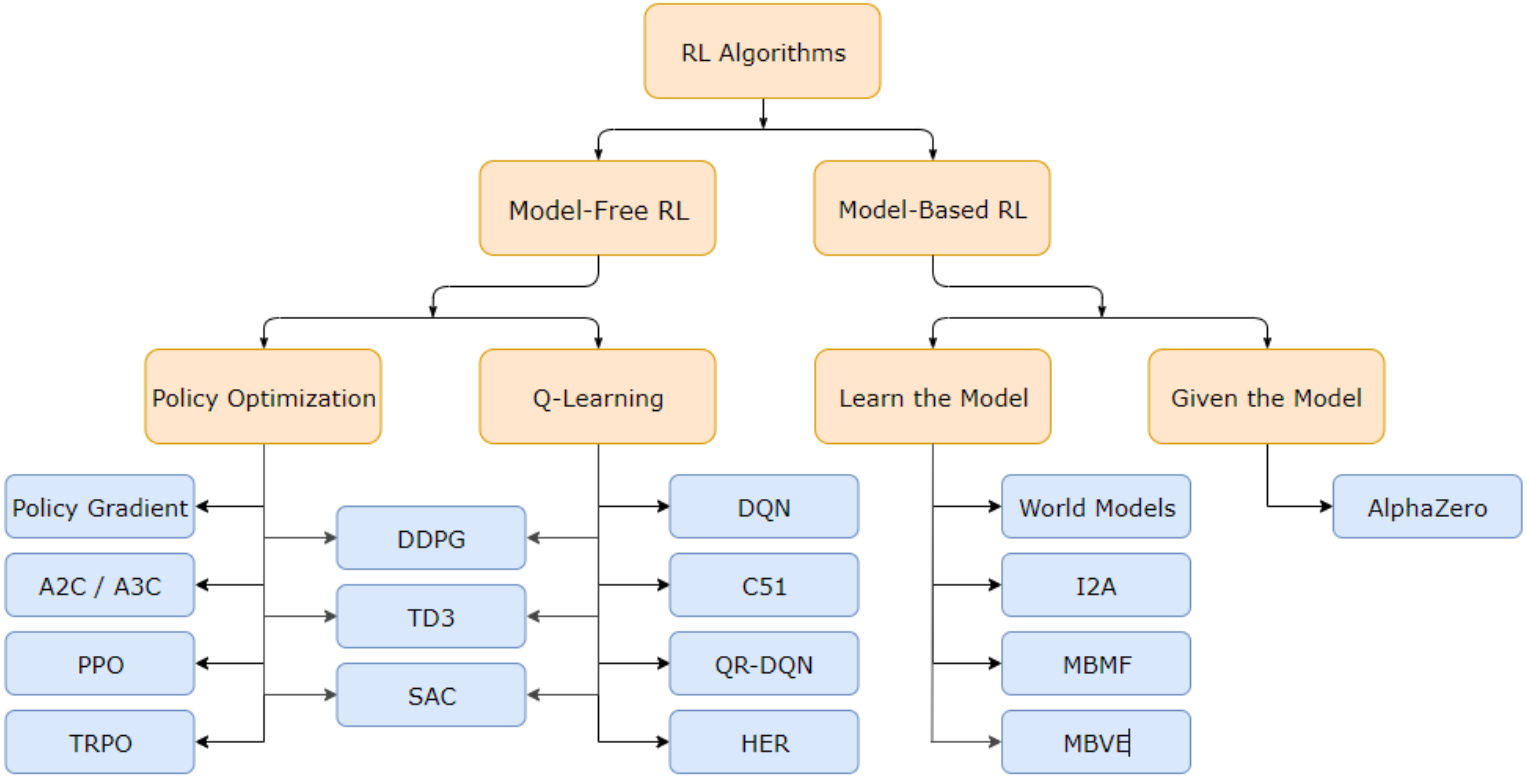
SCALIAN

MC_SARSA_ACTEUR CRITIQUE

RAPPEL

RL = RÉSOUDRE UN MDP



RAPPEL

Model Based

La **Transition** est connue pour chaque état ainsi que la **récompense**. (Peut nécessiter une phase d'observation pour la déduire)
Construit ensuite un modèle pour définir la **policy** à partir de ces éléments (Bellman Equation)

VS

Model Free

Calcule la policy (l'action) sans avoir besoin de connaître la transition et la récompense

RAPPEL

VS

Model Based

Décompose le modèle pour calculer la transition et la reward.

Model Free

Cherche directement à calculer la meilleure policy.

