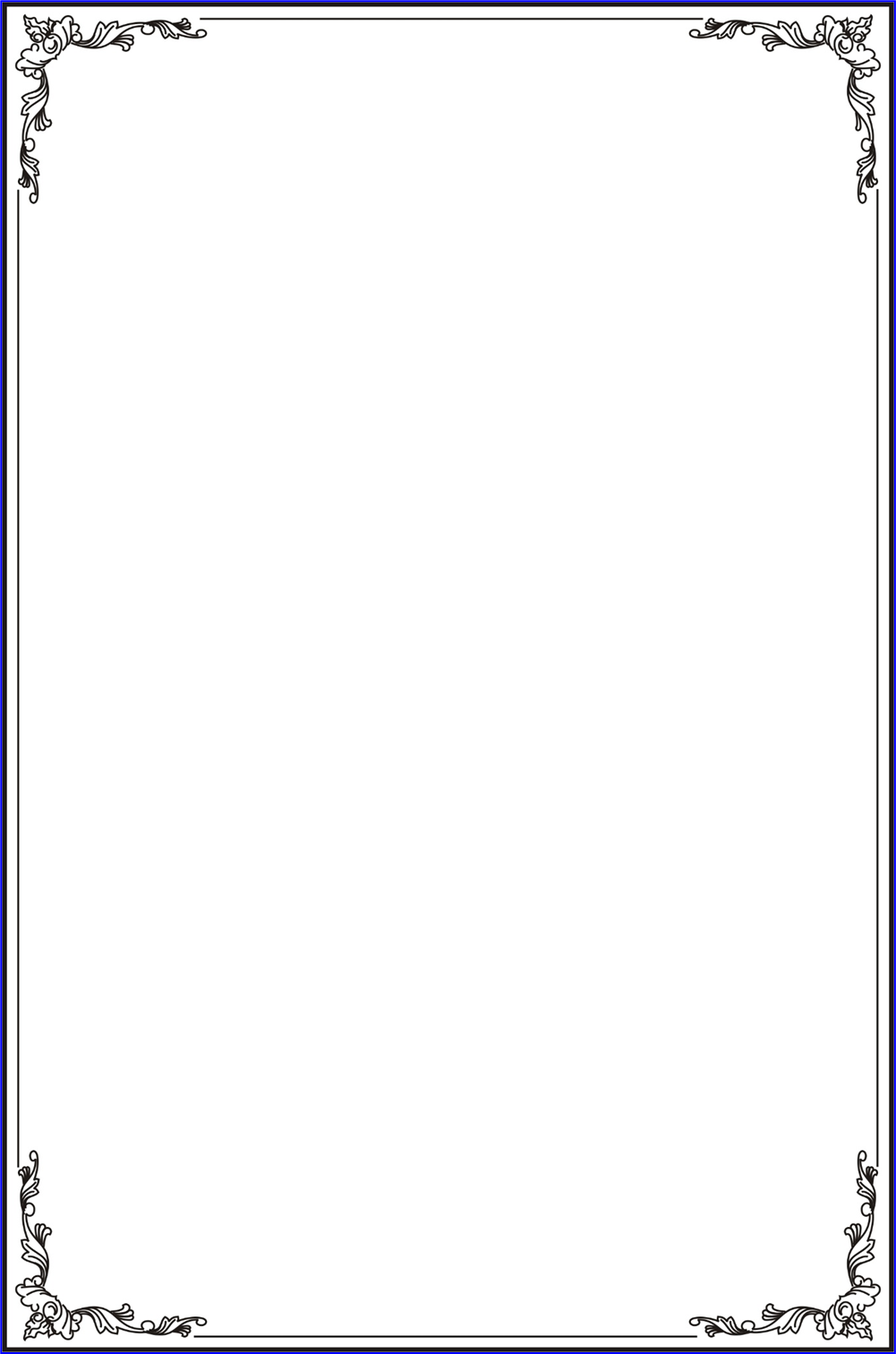
****

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

**🕯✡🕮🕮✡🕯**

****

**Đề tài**

**REINFORCEMENT LEARNING**

**BASIC CONCEPTS, FARAMA GYMNASIUM PACKAGE**

**MÔN HỌC: TRÍ TUỆ NHÂN TẠO**

**GVHD: ThS. Lê Minh Tân**

|  |  |
| --- | --- |
| ***Nhóm sinh viên thực hiện :*** | |
| Hoàng Mai Hiếu | 21110882 |
| Trương Nguyễn Thùy Trang | 21110691 |
| Đặng Công Tuấn | 21110709 |
| Nguyễn Văn Tín | 21110931 |
| Nguyễn Thành Lộc | 21110897 |
| Trần Hoàng Phúc  Nguyễn Hoàng Phương  Mai Đình Hồng Sơn | 21110606  21110609  21110465 |

**TP Hồ Chí Minh , tháng 5 năm 2023**

# MỤC LỤC

[I. KHÁI QUÁT CƠ BẢN VỀ REINFORCEMENT LEARNING VÀ LEARNING FROM REWARD 1](#_Toc134639251)

[1.Giới thiệu về học tăng cường và học từ phần thưởng 1](#_Toc134639252)

[2. Các thành phần chính của một bài toán Learning from Rewards 1](#_Toc134639253)

[3. Các phương pháp học từ phần thưởng 2](#_Toc134639254)

[4. Các thuật toán trong học tăng cường 2](#_Toc134639255)

[II. GYMNASIUM PACKAGE 3](#_Toc134639256)

[1. Cơ bản về Gymnasium Package 3](#_Toc134639257)

[2. Bắt đầu với Gymnasium 3](#_Toc134639258)

[3. Các môi trường cung cấp bởi Gymnasium Package 4](#_Toc134639259)

[4. Các thành phần chính của Gymnasium Package 6](#_Toc134639260)

[III.  Thực nghiệm với môi trường Frozen Lake 7](#_Toc134639261)

[1. Các thành phần của môi trường Frozen Lake trong Gymnasium 7](#_Toc134639262)

[2. Giải quyết môi trường Frozen Lake với chiến lược Epsilon-greedy và thuật toán Q-learning 8](#_Toc134639263)

