NGUYỄN THỊ THUẬN

CÔNG NGHỆ THÔNG TIN

2004-2006

HÀ NỘI
2006

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

LUẬN VĂN THẠC SĨ KHOA HỌC

NGÀNH: CÔNG NGHỆ THÔNG TIN

PHƯƠNG PHÁP HỌC TĂNG CƯỜNG

NGUYỄN THỊ THUẬN

HÀ NỘI 2006

LỜI CẢM ƠN

Trong suốt quá trình học tập cũng như quá trình làm luận văn, em đã nhận được sự giúp đỡ của các thầy cô giáo trong bộ môn, đặc biệt là sự chỉ bảo hướng dẫn tận tình của thầy giáo hướng dẫn TS Nguyễn Linh Giang. Với lòng biết ơn sâu sắc, em xin chân thành cảm ơn các thầy cô giáo trong bộ môn đặc biệt là thầy giáo TS Nguyễn Linh Giang đã giúp đỡ để em hoàn thành luận văn thạc sỹ khoa học này.

Em cũng xin gửi lời cảm ơn tới ban lãnh đạo cũng như các đồng nghiệp nơi em đang công tác đã tạo điều kiện giúp em có một môi trường nghiên cứu và làm việc tốt.

Cuối cùng, em xin gửi lời cảm ơn tới gia đình, bạn bè, những người thân đã luôn động viên, khích lệ và giúp đỡ em trong suốt quá trình học tập và làm luận văn vừa qua.

Hà Nội, tháng 10 năm 2006
Học viên
Nguyễn Thị Thuận
Lớp: Cao học CNTT 2004-2006

MỤC LỤC

DANH MỤC CÁC KÝ HIỆU, CHỮ VIẾT TẮT

Thuật ngữ	*Viết tắt*
Học tăng cường (Reinforcement Learning)	RL
Phương pháp lập trình động (Dynamic Programming)	DP
Phương pháp Monte Carlo	MC
Phương pháp Temporal Difference	TD

MỞ ĐẦU

- **Tính cấp thiết của đề tài**

Xã hội ngày càng hiện đại, các kỹ thuật công nghệ ngày càng phát triển, đi cùng với nó là các nghiên cứu phát triển không ngừng về lĩnh vực trí tuệ nhân tạo và học máy, cho ra đời các hệ thống máy móc thông minh ứng dụng rộng rãi trong hầu hết các lĩnh vực đời sống như máy truy tìm dữ liệu, chẩn đoán y khoa, phát hiện thẻ tín dụng giả, phân tích thị trường chứng khoán, phân loại chuỗi DNA, nhận dạng tiếng nói và chữ viết, … đặc biệt là trong lĩnh vực điều khiển.

Các phương pháp tự đào tạo (học) đã được đưa ra từ rất lâu để chỉ khả năng các hệ thống thông minh trong quá trình hoạt động tự tích luỹ, phân tích các thông tin thu được từ đó tự nâng cao khả năng của bản thân, đây chính là mục đích quan trọng trong lỹ thuyết quyết định cũng như trong các bài toán tự động hoá và điều khiển tối ưu.

Chúng ta có nhiều loại thuật toán học như học có giám sát, học không có giám sát, học tăng cường, mỗi loại thuật toán thích ứng với từng loại bài toán cụ thể. Trong phạm vi đề tài này, chúng ta sẽ nghiên cứu và tìm hiểu các vấn đề liên quan đến phương pháp học tăng cường. Đây là một thuật toán học có khả năng giải quyết được những bài toán thực tế khá phức tạp trong đó có sự tương tác giữ hệ thống và môi trường. Với những tình huống môi trường không chỉ đứng yên, cố định mà thay đổi phức tạp thì các phương pháp học truyền thống không còn đáp ứng được mà phải sử dụng phương pháp học tăng cường. Những bài toán với môi trường thay đổi trong thực tế là không nhỏ và ứng dụng nhiều trong các lĩnh vực quan trọng.

- **Mục đích**

Qua quá trình làm luận văn sẽ tổng hợp và nắm vững các kiến thức về phương pháp học tăng cường nói chung. Hiểu rõ ý tưởng, cơ chế hoạt động các thuật toán học tăng cường và ứng dụng trong các bài toán điển hình cụ thể. Đồng thời cũng thực hiện mô phỏng bài toán thử nghiệm, đo đạc thống kê và đánh giá kết quả thử nghiệm về các thuật toán RL.

- **Giới hạn vấn đề**

Do những hạn chế về điều kiện và thời gian thực hiện, đề tài nghiên cứu mới chỉ ở mức lý thuyết và cài đặt thử nghiệm, chưa được ứng dụng vào thực tiễn.

- **Hướng phát triển**

Trong thời gian tới, sẽ cố gắng ứng dụng các kiến thức về phương pháp học tăng cường, xây dựng bài toán thực tiễn cụ thể và ứng dụng rộng rãi.

- **Bố cục của luận văn**

Luận văn gồm 3 chương với những nội dung chính như sau:

Chương 1: Trình bày lý thuyết tổng quan về phương pháp học tăng cường, mô hình bài toán quyết định Markov, bên cạnh đó cũng giới thiệu sơ lược về sự ra đời, cũng như lịch sử phát triển của phương pháp học tăng cường, các lĩnh vực ứng dụng trong thực tiễn.

Chương 2: Trình bày chi tiết về đặc điểm, các bước thực hiện của từng loại giải thuật học tăng cường đã và đang được sử dụng hiện nay.

Chương 3: Trình bày về bài toán lựa chọn thử nghiệm, giới thiệu lại sơ qua về loại thuật toán học tăng cường lựa chọn áp dụng trong bài toán thử nghiệm. Các kịch bản thử nghiệm và các kết quả thu được. Trên cơ sở đó, kết luận đánh giá và đưa ra giải pháp cải tiến.

Chương 1 BÀI TOÁN QUYẾT ĐỊNH MARKOV VÀ PHƯƠNG PHÁP HỌC TĂNG CƯỜNG

Phương pháp học tăng cường là một phương pháp phổ biến để giải các bài toán quyết định Markov. Bài toán quyết định Markov có rất nhiều ứng dụng trong các lĩnh vực kỹ thuật như lý thuyết quyết định, quy hoạch toán học, điều khiển tối ưu, ... Trong phần này, chúng ta sẽ trình bày về quá trình quyết định Markov trong đó tập trung vào các khái niệm của quá trình Markov có số bước vô hạn và có số bước hữu hạn.

1.1 PHÁT BIỂU BÀI TOÁN

Bài toán quyết định Markov là bài toán học từ các tác động để đạt được mục đích. Người học và người ra quyết định được gọi là tác tử. Tất cả những gì mà chúng tương tác với, bao gồm mọi thứ bên ngoài tác tử được gọi là môi trường. Các tác động thực hiện một cách liên tục, tác tử lựa chọn các hành động, môi trường đáp ứng lại các hành động đó và chuyển từ trạng thái hiện thời sang trạng thái mới. Môi trường cũng đem lại các mục tiêu, các giá trị bằng số mà tác tử cố gắng cực đại hoá qua thời gian. Một đặc tả hoàn thiện về môi trường được coi là một "nhiệm vụ", một thực thể của bài toán quyết định Markov.

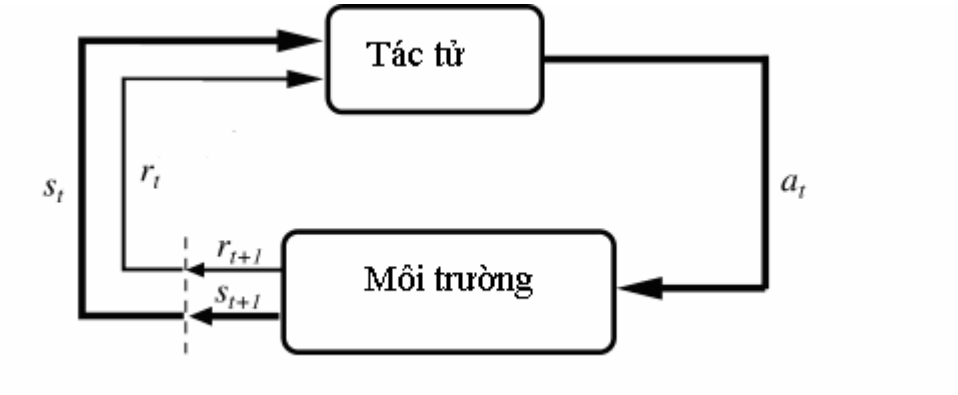
Tóm lại, bài toán quyết định Markov liên quan đến lớp bài toán trong đó một tác tử rút ra kết luận trong khi phân tích một chuỗi các hành động của nó cùng với tín hiệu vô hướng được đưa ra bởi môi trường.

Trong khái niệm chung này có thể thấy hai đặc tính quan trọng:

- Tác tử tương tác với môi trường và cặp "Tác tử + Môi trường" tạo thành một hệ thống động.

- Tín hiệu tăng cường, được nhận biết dựa vào mục tiêu, cho phép tác tử thay đổi hành vi của nó.

Lược đồ tương tác tác tử-môi trường như sau:



Hình 1.1: Mô hình tương tác giữa tác tử và môi trường

Trong lược đồ trên, tác tử và môi trường tác động lẫn nhau tại mỗi bước trong chuỗi các bước thời gian rời rạc, t = 0, 1, 2, 3, …Tại mỗi bước thời gian t, tác tử nhận một số biểu diễn về trạng thái của môi trường, $s_t \in S$, với S là tập các trạng thái có thể, và trên đó lựa chọn một hành động $a_t \in A(s_t)$, với $A(s_t)$ là tập các hành động có hiệu lực trong trạng thái s_t. Mỗi bước thời gian tiếp theo, tác tử nhận một giá trị tăng cường $r_{t+1} \in R$ và tự nó tìm ra một trạng thái mới s_{t+1}.

Tại mỗi bước tác tử thực hiện ánh xạ từ các trạng thái đến các hành động có thể lựa chọn. Phép ánh xạ này được gọi là chiến lược của tác tử, kí hiệu là π_t với $\pi_t(s,a)$ là xác suất thực hiện hành động $a_t=a$ khi $s_t=s$. Như vậy, bài toán quyết định Markov thực chất có thể được phát biểu như sau:

Biết	- Tập các trạng thái: S - Tập các hành động có thể: A - Tập các tín hiệu tăng cường (mục tiêu).
Bài toán	Tìm π:S→A sao cho R lớn nhất

Với mô hình bài toán quyết định Markov như trên, chúng ta có thể xem xét qua một số ví dụ quen thuộc.

Ví dụ 1: Máy bán hàng tự động

- Trạng thái: cấu hình các khe.
- Hành động: thời gian dừng lại.
- Mục tiêu: kiếm được nhiều tiền.
- Bài toán: tìm π:S$\rightarrow$A sao cho R lớn nhất.

Ví dụ 2: Tic-Tac-Toe

Đây là một trò chơi quen thuộc của giới trẻ. Hai người chơi thực hiện chơi trên một bảng kích thước 3x3. Một người ghi kí hiệu X và một người ghi kí hiệu O, đến tận khi có người thắng nhờ ghi 3 dấu trên cùng một hàng dọc hoặc hàng ngang hoặc hàng chéo, như người ghi dấu X trong hình vẽ:

X	O	O
O	X	X
		X

Nếu bảng bị lấp đầy mà không người chơi nào ghi được 3 dấu trong cùng một hàng thì trận đấu sẽ hoà. Bài toán tic-tac-toe được tiếp cận sử dụng RL như sau:

- Trạng thái: bảng 3x3.
- Hành động: phép di chuyển tiếp theo.
- Mục tiêu: 1 nếu thắng, -1 nếu thua, 0 nếu hoà.
- Bài toán: tìm π:S$\rightarrow$A sao cho R lớn nhất.

Ví dụ 3:Robot di động

- Trạng thái: vị trí của Robot và của người.
- Hành động: sự di chuyển.
- Mục tiêu: số các bước đối mặt thành công.