Question à se poser :
Après un apprentissage,
Est-ce que l'agent fait des prédictions sur le prochain état et sur la récompense engendrée pour prédire sa prochaine action ?
Si oui, c'est un "**model based**" sinon "**model free**".

RAPPEL

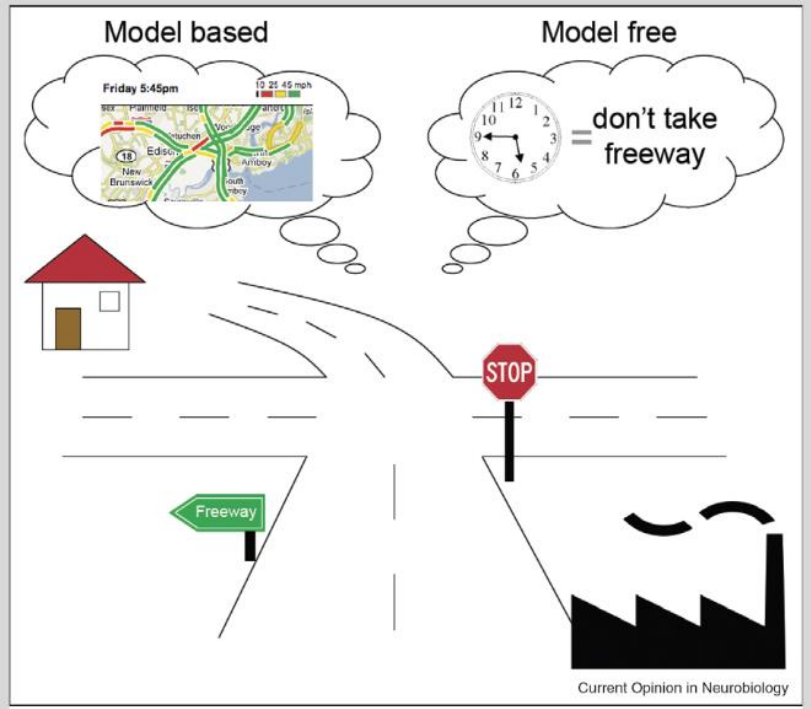


Figure 1: Two ways to choose which route to take when traveling home from work on friday evening.

OPTIMISATION UNE CAMPAGNE SUR LES RÉSEAUX SOCIAUX

EXEMPLE

Model Based

?

VS

Model Free

?

OPTIMISATION UNE CAMPAGNE SUR LES RÉSEAUX SOCIAUX

EXEMPLE

Model Based

VS

Model Free

Des études poussées en psychologies permettraient de bien comprendre l'environnement. On utiliserait ces connaissances pour poster des messages.

On part d'aucune hypothèse, on essaye et on regarde la récompense. On affine au fur et à mesure?

MODEL FREE ET ÉCHANTILLONAGE

On part d'échantillons pour déduire une théorie générale (approximation) :

↓

<u>Faits particuliers :</u> la majorité des parties que j'ai gagné au monopoly était lorsque j'achetais les gares