[IV. ỨNG DỤNG CỦA HỌC TĂNG CƯỜNG TRONG LĨNH VỰC CỤ THỂ, ƯU VÀ NHƯỢC ĐIỂM 9](#_Toc134639264)

[1. Ứng dụng 9](#_Toc134639265)

[2. Ưu điểm và nhược điểm của học tăng cường 9](#_Toc134639266)

[V. KẾT LUẬN 10](#_Toc134639267)

# I. KHÁI QUÁT CƠ BẢN VỀ REINFORCEMENT LEARNING VÀ LEARNING FROM REWARD

## **1.Giới thiệu về học tăng cường và học từ phần thưởng**

Reinforcement Learning hay còn gọi là “học tăng cường” là một nhánh quan trọng của Machine Learning, đóng vai trò quan trọng trong nhiều ứng dụng của trí tuệ nhân tạo, bao gồm cả trò chơi, robot, học tập và các tác vụ tự động hóa khác

Reinforcement Learning tập trung vào việc huấn luyện các mô hình máy tính để tương tác với một môi trường để đạt được một mục tiêu cụ thể thông qua phương thức phản hồi thích nghi. Trong Reinforcement Learning, mô hình máy tính cố gắng tìm ra các hành động tối ưu dựa trên các phản hồi của môi trường.

Learning from Rewards là một phương pháp học tăng cường (reinforcement learning) trong đó tác tử (agent) học cách lựa chọn hành động trong một môi trường (environment) để tối đa hóa tổng phần thưởng (reward) nhận được từ môi trường.

Khi tác tử thực hiện một hành động trong một trạng thái, môi trường sẽ cung cấp phản hồi dưới dạng phần thưởng, cho biết giá trị của hành động đó trong trạng thái hiện tại. Tác tử sử dụng thông tin phần thưởng này để cập nhật chính sách (policy), tăng cường khả năng lựa chọn hành động trong tương lai.

