Faculté des sciences économiques et de gestion
Université de Strasbourg

PROJET REINFORCEMENT LEARNING

RAPPORT FROZEN LAKE

2024 - 2025

REALISE PAR :

RAZAFINDRAKOTO Tsiba
SOW Oumou

Sommaire

Liste des figures

Liste des tableaux

INTRODUCTION

L'apprentissage par renforcement (RL) est un type d'apprentissage automatique axé sur la prise de décision par des agents autonomes. Un agent autonome est tout système capable de prendre des décisions et d'agir en réponse à son environnement indépendamment de l'instruction directe d'un utilisateur humain. Autrement dit, le « reinforcement learning » désigne l'ensemble des méthodes qui permettent à un agent d'apprendre à choisir quelle action prendre de manière autonome. Plongé dans un environnement donné, il apprend en recevant des récompenses ou des pénalités en fonction de ses actions. Au travers de son expérience, l'agent cherche à trouver la stratégie décisionnelle optimale qui puisse lui permettre de maximiser les récompenses accumulées au cours du temps.

Ainsi, c'est dans cet ordre d'idée que nous avons décidé d'entraîner un agent qui doit naviguer sur un lac gelé où il commence à une case initiale et doit atteindre une case d'objectif sans tomber dans des trous ou pièges. Ce jeu s'appelle **Frozen Lake**. Cependant, étant un jeu relativement simple, nous avons décidé d'enrichir l'environnement du jeu en incluant plus d'obstacles pour le rendre plus complexe et interactif.

L'objectif de ce projet est d'entraîner un agent à résoudre le dilemme du lac gelé en utilisant le « reinforcement learning ». Ainsi, le but est de développer une stratégie (ou politique) efficace pour que l'agent atteigne la case de sortie en minimisant le risque de tomber dans les trous. Concrètement, cela signifie qu'il doit apprendre à choisir le meilleur mouvement en fonction de sa position actuelle, en maximisant les chances de réussite c'est-à-dire une politique optimale qui va guider l'agent vers la sortie tout en minimisant les risques et en maximisant ses récompenses cumulées.

I. Q-Learning et Apprentissage par Renforcement

Le Q-learning est l'une des approches d'apprentissage par renforcement représentatif les plus appliquées et l'une des stratégies hors politique. Il ne nécessite aucun modèle initial de l'environnement. La lettre « Q » désigne la fonction qui mesure la qualité d'une action exécutée dans un état donné du système. C'est un algorithme d'apprentissage par renforcement sans modèle qui cherche à trouver la meilleure action à entreprendre compte tenu de l'état actuel. Il s'agit d'apprendre une fonction qui nous donnera la meilleure action pour maximiser la récompense future totale. Cet algorithme Q-learning fonctionne par **essais et erreurs**. En particulier, l'agent vérifie son environnement, choisissant parfois des chemins au hasard pour découvrir de nouvelles façons d'y aller. Après avoir fait un mouvement, l'agent voit ce qui se passe et quel type de récompense il reçoit. Un bon mouvement, comme se rapprocher du but, rapporte une récompense positive. Un mouvement pas bon, comme tomber dans un trou, signifie une récompense négative. Sur la base de ce qu'il apprend, l'agent met à jour son guide, en augmentant les scores pour les bons mouvements et en les réduisant pour les mauvais. Au fur et à mesure que l'agent continue d'explorer et de mettre à jour son guide, il devient plus précis dans la sélection des meilleurs mouvements. L'objectif du Q-learning est de permettre à l'agent de déterminer quelle action choisir dans chaque situation pour maximiser ses récompenses à long terme. Cette méthode repose sur la mise à jour des valeurs dites Q (valeurs de qualité), qui sont des estimations de la qualité d'une action dans un certain état. Ces valeurs permettent à l'agent de construire une politique pour choisir les actions les plus avantageuses.

Dans l'environnement de Frozen Lake, on trouve un total de 16 cases, ce qui veut dire que notre agent peut se trouver dans 16 positions distinctes, appelées **états**. Pour chaque état, l'agent a 4 choix possibles: aller à gauche, en bas, à droite ou en haut. Apprendre à naviguer dans Frozen Lake revient donc à déterminer quelle action prendre dans chaque état. Afin de savoir quelle action est optimale dans un état donné, nous allons associer une valeur de qualité à chaque action. Avec nos 16 états et 4 actions possibles, cela signifie que nous aurons besoin de calculer **16 x 4 = 64 valeurs de qualité**.

Une façon efficace de représenter ces valeurs est d'utiliser une table Q, où chaque ligne correspond à un état ***s*** et chaque colonne à une action ***a***. Dans cette table, chaque cellule contient une valeur ***Q(s, a)***, qui indique la qualité de l'action *a* dans l'état *s*. Une valeur élevée (proche de 1) signifie que l'action est favorable, tandis qu'une faible valeur (proche de 0) indique qu'elle est peu avantageuse. Quand notre agent se trouve dans un état particulier *s*, il peut simplement consulter cette table pour voir quelle action présente la meilleure récompense *r*.