↓ **Déduction**

<u>Théorie :</u> Il faut acheter les gares pour gagner au monopoly

SARSA

STATE-ACTION-REWARD-STATE-ACTION

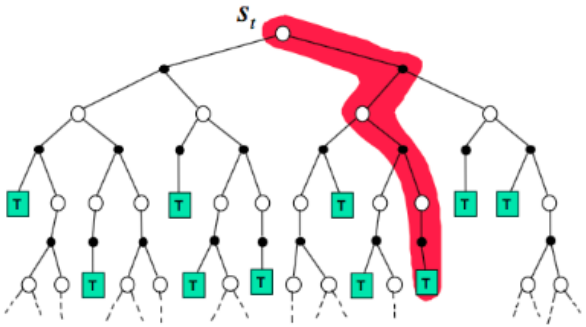
TEMPORAL-DIFFERENCE LEARNING

SCALIAN

Model-free | **Model-based**

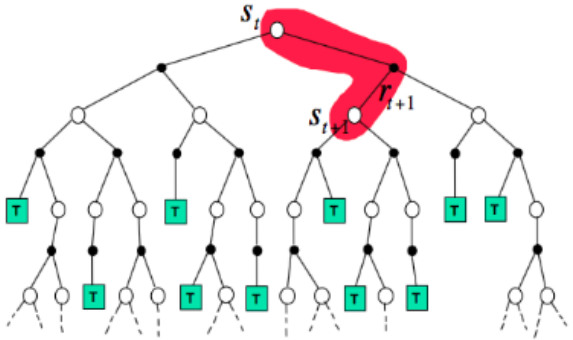
Monte-Carlo

$V(S_t) \leftarrow V(S_t) + \alpha\,(G_t - V(S_t))$



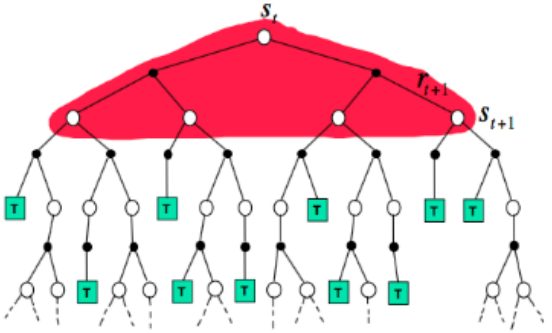
Temporal-Difference

$V(S_t) \leftarrow V(S_t) + \alpha\,(R_{t+1} + \gamma V(S_{t+1}) - V(S_t))$



Dynamic Programming

$V(S_t) \leftarrow \mathbb{E}_\pi\,[R_{t+1} + \gamma V(S_{t+1})]$



runs, expérimentations **complètes**

runs, expérimentations **incomplètes**

OFF-POLICY VS ON POLICY METHOD

SCALIAN

Target policy :
C'est la politique que l'on souhaite apprendre.

Behavior policy:
C'est la politique utilisée par l'agent pour sélectionner les actions à faire dans l'environnement

SARSA