- Bài toán: tìm π:S$\rightarrow$A sao cho R lớn nhất.

Để hiểu rõ ràng về các bài toán trong thực tế, ở đây chúng ta xét ví dụ một cuộc đối thoại về mối quan hệ giữa tác tử và môi trường như sau:

Môi trường: Bạn đang ở trạng thái 65. Bạn có 4 hành động để lựa chọn.

Tác tử: Tôi lựa chọn hành động 2.

Môi trường: Bạn nhận được một giá trị tăng cường là 7 đơn vị.

Hiện tại bạn đang ở trạng thái 15.

Bạn có 2 hành động để lựa chọn.

Tác tử: Tôi lựa chọn hành động 1.

Môi trường: Bạn nhận được một giá trị tăng cường là -4 đơn vị.

Hiện tại bạn đang ở trạng thái 65.

Bạn có 4 hành động để lựa chọn.

Tác tử: Tôi lựa chọn hành động 2.

Môi trường: Bạn nhận được một giá trị tăng cường là 5 đơn vị.

Hiện tại bạn đang ở trạng thái 44.

Bạn có 5 hành động để lựa chọn.

1.2 CÁC PHẦN TỬ CỦA BÀI TOÁN QUYẾT ĐỊNH MARKOV

Dựa vào tác tử và môi trường, chúng ta có thể định nghĩa 4 phần tử con của một bài toán quyết định Markov: chiến lược (*policy*), hàm phản hồi (*reward function*), hàm giá trị (*value function*), và không bắt buộc, một mô hình về môi trường.

Chiến lược định nghĩa cách thức tác tử học từ hành động tại thời điểm đưa ra. Chiến lược là một ánh xạ từ tập các trạng thái của môi trường đến tập các hành động được thực hiện khi môi trường ở trong các trạng thái đó. Nó tương ứng với

tập các luật nhân quả trong lĩnh vực tâm lí học. Trong một số trường hợp, chiến lược có thể là một hàm đơn giản hoặc một bảng tra cứu, trong những trường hợp khác, nó có thể liên quan đến các tính toán mở rộng ví dụ như một tiến trình tìm kiếm. Chiến lược là nhân của một tác tử với nhận thức rằng một mình nó đủ quyết định hành động.

Hàm phản hồi định nghĩa mục tiêu trong bài toán quyết định Markov. Nó ánh xạ mỗi trạng thái quan sát được (hoặc một cặp hành động-trạng thái) của môi trường với một giá trị phản hồi để chỉ ra mong muốn thực chất về trạng thái đó. Mục đích duy nhất của tác tử là cực đại hoá tổng giá trị phản hồi nó nhận được trong suốt thời gian chạy. Hàm phản hồi định nghĩa sự kiện nào là tốt hay xấu cho tác tử. Trong một hệ thống thuộc lĩnh vực sinh vật học, không phù hợp để định nghĩa các giá trị phản hồi với niềm vui và sự đau đớn. Chúng là các đặc tính tức thì và được định nghĩa là các vấn đề mà tác tử cần đối mặt. Như thế, hàm phản hồi cần phải có khả năng thay đổi bởi tác tử. Tuy nhiên, nó có thể phục vụ dưới dạng một yếu tố cơ bản để thay đổi chiến lược. Ví dụ, nếu hành động lựa chọn bởi chiến lược được theo sau bởi một hàm phản hồi thấp, thì chiến lược có thể được thay đổi để lựa chọn hành động khác thay thế trong tương lai.

Trong khi một hàm phản hồi chỉ ra cái gì là tốt cho một ý thức tức thì, một hàm giá trị sẽ đặc tả cái gì là tốt trong suốt một giai đoạn thời gian. Nói cách khác, giá trị của một trạng thái là tổng số các hàm phản hồi một tác tử có thể kỳ vọng để tích luỹ trong tương lai, bắt đầu từ trạng thái đó. Trong khi các giá trị phản hồi quyết định mong muốn thực chất tức thì về các trạng thái môi trường, thì các hàm giá trị chỉ ra mong muốn trong cả quá trình về các trạng thái sau khi đưa vào bản miêu tả các trạng thái tiếp theo, và các mục tiêu hiệu quả trong các trạng thái đó. Ví dụ, một trạng thái có thể thường xuyên mang lại một hàm phản

hồi tức thì thấp, nhưng vẫn có một hàm giá trị cao, vì nó thường được theo sau bởi các trạng thái khác mà mang lại các giá trị phản hồi cao, hoặc ngược lại. Để tạo ra các mô hình tương tự con người, các giá trị phản hồi giống như là sự hài lòng (khi hàm phản hồi có giá trị lớn) và hình phạt (khi hàm phản hồi có giá trị thấp), trong khi các hàm giá trị tương ứng với một sự phán đoán tinh tế hơn và nhìn xa trông rộng hơn về việc chúng ta hài lòng hay không hài lòng như thế nào khi môi trường ở trong một trạng thái riêng biệt. Biểu diễn theo cách này, chúng ta kỳ vọng rằng các hàm giá trị rõ ràng là một ý tưởng khuôn mẫu thân thiện và căn bản.

Các hàm phản hồi là trong một ngữ cảnh chính, trong khi các hàm giá trị, như là các tiên đoán của các giá trị phản hồi, là nhân tố thứ hai. Không có các giá trị phản hồi thì sẽ không có các hàm giá trị. Mục đích duy nhất của việc ước lượng các hàm giá trị là để đạt được các giá trị phản hồi lớn hơn. Tuy nhiên, chính các hàm giá trị là đối tượng mà chúng ta đề cập đến nhiều nhất khi ra quyết định và đánh giá quyết định. Việc lựa chọn quyết định dựa trên sự phán đoán về hàm giá trị. Chúng ta tìm kiếm các hành động mà đem lại các trạng thái với giá trị lớn nhất, chứ không phải là các phản hồi lớn nhất, bởi vì các hành động này chứa số lượng phản hồi lớn nhất cho chúng ta trong cả giai đoạn. Trong ra quyết định và lập kế hoạch, con số được kế thừa được gọi là “giá trị” là một đối tượng mà chúng ta quan tâm nhiều nhất. Thật không may, việc xác định giá trị khó hơn nhiều so với xác định giá trị phản hồi. Các giá trị phản hồi về cơ bản được đưa ra trực tiếp bởi môi trường, nhưng các hàm giá trị cần phải được ước lượng và ước lượng lại từ chuỗi các quan sát tác tử có được qua toàn bộ thời gian sống của nó. Thực tế, thành phần quan trọng nhất của tất cả các thuật toán học tăng cường là một phương pháp để ước lượng các hàm giá trị một cách hiệu quả nhất. Vai trò

trung tâm của phép ước lượng hàm giá trị có thể xem là điều quan trọng nhất mà chúng ta học về phương pháp học tăng cường trong suốt các thập kỷ gần đây.

Mặc dù hầu hết các phương pháp học tăng cường được xem xét tuân theo cấu trúc xung quanh việc ước lượng các hàm giá trị, tuy nhiên đây cũng không phải là nhân tố bắt buộc để giải quyết được các bài toán quyết định Markov. Ví dụ, có thể sử dụng các phương pháp tìm kiếm như các thuật toán phát sinh, lập trình phát sinh, huấn luyện tái tạo và các phương pháp tối ưu hoá chức năng khác được sử dụng để giải quyết các bài toán quyết định Markov. Các phương pháp này tìm kiếm trực tiếp trong không gian các chiến lược mà không phải sử dụng các hàm giá trị. Chúng ta gọi đây là "các phương pháp tiến hoá" bởi vì hoạt động của chúng tương tự như cách mà phép tiến hoá sinh vật học tạo ra các sinh vật với các hành động có kỹ năng thậm chí khi chúng không học trong suốt chu kỳ sống cá thể của chúng. Nếu không gian các chiến lược là đủ nhỏ hoặc có thể định cấu trúc, nhờ đó các chiến lược tốt là phổ biến hoặc dễ tìm kiếm, thì các phương pháp "tiến hoá" có thể hiệu quả. Ngoài ra, các phương pháp "tiến hoá" có ưu điểm trong những bài toán ở đó tác tử học không thể phán đoán chính xác trạng thái của môi trường.

Tuy nhiên, những gì chúng ta đề cập đến phương pháp học tăng cường liên quan đến việc học trong quá trình tương tác với môi trường, do đó các phương pháp tiến hoá không thực hiện được. Chúng ta tin tưởng rằng các phương pháp có khả năng nắm bắt những ưu điểm trong tác động thuộc hành vi có thể hiệu quả hơn là các phương pháp tiến hoá trong nhiều tình huống. Các phương pháp tiến hoá bỏ qua rất nhiều cấu trúc có ích của bài toán quyết định Markov: chúng không sử dụng một thực tế rằng chiến lược mà chúng đang tìm kiếm là một hàm từ các trạng thái đến hành động., chúng không chú ý đến trạng thái nào cá thể

trải qua trong suốt chu kỳ sống hoặc hành động nào nó lựa chọn. Trong một số trường hợp, thông tin này có thể là sai lạc (ví dụ, khi các trạng thái không được quan sát), nhưng thường xuyên hơn, nó có thể cho phép tìm kiếm hiệu quả hơn. Mặc dù việc "học" và "tiến hoá" chia sẻ nhiều đặc tính và có thể kết hợp cùng với nhau, như chúng thực hiện trong tự nhiên, chúng ta không xem xét các phương pháp tiến hoá đặc biệt là trong các bài toán quyết định Markov. Một cách đơn giản trong tài liệu này khi chúng ta sử dụng thuật ngữ "học tăng cường", chúng ta không bao gồm các phương pháp tiến hoá.

Phần tử thứ 4 và cũng là phần tử cuối cùng của bài toán quyết định Markov đó là mô hình của môi trường. Đây là đối tượng để bắt chước hành vi của môi trường. Ví dụ, khi đưa ra một trạng thái và hành động, mô hình có thể dự đoán tổng hợp trạng thái tiếp theo và giá trị phản hồi tiếp theo. Các mô hình được sử dụng để lập kế hoạch, nhờ đó chúng ta dự định cho quyết định bất kỳ trên một tiến trình của hành động bằng cách xem xét các tình huống trong tương lai có thể xảy ra trước khi chúng có kinh nghiệm thực sự. Sự hợp nhất giữa các mô hình và kế hoạch trong các hệ thống học tăng cường là một phát triển mới. Các hệ thống học tăng cường ban đầu là những người học "thử và lỗi", với cách tiếp cận này những gì chúng thực hiện được xem như là đối lập với kế hoạch. Tuy nhiên, ngày càng rõ ràng rằng các phương pháp học tăng cường có liên quan gần gũi với các phương pháp quy hoạch động, trong đó cũng sử dụng các mô hình và chúng cũng lần lượt có liên quan gần gũi với các phương pháp lập kế hoạch không gian trạng thái. Các phương pháp học tăng cường hiện đại mở rộng sự phân bố từ học thử và lỗi mức thấp sang việc lập kế hoạch có tính thảo luận mức cao.

1.2.1 Hàm phản hồi

Mục đích của tác tử là cực đại hoá các mục tiêu được tích luỹ trong tương lai. Hàm phản hồi R(t) được biểu diễn dưới dạng hàm số đối với các mục tiêu. Trong các bài toán quyết định Markov, hàm phản hồi sử dụng biểu thức dạng tổng. Các nhà nghiên cứu đã tìm ra ba biểu diễn thường được sử dụng của hàm phản hồi:

Trong các bài toán số bước hữu hạn

Với những bài toán này ta có một số hữu hạn các bước trong tương lai. Sẽ tồn tại một trạng thái kết thúc và một chuỗi các hành động giữa trạng thái đầu tiên và trạng thái kết thúc được gọi là một giai đoạn.

