ♠ 11 ou 1 POINTS AU CHOIX

10 POINTS

J ♦ Q ♦ K ♦

LES CARTES DE 2 À 10 COMPTENT POUR LEUR VALEUR FACIALE

2 ♠ 3 ♠ 4 ♠ 5 ♠ 6 ♠ 7 ♠ 8 ♠ 9 ♠ 10 ♠

LUCKY BONUS



3 - MÉTHODOLOGIE

Comparaison entre 4 agents différents:

- Random agent
- Simple agent
- Q-Learning
- Sarsa



RANDOM AGENT

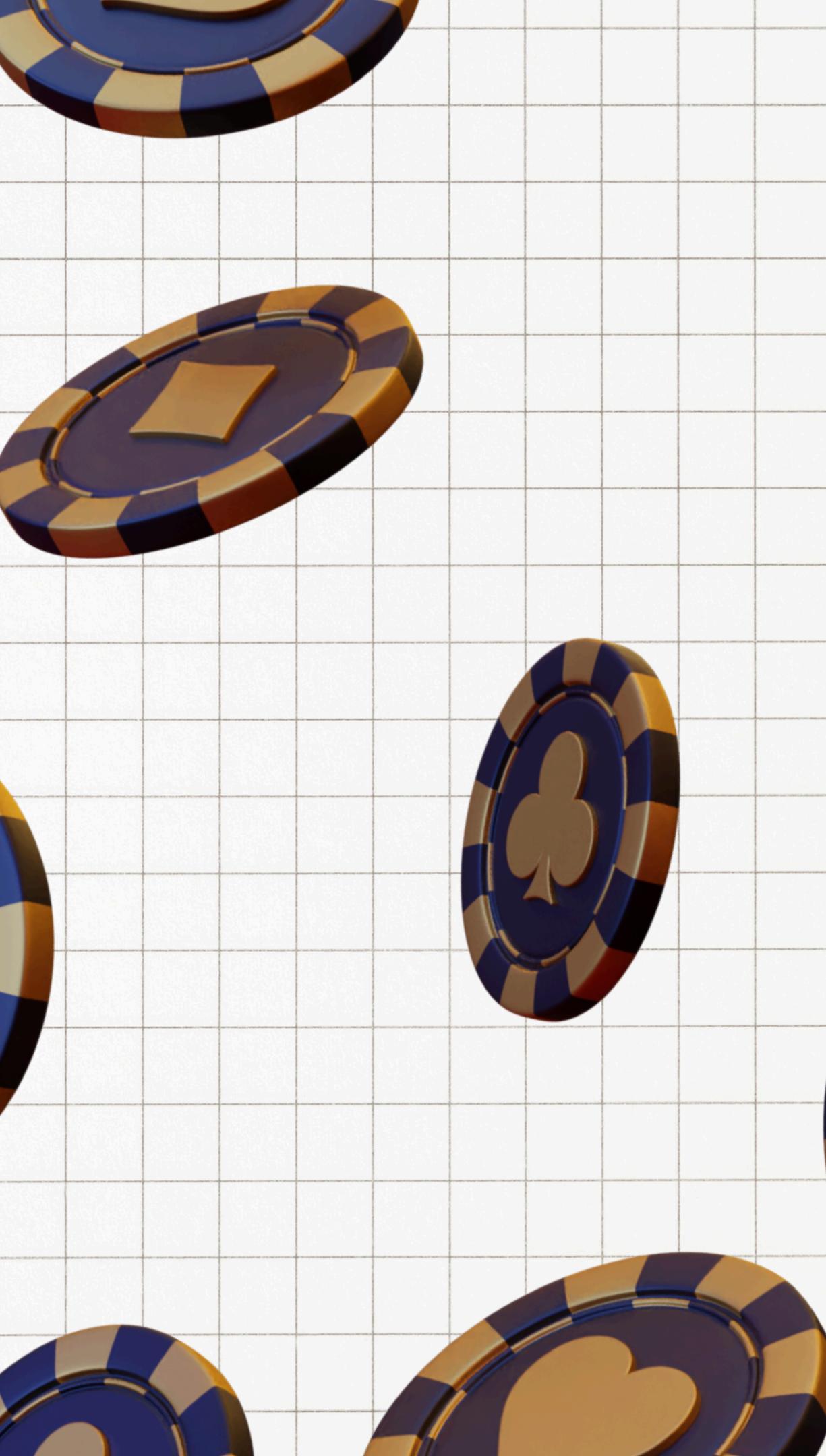
Stratégie utilisée :