Dans l'apprentissage **on-policy**, la fonction Q(s,a) est apprise à partir des actions que nous avons prises en utilisant notre politique actuelle $\pi(a|s)$.

apprendre à quel point il est bon de faire quelque chose en le faisant
Target policy == behavior policy

Q-Learning

Dans l'apprentissage **off-policy**, la fonction Q(s,a) est apprise en prenant différentes actions (par exemple, des actions aléatoires).

apprendre à quel point il est bon de faire quelque chose en faisant autre chose.
Target policy != behavior policy

SARSA VS Q-LEARNING

SCALIAN

Questions	SARSA	Q-Learning
Model free / update pendant l'épisode	oui	oui
Policy	On policy	Off policy
Solution trouvée	Proche de l'optimale	optimale
Variance	oui	Forte variance
Comportement	Prudent	+ explorateur
Durée d'entraînement	longue	Très longue

SARSA VS Q-LEARNING

SCALIAN

Prend en compte la meilleure action possible (même si elle ne sera pas prise)

Q Learning:

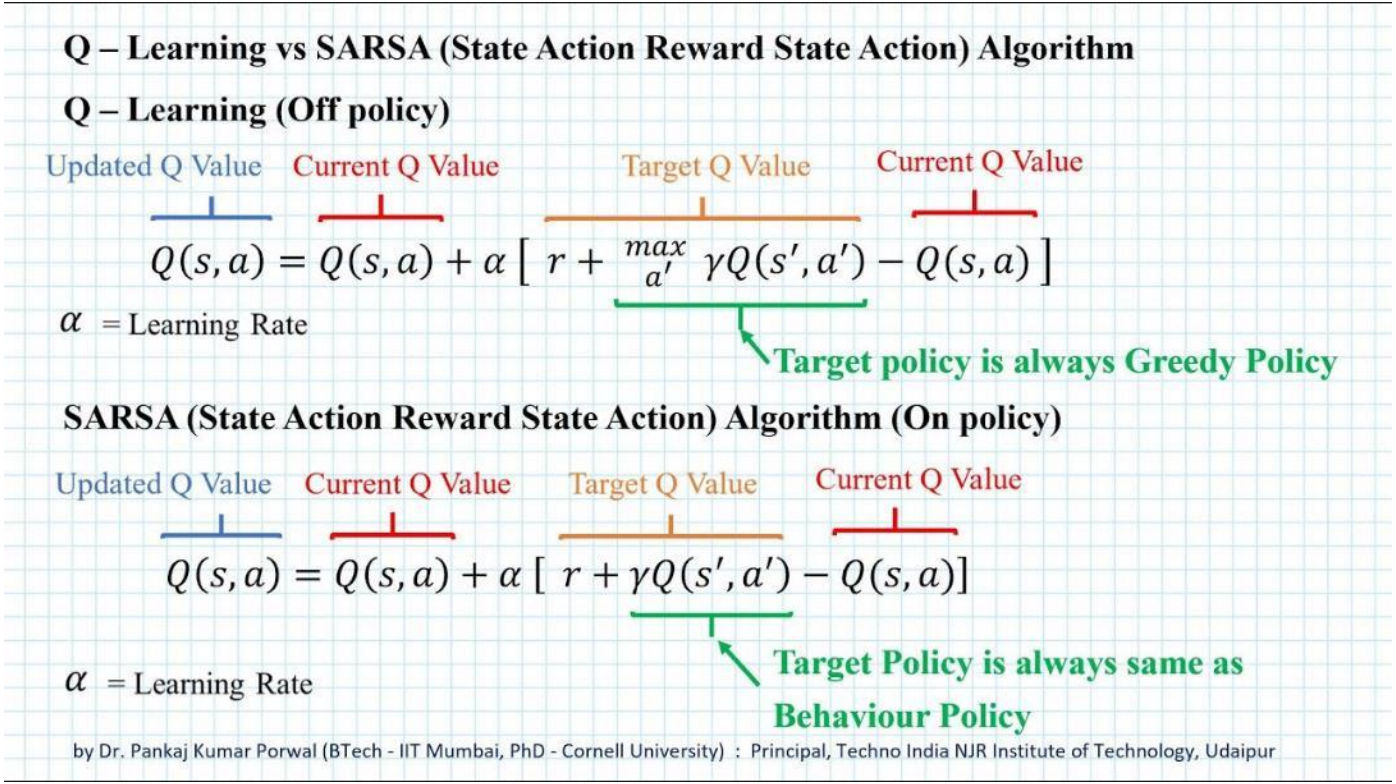
$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t)]$$