Quá trình này được lặp lại nhiều lần, cho đến khi tác tử học được một chính sách tối ưu để tối đa hóa phần thưởng trong môi trường. Trong quá trình này, hàm giá trị (value function) được sử dụng để đánh giá chất lượng của các hành động trong một trạng thái, giúp tác tử cập nhật chính sách một cách hiệu quả.

## **2. Các thành phần chính của một bài toán Learning from Rewards**

Một bài toán Learning from Rewards bao gồm các thành phần chính sau:

1. Môi trường (Environment): Môi trường là nơi mà hành động xảy ra và cung cấp thông tin về trạng thái hiện tại của hệ thống và các tác động của các hành động.
2. Trạng thái (State): Trạng thái là một tập hợp các đặc trưng của môi trường được sử dụng để mô tả tình trạng của hệ thống.
3. Hành động (Action): Hành động là các quyết định mà hệ thống có thể thực hiện để tương tác với môi trường.
4. Phần thưởng (Reward): Phần thưởng là một giá trị số được cung cấp cho hệ thống bởi môi trường để đánh giá hiệu quả của các hành động.
5. Chính sách (Policy): Chính sách là một quy tắc quyết định được sử dụng để lựa chọn hành động dựa trên trạng thái hiện tại của hệ thống.
6. Hàm giá trị (Value function): Hàm giá trị đánh giá giá trị của một trạng thái (state) hoặc một cặp trạng thái-hành động (state-action pair) dựa trên các phần thưởng (rewards) có được khi đi từ trạng thái đó hoặc khi thực hiện hành động đó từ trạng thái đó.
7. Mục tiêu (Objective): Mục tiêu là một tiêu chí được sử dụng để đánh giá hiệu quả của hệ thống và định hướng quá trình học.

## **3. Các phương pháp học từ phần thưởng**

- MODEL-BASED REINFORCEMENT LEARNING:là một phương pháp học tăng cường trong đó tác tử cố gắng xây dựng một mô hình của môi trường mà nó đang tương tác. Mô hình này được sử dụng để dự đoán trạng thái tiếp theo của môi trường và tính toán giá trị của các hành động.

- MODEL-FREE REINFORCEMENT LEARNING: là một phương pháp học tăng cường không sử dụng một mô hình của môi trường để tính toán giá trị của hành động. Thay vào đó, nó tập trung vào việc cập nhật hàm giá trị (value function) hoặc chính sách (policy) mà không cần biết một mô hình rõ ràng của môi trường:

+ ACTION-UTILITY LEARNING: là một phương pháp học tăng cường cho phép tác tử học cách chọn hành động tối ưu dựa trên độ ưu tiên của các mục tiêu cần đạt được. Điều này được thực hiện bằng cách ước lượng giá trị của từng hành động, dựa trên mức độ đóng góp của hành động đó đối với các mục tiêu cần đạt được.

+ POLICY SEARCH: là một phương pháp học tăng cường trong đó mục tiêu là tìm ra chính sách tối ưu trực tiếp, thay vì ước lượng hàm giá trị của một chính sách đã cho. Trong policy search, chúng ta cố gắng tìm kiếm chính sách mà cho phép tác tử đạt được hiệu suất tốt nhất trong một nhiệm vụ cụ thể. Chính sách có thể được biểu diễn dưới dạng một bản đồ từ trạng thái sang hành động, hoặc bằng các phương pháp khác như mạng nơ-ron.

## **4. Các thuật toán trong học tăng cường**

Thuật toán học tăng cường là một lĩnh vực quan trọng trong Machine Learning, nó tập trung vào việc giúp tác tử tự động học cách tương tác với một môi trường bằng cách đưa ra các hành động để đạt được các mục tiêu nhất định.

Một trong những thuật toán MODEL-BASED REINFORCEMENT LEARNING đáng chú ý là Dyna-Q, thuật toán này kết hợp MODEL-BASED và MODEL-FREE REINFORCEMENT LEARNING bằng cách sử dụng mô hình của môi trường để dự đoán trạng thái tiếp theo và tính toán giá trị của các hành động. Monte Carlo Tree Search (MCTS) là một thuật toán khác của MODEL-BASED REINFORCEMENT LEARNING, được sử dụng trong các trò chơi như cờ vua.