En outre, l'algorithme de Q-learning repose sur plusieurs paramètres fondamentaux qui influencent la façon dont l'agent apprend de ses expériences et ajuste ses valeurs Q. Ces paramètres permettent de contrôler la vitesse d'apprentissage, l'équilibre entre exploration et exploitation, ainsi que la prise en compte des récompenses futures. Le **taux d'apprentissage**, noté (**α**), est un paramètre qui contrôle la proportion d'une nouvelle information qui est utilisée pour mettre à jour les valeurs Q existantes. Il est compris entre 0 et 1. Une valeur élevée de α (proche de 1) signifie que l'agent accorde une grande importance aux nouvelles informations, modifiant rapidement ses valeurs Q. Une valeur faible, en revanche, rend l'apprentissage plus lent, car l'agent conserve davantage de l'ancienne information.

Le **facteur de réduction**, noté (γ), est un paramètre qui détermine l'importance accordée aux récompenses futures par rapport aux récompenses immédiates. Il est également compris entre 0 et 1. Si γ est proche de 1, l'agent valorise les récompenses futures, favorisant des actions qui maximisent les gains à long terme. À l'inverse, si γ est proche de 0, l'agent se concentre uniquement sur les récompenses immédiates, sans tenir compte de leur impact futur.

L'algorithme de Q-learning repose également sur une stratégie d'exploration-exploitation pour que l'agent ne se limite pas aux actions connues, mais explore aussi de nouvelles possibilités. Ce comportement est souvent réglé par le paramètre **epsilon (ε)** dans une stratégie appelée **epsilon-greedy**. ε est la probabilité que l'agent choisisse une action aléatoire pour explorer de nouvelles options. L'exploration consiste à essayer de nouvelles actions, même si elles ne semblent pas optimales selon les connaissances actuelles de l'agent. L'objectif est de permettre à l'agent de découvrir de nouvelles options ou des stratégies potentiellement meilleures. Une valeur élevée de ε favorise l'exploration, permettant à l'agent de découvrir des informations sur l'environnement. À l'inverse, une valeur faible favorise l'exploitation des connaissances existantes qui consiste à tirer parti de l'expérience accumulée par l'agent en sélectionnant l'action qui semble la plus prometteuse (c'est-à-dire celle ayant la meilleure valeur Q connue dans un état donné). L'exploitation maximise les gains immédiats, mais elle peut limiter la capacité de l'agent à découvrir des stratégies globalement optimales.

Ensuite nous avons utilisé l'équation de Bellman optimale qui permet de mettre à jour la valeur Q de chaque état-action (s,a) en tenant compte à la fois de la récompense immédiate reçue et de la valeur de la meilleure action dans l'état suivant. Il fonctionne en ajustant progressivement la valeur Q(s,a) en fonction de l'erreur de prédiction entre la récompense observée et l'estimation actuelle de la meilleure récompense future. Ce processus permet à l'agent de mettre à jour sa connaissance de la valeur de chaque action dans chaque état en tenant compte des nouvelles informations issues de ses interactions avec l'environnement.

L'équation de mise à jour du Q-learning est basée sur l'équation de Bellman optimale et s'écrit ainsi :

$$Q(s, a) = Q(s, a) + \alpha [r + \gamma \max_{a'} Q(s', a') - Q(s, a)]$$

Où :

- Q(s, a) est la valeur actuelle de l'état-action (s, a),

- α est le taux d'apprentissage, qui contrôle la vitesse de mise à jour,

- r est la récompense immédiate obtenue après avoir pris l'action a dans l'état s,

- γ est le facteur de réduction (ou discount factor), qui détermine l'importance des récompenses futures,

- max a' Q(s', a') est la valeur maximale estimée des actions possibles dans l'état suivant s'.

Cette équation de Bellman optimale pour Q-learning fonctionne en ajustant progressivement la valeur Q(s, a) en fonction de l'erreur de prédiction entre la récompense observée et l'estimation actuelle de la meilleure récompense future. Ce processus permet à l'agent de mettre à jour sa connaissance de la valeur de chaque action dans chaque état en tenant compte des nouvelles informations issues de ses interactions avec l'environnement.

II. Présentation de l'Environnement Frozen Lake

Comme annoncé plus haut, nous avons modifié l'environnement de Frozen Lake afin d'enrichir les interactions de l'agent. Ce nouvel environnement de Frozen Lake est une grille personnalisée de **4x4 cases**, où chaque type de case a un rôle spécifique.

La case *S* (rose pâle) est le point de départ de l'agent, symbolisé par un robot.

Les cases *F* (bleu clair) sont des surfaces glacées sans danger que l'agent peut traverser librement.

Les cases *H* (bleu foncé) sont des trous : si l'agent tombe dedans, l'épisode se termine prématurément.

Les cases *W* (bleu-gris) représentent des murs infranchissables, bloquant le passage de l'agent.
La case *P* (turquoise clair) offre un power-up, une récompense bonus pour l'agent.