SARSA:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]$$

Prend en compte l'action "réelle" prise

SARSA VS Q-LEARNING



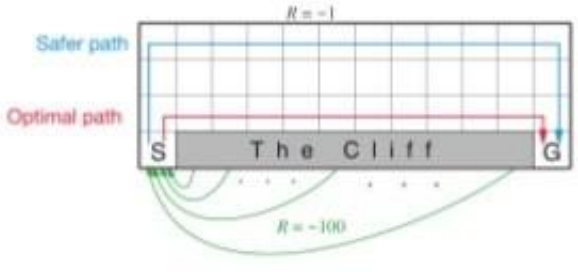
source

SARSA VS Q-LEARNING

SCALIAN

Cliff Walking Example: Sarsa vs. Q-learning

- Q-learning learns optimal policy
- Sarsa learns safe policy
- Q-learning has worse online performance
- Both reach optimal policy with ε-decay



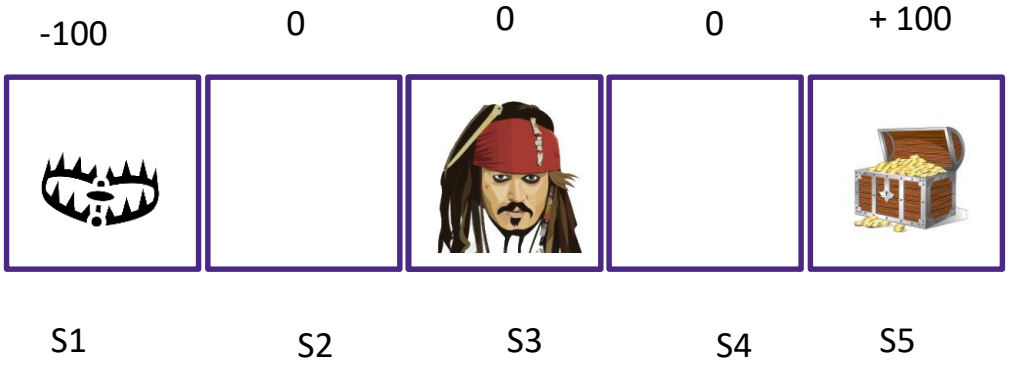
Sutton & Barto book

MONTE CARLO

LES MÉTHODES DE MONTE CARLO

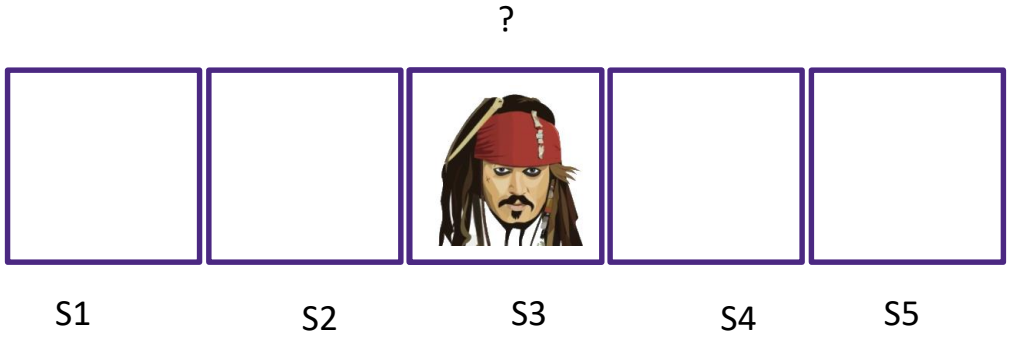
Monte Carlo = Casino = hasard

MDP



S1 S2 S3 S4 S5

Comment déduire la value pour l'ensemble des states S ?
Avez-vous une idée ?



Calculons la valeur pour le State 3

ALGORITHME MC

PRÉDICTION => POUR APPROXIMER V(S)

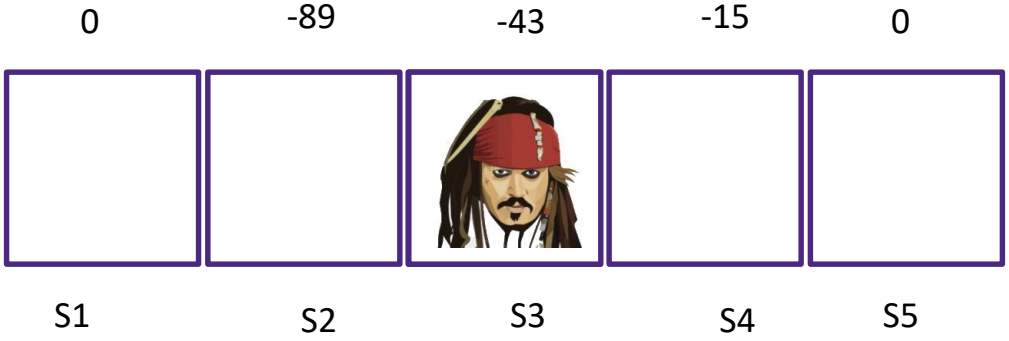
Algorithm 1: Monte-Carlo Prédiction

```
Input: π
Init V(s) = 0 pour chaque s ∈ S

repeat
    collecte d'expérience avec π (un épisode)
    foreach s do
        V(s) ← moyenne(returns(s))
    end
until V ne change presque plus

return V ≈ v_π
```

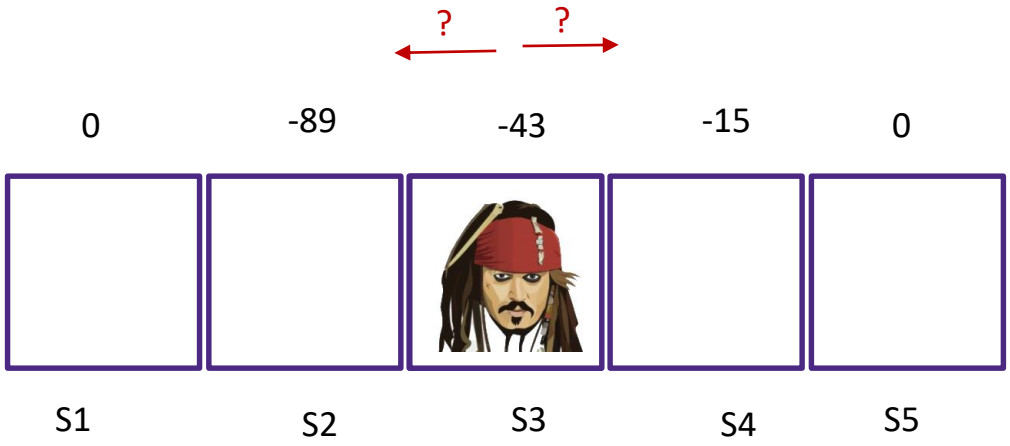
2 contraintes :

Il faut aller au bout d'un "run" pour connaître la finalité du jeu, les environnements doivent pouvoir se terminer (perdu / gagné)

Il faut pouvoir lancer suffisamment de runs pour que ça fonctionne bien)



Nous avons approximé ***V(s)*** pouvant nous faire de même avec ***Q(S,A) ?***



Q(S,A) = motenne (récompense (S,A))

ALGORITHME MC

CONTRÔLE => POUR APPROXIMER Q(S,A)