Trong MODEL-FREE REINFORCEMENT LEARNING, có nhiều thuật toán đáng chú ý được sử dụng như Q-Learning và SARSA. Cả hai đều tập trung vào cập nhật hàm giá trị của từng hành động để tìm kiếm hành động tối ưu. Deep Q-Networks (DQN) là một trong những thuật toán phổ biến khác của MODEL-FREE REINFORCEMENT LEARNING, nó sử dụng mạng nơ-ron sâu để học các hàm giá trị và tối ưu hành động.

# II. GYMNASIUM PACKAGE

## **1. Cơ bản về Gymnasium Package**

Thư viện Gymnasium là một thư viện mã nguồn mở cho phép tạo ra các môi trường tương tác để huấn luyện các mô hình học tăng cường (reinforcement learning). Nó cung cấp một bộ các môi trường phổ biến và đa dạng cho các bài toán học tăng cường, bao gồm các môi trường về trò chơi, robot và điều khiển tối ưu.

Gymnasium cung cấp một bộ sưu tập các môi trường hoặc "nhiệm vụ" có thể được sử dụng để kiểm tra và phát triển các thuật toán học tăng cường. Các môi trường này thường giống như các trò chơi, với các quy tắc được định nghĩa rõ ràng và một cấu trúc phần thưởng, làm cho chúng hữu ích để đánh giá và so sánh các thuật toán học tăng cường khác nhau.

Các môi trường trong Gymnasium Package đều cung cấp cho các tác tử một trạng thái (state) của môi trường, các hành động (actions) có thể thực hiện, và các phần thưởng (rewards) nhận được sau mỗi hành động. Tác tử sử dụng các thuật toán học tăng cường để tìm hiểu cách chọn hành động tối ưu dựa trên trạng thái hiện tại của môi trường để đạt được phần thưởng cao nhất.

Vì vậy, các môi trường trong Gymnasium Package rất phù hợp cho việc nghiên cứu và phát triển các thuật toán học tăng cường.

## **2. Bắt đầu với Gymnasium**

Quy trình cơ bản để sử dụng Gymnasium để huấn luyện mô hình Học tăng cường như sau:

Bước 1: Xác định môi trường bạn muốn làm việc.

Bước 2: Khởi tạo môi trường.

Bước 3: Xác định chính sách của tác tử (tức là cách nó quyết định hành động nào sẽ được thực hiện).

Bước 4: Tương tác với môi trường, thực hiện các hành động và nhận phần thưởng.

Bước 5: Cập nhật chính sách của tác tử dựa trên các phần thưởng mà nó nhận được.

Bước 6: Dừng việc huấn luyện nếu đã huấn luyện đủ số lần lặp hoặc tác tử đã đạt hiệu suất hài lòng. Nếu không, lặp lại bước 4 và 5 cho đến  khi trò chơi kết thúc và reset môi trường.

## **3. Các môi trường cung cấp bởi Gymnasium Package**

**3.1. Classic Control**

**Ảnh có chứa biểu đồ

Mô tả được tạo tự động**

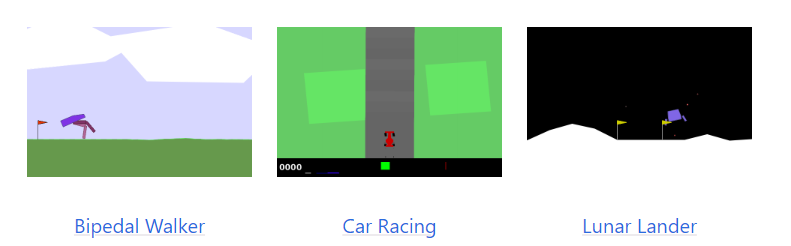
Có năm môi trường điều khiển cổ điển: Acrobot, CartPole, Mountain Car, Continuous Mountain Car, and Pendulum. Tất cả các môi trường này đều ngẫu nhiên về trạng thái ban đầu của chúng, trong một phạm vi nhất định.