Ta có:

$$R(t) = r_t + r_{t+1} + \ldots + r_{t+K-1}$$

Trong đó K là số các bước trước trạng thái kết thúc

Trong các bài toán số bước vô hạn

Với những bài toán này ta có chuỗi các hành động là vô hạn. Một hệ số suy giảm γ, 0≤γ≤1 được đưa ra và hàm phản hồi được biểu diễn dưới dạng tổng của các giá trị mục tiêu giảm dần:

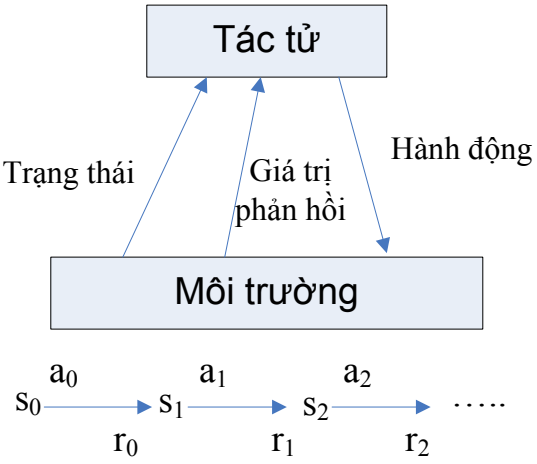
$$R(t) = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$$

Hệ số γ cho phép xác định mức độ ảnh hưởng của những bước chuyển trạng thái tiếp theo đến giá trị phản hồi tại thời điểm đang xét. Giá trị của γ cho phép điều chỉnh giai đoạn tác tử lấy các hàm tăng cường. Nếu γ = 0, thì tác tử chỉ xem xét mục tiêu gần nhất, giá trị γ càng gần với 1 thì tác tử sẽ quan tâm đến các mục tiêu xa hơn trong tương lai.

Như vậy, thực chất bài toán quyết định Markov trong trường hợp này chính là việc lựa chọn các hành động để làm cực đại biểu thức R:

$R = r_0+\gamma r_1+\gamma^2 r_2+\ldots$ với $0<\gamma<1$.

Như trong hình vẽ minh hoạ sau:



Hình 1.2: Mô hình tương tác giữa tác tử và môi trường trong bài toán có số bước vô hạn

Trong các bài toán số bước vô hạn mà hàm phản hồi không hội tụ

Trường hợp này xảy ra khi $\gamma = 1$. Giá trị trung bình của hàm phản hồi trên một bước thực hiện có thể hội tụ khi số bước tiến tới vô hạn. Trong trường hợp này hàm phản hồi được xác định bằng cách lấy trung bình của các giá trị tăng cường trong tương lai:

$$R(t) = \lim_{n\to\infty} \frac{1}{n} \sum_{k=0}^{n} r_{t+k}$$

1.2.2 Hàm giá trị

Trong mọi trạng thái s_t, một tác tử lựa chọn một hành động dựa theo một chiến lược điều khiển, π: $a_t = \pi(s_t)$. Hàm giá trị tại một trạng thái của hệ thống được tính bằng kỳ vọng toán học của hàm phản hồi theo thời gian. Hàm giá trị là hàm của trạng thái và xác định mức độ thích hợp của chiến lược điều khiển π đối

với tác tử khi hệ thống đang ở trạng thái s. Hàm giá trị của trạng thái s trong chiến lược π được tính như sau:

$V^{\pi}(s) = E_{\pi} \{R_t \mid s_t = s\}$

Bài toán tối ưu bao gồm việc xác định chiến lược điều khiển π^* sao cho hàm giá trị của trạng thái hệ thống đạt cực đại sau một số vô hạn hoặc hữu hạn các bước.

$\pi^* = \{\pi_0(s_0), \pi_1(s_1), \ldots, \pi_{N-1}(s_{N-1})\}$

Đối với bài toán có số bước vô hạn ta có hàm giá trị trạng thái:

$$V^{\pi}(s) = E_{\pi}\{R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s\}$$

Sử dụng các phép biến đổi:

$$\begin{aligned}
V^{\pi}(s) &= E_{\pi}\{R_t \mid s_t = s\} \\
&= E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \;\middle|\; s_t = s\right\} \\
&= E_{\pi}\left\{r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \;\middle|\; s_t = s\right\} \\
&= \sum_{a} \pi(s,a) \sum_{s'} \mathcal{P}^a_{ss'} \left[\mathcal{R}^a_{ss'} + \gamma E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \;\middle|\; s_{t+1} = s'\right\}\right] \\
&= \sum_{a} \pi(s,a) \sum_{s'} \mathcal{P}^a_{ss'} \left[\mathcal{R}^a_{ss'} + \gamma V^{\pi}(s')\right],
\end{aligned}$$

Như vậy, hàm $V^{\pi}(s)$ có thể được viết lại một cách đệ qui như sau:

$$V^{\pi}(s) = E_{\pi}\{r_{t+1} + \gamma V^{\pi}(s_{t+1}) \mid s_t = s\}$$

Hay:

$$V^{\pi}(s) = R(s, a) + \gamma \sum_{s' \in S} P_{ss'}^{a} V^{\pi}(s') \qquad (*)$$

Với $P_{ss'}^{a}$ là xác xuất để chuyển từ trạng thái s sang s' khi áp dụng hành động a.

Chúng ta có thể tính hàm $V^{\pi}(s)$ ngoại tuyến nếu biết trạng thái bắt đầu và xác suất mọi phép chuyển đổi theo mô hình. Vấn đề đặt ra là sau đó giải quyết hệ thống các phương trình tuyến tính trong công thức (*). Chúng ta biết rằng tồn tại một chiến lược tối ưu, kí hiệu π^{*}, được định nghĩa như sau:

$$V^{\pi^{*}}(s) \geq V^{\pi}(s)$$

$$\pi^{*} = \arg\max_{\pi} \{V^{\pi}(s)\}$$

Để đơn giản chúng ta viết $V^{*} = V^{\pi^{*}}$. Hàm giá trị tối ưu của một trạng thái tương ứng với chiến lược tối ưu là:

$$V^{\pi}(s) = \max_{\pi} \{V^{\pi}(s)\}$$

Đây là phương trình tối ưu Bellman (hoặc phương trình của quy hoạch động).

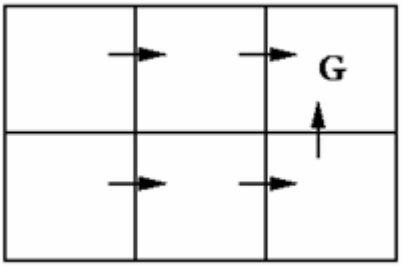
Tóm lại *V^{π} là hàm giá trị trạng thái cho chiến lược π.* Giá trị của trạng thái kết thúc thường bằng 0. Tương tự, định nghĩa $Q^{\pi}(s,a)$ là giá trị của việc thực hiện hành động a trong trạng thái s dưới chiến lược điều khiển π, được tính bằng kỳ vọng toán học của hàm phản hồi bắt đầu từ trạng thái s, thực hiện hành động a trong chiến lược π:

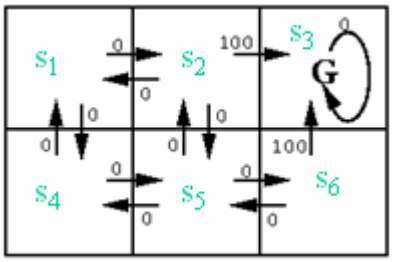
$$Q^{\pi}(s,a) = E_{\pi}\{R_t | s_t = s, a_t = a\} = E_{\pi}\left\{ \sum_{k=0}^{\infty} \gamma^{k} r_{t+k+1} \;\middle|\; s_t = s, a_t = a \right\}.$$

Q^{π} được gọi là hàm giá trị hành động cho chiến lược π. Và các hàm giá trị V^{π}, Q^{π} có thể được ước lượng từ kinh nghiệm.

Ví dụ minh họa cách tính toán các hàm giá trị

Chúng ta xét một ví dụ đơn giản để minh họa cho cách tính toán các hàm giá trị V và Q. Cho một lưới các ô vuông, mỗi ô vuông tương ứng với một trạng thái về môi trường. Ta có tập các trạng thái $\{s_1, s_2, s_3, s_4, s_5, s_6\}$ trong đó s_3 là trạng thái kết thúc. Tại mỗi ô, có 4 hành động có thể xảy ra đó là di chuyển lên trên, xuống dưới, sang trái, sang phải. Mỗi bước di chuyển đến trạng thái kết thúc có giá trị phản hồi 100, các bước di chuyển còn lại giá trị phản hồi đều bằng 0, minh họa như hình vẽ:





Ta có công thức tính V* cho π*:

$V^*(s_t) = r_t + \gamma V^*(s_{t+1})$

$V^*(s_6) = 100 + 0.9 * 0 = 100$

$V^*(s_5) = 0 + 0.9 * 100 = 90$

$V^*(s_4) = 0 + 0.9 * 90 = 81$

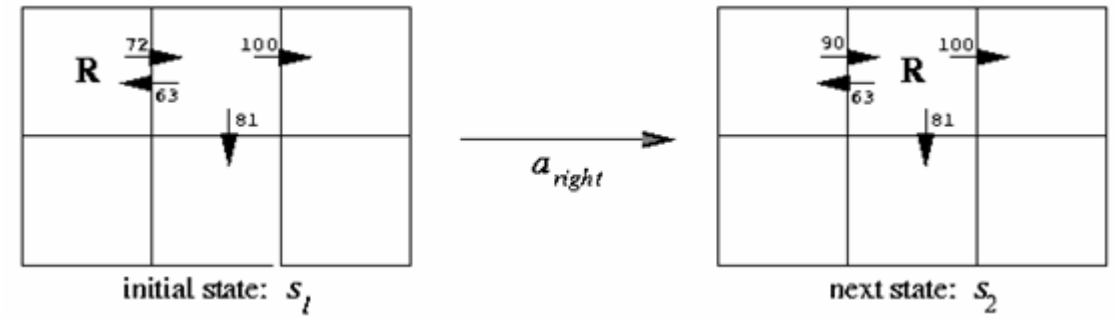
Tính V^α cho π^α như sau:

$V^\alpha(s_6) = 0.5 * (100 + 0.9 * 0) + 0.5 * (0 + 0.9 * 0) = 50$

$V^\alpha(s_5) = 0.66 * (0 + 0.9 * 50) + 0.33 * (0 + 0.9 * 0) = 30$

$V^\alpha(s_6) = 0.5 * (0 + 0.9 * 30) + 0.5 * (0 + 0.9 * 0) = 13.5$

Nếu tính cho tất cả các trạng thái thì bắt đầu lại và lặp đến tận khi giá trị hội tụ



Với hàm giá trị trạng thái-hành động Q, công thức tính như sau:

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

$Q(s_1, right) = r + \gamma \max_{a'} Q(s_2, a')$ (lấy $\gamma = 0.9$)

$= 0 + 0.9 \max\{63, 81, 100\}$

$= 90$

1.3 CẤU TRÚC TOÁN HỌC CỦA BÀI TOÁN QUYẾT ĐỊNH MARKOV

Trước hết chúng ta xem xét khái niệm "Thuộc tính Markov" được đưa ra trong các bài toán quyết định. Trong bài toán quyết định, tác tử ra quyết định do một tín hiệu từ môi trường gọi là trạng thái của môi trường. Chúng ta định nghĩa thuộc tính môi trường và các tính hiệu trạng thái của chúng là thuộc tính Markov.

Trạng thái được hiểu là bất cứ thông tin gì có ích với tác tử, giả thiết trạng thái được đưa ra bởi một số hệ thống tiền xử lý của môi trường. Chúng ta sẽ định nghĩa thuộc tính Markov cho bài toán quyết định. Để đơn giản biểu thức toán học, chúng ta giả sử tập các trạng thái và các mục tiêu là hữu hạn. Quan sát cách thức một môi trường tổng quát có thể đáp ứng tại thời điểm t+1 đối với hành động được thực hiện tại thời điểm t. Trong hầu hết các trường hợp, nguyên nhân của sự đáp ứng này có thể phụ thuộc vào mọi thứ đã xảy ra trước đó. Khi đó biến

động của môi trường có thể được định nghĩa bằng cách đặc tả xác suất phân bố khả năng như sau:

$Pr\{s_{t+1} = s', r_{t+1} = r \mid s_t, a_t, r_t, s_{t-1}, a_{t-1}, \ldots, r_1, s_0, a_0\}$; với mọi s', r và mọi giá trị có thể của các sự kiện trước s_t, a_t, r_t, …, r_1, s_0, a_0.