Algorithm 1: Monte-Carlo Contrôle

Init $Q(s,a) = 0$ pour chaque $s \in S$ et $a \in A$, $\pi(a|s)$ arbitrairement

repeat

  collecte d'expérience avec π (un épisode)

  foreach s, a **do**

    $Q(s,a) \leftarrow$ moyenne(returns(s,a))

  end

  foreach s **do**

    $\pi(s) \leftarrow \text{argmax}_a\, Q(s,a)$

  end

until Q ne change presque plus

return $Q \approx q_*$, $\pi \approx \pi_*$

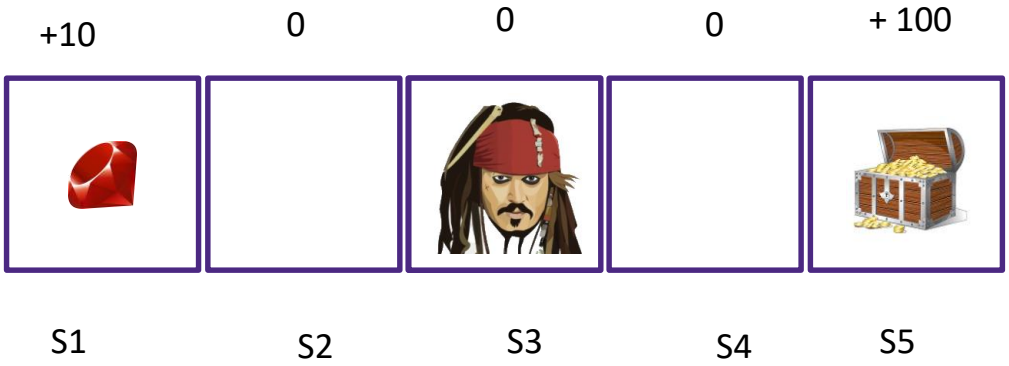
2 contraintes (identiques) :

Il faut aller au bout d'un "run" pour connaître la finalité du jeu, les environnements doivent pouvoir se terminer (perdu / gagné)

Il faut pouvoir lancer suffisamment de runs pour que ça fonctionne bien)

NOUVEAU MDP

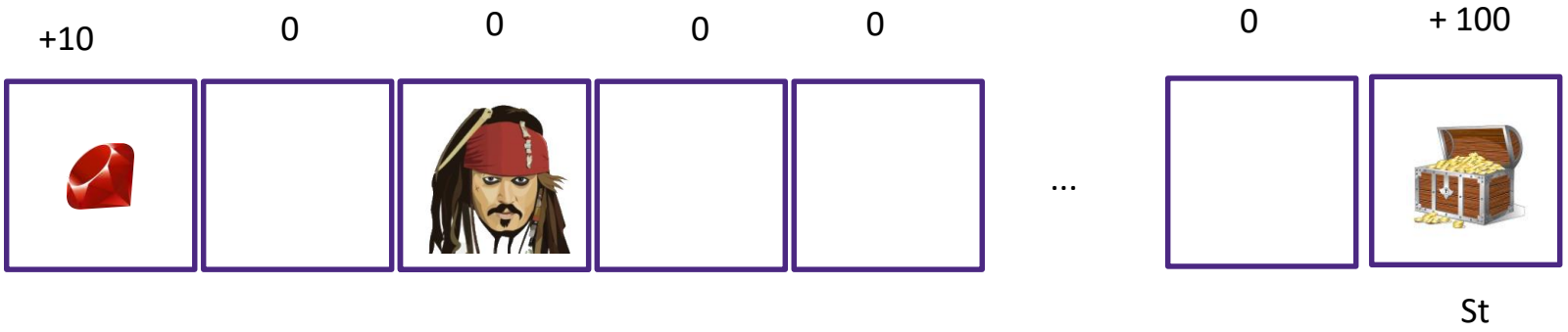
Comment s'assurer d'être dans la récompense optimale ?



NOUVEAU MDP

SCALIAN

Comment s'assurer d'être dans la récompense optimale ?