Trong số các môi trường Gymnasium, tập hợp các môi trường này có thể được coi là những môi trường dễ giải quyết hơn bằng chính sách.

Tất cả các môi trường đều có thể định cấu hình cao thông qua các đối số được chỉ định trong tài liệu của từng môi trường.

Xem thêm tại: https://gymnasium.farama.org/environments/classic\_control/

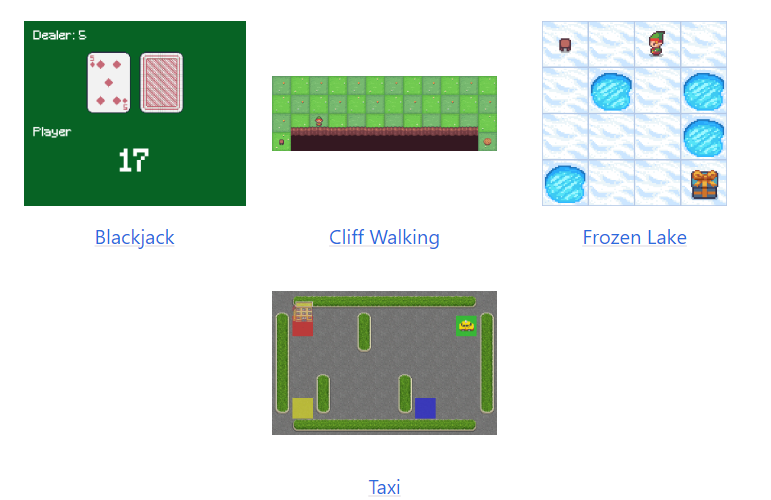
**3.2. Box2D**

****

Tất cả các môi trường này đều liên quan đến các trò chơi đồ chơi dựa trên điều khiển vật lý, sử dụng vật lý dựa trên Box2D và kết xuất dựa trên PyGame. Những môi trường này đã được Oleg Klimov đóng góp vào những ngày đầu của OpenAI Gym và đã trở thành tiêu chuẩn môi trường trò chơi phổ biến kể từ đó. Tất cả các môi trường đều có thể định cấu hình cao thông qua các đối số được chỉ định trong tài liệu của từng môi trường.

Xem thêm tại: https://gymnasium.farama.org/environments/box2d/

**3.3. Toy Text**

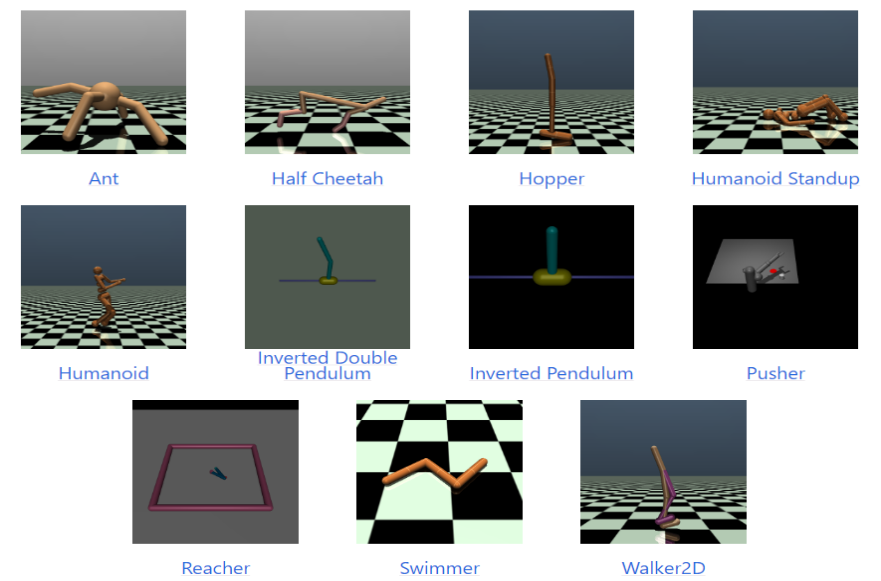
****

Môi trường văn bản đồ chơi được thiết kế cực kỳ đơn giản, với các không gian hành động và trạng thái riêng biệt nhỏ, do đó dễ học. Do đó, chúng phù hợp để gỡ lỗi triển khai các thuật toán học tăng cường.