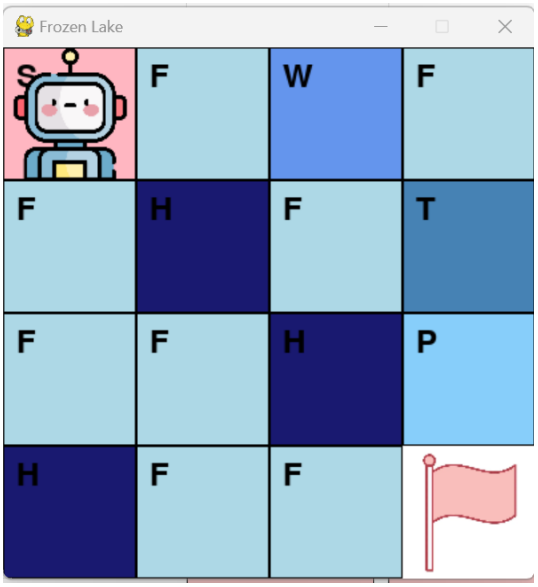
La case *T* (bleu acier) est un piège qui inflige une pénalité à l'agent.

Enfin, la case *G* (jaune clair) est l'objectif final à atteindre pour que l'agent réussisse l'épisode.

Le défi pour l'agent est de naviguer de la case de départ *S* jusqu'à l'objectif *G* tout en évitant les trous, les murs, et en tirant parti des cases bonus ou pièges selon la meilleure stratégie d'apprentissage.

Nous pouvons voir un aperçu de notre environnement :

Figure 1: L'environnement Frozen Lake amélioré



III. Mise en œuvre du Q-learning

Dans cette étape, on va d'abord initialiser une table Q qui associe chaque état (position de l'agent) à une valeur pour chaque action possible (haut, bas, gauche, droite). Cette table sera mise à jour à chaque étape de l'apprentissage. On va également configurer les paramètres essentiels pour Q-learning, y compris la stratégie d'exploration ε-greedy.

Afin de privilégier une mise à jour progressive des valeurs Q pour éviter des fluctuations trop importantes, nous avons choisi un taux d'apprentissage égal à 0.1 qui permet à l'agent de construire lentement une politique plus précise, particulièrement utile dans cet environnement complexe où des ajustements trop rapides pourraient entraîner une instabilité dans la politique d'apprentissage. Ensuite, nous avons fixé une valeur de 0.95 pour le facteur de réduction pour que l'agent donne une importance élevée aux récompenses futures, en privilégiant des stratégies à long terme. L'objectif de l'agent étant de trouver une sortie du lac en évitant les trous et les obstacles, un facteur de discount élevé l'encourage à évaluer chaque action en fonction de son impact futur et non seulement de la récompense immédiate.

Le taux d'exploration étant fixé à 1.0, on force l'agent à explorer largement l'environnement pour découvrir les conséquences des différentes actions dans chaque état. Cette exploration intensive est cruciale au début, car l'agent n'a aucune connaissance initiale de l'environnement.

Nous avons aussi fixé une valeur maximale pour epsilon à 1.0 permettant à l'agent de commencer avec une exploration totale et une valeur minimale pour epsilon de 0.01 garantissant qu'il y a toujours une petite probabilité d'exploration, même après un long apprentissage, pour éviter que l'agent ne reste bloqué dans une politique sous-optimale. Cette légère exploration résiduelle peut lui permettre de s'adapter si l'environnement change ou si certains états n'ont pas été suffisamment explorés au début.

Le taux de décroissance de l'exploration contrôle la vitesse à laquelle le taux d'exploration diminue avec le temps. Une décroissance lente (0.001) signifie que ε diminue progressivement,

ce qui permet à l'agent de passer plus de temps en phase d'exploration avant de basculer vers une exploitation plus systématique de ses connaissances.

Une fois nos paramètres établis, nous allons exécuter plusieurs épisodes d'apprentissage (5000) pour que l'agent puisse explorer l'environnement, mettre à jour la table Q et apprendre la meilleure stratégie. Cette boucle va suivre les étapes suivantes:

a) Pour chaque épisode :

- Initialiser l'état.
- Boucle d'action jusqu'à atteindre l'objectif ou tomber dans un trou.

b) Choisir l'action : Utiliser la stratégie ε-greedy pour décider entre exploration et exploitation.

c) Effectuer l'action et observer le nouvel état et la récompense.

d) Mettre à jour la table Q en utilisant l'équation de Bellman.

Configuration des Récompenses et des Obstacles

Pour guider l'agent à travers l'environnement de manière optimale, nous avons introduit des récompenses et des pénalités spécifiques en fonction des différents types de cases dans la grille. Chaque valeur de récompense ou de pénalité est choisie pour encourager ou dissuader certains comportements, alignant l'agent vers des actions bénéfiques et l'éloignant des situations indésirables.

- +10 pour atteindre l'objectif (G) : Cette récompense élevée incite fortement l'agent à atteindre l'objectif final, qui est la tâche principale dans cet environnement. En attribuant une récompense positive importante (+10), l'agent comprend progressivement que cette action est très bénéfique. En répétant des épisodes, l'agent apprend que certaines actions le rapprochent de cette grande récompense. Cela le pousse à former des stratégies pour atteindre l'objectif le plus efficacement possible, en maximisant sa récompense cumulative.