- Action aléatoire : À chaque tour, l'agent décide aléatoirement entre :
 - Tirer une carte (hit).
 - S'arrêter (stand).

SIMPLE AGENT

Stratégie utilisée :

- Seuil fixe : L'agent utilise une stratégie déterministe basée sur la valeur de sa main.
- Si la valeur de la main est inférieure à 17, l'agent tire une carte (hit).
- Si la valeur est 17 ou plus, l'agent s'arrête (stand).

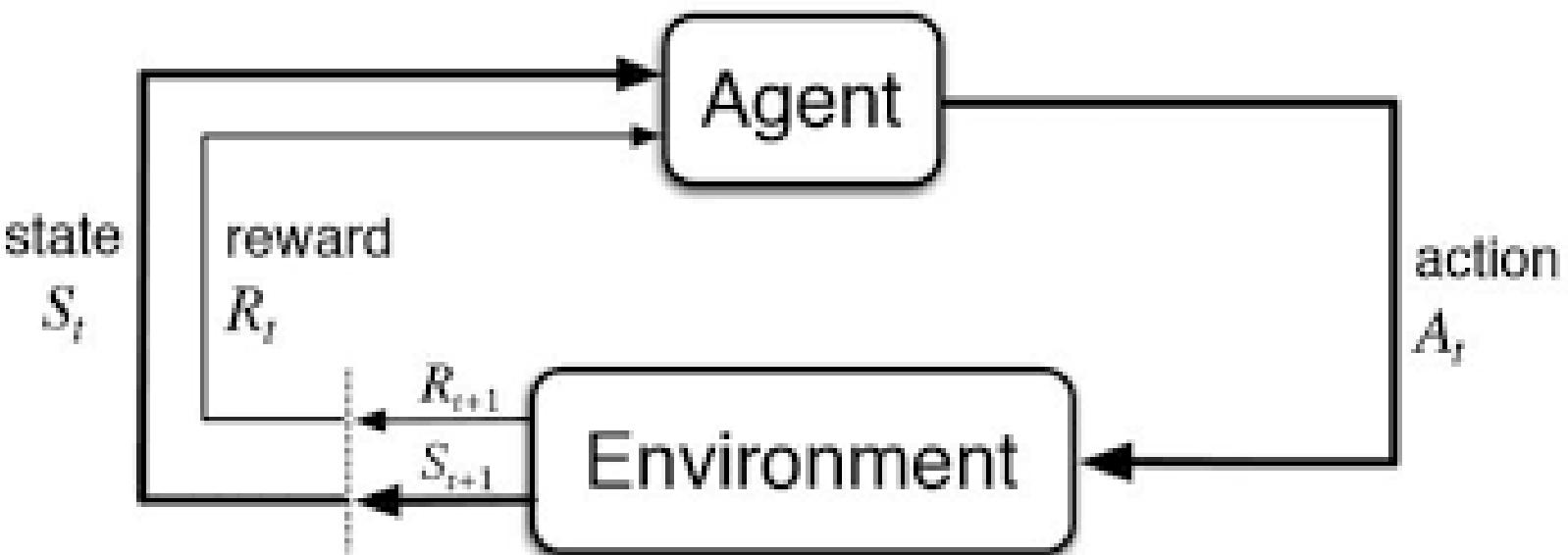


Q-LEARNING

Stratégie utilisée :

- L'agent apprend à jouer au Blackjack en mettant à jour une table Q qui associe des états (valeurs de mains, présence d'un As, carte visible du croupier) aux actions possibles (hit ou stand).
- Utilise une stratégie epsilon-greedy pour équilibrer l'exploration et l'exploitation.