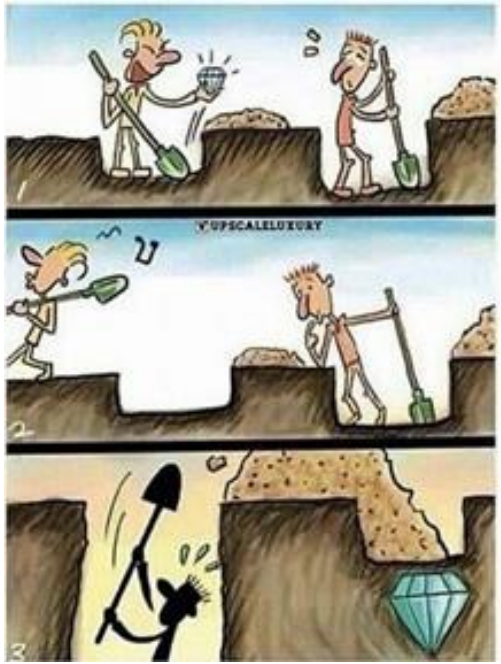


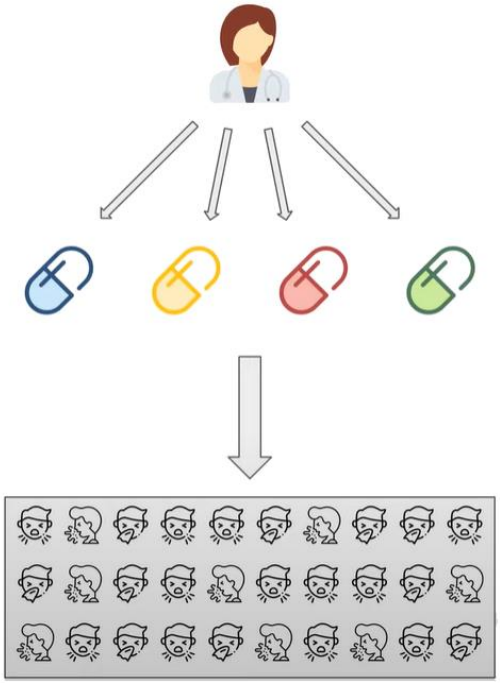
Quelle problématique cela soulève ?

EXPLORATION VS EXPLOITATION

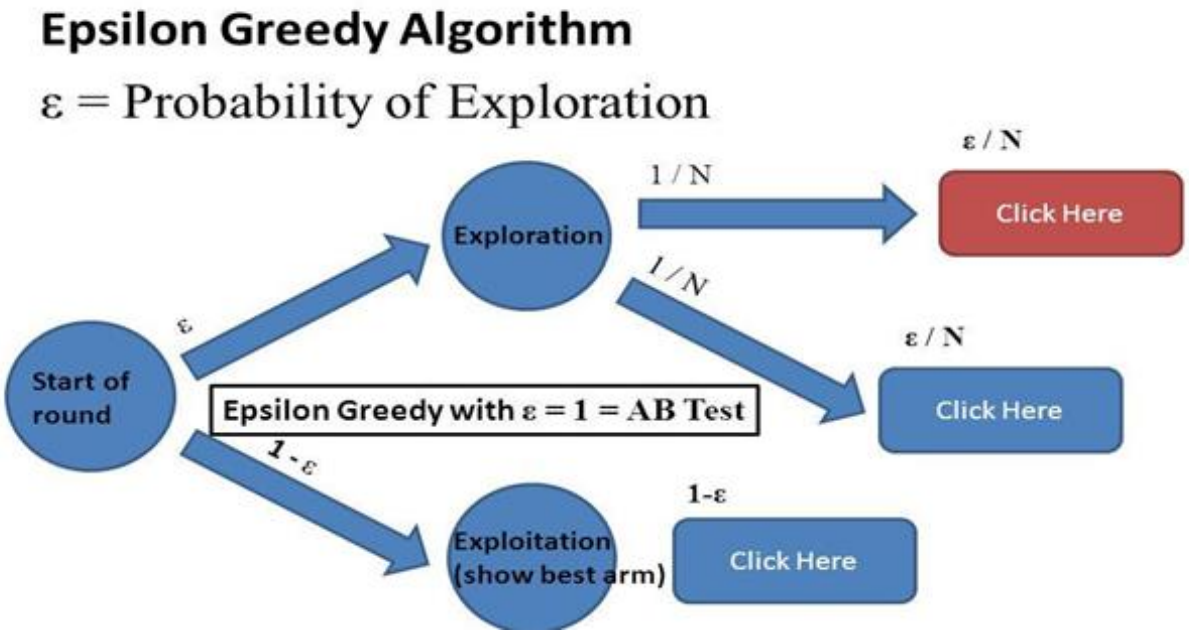
SCALIAN

Comment s'assurer d'être dans la récompense optimale ?





ALGORITHME E(EPSILON)-**GREEDY**



On détermine une valeur pour Epsilon, valeur utilisée par la suite pour détermine si dans le prochain "run" on va explorer ou exploiter. Epsilon est initialiasié avec une valeur élevée au début puis va diminuer au fur et à mesure.