Nếu tín hiệu trạng thái có thuộc tính Markov thì đáp ứng của môi trường tại thời điểm t+1 chỉ phụ thuộc vào trạng thái và hành động tại thời điểm t, trong trường hợp này, biến động của môi trường được thể hiện như sau:

$Pr\{s_{t+1} = s', r_{t+1} = r \mid s_t, a_t\}$; với mọi s', r, s_t, a_t.

Nói cách khác, một tín hiệu trạng thái có thuộc tính Markov và là một trạng thái Markov khi và chỉ khi giá trị ở hai biểu thức trên bằng nhau với mọi s', r và s_t, a_t, r_t, …, r_1, s_0, a_0. Trong trường hợp này môi trường cũng được gọi là có thuộc tính Markov.

Nếu một môi trường có thuộc tính Markov thì biến động tại mỗi bước của nó sẽ cho phép dự đoán trạng thái và mục tiêu kỳ vọng tiếp được đưa ra từ trạng thái và hành động hiện tại. Bằng cách lặp phương trình này, chúng ta có thể dự đoán tất cả các trạng thái và mục tiêu kỳ vọng trong tương lai mà chỉ với kiến thức từ trạng thái hiện tại trong thời điểm hiện tại. Các trạng thái Markov cung cấp khả năng tốt nhất cho việc lựa chọn hành động, khi đó chiến lược tốt nhất cho việc lựa chọn hành động sẽ là hàm của một trạng thái Markov.

Nhiều trường hợp trong học tăng cường khi tín hiệu trạng thái không có thuộc tính Markov, chúng ta cũng sẽ xấp xỉ trạng thái này thành trạng thái Markov vì chúng ta luôn mong muốn trạng thái là tốt để dự đoán hàm mục tiêu cũng như việc lựa chọn hành động trong tương lai. Với tất cả những lý do đó, cách tốt nhất là xem trạng thái tại mỗi bước thời gian như là một xấp xỉ của trạng thái Markov mặc dù nó không hoàn toàn thoả mãn thuộc tính Markov.

Thuộc tính Markov là rất quan trọng trong các bài toán quyết định vì các quyết định và các giá trị được giả thiết chỉ là hàm phụ thuộc vào trạng thái hiện tại. Giả thiết này không có nghĩa là áp dụng hoàn toàn cho mọi tình huống học tăng cường kể cả những tình huống không thoả mãn Markov. Tuy nhiên lý thuyết phát triển cho các thuộc tính Markov vẫn giúp chúng ta có thể hiểu được hành vi của các giải thuật học tăng cường và các giải thuật thì vẫn có thể áp dụng thành công cho mọi nhiệm vụ với các trạng thái không thoả mãn Markov. Kiến thức về lý thuyết Markov là cơ sở nền tảng để mở rộng trong những trường hợp phức tạp hơn kể cả những trường hợp không thoả mãn thuộc tính Markov.

Với giả thiết như vậy, tương tác giữa tác tử và môi trường có thể được mô hình dưới dạng bài toán quyết định Markov. Việc tìm kiếm sách lược điều khiển tối ưu trong các bài toán quyết định Markov tương ứng với những tiêu chí tối ưu khác nhau dẫn tới việc xây dựng các phương trình tối ưu Bellman và các thuật toán quy hoạch động. Thông thường, quy hoạch động là phương pháp giải các phương trình tối ưu Bellman khi biết các thuộc tính thống kê của môi trường. Khác với quy hoạch động, phương pháp học tăng cường tìm kiếm trực tiếp các chiến lược quyết định tối ưu từ các giá trị phản hồi thu nhận được trong các quá trình tương tác với môi trường và trạng thái của môi trường.

Bài toán quyết định Markov bao gồm một tập các trạng thái ($s_1,s_2,\ldots,s_n$) và một tập các hành động ($a_1,a_2,\ldots,a_n$). Mỗi trạng thái có một giá trị mục tiêu ($r_1,r_2,\ldots,r_n$). Trong bài toán quyết định Markov, các phép chuyển đổi từ trạng thái i sang trạng thái j chỉ phụ thuộc vào các hành động có thể tại trạng thái i. Hàm đo khả năng chuyển đổi hay còn gọi là xác suất của phép chuyển đổi được biểu diễn như sau:

$$P^k_ij = (\text{tiếp theo} = s_j \mid \text{hiện tại} = s_i \text{ và thực hiện hành động } a_k)$$

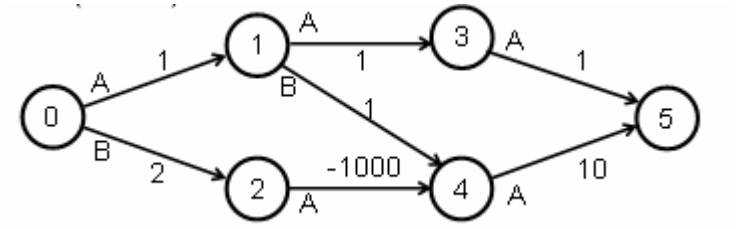
Tại mỗi bước, hệ thống sẽ thực hiện các công việc như sau:

- 0) Giả sử trạng thái hiện tại là s_i
- 1) Giá trị phản hồi r_i
- 2) Lựa chọn hành động a_k
- 3) Chuyển đến trạng thái s_j với khả năng P_{ij}^k
- 4) Tất cả các giá trị phản hồi trong tương lai được biểu diễn theo hệ số suy giảm γ

Mục tiêu của bài toán quyết định Markov là với mọi trạng thái bắt đầu, tìm ra một chiến lược tốt nhất (một chuỗi các hành động) để cực đại hoá giá trị phản hồi. Để hiểu rõ cách tính toán hàm giá trị V và hàm giá trị trạng thái Q ta xét một số ví dụ bài toán Markov sau đây:

Ví dụ 1:

Xét ví dụ một bài toán quyết định Markov có mô hình:



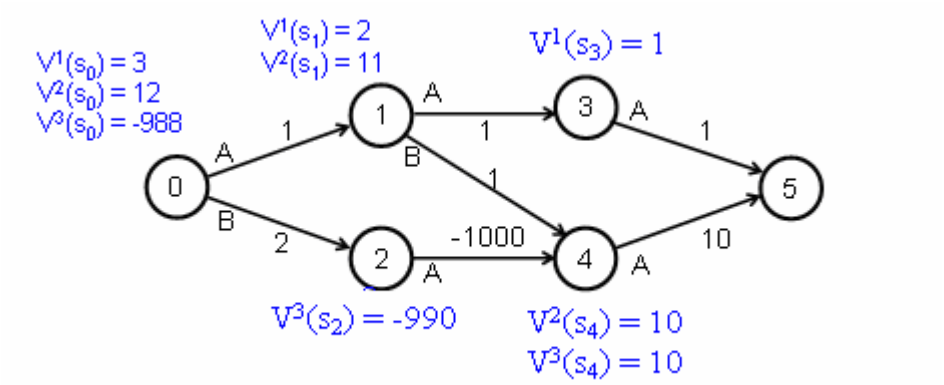
Trong bài toán này, tập các trạng thái bao gồm {0, 1, 2, 3, 4, 5} trong đó 0 là trạng thái bắt đầu, 5 là trạng thái kết thúc. Mỗi bước chuyển trạng thái được biểu diễn bằng một mũi tên và giá trị phản hồi (tăng cường) của nó được biểu hiện bằng trọng số trên ghi trên mũi tên tương ứng. Tập {A, B} là tập các hành động có thể thực hiện. Chúng ta có thể thấy có 3 chiến lược cho bài toán này.

1. $0 \rightarrow 1 \rightarrow 3 \rightarrow 5$
2. $0 \rightarrow 1 \rightarrow 4 \rightarrow 5$
3. $0 \rightarrow 2 \rightarrow 4 \rightarrow 5$

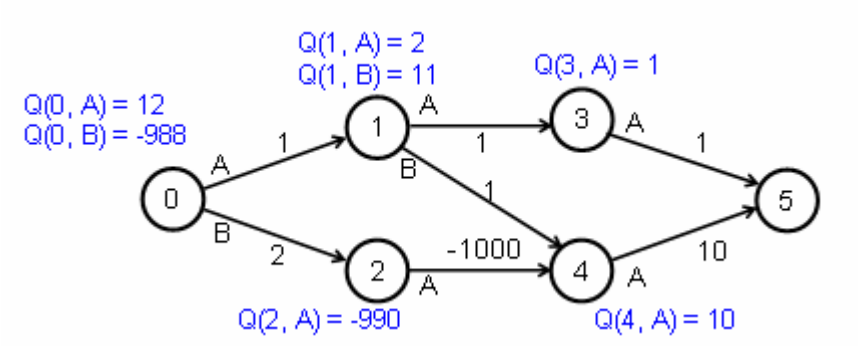
So sánh các chiến lược, chúng ta sắp xếp các chiến lược theo tổng giá trị phản hồi mà nó thu được:

1. $0 \rightarrow 1 \rightarrow 3 \rightarrow 5 = 1 + 1 + 1 = 3$
2. $0 \rightarrow 1 \rightarrow 4 \rightarrow 5 = 1 + 1 + 10 = 12$
3. $0 \rightarrow 2 \rightarrow 4 \rightarrow 5 = 2 + (-1000) + (10) = -988$

Chúng ta có thể kết hợp một giá trị với mỗi trạng thái. Với một chiến lược cố định, hàm giá trị trạng thái V xác định mức độ thích hợp của việc thực hiện chiến lược π đối với trạng thái s. Hình vẽ sau đây chỉ ra chiến lược cần thực hiện tại mỗi trạng thái.

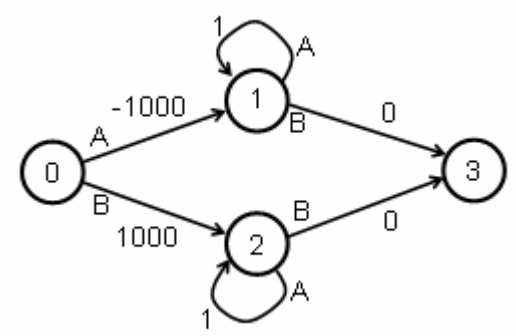


Chúng ta cũng có thể định nghĩa giá trị mà không cần đặc tả chiến lược bằng cách định nghĩa giá trị của việc lựa chọn hành động a từ trạng thái s và sau đó thực hiện tối ưu, đây chính là hàm giá trị trạng thái- hành động Q. Hình vẽ sau đây chỉ ra hành động cần thực hiện tại mỗi trạng thái.



Ví dụ 2:

Xét một ví dụ khác, bài toán quyết định Markov có mô hình như sau:



Trong bài toán này, tập các trạng thái bao gồm {0, 1, 2, 3} trong đó 0 là trạng thái bắt đầu, 3 là trạng thái kết thúc. Mỗi bước chuyển trạng thái được biểu diễn bằng một mũi tên và giá trị phản hồi (tăng cường) của nó được biểu hiện bằng trọng số trên ghi trên mũi tên tương ứng. Tập {A, B} là tập các hành động có thể thực hiện. Quan sát bước đi tại lặp lại tại trạng thái 1 và 2 có thể thấy:

- Số các bước của bài toán là không giới hạn vì phép lặp.
- Giá trị của trạng thái 1 và 2 là không giới hạn cho một số chiến lược.