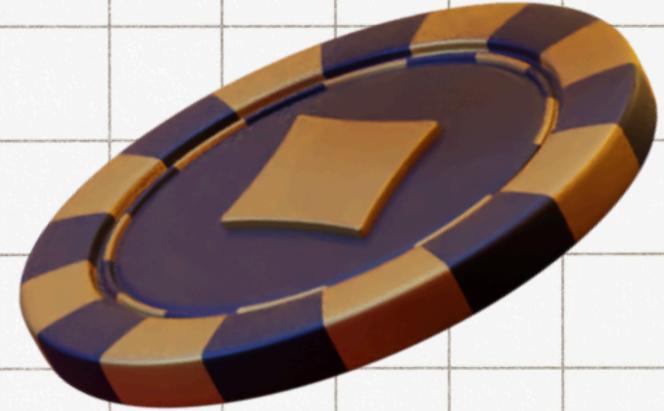
SARSA



Stratégie utilisée :

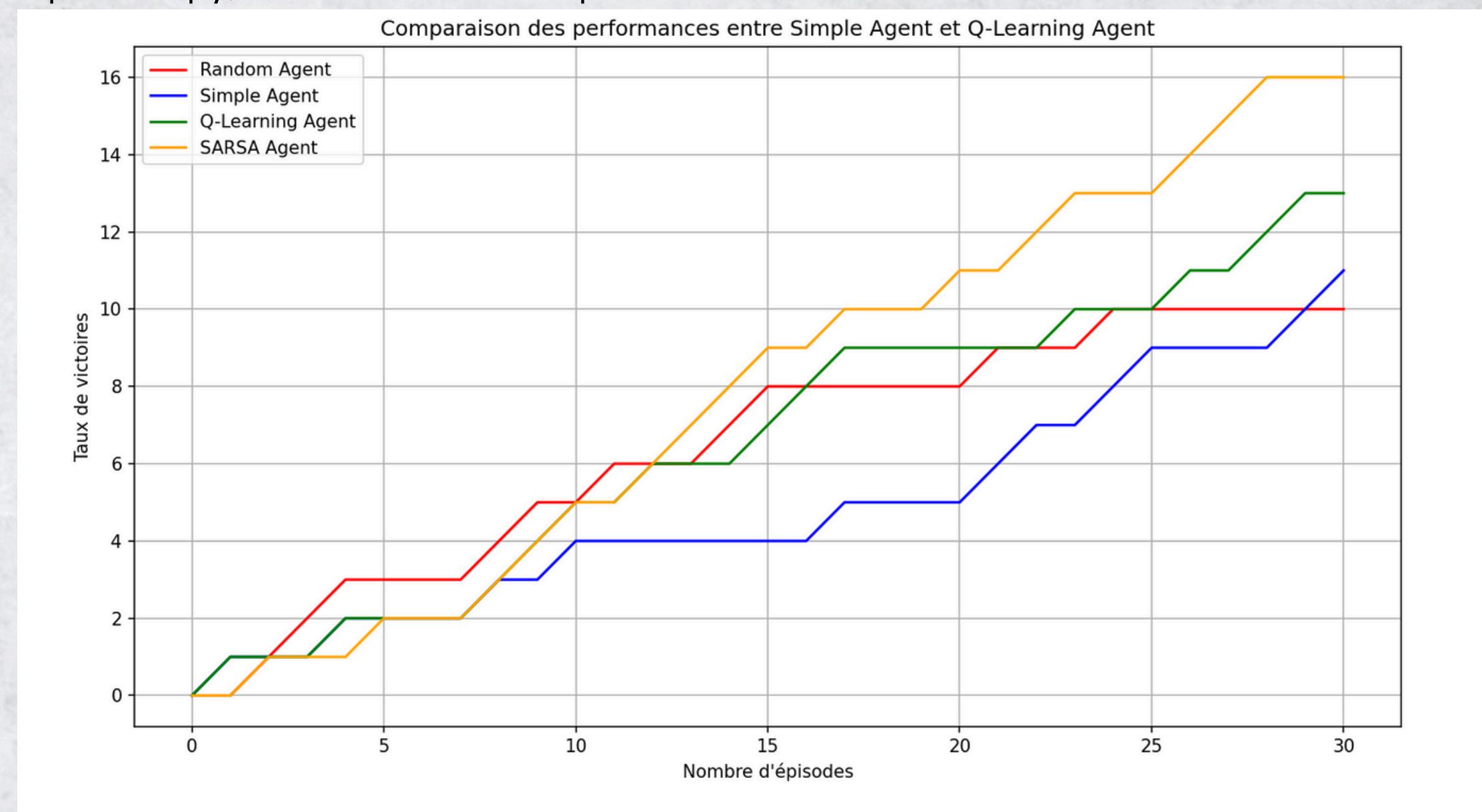
- L'agent utilise l'algorithme SARSA (State-Action-Reward-State-Action) pour apprendre à jouer au Blackjack.
- Contrairement au Q-Learning, SARSA prend en compte la prochaine action choisie pour mettre à jour les valeurs Q.