- -10 pour tomber dans un trou (H) : Cette forte pénalité dissuade l'agent de se déplacer vers les cases contenant un trou, qui représentent des situations d'échec dans cet environnement. Avec cette pénalité de -10, l'agent est "puni" de manière significative chaque fois qu'il tombe dans un trou. Cette pénalité lourde permet à l'agent de comprendre rapidement que ces cases doivent être évitées. Au fur et à mesure des épisodes, l'agent mémorise ces positions comme des zones dangereuses, minimisant ainsi les actions qui le conduiraient vers un trou.

- +1.5 pour un power-up (P) : Incite l'agent à explorer certaines cases pour un gain temporaire. Ces cases sont stratégiquement placées pour guider l'agent vers des zones qui pourraient ne pas faire partie du chemin direct vers l'objectif. Avec une récompense modérée de +1.5, ces power-ups offrent un intérêt pour l'exploration sans pour autant

détourner complètement l'agent de son objectif principal. Ce mécanisme permet de diversifier les expériences de l'agent, enrichissant ainsi son apprentissage et l'aidant à mieux estimer la valeur des actions dans différentes situations.

- -1 pour un piège (T) : Pénalité légère pour dissuader l'agent d'explorer les zones dangereuses. Ces cases représentent des pièges, mais avec une pénalité moindre (-1), l'agent apprend que ces zones sont moins désastreuses qu'un trou mais restent à éviter. En attribuant une légère pénalité, on encourage l'agent à évaluer les alternatives et à prioriser les chemins plus sûrs, tout en lui laissant la possibilité d'explorer les pièges pour voir si des récompenses plus importantes peuvent être trouvées à proximité.

- Murs (W) : Les murs agissent comme des obstacles physiques qui restreignent le mouvement de l'agent. Lorsqu'un mur est rencontré, l'agent reste sur sa position actuelle, et une pénalité de -0.5 est appliquée. Cette pénalité légère dissuade l'agent d'insister dans une direction bloquée, sans être trop punitive. En expérimentant, l'agent apprend progressivement que les murs sont infranchissables et ajuste sa stratégie pour éviter ces actions inefficaces.

IV. Analyse des résultats

Dans cette section, nous allons évaluer la performance de l'agent entraîné sur l'environnement **Frozen Lake**, en observant plusieurs aspects de l'analyse : la table Q apprise, le taux de réussite global, le nombre moyen de pas par épisode, les résultats des épisodes de validation, et enfin le taux de réussite de la politique apprise. Cette analyse détaillée nous permettra de mieux comprendre le comportement de l'agent et d'identifier des pistes d'amélioration pour les apprentissages futurs.

1. Analyse de la table Q après l'apprentissage

Après avoir appliqué l'algorithme de Q-learning et introduit les paramètres d'apprentissage définis précédemment, nous avons obtenu la table Q suivante:

Tableau 1: Table Q après l'apprentissage

```
Table Q après l'apprentissage :
[[  7.35   7.74   6.98   7.35]
 [  7.35  -9.66   4.52   6.37]
 [  0.     0.     0.     0.  ]
 [ -0.39  -0.92   0.    -0.  ]
 [  7.74   8.15 -10.     7.35]
 [  0.     0.     0.     0.  ]
 [ -5.36  -3.2   -0.78  -0.85]
 [  0.     0.     0.     0.  ]
 [  8.15 -10.     8.57   7.74]
 [  8.15   9.02  -0.98 -10.  ]
 [  7.22   9.5    1.48  -0.58]
 [  0.     0.     0.     0.  ]
 [  0.     0.     0.     0.  ]
 [-10.     9.02   9.5    8.57]
 [  9.02   9.5   10.    -0.98]
 [  0.     0.     0.     0.  ]]
```

Chaque ligne dans la table Q correspond à un état (ou position) dans la grille et chaque colonne représente une action possible comme:

- 0: Haut
- 1: Droite
- 2: Bas
- 3: Gauche

Les valeurs positives dans la table indiquent des actions bénéfiques, tandis que les valeurs négatives indiquent des actions à éviter (comme les trous ou les pièges).
Nous allons examiner chaque état clé pour observer la stratégie apprise par l'agent et comprendre ses décisions en fonction des valeurs Q:

⇒ État 0 (Départ, S) : Valeurs [7.35, 7.74, 6.98, 7.35]

- **Décision de l'agent** : L'agent choisit d'aller vers la droite (valeur Q : 7.74), car cette direction a la plus grande valeur Q.
- **Interprétation** : Bien que cette case ne mène pas directement à l'objectif, l'agent a appris que se déplacer vers la droite est bénéfique à long terme. Cela suppose qu'il a identifié cette direction comme sécurisée et propice pour progresser vers l'objectif tout en évitant les dangers.