Các môi trường có thể định cấu hình thông qua các đối số được chỉ định trong tài liệu của từng môi trường.

Xem thêm tại: https://gymnasium.farama.org/environments/toy\_text/

**3.4 MuJoCo**

****

MuJoCo là viết tắt của Multi-Joint dynamics with Contact. Nó là một công cụ vật lý để hỗ trợ nghiên cứu và phát triển về rô-bốt, cơ chế sinh học, đồ họa và hoạt hình cũng như các lĩnh vực khác cần mô phỏng nhanh và chính xác.

Có 11 môi trường Mujoco: Ant, HalfCheetah, Hopper, Humanoid, HumanoidStandup, InvertedDoublePendulum, InvertedPendulum, Pusher, Reacher, Swimmer và Walker2d.

Xem thêm tại: https://gymnasium.farama.org/environments/mujoco/

**3.5. Atari**

Một tập hợp các môi trường Atari 2600 được mô phỏng thông qua Stella và Môi trường học tập Arcade.

Ảnh có chứa văn bản

Mô tả được tạo tự động Xem thêm tại: https://gymnasium.farama.org/environments/atari/

## **4. Các thành phần chính của Gymnasium Package**

Observation and Action Spaces: Không gian quan sát (observation space) là tập hợp các trạng thái có thể mà một tác tử có thể quan sát được trong môi trường. Còn không gian hành động (action space) là tập hợp các hành động có thể mà một tác tử có thể thực hiện trong môi trường.

Episode: Một Episode là một vòng chơi hoàn chỉnh của một môi trường, bắt đầu từ trạng thái ban đầu và tiếp tục cho đến khi đạt được trạng thái kết thúc. Mỗi tập được tạo thành bởi một chuỗi các trạng thái, hành động và phần thưởng.

Wrapper: Một wrapper là một công cụ trong Gymnasium cho phép bạn thay đổi hành vi của một môi trường mà không cần thay đổi mã nguồn của nó. Các wrapper có thể được sử dụng để thêm các tính năng như giới hạn thời gian, hình dạng phần thưởng và chặn hành động.

Benchmark: Gymnasium cung cấp một bộ các môi trường đánh giá chuẩn để đánh giá và so sánh các thuật toán học tăng cường. Các môi trường đánh giá này bao gồm các nhiệm vụ điều khiển cổ điển, các trò chơi Atari và các nhiệm vụ robot.

**III.  Thực nghiệm với môi trường Frozen Lake**

**1. Các thành phần của môi trường Frozen Lake trong Gymnasium**

**1.1. Những quy tắc cơ bản của Frozen Lake:**

Môi trường FrozenLake trong Gymnasium là môi trường mô phỏng một hồ đóng băng. Nhiệm vụ của một tác nhân trong môi trường này là di chuyển từ vị trí khởi đầu đến vị trí đích một cách an toàn mà không rơi vào một trong các ô trên hồ đóng băng (các ô này được đại diện bởi các ký tự ‘H’, các ô bình thường được kí hiệu là ‘F’, đích đến kho báu kí hiệu là ‘G’).

**1.2. Action Space**

Trong môi trường FrozenLake của Gymnasium, agent sử dụng một vector gồm 1 phần tử để thực hiện hành động. Không gian hành động được định nghĩa bằng các hướng đi mà agent có thể di chuyển, bao gồm:

·         0: LEFT (trái)

·         1: DOWN (xuống)

·         2: RIGHT (phải)

·         3: UP (lên)

**1.3. Observation Space**

Một quan sát được biểu diễn bằng một giá trị thể hiện vị trí hiện tại của người chơi dưới dạng current\_row \* nrows + current\_col (trong đó cả hàng và cột bắt đầu từ 0).