$$2^{280} = 1.94 * 10^{84} \text{ possible policies}$$



4 - RÉSULTATS ET PERFORMANCES

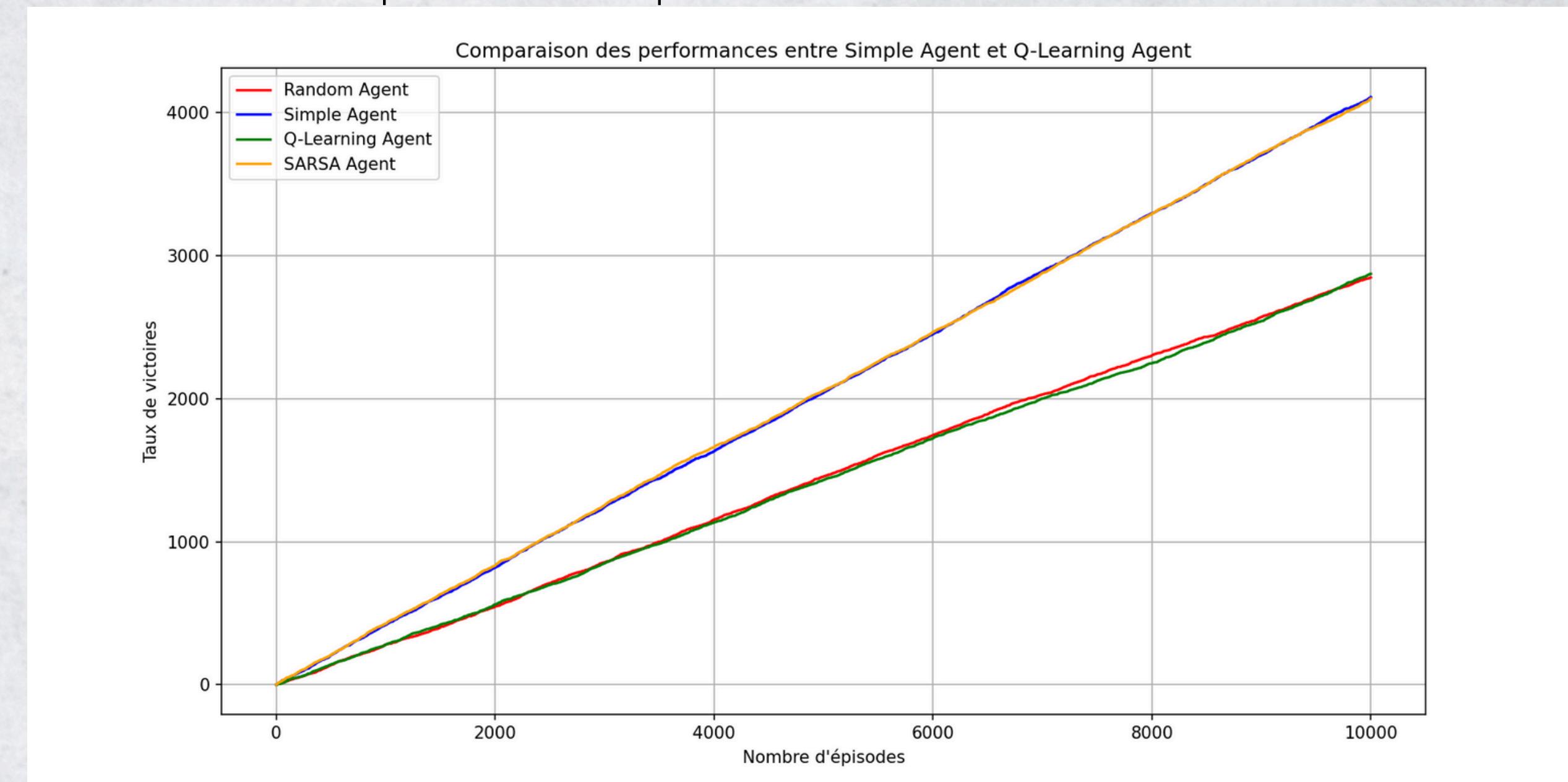
Avec le code réalisé comparison.py, nous obtenons les performances suivantes :