⇒ État 1 : Valeurs [7.35, -9.66, 4.52, 6.37]

- **Décision de l'agent** : L'agent préfère aller vers le haut (valeur Q : 7.35).
- **Interprétation** : La valeur très négative pour aller vers la droite (-9.66) indique qu'il y a probablement un danger (comme un trou) dans cette direction. L'agent a appris à éviter cette direction risquée, préférant revenir vers une position plus sûre, même si cela ne le rapproche pas immédiatement de l'objectif.

⇒ État 4 : Valeurs [7.74, 8.15, -10.0, 7.35]

- **Décision de l'agent** : L'agent choisit d'aller vers la droite (valeur Q : 8.15).
- **Interprétation** : La direction droite est celle qui offre la meilleure progression. La valeur fortement négative pour aller vers le bas (-10.0) indique que l'agent a identifié un trou dans cette direction et a appris à l'éviter systématiquement pour réduire les pénalités.

⇒ État 8 : Valeurs [8.15, -10.0, 8.57, 7.74]

- **Décision de l'agent** : L'agent choisit d'aller vers le bas (valeur Q : 8.57).
- **Interprétation** : L'agent a appris à éviter la direction de droite, qui a une valeur Q de -10.0 (indiquant un danger). La direction choisie (Bas) est perçue comme avantageuse et permet de progresser de manière sécurisée.

⇒ État 9 : Valeurs [8.15, 9.02, -0.98, -10.0]

- **Décision de l'agent** : L'agent préfère aller vers la droite (valeur Q : 9.02).
- **Interprétation** : Cette décision indique que l'agent voit la direction de droite comme un chemin sûr vers l'objectif. La valeur fortement négative pour aller vers le bas (-10.0) montre qu'il a appris à éviter cette direction, où un danger est probablement présent.

⇒ État 13 : Valeurs [-10.0, 8.15, 9.5, 8.57]

- **Décision de l'agent** : L'agent choisit d'aller vers le bas (valeur Q : 9.5).
- **Interprétation** : Ici, l'agent privilégie la direction vers le bas comme étant la plus sécurisée et la plus proche de l'objectif. La valeur très négative pour aller vers la gauche (-10.0) indique un danger que l'agent a appris à éviter.

⇒ État 14 (Proche de l'Objectif) : Valeurs [9.02, 9.5, 10.0, -0.98]

- **Décision de l'agent** : L'agent choisit d'aller vers le bas (valeur Q : 10.0), ce qui le mène directement à l'objectif.
- **Interprétation** : Cette valeur élevée montre que l'agent reconnaît cette direction comme la meilleure pour atteindre l'objectif, maximisant ainsi sa récompense. Cela démontre une compréhension claire de la stratégie optimale dans cet état.

D’après ces premières observations, on peut en déduire que:

- **L’agent a appris à éviter les dangers** : Les valeurs fortement négatives pour certaines directions montrent que l’agent a appris à éviter les trous ou les pièges. Par exemple, les valeurs autour de -10 indiquent des trous que l'agent évite, privilégiant des directions avec des valeurs positives ou neutres.
- **La stratégie est optimisée vers l'objectif** : Les valeurs élevées dans les états proches de l’objectif (comme l’état 14 avec une valeur de 10 pour l’action vers le bas) montrent que l’agent a appris à identifier et atteindre l’objectif de manière efficace.

- **L’agent s’adapte à l’environnement** : L’agent a ajusté ses choix en fonction des pénalités et récompenses des cases. Il préfère les directions ayant des valeurs Q positives et tend à éviter celles avec des valeurs Q négatives, s'adaptant ainsi aux caractéristiques de l’environnement.

2. Analyse des résultats d’apprentissage et de test

Les résultats obtenus comme le taux de réussite et le nombre moyen de pas par épisode révèlent des informations importantes sur l'évolution de la stratégie de l'agent au fil du temps.

Taux de Réussite Moyen et Taux de Réussite dans les Derniers Épisodes

- **Taux de réussite global (4.73%)** : Ce faible taux de réussite indique que l’agent a souvent échoué à atteindre l'objectif pendant la phase d'apprentissage. Dans un environnement comme celui du lac gelé, avec des obstacles (murs) et des pièges (trous), ce taux est typique, surtout au début de l'apprentissage.

En phase d'apprentissage, l'agent utilise une stratégie epsilon-greedy pour explorer l'environnement, ce qui implique une proportion importante d'actions aléatoires au départ. Ce taux de réussite global montre que l'agent passe beaucoup de temps à explorer et n'a pas encore consolidé une politique stable et optimale. Cette exploration intensive est nécessaire pour découvrir des actions qui mènent à l'objectif et éviter les pièges.