$Q(1, A) = 1 + Q(1, A)$

$= 1 + 1 + Q(1, A)$

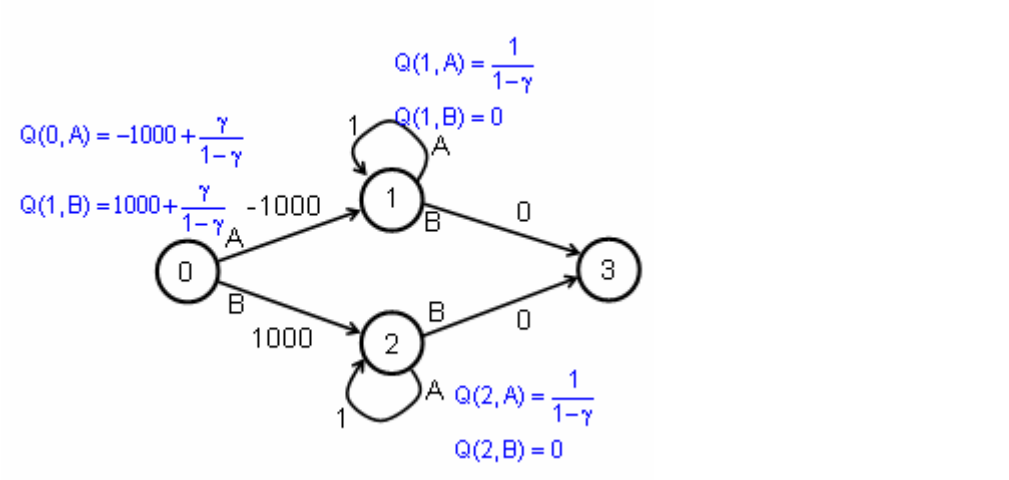
$= ! + 1 + 1 + Q(1, A)$

$=\ldots..$

Trong những bài toán số bước vô hạn như trường hợp này, như đã trình bày trong mục 4.2, người ta đưa thêm hệ số suy giảm γ vào khi tính hàm phản hồi. $0 \le \gamma \le 1$. Ta có công thức tính hàm giá trị và hàm giá trị trạng thái-hành động

$$V^{\pi}(s) = R(s, \pi(s), s') + \gamma V^{\pi}(s')$$
$$Q(s, a) = R(s, a, s') + \gamma \max_{a'} Q(s', a')$$

Khi đó, tính toán giá trị Q cho từng cặp trạng thái-hành động như sau:



Nhìn từ kết quả trên ta có các chiến lược tối ưu: π(0) = B; π(1) = A; π(2) = A.

1.4 PHƯƠNG PHÁP HỌC TĂNG CƯỜNG

1.4.1 Ý tưởng chung

Có hai phương pháp thường được sử dụng để giải các bài toán quyết định đó là tìm kiếm trong không gian chiến lược và tìm kiếm trong không gian hàm giá trị hay còn gọi là "phép lặp chiến lược" và "phép lặp giá trị". Hai phương pháp này chính là các giải thuật học tăng cường đặc trưng. Ngoài ra còn xuất hiện một phương pháp lai giữa hai phương pháp trên: Actor-Critic learning.

Cơ chế chung của phép lặp chiến lược và phép lặp giá trị như sau:

- **Phép lặp chiến lược**

Ý tưởng là ở chỗ, bắt đầu từ một chiến lược bất kỳ π và cải thiện nó sử dụng V^{π} để có một chiến lược tốt hơn π'. Chúng ta sau đó có thể tính $V^{\pi'}$ và cải thiện nó với một chiến lược tốt hơn nữa π'',…Kết quả của tiến trình lặp này, chúng ta có thể đạt được một chuỗi các bước cải thiện chiến lược và các hàm giá trị.

Thuật toán lặp chiến lược:

(a) Bắt đầu với một chiến lược bất kỳ π.

(b) Lặp

Đánh giá chiến lược π.

Cải tiến chiến lược tại mỗi trạng thái.

Đến tận khi chiến lược không có khả năng thay đổi.

Trong thuật toán lặp chiến lược ở trên có đề cập đến một số khái niệm liên quan đó là *đánh giá chiến lược* và *cải tiến chiến lược*.

Đánh giá chiến lược

Chính là quá trình tính toán hàm giá trị trạng thái V^{π} cho một chiến lược π bất kỳ. Nó được biết đến là phương trình Bellman:

$$V^{\pi}(s) = \sum_{a} \pi(s,a) \sum_{s'} P^{a}_{ss'} \left[R^{a}_{ss'} + \gamma V^{\pi}(s') \right].$$

Đây là một hệ thống các phương trình tuyến tính đồng thời. Lời giải của nó không quá phức tạp và có thể tìm được bằng cách sử dụng một trong các phương pháp giải hệ thống các phương trình tuyến tính. Lời giải có thể tìm được bằng việc tạo ra một chuỗi các hàm giá trị xấp xỉ $V_0, V_1, V_2, \ldots$

Xấp xỉ khởi tạo V_0 được chọn ngẫu nhiên. Nếu có một trạng thái kết thúc nó sẽ nhận giá trị 0. Mỗi xấp xỉ thành công đạt được bằng cách sử dụng phương trình Bellman cho V^{π} như là một luật cập nhật:

$$V_{k+1}(s) = \sum_a \pi(s,a) \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V_k(s') \right].$$

Bước lặp kết thúc khi độ lệch cực đại giữa hai hàm giá trị thành công nhỏ hơn một giá trị đủ nhỏ ε.

Cải tiến chiến lược

Chính là quá trình tạo một chiến lược mới cải tiến dựa trên chiến lược gốc bằng cách sử dụng thuật toán tham lam đối với hàm giá trị của chiến lược gốc. Với một chiến lược π cho trước, có thể tìm ra một chiến lược tốt hơn π' sao cho $V^{\pi'} > V^{\pi}$. Điều này đạt được bằng cách chọn theo tiên đoán một hành động tại một trạng thái riêng biệt hoặc bằng cách xem xét sự thay đổi tại tất cả các trạng thái và đối với tất cả các hành động có thể, lựa chọn tại mỗi trạng thái hành động xuất hiện tốt nhất dựa theo $Q^{\pi}(s,a)$. Chiến lược π' là tham lam nếu:

$$\pi'(s) = \arg\max_a \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V^{\pi}(s') \right].$$

Trong phương trình trên, *arg max*$_a$ chỉ ra giá trị của a tại đó biểu thức đạt cực đại. Chiến lược tham lam thực hiện hành động tốt nhất sau mỗi bước dựa theo V^{π}.

Tóm lại, trong phép lặp chiến lược, giá trị của chiến lược chính là kết quả của hệ thống các phương trình tuyến tính. Sau đó, với mọi trạng thái, chúng ta sẽ quan sát liệu rằng có thể cải thiện chiến lược trong khi chỉ thay đổi hành động bắt đầu hay không. Phép lặp chiến lược là nhanh khi không gian hành động là nhỏ và đôi khi chỉ cần vài bước lặp là đủ, mặt khác phương pháp này là khá đắt thậm chí khó thực hiện trong trường hợp không gian hành động lớn.

- **Phép lặp giá trị**

Trong phương pháp này, chúng ta không cố gắng quyết định chiến lược một cách rõ ràng, mà sẽ quyết định hành động có giá trị tối ưu cho mọi trạng thái. Thuật toán lặp giá trị sinh ra từ dạng đệ qui của hàm giá trị trạng thái tối ưu Bellman. Phương trình chi phối thuật toán lặp giá trị như sau:

$$V_{k+1}(s) = \max_a \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V_k(s') \right].$$

Người ta đã chứng minh được rằng giải thuật này hội tụ tại một số hữu hạn các bước lặp để đạt tới đích là chiến lược tối ưu, chuỗi $\{V_k\}$ hội tụ đến giá trị trạng thái tối ưu V^*. Phép lặp giá trị kết hợp một cách hiệu quả cả việc đánh giá chiến lược và cải thiện chiến lược.

Thuật toán lặp giá trị

(a) Khởi tạo V ngẫu nhiên cho mọi trạng thái

(b) Lặp

Với mỗi trạng thái s:

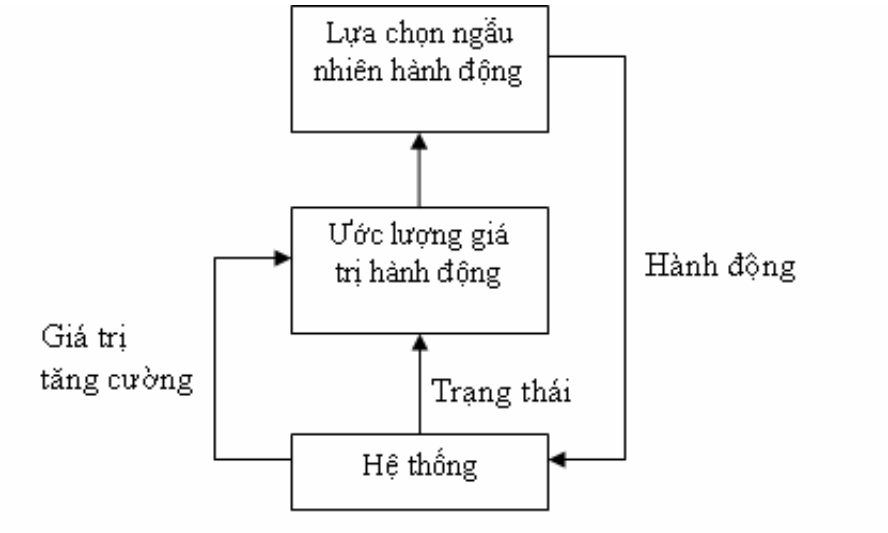
$$V(s) \leftarrow \max_a \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V(s') \right]$$

Đến tận khi Độ lệch cực đại giữa hai hàm giá trị thành công nhỏ hơn một giá trị đủ nhỏ ε

(c) *Đầu ra*: Một chiến lược π sao cho

$$\pi(s) = \arg\max_a \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V(s') \right)$$

Kiến trúc của các thuật toán dựa trên lặp giá trị được biểu diễn trong hình sau:



Hình 1.3: Kiến trúc của thuật toán lặp giá trị

1.4.2 Một số thuật ngữ

1.4.2.1 Khảo sát và khai thác

Trong phương pháp học tăng cường, đặc biệt là với các bài toán quyết định trực tiếp, có một vấn đề về khảo sát và khai thác. Một mặt tác tử muốn khảo sát môi trường để tìm ra bài toán tối ưu, mặt khác cực tiểu hoá chi phí cho việc học bằng cách khai thác môi trường.

Có một số phương pháp cân bằng giữa khảo sát và khai thác. Kỹ thuật phổ biến nhất là sử dụng một trong các chiến lược lựa chọn hành động ε-soft, ε-greedy và softmax.

1.4.2.2 Kỹ thuật ε-greedy, ε-soft và softmax

Chiến lược lựa chọn hành động ε-greedy

Đây là cách đơn giản và phổ biến nhất để cân bằng giữa khảo sát và khai thác. Trong phương pháp này, hành động có ước lượng về giá trị phản hồi lớn nhất sẽ được lựa chọn trong hầu hết thời gian, gọi là hành động tham lam. Nhưng bất cứ

khi nào với khả năng rất nhỏ ε, hành động được lựa chọn ngẫu nhiên, giống nhau và độc lập với các ước lượng về giá trị hành động.

Trong hầu hết các trường hợp với khả năng của hành động là 1-ε thì giá trị hành động được ước lượng lớn nhất Q(s,a) được lựa chọn.

Giả sử A là tập tất cả các hành động và N là số hành động. Giả sử thêm nữa P_g^a là khả năng lựa chọn một hành động tham lam a, và P_{ng}^a là khả năng lựa chọn một hành động không tham lam a. Trong phương pháp lựa chọn hành động ε-greedy, khả năng lựa chọn một hành động không tham lam được cho bởi công thức:

$$P_{ng}^a = \frac{\epsilon}{N}$$

Từ đó dễ dàng chỉ ra rằng khả năng lựa chọn một hành động tham lam:

$$P_g^a = 1 - \epsilon + \frac{\epsilon}{N} = 1 - \epsilon + P_{ng}^a$$

Phương pháp này chỉ ra rằng nếu phép thử là đủ, mỗi hành động sẽ được thử một số vô hạn các lần thì đảm bảo rằng sẽ tìm ra được các hành động tối ưu.