Avec un entraînement de 10 000 épisodes pour les agents Q_learning et SARSA

4 - RÉSULTATS ET PERFORMANCES

Puis comme nous n'étions pas satisfait des résultats précédents, nous avons optimisé au mieux les paramètres des agents et les récompenses liées à son environnement pour obtenir des performances meilleures :



Malgré ces modifications, la part de hasard pour tirer une carte est trop importante ; les agents implémentés apprennent une défaite selon une égalité ou une valeur en main > 21. Donc ils revoyent leurs exigences à la baisse.

6- CONCLUSION ET PERSPECTIVES

SUR CE JEU DE BLACKJACK, NOUS NOUS SOMMES RENDU COMPTE QUE LE REINFORCEMENT LEARNING APPLIQUÉ AU BLACKJACK PERMET D'ATTEINDRE LA SITUATION DE L'AGENT SIMPLE MAIS NE PERMET PAS DE LE DÉPASSER A CAUSE DE L'ASPECT FLUCTUANT DES RÉSULTATS DU JEU.

**POUR ALLER PLUS LOIN, NOUS POURRIONS DÉFINIR CE QUE
L'AGENT DOIT CHOISIR COMME ACTION SELON LA VALEUR
DES CARTES OBTENUES PAR LE DEALER**

7- RÉFÉRENCES

ENVIRONNEMENT BLACKJACK :

[HTTPS://GITHUB.COM/SHEETALBONGALE/BLACKJACK-PYTHON/BLOB/MASTER/BLACKJACK.PY](https://github.com/SheetalBongale/blackjack-python/blob/master/blackjack.py)

ARTICLE D'APPRENTISSAGE Q LEARNING DE LA MEILLEURE STRATÉGIE :

[HTTPS://LUCASPAUKER.COM/ARTICLES/REINFORCEMENT-LEARNING-APPLIED-TO-BLACKJACK/](https://lucaspauker.com/articles/reinforcement-learning-applied-to-blackjack/)

MERCI