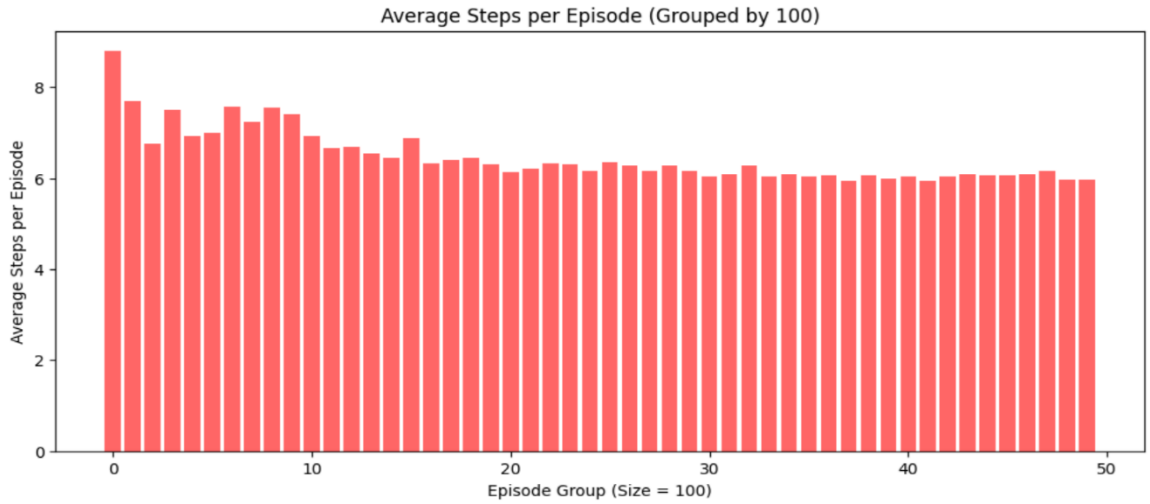
- **Taux de réussite dans les 100 derniers épisodes (9.50%)** : La légère amélioration du taux de réussite dans les derniers épisodes suggère que l'agent commence à stabiliser sa politique d'action. Il utilise de moins en moins d'actions aléatoires à mesure que le taux d'exploration diminue. Bien que le taux de réussite ait augmenté, l'écart reste faible entre le taux global et celui des derniers épisodes. Cela indique que l’agent rencontre encore des difficultés persistantes dans certaines situations, notamment pour naviguer entre les pièges et les murs. L'agent commence à optimiser sa stratégie, mais l'exploration reste active à un certain degré, ce qui conduit parfois à des actions non optimales.

Nombre Moyen de Pas par Épisode

- **Nombre moyen de pas global (6.50)** : Ce nombre modéré de pas par épisode montre que l'agent termine chaque épisode assez rapidement, que ce soit en atteignant l'objectif ou en tombant dans un trou.

Le nombre de pas relativement faible, combiné avec le faible taux de réussite, suppose que l'agent termine souvent les épisodes en tombant dans des pièges. Cela peut être dû au fait qu'il explore encore de nombreuses directions, y compris celles menant aux échecs. En théorie, si l’agent apprenait à éviter systématiquement les dangers, on s'attendait à une augmentation progressive du nombre de pas par épisode, signe d’une meilleure exploration avant d'atteindre l'objectif.

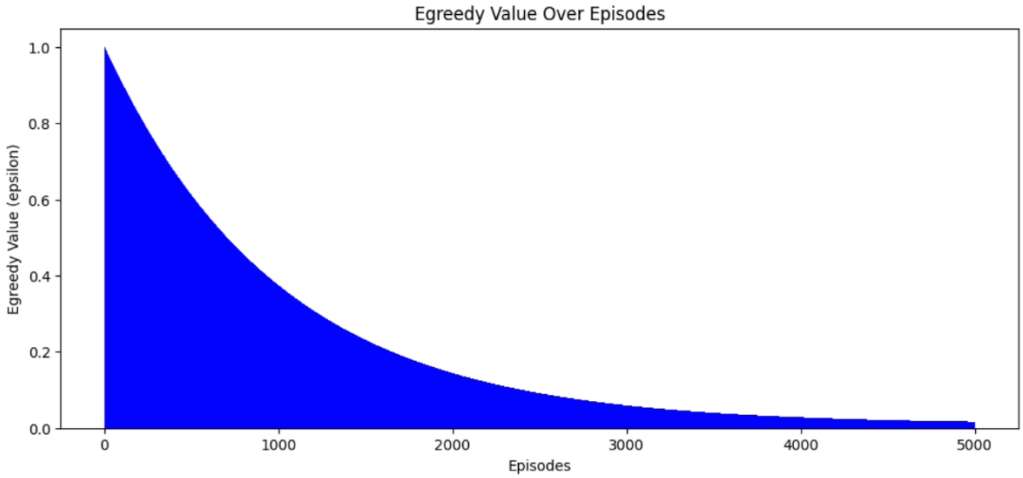
Figure 2: Moyenne des Pas par Épisode (Agrégée par Groupes de 100)



Ce graphique illustre l'évolution du nombre moyen de pas par épisode, regroupé par tranches de 100 épisodes. Cela permet de suivre la progression de l'agent en termes de rapidité d'atteinte de l'objectif.

Au début de l'apprentissage, le nombre moyen de pas est plus élevé, indiquant que l'agent explore davantage. Au fil du temps, la moyenne des pas par épisode tend à diminuer, ce qui suggère que l'agent améliore sa politique et devient plus efficace. La stabilisation de cette moyenne dans les derniers groupes d'épisodes montre que l'agent a probablement atteint une politique stable, lui permettant d'atteindre l'objectif avec un nombre de pas optimal.