Chiến lược lựa chọn hành động ε-soft

Tương tự như phương pháp ε-greedy, hành động tốt nhất được lựa chọn với khả năng 1-ε và trong các trường hợp khác thực hiện lựa chọn hành động một cách ngẫu nhiên giống nhau.

Chiến lược lựa chọn hành động softmax

Kỹ thuật ε-greedy và ε-soft có hạn chế là trong một số tình huống chúng lựa chọn các hành động ngẫu nhiên giống nhau, như vậy hành động có khả năng tồi nhất có thể được lựa chọn như là hành động tốt thứ hai. Kỹ thuật softmax khắc phục nhược điểm này bằng cách gán thứ hạng hoặc trọng số cho mỗi hành động,

như vậy các hành động tồi nhất sẽ chắc chắn không được chọn. Như vậy trong kỹ thuật này, hành động tham lam vẫn đem lại khả năng lựa chọn cao nhất. Tất cả các hành động khác được phân hạng và định lượng phụ thuộc vào giá trị ước lượng của nó. Phép phân bố Boltzmann được sử dụng để tính toán khả năng lựa chọn hành động.

Cho A là tập tất cả các hành động. Khả năng thực hiện một hành động aЄA được cho bởi phương trình sau:

$$P(a) = \frac{e^{Q(s,a)/\tau}}{\sum_{a' \in A} e^{Q(s,a')/\tau}}.$$

Tham số τ được gọi là "nhiệt độ" và luôn dương. Nhiệt độ cao gây ra hành động có xác suất ngang nhau. Nhiệt độ thấp gây ra sự khác nhau lớn hơn trong khả năng lựa chọn hành động chính là sự khác nhau trong các ước lượng giá trị của chúng.

1.4.2.3 Khái niệm học on-policy và off-policy

Học On-policy

Đây là phương pháp học dựa trên giá trị của chiến lược được sử dụng để ra quyết định. Các hàm giá trị được cập nhật sử dụng các kết quả từ việc thực hiện các hành động được quyết định bởi một số chiến lược. Các chiến lược này thường xuyên là "soft" và không mặc định, "soft" ở đây có nghĩa là nó đảm bảo luôn luôn có một phần tử thăm dò đối với chiến lược. Chiến lược không quá nghiêm khắc mà nó thường lựa chọn các hành động đưa ra giá trị tăng cường tốt nhất. Có 3 chiến lược thông thường được sử dụng ε-soft, ε-greedy hoặc softmax như đã trình bày ở trên.

Học Off-policy

Đây là phương pháp học các chiến lược khác nhau cho hành vi và ước lượng. Có thể cập nhật các hàm giá trị ước lượng sử dụng các hành động giả thiết mà không cần phải thử trong thực tế. Điều này đối lập với chiến lược trên ở chỗ cập nhật các hàm giá trị chỉ dựa trên kinh nghiệm.

1.4.3 Phân loại thuật toán học tăng cường

Các thuật toán học tăng cường được chia thành hai loại chính đó là: học dựa trên mô hình (model based) và học không có mô hình (model free). Đại điện cho kiểu học dựa trên mô hình phải kể đến phương pháp quy hoạch động (Dynamic Programming-DP), đại diện cho kiểu học không có mô hình là phương pháp Monte Carlo và phương pháp TD (Temporal Difference).

1.4.3.1 Học dựa trên mô hình

Phương pháp này thực hiện học theo mô hình và sử dụng nó để quyết định chính sách tối ưu. Tác tử ước lượng mô hình từ các quan sát về cả khả năng chuyển đổi trạng thái và hàm tăng cường. Sau đó sẽ sử dụng mô hình ước lượng này như là mô hình thực tế để tìm ra chiến lược tối ưu .

Một cách cụ thể, tác tử tiến hành lập kế hoạch và biên dịch kết quả sang một tập các phản hồi nhanh hoặc các luật tình huống – phản hồi, sau đó sẽ được sử dụng trong quyết định thời gian thực. Cách tiếp cận này tuy nhiên bị giới hạn vào sự phục thuộc của nó vào một mô hình hoàn thiện về môi trường.

1.4.3.2 Học không có mô hình

Phương pháp này tìm thấy chính sách tối ưu mà không phải học theo mô hình. Tác tử học các giá trị hành động mà không có mô hình về môi trường được mô tả bởi $P^a_{ss'}$ và $R^a_{ss'}$. Trong phương pháp này tác tử tương tác trực tiếp với môi

trường và biên dịch thông tin nó thu thập được thành một cấu trúc phản hồi mà không có học từ mô hình. Trong phương pháp này, các bước chuyển đổi trạng thái và các giá trị phản hồi tác tử quan sát thay thế cho mô hình môi trường.

Một trong các khó khăn lớn nhất gặp phải đó là làm cách nào để tính toán được mối liên kết giữa hành động hiện tại và các hệ quả trong tương lai. Để giải quyết khó khăn này có hai cách tiếp cận: thứ nhất là đợi đến khi kết thúc và thực hiện thưởng/phạt mọi hành động được thực hiện trong quá khứ, dựa trên kết quả cuối cùng. Trong đó phương pháp Monte Carlo là một ví dụ. Vấn đề hạn chế trong cách tiếp cận này đã được Kaelbling và các cộng sự chỉ ra vào năm 1996, đó là khó khăn trong việc nhận biết khi nào kết thúc trong chuỗi liên tiếp các sự việc đang xảy ra. Thậm chí nếu biết được nó thì cũng yêu cầu một lượng lớn về bộ nhớ.

Cách tiếp cận khác là phương pháp TD được đưa ra bởi Sutton vào năm 1988. Trong phương pháp này, một mạng đặc biệt được điều chỉnh để học kết hợp các giá trị tăng cường cục bộ với các trạng thái tức thì giữa hành động và giá trị tăng cường bên ngoài. Ý tưởng quan trọng của phương pháp này là giá trị tăng cường cục bộ của một trạng thái tức thì hồi quy về giá trị tăng cường thành công.

Sau đây chúng ta sẽ đi tìm hiểu một số giải thuật RL điển hình với những đặc điểm riêng, bao gồm phương pháp quy hoạch động, phương pháp Monte Carlo và phương pháp TD. Với phương pháp quy hoạch động, nó đòi hỏi một mô hình hoàn hảo về môi trường, điều này không phù hợp trong những tình huống học của robot trong thực tế nên thường được dùng trong lý thuyết trò chơi, toán học,…Phương pháp Monte Carlo không đòi hỏi mô hình về môi trường và không cần có cơ chế tự cập nhật mà bắt đầu từ việc thăm dò. Phương pháp TD

cũng không đòi hỏi mô hình môi trường nhưng có cơ chế tự mồi nghĩa là chiến lược sẽ được cập nhật tại mỗi bước thời gian thay vì mỗi giai đoạn.

Chúng ta đã trình bày các vấn đề chính trong phương pháp học tăng cường bao gồm mô hình bài toán, các phần tử cấu thành và các loại thuật toán học tăng cường. Phần cuối chương này, đề tài xin giới thiệu sơ lược một số thông tin về lịch sử phát triển cũng như lĩnh vực ứng dụng của phương pháp học tăng cường.

1.4.4 Lịch sử phát triển và các lĩnh vực ứng dụng

"Học tăng cường" thực chất là một loại giải thuật được áp dụng trong "Học máy"- *machine learning*. Chúng ta biết đến học máy là một lĩnh vực của trí tuệ nhân tạo liên quan đến việc phát triển các kĩ thuật cho phép các máy tính có thể "học". Cụ thể hơn, học máy là một phương pháp để tạo ra các chương trình máy tính bằng việc phân tích các tập dữ liệu.

Cho trước một *bài toán* cụ thể để giải quyết, và một *lớp* các hàm *F*, việc học có nghĩa là sử dụng một tập các quan sát để tìm hàm $f^* \in F$ giải được bài toán một cách tốt nhất. Việc đó đòi hỏi định nghĩa một hàm chi phí $C : F \to \mathbb{R}$ sao cho, với lời giải tối ưu f^*, $C(f^*) \leq C(f) \forall f \in F$. Hàm chi phí *C* là một khái niệm quan trọng trong học máy, do nó là một phép đo khoảng cách tới lời giải tối ưu cho bài toán cần giải quyết.

Các thuật toán học tìm kiếm trong không gian lời giải để được một hàm có chi phí nhỏ nhất có thể. Chúng được phân loại theo kết quả mong muốn của thuật toán. Có ba kiểu học chính, đó là *học có giám sát, học không có giám sát* và *học tăng cường*.

Trong học có giám sát, ta được cho trước một tập ví dụ gồm các cặp $(x, y), x \in X, y \in Y$ và mục tiêu là tìm một hàm *f* (trong lớp các hàm được

phép) khớp với các ví dụ. Nói cách khác, ta muốn tìm ánh xạ mà dữ liệu đầu vào đã hàm ý, với hàm chi phí đo độ không khớp giữa ánh xạ của ta và dữ liệu.

Trong học không có giám sát, ta được cho trước một số dữ liệu x, và hàm chi phí cần được cực tiểu hóa có thể là một hàm bất kỳ của dữ liệu x và đầu ra, f. Hàm chi phí được quyết định bởi phát biểu của bài toán. Phần lớn ứng dụng nằm trong vùng các bài toán ước lượng như mô hình hóa thống kê, nén, lọc,…

Trong học tăng cường, dữ liệu x thường không được cho trước mà được tạo ra trong quá trình một tác tử tương tác với môi trường. Tại mỗi thời điểm t, tác tử thực hiện hành động y_t và môi trường tạo một quan sát x_t và một chi phí tức thời c_t, theo một quy trình động nào đó (thường là không được biết). Mục tiêu là tìm một chiến lược lựa chọn hành động để cực tiểu hóa một chi phí dài hạn nào đó, nghĩa là chi phí tích lũy mong đợi. Quy trình động của môi trường và chi phí dài hạn cho mỗi sách lược thường không được biết, nhưng có thể ước lượng được. Các bài toán thường được giải quyết bằng học tăng cường là các bài toán điều khiển, trò chơi và các nhiệm vụ quyết định tuần tự khác.

Ý tưởng học qua tác động với môi trường xuất hiện lần đầu tiên khi chúng ta nghĩ đến thế giới tự nhiên. Khi một đứa bé chơi, vẫy tay, hoặc nhìn mọi vật, nó không có một người dạy trực tiếp nào cả, nhưng nó có một mối quan hệ trực tiếp giữa cảm nhận và vận động đối với môi trường. Sự tập luyện dựa trên mối quan hệ này sẽ sản xuất ra một lượng thông tin giàu có về nguyên nhân và ảnh hưởng, về các hệ quả của hành động, và về việc "Phải làm gì ?" để đạt được các mục đích. Trong toàn bộ cuộc sống của chúng ta, các tác động lẫn nhau như vậy rõ ràng là một nguồn tài nguyên chính của nhận thức về môi trường của mỗi người. Chẳng hạn việc chúng ta học lái một chiếc xe hoặc thực hiện một cuộc hội thoại nghĩa là chúng ta đã nhận thức sâu sắc về cách thức mà môi trường phản ứng lại

với những gì mà chúng ta làm, và chúng ta tìm kiếm sự tác động đến những gì xảy ra qua hành động của chúng ta. Học từ tác động qua lại là một ý tưởng cơ bản dựa trên hầu hết các lý thuyết của học và trí tuệ nhân tạo.