Ví dụ, vị trí mục tiêu trên bản đồ 4x4 có thể được tính như sau: 3 \* 4 + 3 = 15. Số lượng quan sát có thể khả thi phụ thuộc vào kích thước của bản đồ. Ví dụ, bản đồ 4x4 có 16 quan sát khả thi. Quan sát được trả về dưới dạng một số nguyên (int()).

**1.4. Starting State**

Một tập hợp các hành động và trạng thái xác định môi trường trò chơi FrozenLake trong thư viện Gymnasium bao gồm các trạng thái của trò chơi. Trạng thái ban đầu của một tập hợp này là trạng thái [0], có nghĩa là người chơi bắt đầu tại vị trí [0,0] trên bản đồ của trò chơi.

**1.5. Rewards**

·         Đến được đích: +1 điểm

·         Rơi vào hố: 0 điểm

·         Đi vào ô băng đóng: 0 điểm

Nói cách khác, agent chỉ nhận được điểm nếu nó đến được đích và không bị trừ điểm nếu đi vào ô băng đóng hoặc hố.

**1.6. Điều kiện kết thúc một Episode**

* Quá trình kết thúc trong các trường hợp sau:

·         Agent di chuyển vào hố.

·         Agent đến được mục tiêu tại vị trí [max(nrow)-1, max(ncol)-1], với max(nrow) và max(ncol) lần lượt là số hàng và số cột lớn nhất trên bản đồ.

·         Cắt ngắn tập huấn luyện (nếu sử dụng time\_limit wrapper): Tổng số bước đi trong một episode bị giới hạn ở 100 bước cho bản đồ 4x4, 200 bước cho bản đồ 8x8.

**2. Giải quyết môi trường Frozen Lake với chiến lược Epsilon-greedy và thuật toán Q-learning**

**2.1. Chiến lược Epsilon-greedy trong môi trường Frozen Lake**

- Chiến lược epsilon-greedy là một phương pháp để giải quyết bài toán đưa ra hành động tốt nhất cho một tình huống dựa trên chính sách hiện tại. Theo chiến lược này, đôi khi tác tử sẽ chọn một hành động tốt nhất (với xác suất 1-epsilon) hoặc một hành động ngẫu nhiên (với xác suất epsilon) để tìm kiếm các chính sách mới, có thể đem lại phần thưởng tốt hơn trong tương lai.

- Trong môi trường Frozen Lake, chiến lược epsilon-greedy có thể được áp dụng để xác định hướng di chuyển của agent trên mỗi ô. Tại mỗi bước, agent có thể chọn hành động được khuyến nghị bởi chính sách hiện tại hoặc một hành động ngẫu nhiên. Chính sách sẽ được học qua thời gian bằng cách cập nhật ước lượng giá trị hành động cho mỗi cặp trạng thái-hành động dựa trên các phần thưởng nhận được trong quá trình di chuyển trên bản đồ.

- Khi di chuyển trên bản đồ nhiều lần, agent sẽ học được chính sách tối ưu nhằm tối đa hóa phần thưởng trung bình. Ban đầu, agent có thể khám phá bằng cách thực hiện các hành động ngẫu nhiên để tìm kiếm các chiến lược mới. Tuy nhiên, khi di chuyển trên bản đồ, agent sẽ bắt đầu tận dụng chính sách tốt nhất đã biết, nhằm tối đa hóa phần thưởng trung bình trong thời gian dài.

**2.2. Huấn luyện tác tử chơi Frozen Lake với thuật toán Q-learning**

Q-Learning là một thuật toán RL có thể áp dụng cho nhiều môi trường khác nhau. Q-learning thực hiện bằng cách lặp đi lặp lại các action và nhận được reward để xây dựng look-up table gọi là Q-table. Các bước xây dựng Q-table:

1. Khởi tạo bảng Q với các giá trị bất kỳ.

2. Dựa vào chiến lược epsilon-greedy để lựa chọn hành động.

3. Thực hiện hành động và quan sát kết quả. Kết quả này sẽ bao gồm trạng thái mới, phần thưởng nhận được và trạng thái tiếp theo của người chơi.