Figure 3: Évolution de la Valeur Epsilon (Exploration) au Fil des Épisodes



Ce graphique présente la diminution de la valeur epsilon au fil des épisodes, illustrant la réduction progressive de l'exploration de l'agent.

La valeur epsilon commence élevée, ce qui signifie que l'agent explore largement les différentes actions au début de l'apprentissage. Au fil des épisodes, epsilon diminue progressivement, ce qui réduit la probabilité d'exploration et favorise l'exploitation de la meilleure stratégie apprise.

Cette transition permet à l'agent de stabiliser sa politique et d'atteindre une performance optimale dans l'environnement.

- **Nombre moyen de pas dans les 100 derniers épisodes (6.07)** : La légère diminution du nombre de pas dans les derniers épisodes indique une stabilisation de la politique d'action, même si elle n'est pas encore entièrement optimisée.

La réduction du nombre moyen de pas suggère que l'agent commence à naviguer de manière plus cohérente, évitant certains pièges qu'il rencontrait souvent au début. Cependant, le fait que cette diminution soit légère montre que l'agent n'a pas encore totalement éliminé les erreurs liées à l'exploration, ce qui peut entraîner des échecs dans certaines configurations.

3. Evaluation et visualisation de la politique apprise:

⇒ Evaluation de la politique apprise

Le taux de réussite de la politique apprise durant la phase de test est de 100%.

Ce taux de réussite dans la phase de test nous indique que l'agent a bien consolidé une politique qui lui permet d'atteindre systématiquement l'objectif lorsqu'il suit les actions optimales apprises dans la table Q.

Dans cette phase de test, l'agent exploite uniquement la politique apprise (choisissant toujours l'action ayant la valeur de Q maximale), sans exploration aléatoire. Ce taux de réussite élevé démontre que l'agent a bien appris à naviguer vers l'objectif en suivant les valeurs Q les plus avantageuses pour chaque état. Cela montre que l'agent peut atteindre l'objectif de manière fiable dans un environnement connu et fixe, où les positions des obstacles et de l'objectif restent inchangées.

Limites de la politique apprise par l'agent:

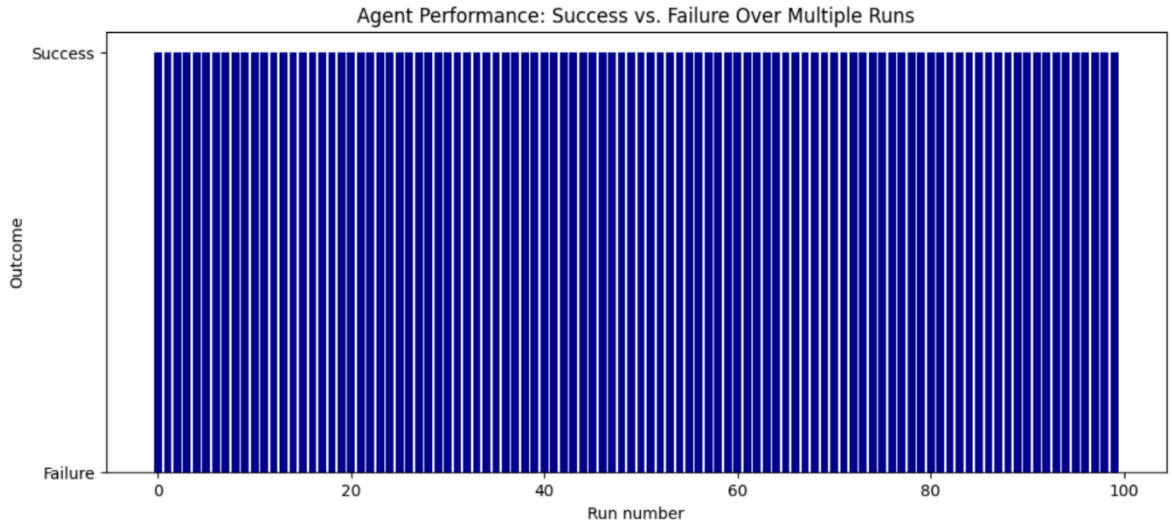
- **Convergence partielle des valeurs Q** : Bien que l'agent ait atteint l'objectif avec un taux de réussite de 100 % en phase de test, certaines valeurs de Q ne sont pas totalement stabilisées, notamment celles associées aux directions des murs et des pièges. Cela signifie que, même si l'agent évite les obstacles principaux dans le chemin optimal, il pourrait prendre des décisions moins optimales si des variations dans l'environnement (comme des changements de positions des obstacles) étaient introduites.
- **La flexibilité de la stratégie est limitée** : La politique apprise est bien adaptée à cet environnement spécifique, mais la faible consolidation des valeurs Q pour certaines actions indique que l'agent pourrait être moins performant dans un environnement légèrement modifié. Cela souligne une limitation dans la généralisation de sa stratégie actuelle.