CONCLUSION

SCALIAN

- **L'algorithme de Monte Carlo appartient à la famille des algorithmes "model free"**
- **Il permet d'axproximer** v_π ou q_π
- **Pour cela on intéragit avec l'environnement (exploration) et on garde en mémoire les récompenses obtenues lorsqu'on est passé par une étape S et par une action + une étape, pour ensuite utiliser la moyenne des récompenses :**

$$V(s) \leftarrow \text{moyenne}(\text{returns}(s))$$

$$Q(s,a) \leftarrow \text{moyenne}(\text{returns}(s,a))$$

- **Nécessite que les runs, expérimentations soient complètes**

ACTEUR CRITIQUE

POLICY GRADIENT

VS VALUE FUNCTION APPROXIMATION

SCALIAN

Value-Based:
Les méthodes vues auparavant comme **Q-Learning** et **SARSA** essayent d'apprendre la fonction de value/action.

Policy-Based:
Cherche à apprendre la police optimale. Cette approche ne nécessite pas de connaître directement la "Value" d'un State ou d'un State + action (Q(s,a)) : π:s-->a

Dans les algorithmes de Policy Graident on est face à un problème de maximisation ou l'on va chercher à **maximiser la fonction Reward** (à l'opposé de la fonciton de coût utilisée en Deep Learning, à l'aide de l'ascension de gradient).

VOCABULAIRE

2 ÉLÉMENTS DIFFÉRENTS : ACTOR-CRITIC

Acteur : l'enfant qui explore le monde

Critique : la mère qui évalue l'enfant (bien / pas bien)

L'acteur va évoluer et explorer. Il va apprendre à réaliser des actions plus intéressantes au fur et à mesure du temps, en fonction des retours de sa mère.
Sa mère va être "surprise" par son comportement et va au fur à mesure améliorer son évaluation sur la situation.

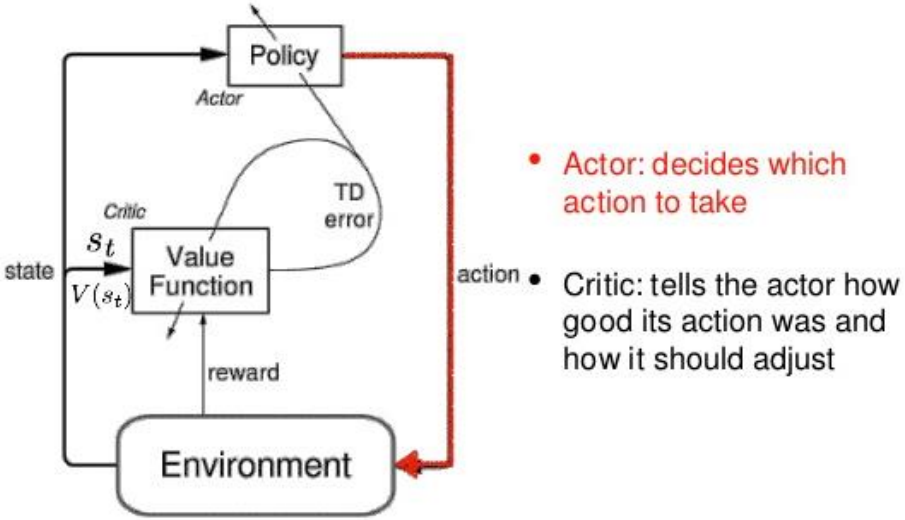
Dans quelques années (fin de l'entrainement) l'enfant aura de "bonnes" actions et la mère saura évaluer parfaitement ses actions.

VOCABULAIRE

2 ÉLÉMENTS DIFFÉRENTS : ACTOR-CRITIC

Acteur : met à jour les paramètres de la politique θ, dans la direction suggérée par le critique, π(a|s;θ). Cela peut être une fonctionne d'approximation qui prédit l'action à faire.

Critique : met à jour les paramètres de la fonction de valeur w et, selon l'algorithme, il peut s'agir d'une valeur d'action Q(a|s;w) ou d'une valeur d'état V(s;w). Cela peut être une fonctionne d'approximation qui évalue l'action faite.



- Actor: decides which action to take
- Critic: tells the actor how good its action was and how it should adjust

Ce modèle combine les approches "Value based" et "policy based". Séparer la tâche en deux tâches facilite l'apprentissage

ENTRAINEMENT

SCALIAN

Les deux réseaux s'entrainent séparamant en optimisant les paramètres à l'aide de l'ascension de gradient. Les mises à jour sont effectuées lors de chaque étape (TD, donc pas besoin d'attendre la fin du run))

EXERCICES