Lịch sử phát triển của RL chia thành hai hướng chính, một hướng quan tâm đến việc học với phương pháp thử và sai, bắt đầu trong lĩnh vực tâm lý học nghiên cứu việc học của động vật. Hướng này xem xét các công việc sơ khai trong trí tuệ nhân tạo và phát triển thời kỳ phục hưng của RL vào đầu những năm 1980. Hướng thứ hai quan tâm đến các bài toán về điều khiển tối ưu và cách giải quyết là sử dụng các hàm giá trị và quy hoạch động. Các ngoại lệ xoay quanh một hướng thứ 3 sử dụng các phương pháp chênh lệch về thời gian (TD). Tất cả các hướng nghiên cứu hợp nhất lại vào cuối những năm 1980, tạo ra một lĩnh vực hiện đại về RL.

Người đầu tiên đi theo hướng tiếp cận sử dụng phương pháp thử và sai có thể kể đến là Edward Thorndike. Thực chất của ý tưởng này là: các hành động mà theo sau đó là một kết quả tốt hay xấu, sẽ có xu hướng thay đổi tương ứng để lựa chọn lại. Thorndike gọi điều này là "luật tác động"-mô tả tác động của các sự kiện lên xu hướng lựa chọn hành động. Luật tác động bao gồm hai khía cạnh quan trọng nhất của phương pháp thử và sai, tính lựa chọn và tính kết hợp. Tính lựa chọn liên quan đến việc cố gắng thay đổi và lựa chọn dựa trên việc so sánh các kết quả. Tính kết hợp thể hiện ở chỗ các thay đổi được kết hợp với các tình huống riêng biệt. Lựa chọn tự nhiên trong tiến hóa là một ví dụ về tính lựa chọn, nhưng nó không có tính kết hợp trong khi, việc học có giám sát mang tính kết hợp nhưng không có tính lựa chọn. Tóm lại, luật tác động là sự kết hợp giữa "tìm kiếm" và "ghi nhớ", tìm kiếm trong các định dạng về phép thử và lựa chọn hành

động trong mỗi tình huống, ghi nhớ các hành động hoạt động tốt nhất trong các tình huống. Sự kết hợp này chính là bản chất trong RL.

Với hướng tiếp cận thứ hai, thuật ngữ "điều khiển tối ưu" bắt đầu được sử dụng vào cuối những năm 1950 để mô tả bài toán thiết kế một bộ điều khiển nhằm cực tiểu hóa phép đo hành vi của một hệ thống động theo thời gian. Một cách tiếp cận cho bài toán này được Richard Bellman và các cộng sự phát triển vào giữa những năm 1950 bằng cách mở rộng lý thuyết của Hamilton và Jacobi ở thế kỷ 19. Cách tiếp cận này sử dụng khái niệm "trạng thái" của một hệ thống động và khái niệm "hàm giá trị" hay "hàm phản hồi tối ưu" để định nghĩa một phương trình hàm hay còn gọi "phương trình Bellman". Lớp các phương pháp để giải quyết bài toán điều khiển tối ưu bằng cách giải phương trình này được gọi là quy hoạch động (Bellman 1957a). Bellman (1957b) cũng giới thiệu một phiên bản bài toán điều khiển tối ưu riêng biệt gọi là quá trình ra quyết định Markov (MDP). Ron Howard (1960) phát minh ra phương pháp lặp chiến lược cho MDP. Tất cả những yếu tố này là những thành phần thiết yếu trong lý thuyết và các giải thuật của RL hiện đại. Quy hoạch động là phương pháp khả thi cho các bài toán điều khiển tối ưu, tuy nhiên nó cũng bị hạn chế ở độ phức tạp tính toán, các yêu cầu tính toán tăng theo cấp số nhân theo số các biến trạng thái. Phương pháp này sau đó cũng đã được nghiên cứu và phát triển mở rộng cho phù hợp với từng yêu cầu.

Hướng tiếp cận thứ ba liên quan đến sự chênh lệch về thời gian (TD). Hướng phát triển này là mới và duy nhất trong RL và đóng một vai trò quan trọng vì chúng có khả năng giải quyết các bài toán với tập trạng thái và hành động liên tục.

Nhiều bài toán khác nhau có thể được giải quyết bởi RL. Do RL tác tử có thể học mà không cần người giám sát nên kiểu bài toán phù hợp với RL là các bài toán phức tạp, xuất hiện cách giải quyết không dễ dàng và mạch lạc. Lĩnh vực ứng dụng RL chủ yếu là phục vụ cho hai lớp người dùng chính:

- *Người chơi game*: việc quyết định bước di chuyển tốt nhất trong trò chơi phụ thuộc vào một số nhân tố khác nhau, do đó số các trạng thái có khả năng tồn tại trong một trò chơi thường rất lớn. Để bao hàm toàn bộ các trạng thái này sử dụng một cách tiếp cận dựa trên các luật chuẩn đòi hỏi phải đặc tả một số lượng lớn các luật mã hoá cứng. RL sẽ giúp lược bỏ điều này, tác tử học đơn giản bằng cách chơi trò chơi, với 2 người chơi ví dụ như trong chơi cờ, tác tử có thể được đào tạo bằng cách chơi với các người chơi hoặc thậm chí là các tác tử chơi khác.

- *Các bài toán điều khiển*: ví dụ như lập chương trình cho thang máy. Sẽ không dễ dàng chỉ ra các chiến lược cung cấp tốt nhất cho hầu hết các lần thang máy phục vụ. Với các bài toán điều khiển kiểu như thế này, tác tử RL có thể được đặt để học trong một môi trường mô phỏng, cuối cùng là chúng sẽ đạt được các chiến lược điều khiển tốt nhất. Một số ưu điểm trong việc sử dụng RL cho các bài toán điều khiển là tác tử có thể đào tạo lại dễ dàng để thích ứng với những thay đổi của môi trường, và được đào tạo liên tục trong khi hệ thống online, cải thiện hiệu năng trên toàn bộ thời gian.

Chương 2 CÁC THUẬT TOÁN HỌC TĂNG CƯỜNG

Trong chương này trình bày chi tiết từng thuật toán học tăng cường đã và đang được sử dụng hiện nay.

2.1 PHƯƠNG PHÁP QUY HOẠCH ĐỘNG (DP)

Thuật ngữ quy hoạch động liên quan đến tập các giải thuật được sử dụng để tính các chiến lược tối ưu với mô hình về môi trường hoàn hảo được đưa ra. Các thuật toán DP cổ điển bị giới hạn trong RL cả về giả thiết một mô hình hoàn hảo về môi trường và cả về phí tổn tính toán của nó tuy nhiên chúng vẫn đóng một vai trò quan trọng về lý thuyết. DP cung cấp một nền tảng thiết yếu để hiểu được các phương pháp khác. Thực tế tất cả các phương pháp khác ra đời đều với mục đích là đạt được cùng hiệu năng như phương pháp DP với ít chi phí tính toán hơn và không cần giả thiết một mô hình hoàn hảo về môi trường.

Để áp dụng được quy hoạch động, chúng ta phải sử dụng các giả thiết sau:

- Môi trường có thể được mô hình dưới dạng một bài toán Markov hữu hạn. Nghĩa là tập các trạng thái và hành động là hữu hạn, và tính động được đưa ra là các khả năng chuyển đổi trạng thái.

$$P_{ss'}^{a} = P\left\{s_{t+1} = s' | s_t = s, a_t = a\right\},$$

- Mục tiêu tức thì được kỳ vọng:

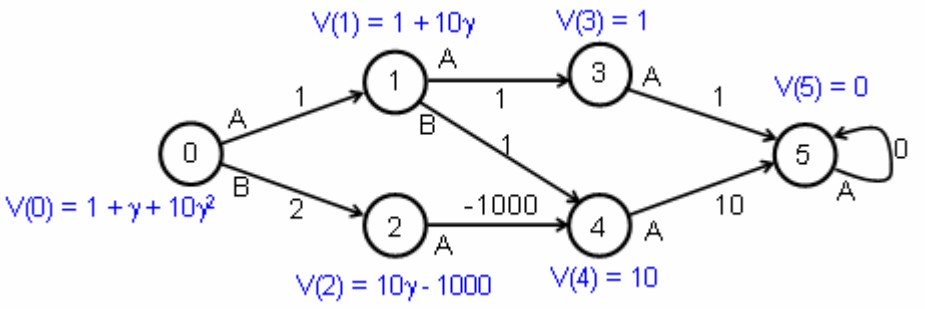
$$R_{ss'}^{a} = E\left\{r_{t+1} | s_t = s, a_t = a, s_{t+1} = s'\right\}.$$

Phương pháp quy hoạch động sử dụng các hàm giá trị để tổ chức và cấu trúc hóa phép tìm kiếm các chính sách tối ưu. Chúng ta có thể dễ dàng thu được các chính sách tối ưu mỗi khi tìm thấy các hàm giá trị tối ưu, V* hoặc Q*, thỏa mãn

phương trình tối ưu Bellman. Các thuật toán DP thu được chính là nhờ phép biến đổi phương trình Bellman.

$$V^{\pi}(s) = \sum_{a} \pi(s,a) \sum_{s'} P^{a}_{ss'} \left[R^{a}_{ss'} + \gamma V^{\pi}(s') \right].$$

Ví dụ với mô hình DP cho trước chúng ta có thể tính các hàm giá trị tối ưu một cách trực tiếp như hình vẽ minh họa sau đây:



2.2 PHƯƠNG PHÁP MONTE CARLO (MC)

Các phương pháp Monte Carlo thích hợp cho việc học từ các kinh nghiệm trong đó không yêu cầu nhận thức trước đó về tính động của môi trường. Chúng giải quyết bài toán quyết định dựa trên việc tính trung bình các giá trị phản hồi mẫu.

Có hai kiểu phương pháp Monte Carlo được áp dụng để ước lượng $V^{\pi}(s)$ và $Q^{\pi}(s,a)$ đó là phương pháp MC kiểm tra toàn bộ và phương pháp MC kiểm tra đầu tiên.

Phương pháp MC kiểm tra toàn bộ ước lượng $V^{\pi}(s)$ bằng trung bình các phản hồi sau tất cả các bước kiểm tra đối với s. $Q^{\pi}(s,a)$ được ước lượng là trung bình các phản hồi sau tất cả các bước kiểm tra đối với cặp (s,a). Phương pháp MC kiểm tra đầu tiên tính trung bình chỉ giá trị phản hồi sau bước kiểm tra đầu tiên

trong phép ước lượng $V^{\pi}(s)$ và $Q^{\pi}(s,a)$. Cả hai phương pháp này đều hội tụ đến $V^{\pi}(s)$ hoặc $Q^{\pi}(s,a)$ như là số các bước thăm đến s hoặc cặp (s,a).

Đánh giá chiến lược sử dụng phương pháp MC

Lặp vô hạn:

(a) Tạo một đoạn mẫu sử dụng chiến lược được ước lượng

$s_0, a_0; s_1, a_1, r_1; \ldots; s_t, r_t$

(b) Với mỗi trạng thái s xuất hiện trong đoạn

$$R_{total}(s) \leftarrow R_{total}(s) + R_{CurrentFirstOccurrence}(s)$$
$$V^{\pi}(s) \leftarrow \frac{R_{total}(s)}{LoopCounter}$$

Chú ý rằng khi tạo từng đoạn, tất cả các trạng thái phải có khả năng tương đương với trạng thái bắt đầu. Nếu mô hình môi trường không sẵn có thì sử dụng ước lượng các giá trị hành động tốt hơn là ước lượng các giá trị trạng thái. Nếu có mô hình môi trường thì các giá trị trạng thái đủ khả năng để quyết định chiến lược. Chúng ta không thể sử dụng các ước lượng giá trị trạng thái để quyết định chiến lược mà không có mô hình về môi trường. Trong khi đó, chúng ta có thể sử dụng các ước lượng giá trị hành động trong việc quyết định chiến lược mà không cần yêu cầu mô hình môi trường.

Với một chiến lược π, chúng ta sẽ chỉ quan sát các giá trị phản hồi đối với chỉ một hành động tại mỗi trạng thái. Như vậy, ước lượng Monte Carlo của các trạng thái khác sẽ không cải tiến theo kinh nghiệm. Đây là một vấn đề quan trọng vì mục đích của các giá trị hành động học là giúp cho việc lựa chọn giữa các giá trị có hiệu lực trong mỗi trạng thái.