⇒ Visualisation de la politique apprise

Ce graphique en barres montre le taux de succès de l'agent sur une série de 100 runs. Chaque barre représente le résultat d'un run individuel, où une valeur de 1 indique un succès (l'agent atteint l'objectif) et une valeur de 0 représenterait un échec. Ici, toutes les barres sont sur la valeur 1, ce qui signifie que l'agent a atteint l'objectif dans chaque run sans aucun échec.

Ce taux de succès de 100 % indique que l'agent a appris une politique efficace et stable qui lui permet d'atteindre l'objectif de manière systématique. Cela montre une consolidation réussie de la stratégie optimale pour naviguer dans l'environnement sans erreurs, démontrant une grande maîtrise de sa politique.

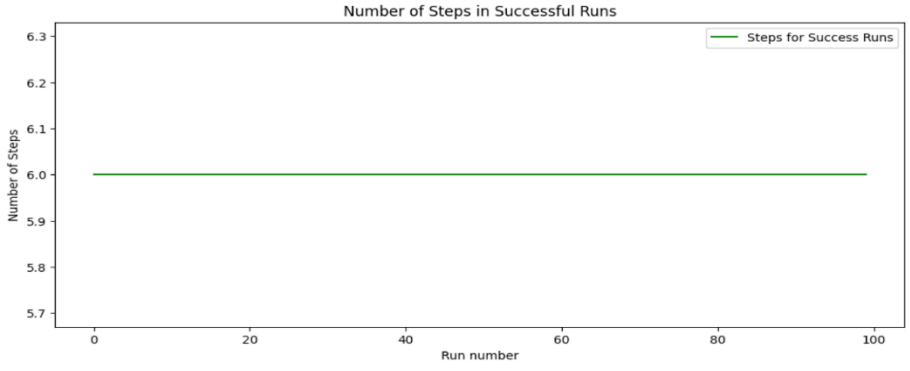
Figure 4: Taux de Succès de l'Agent sur 100 Runs



Ce graphique linéaire montre le nombre de pas que l'agent utilise pour atteindre l'objectif lors des runs réussis. La ligne est constante autour de la valeur 6, ce qui signifie que l'agent utilise systématiquement le même nombre de pas pour atteindre l'objectif à chaque run.

Cela indique que l'agent a non seulement appris à atteindre l'objectif sans échec, mais qu'il suit également un chemin optimal ou quasi optimal en termes de nombre de pas. Ce comportement constant montre que l'agent n'explore plus des actions non optimales et suit une trajectoire stable, ce qui reflète une stratégie optimisée dans cet environnement.

Figure 5: Nombre de Pas pour les Runs Réussis



CONCLUSION

Cette étude a exploré l'utilisation de l'apprentissage par renforcement, et plus particulièrement de l'algorithme de Q-learning, pour permettre à un agent de naviguer dans un environnement complexe, celui du lac gelé (Frozen Lake) avec de nouvelles configurations, en évitant des obstacles et en atteignant un objectif défini. Grâce à une table de valeurs Q associée à chaque état et action, l'agent a progressivement appris une stratégie pour maximiser sa récompense en minimisant les risques.
En apprenant progressivement à maximiser ses récompenses et à éviter les dangers, l'agent a pu développer une stratégie efficace pour atteindre son objectif.

Cette démarche met en évidence la capacité du Q-learning à transformer une exploration aléatoire en une politique optimale dans un environnement donné. La transition de l'exploration initiale vers une exploitation des actions optimales témoigne de la puissance de l'apprentissage par renforcement dans des environnements structurés et prévisibles.

Cependant, bien que le taux de réussite en phase de test atteigne 100 %, certaines valeurs Q restent partiellement convergées, notamment pour les directions impliquant des obstacles. Cela indique que la politique apprise est adaptée spécifiquement à cet environnement, mais pourrait manquer de flexibilité dans des configurations légèrement modifiées.

En conclusion, cette étude confirme l'efficacité de l'apprentissage par renforcement pour des tâches de navigation et d'optimisation dans des environnements fixes et modérément complexes, tout en ouvrant la voie à des améliorations et extensions qui pourraient rendre ces techniques applicables à des contextes plus variés et réalistes. Néanmoins, pour des applications dans des contextes plus dynamiques, des approches plus avancées, comme le Deep Q-learning, pourraient améliorer la robustesse et la généralisation de la stratégie apprise.

BIBLIOGRAPHIE

- Sarjit07. (2021). *Reinforcement Learning Using Q-Table - FrozenLake*. Récupéré de **https://www.kaggle.com/code/sarjit07/reinforcement-learning-using-q-table-frozenlake**

- Sutton, R. S. (2018). Reinforcement learning: An introduction. *A Bradford Book.*

- Watkins, C. J., & Dayan, P. (1992). Q-learning. *Machine learning*, *8*, 279-292.

- Reinforcement Learning 101: Q-Learning, **https://towardsdatascience.com/reinforcement-learning-101-q-learning-27add4c8536d**
- Q-Learning Algorithms: A Comprehensive Classification and Applications, **https://www.researchgate.net/publication/335805245_QLearning_Algorithms_A_Comprehensive_Classification_and_Applications**
- Gym documentation, Frozen Lake, **https://www.gymlibrary.dev/environments/toy_text/frozen_lake/**