4. Sử dụng công thức Q-learning để cập nhật giá trị trong Q-table theo công thức :  
  
Ảnh có chứa biểu đồ

Mô tả được tạo tự động

5.  Lặp lại các bước từ 2 đến 4 cho đến khi đạt được một điều kiện dừng nhất định, ví dụ như đạt đến số lần lặp tối đa.

# IV. ỨNG DỤNG CỦA HỌC TĂNG CƯỜNG TRONG LĨNH VỰC CỤ THỂ, ƯU VÀ NHƯỢC ĐIỂM

## **1. Ứng dụng**

Các ứng dụng của Reinforcement Learning rất đa dạng và phong phú. Chúng ta có thể kể đến một số ứng dụng như: quản lý xe ô tô tự lái, giải quyết vấn đề tiêu thụ năng lượng, điều khiển tín hiệu giao thông, chăm sóc sức khỏe, phát triển người máy, marketing, game, hỗ trợ điều khiển nhập học trong viễn thông và hỗ trợ phi công trực thăng, và cả trong khoa học để giúp hiểu các phản ứng hóa học. Các ứng dụng này sử dụng các thuật toán Reinforcement Learning để học từ các tương tác với môi trường, giúp tối ưu hóa các hành động và đưa ra quyết định chính xác. Thuật toán Deep Learning như AlphaGo, Alpha Zero cũng được áp dụng để tạo ra các máy chơi game và phát hiện lỗi trong môi trường trò chơi.

## **2. Ưu điểm và nhược điểm của học tăng cường**

**Ưu điểm:**

- Nó có thể giải quyết các vấn đề phức tạp và bậc cao hơn. Ngoài ra, các giải pháp thu được sẽ rất chính xác.

- Có thể sửa các lỗi xảy ra trong quá trình đào tạo.

- Kể cả khi không có dữ liệu đào tạo, nó sẽ học được thông qua kinh nghiệm mà nó có được từ việc xử lý dữ liệu đào tạo

- Đối với các vấn đề khác nhau, nó cung cấp các mô hình hoàn hảo để giải quyết những vấn đề phức tạp.

**Nhược điểm:**

- Không thích hợp để giải quyết các vấn đề đơn giản hơn. Lý do là, các mô hình thường giải quyết các vấn đề phức tạp. Và cũng không thích hợp vì sẽ lãng phí sức mạnh xử lý và không gian không cần thiết

- Reinforcement Learning cần nhiều dữ liệu để cung cấp cho mô hình tính toán. Điều này tiêu tốn thời gian

- Chi phí bảo trì rất cao.

- Việc đào tạo quá nhiều có thể dẫn đến quá tải các trạng thái của mô hình, có thể làm giảm kết quả.

# V. KẾT LUẬN

Tóm lại, Learning from Rewards là một phương pháp trong Học máy, trong đó tác tử học cách tương tác với môi trường và tối ưu hóa hành động của mình dựa trên phần thưởng hoặc hình phạt. Phương pháp này có ứng dụng rộng rãi trong nhiều lĩnh vực như robotica, trò chơi điện tử, tối ưu hóa tự động và hệ thống điều khiển.

Gymnasium Package là một thư viện mã nguồn mở cung cấp một môi trường mô phỏng các vấn đề tối ưu hóa cho phép các nhà nghiên cứu và lập trình viên có thể thử nghiệm các thuật toán học tăng cường và tạo ra các mô hình học máy.

Sử dụng Gymnasium Package, các nhà nghiên cứu và lập trình viên có thể tạo và kiểm tra các thuật toán học tăng cường trên một loạt các môi trường khác nhau, từ đơn giản đến phức tạp, để giải quyết các vấn đề từ học tăng cường đơn giản đến các vấn đề phức tạp như tự động hóa sản xuất và các bài toán tối ưu hóa trong các lĩnh vực kinh tế và tài chính.

Hướng phát triển tiềm năng của Learning from Rewards và Gymnasium Package có thể bao gồm mở rộng các mô hình học tăng cường để giải quyết các vấn đề khó khăn hơn trong các môi trường mô phỏng, tập trung vào các ứng dụng thực tế và tích hợp các công nghệ mới vào thư viện.