Kết quả là chúng ta cần ước lượng giá trị của tất cả các hành động từ mỗi trạng thái. Để giải quyết vấn đề này, chúng ta có thể bắt đầu mỗi đoạn tại một

cặp hành động - trạng thái, mọi cặp như vậy sẽ có khả năng lựa chọn <>0 khi bắt đầu. Giải pháp khác là sử dụng chiến lược ngẫu nhiên với khả năng lựa chọn tất cả các hành động khác 0. Điều này đảm bảo rằng tất cả các cặp hành động – trạng thái sẽ được kiểm tra một số lần vô hạn trong giới hạn là có vô hạn các đoạn.

Chiến lược tối ưu sử dụng phương pháp MC

Bắt đầu với một chiến lược π ngẫu nhiên và Q(s,a) ngẫu nhiên

Lặp vô hạn:

(a) Tạo một đoạn mẫu sử dụng π với khả năng lựa chọn tất cả các hành động là khác 0, độc lập với π tại thời điểm bắt đầu: s_0, a_0; s_1, a_1, r_1; …;s_t, r_t

(b) Với mỗi cặp s, a xuất hiện trong đoạn

$$R_{total}(s,a) \leftarrow R_{total}(s,a) + R_{CurrentFirstOccurrence}(s,a)$$
$$Q(s,a) \leftarrow \frac{R_{total}(s,a)}{LoopCounter}$$

(c) Với mỗi s trong đoạn

$$\pi(s) \leftarrow \arg\max_a Q(s,a)$$

Tóm lại, một vấn đề chính trong khi sử dụng phương pháp MC là đảm bảo rằng tất cả các hành động được lựa chọn không giới hạn. Để đảm bảo điều này, chúng ta sử dụng các chiến lược soft với π(s,a) > 0 cho tất cả các trạng thái và hành động. Khả năng thực hiện có thể được chuyển dần chiến lược hướng đến chiến lược tối ưu. Ví dụ, có thể áp dụng phương pháp lựa chọn hành động ε-greeady và softmax để thực hiện khả năng trên.

2.2.1 Phương pháp MC on-policy

Trong phương pháp này, chiến lược điều khiển tác tử sẽ được cải thiện. Một chiến lược soft sử dụng phương pháp lựa chọn hành động ε-greeady là một chiến lược ngẫu nhiên với:

$$\pi(s,a) \quad = \quad \begin{cases} 1-\epsilon+P_{ng}^{a} & \text{if } a = a^* \\ P_{ng}^{a} & \text{if } a \neq a^* \end{cases}$$

Chúng ta có thể thay đổi thuật toán cho chiến lược tối ưu với giả sử rằng phép lựa chọn tất cả các hành động độc lập với π tại thời điểm bắt đầu sử dụng các chiến lược soft. Các chiến lược soft đảm bảo phép lựa chọn tất cả các hành động tại tất cả các bước.

Bắt đầu với một chiến lược soft bất kỳ π và Q(s,a) bất kỳ.

Lặp vô hạn:

(a) Tạo ra một đoạn sử dụng π: $s_0, a_0; s_1, a_1, r_1; \ldots; s_T, r_T$

(b) Với mỗi cặp s, a xuất hiện trong đoạn

$$R_{total}(s,a) \leftarrow R_{total}(s,a) + R_{CurrentFirstOccurrence}(s,a)$$
$$Q(s,a) \leftarrow \frac{R_{total}(s,a)}{LoopCounter}$$

(c) Với mỗi s trong đoạn

$$a^* \leftarrow \arg\max_a Q(s,a)$$

Cho tất cả các hành động a:

$$\pi(s,a) \quad \leftarrow \quad \begin{cases} 1-\epsilon+P_{ng}^{a} & \text{if } a = a^* \\ P_{ng}^{a} & \text{if } a \neq a^* \end{cases}$$

2.2.2 Phương pháp MC off-policy

Trong phương pháp này, chiến lược được sử dụng để tạo hành vi khác so với chiến lược được ước lượng và cải tiến. Chiến lược được sử dụng để tạo hành vi được gọi là chiến lược hành vi và chiến lược khác được gọi là chiến lược ước lượng.

Một đặc điểm quan trọng của chiến lược hành vi đó là chiến lược cần phải có khả năng lựa chọn tất cả các hành động được lựa chọn bởi chiến lược ước lượng là khác 0.

2.3 PHƯƠNG PHÁP TEMPORAL DIFFERENCE (TD)

Phương pháp này được sử dụng để ước lượng các hàm giá trị. Nếu các hàm giá trị có thể tính toán mà không cần ước lượng, tác tử cần phải đợi đến tận khi nhận được giá trị phản hồi cuối cùng trước khi các giá trị của cặp trạng thái-hành động được cập nhật tương ứng. Phương pháp này được biểu diễn hình thức như sau:

$$V(s_t) \leftarrow V(s_t) + \alpha [R_t - V(s_t)]$$

với s_t là trạng thái được xem xét tại thời điểm t, r_t là giá trị phản hồi sau thời gian t và α là một hằng số.

Mặt khác với phương pháp TD, một ước lượng của giá trị phản hồi cuối cùng được tính tại mỗi trạng thái và giá trị trạng thái-hành động được cập nhật cho mọi bước. Biểu diễn hình thức:

$$V(s_t) \leftarrow V(s_t) + \alpha [r_{t+1} + \gamma V(s_{t+1}) - V(s_t)]$$

với r_{t+1} là giá trị phản hồi thu được tại thời điểm t+1.

Phương pháp TD được gọi là phương pháp "tự cập nhật", bởi vì giá trị được cập nhật từng phần sử dụng một xấp xỉ tồn tại mà không phải là giá trị trả về cuối cùng. Với mọi chiến lược cố định π, thuật toán TD được chứng minh là hội tụ đến V^{π}. TD(0) là phương pháp TD đơn giản nhất, đây là một dạng TD dự báo. SARSA và Q-Learning là các thuật toán TD điều khiển. Hiện nay các phương pháp này chính là các phương pháp RL được sử dụng phổ biến nhất.

2.3.1 TD(0)

Chúng ta xem xét tình huống khi mà tác tử không biết được các thông tin về môi trường của nó (hoặc không biết các thông tin mong muốn về giá trị tăng cường tức thì, hoặc không biết xác suất của phép chuyển đổi từ một trạng thái đến trạng thái khác tuỳ theo hành động được lựa chọn). Trong tình huống này, chúng ta có thể xây dựng một mô hình và sử dụng nó để học hàm giá trị, đây được gọi là phương pháp gián tiếp. Chúng ta cũng có thể ước lượng trực tiếp hàm giá trị từ kinh nghiệm, áp dụng các phương pháp không dùng đến mô hình. Ý tưởng được hình thức hoá trong phương pháp TD như sau:

Vì giá trị của một trạng thái là không được biết, để sử dụng các ước lượng thành công của V^{*}. Tại thời gian bước t, $V_t(s_t)$ biểu diễn giá trị của $V^{*}(s_t)$: hành động a_t thực hiện chuyển từ trạng thái s_t sang s_{t+1} và giá trị tăng cường tức thì là r_t. Như vậy ngay lập tức ta có một giá trị mới của hàm giá trị:

$r_t + \gamma V_t(s_{t+1})$

Do tất cả các phép chuyển đổi trạng thái có thể sang trạng thái khác s_{t+1} là 0 và $V_t(s_{t+1})$ biểu diễn giá trị được thực hiện tại thời điểm t của trạng thái s_{t+1} nên sự chênh lệch thời gian giữa hai giá trị thành công của s_t là $r_t + \gamma V_t(s_{t+1}) - V_t(s_t)$. Sự khác nhau này được sử dụng để cập nhật hàm giá trị:

$V_{t+1}(s_t) = V_t(s_t) + \alpha\{r_t + \gamma V_t(s_{t+1}) - V_t(s_t)\}$

Hệ số học α, $0 \le \alpha \le 1$, giảm từ từ đến 0. Các hàm giá trị hội tụ đến hàm giá trị tối ưu.

2.3.2 TD(λ)

Thuật toán TD(0), chỉ quan tâm đến trạng thái hiện tại và chỉ tính hàm giá trị theo trạng thái mới. Sutton mở rộng sự ước lượng cho mọi trạng thái tuỳ theo sự phù hợp của chúng bằng cách đưa ra khái niệm về bậc quan sát của một trạng thái trong quá khứ gần còn gọi là tính phù hợp. Tính phù hợp có thể được định nghĩa theo một vài cách. Tính phù hợp tích luỹ được định nghĩa như sau:

$$e_t(s) = \begin{cases} 1 + \gamma\lambda e_{t-1}(s) & \text{khi } s = s_t \\ \gamma\lambda e_{t-1}(s) & \text{trường hợp còn lại} \end{cases} \quad \text{(I)}$$

Trong khi có một cách định nghĩa khác:

$$e_t(s) = \begin{cases} 1 & \text{khi } s = s_t \\ \gamma\lambda e_{t-1}(s) & \text{trường hợp còn lại} \end{cases} \quad \text{(II)}$$

Theo dõi tính phù hợp tạo thành một bộ nhớ ngắn về chuỗi quan sát một trạng thái. Nó giảm theo hàm mũ theo thời gian, nếu không được kích hoạt bởi một quan sát mới. Sau khi giới thiệu về theo dõi tính phù hợp, việc cập nhật giá trị $V_{t+1}(s_t)$ như sau:

$V_{t+1}(s_t) = V_t(s_t) + \alpha\{r_t + \gamma V_t(s_{t+1}) - V_t(s_t)\}\ e_t(s)$ (III)

Ở đây xuất hiện một tham số mới λ, $0 \le \lambda \le 1$ và giải thuật tương ứng là TD(λ). Hiệu năng tốt nhất là khi λ giảm từ các giá trị gần 1 đến 0. Phương pháp TD(λ) được sử dụng bởi tất cả các giải thuật của RL mà không cần có mô hình, đôi khi

với $\lambda = 0$, đôi khi với $\lambda > 0$, bởi vì trong rất nhiều trường hợp việc hội tụ sẽ đến nhanh hơn.

Thuật toán TD(λ)

$t = 0$, $V_t(s) = 0$, $e_t(s) = 0$ với mọi s∈S.

Lặp

Quan sát bước chuyển trạng thái $s_t \rightarrow s_{t+1}$.

Tính $e_t(s)$ theo phương trình (I) hoặc (II).

Từ đó tính V(s) cho mọi s, theo phương trình (III).

$t \leftarrow t+1$

Đến tận khi điều kiện kết thúc.

2.3.3 Q-Learning

Đây là một thuật toán off-policy dùng cho TD-Learning.

```
Initialize Q(s, a) arbitrarily
Repeat (for each episode):
   Initialize s
   Repeat (for each step of episode):
      Choose a from s using policy derived from Q
         (e.g., ε-greedy)
      Take action a, observe r, s'
      Q(s, a) <-- Q(s, a) + α [r + γ max_α,Q(s', a') - Q(s, a)]
      s <-- s';
   until s is terminal
```

Các tham số được sử dụng trong quá trình cập nhật giá trị Q là:

- α: hệ số học, nằm trong khoảng 0,1. Bằng 0 có nghĩa là giá trị Q không bao giờ được cập nhật, nghĩa là không có gì được học. Giá trị lớn, ví dụ 0.9 nghĩa là việc học xảy ra nhanh.
- γ: hệ số suy giảm, cũng nằm trong khoảng 0,1.

- max_{α} : giá trị tăng cường lớn nhất có thể đạt được trong trạng thái theo sau trạng thái hiện tại.

Giải thích thuật toán trên:

1. Khởi tạo bảng giá trị Q, Q(s,a).
2. Quan sát trạng thái hiện tại s.
3. Lựa chọn hành động a cho trạng thái dựa vào một trong các chiến lược lựa chọn hành động (ε-soft, ε-greedy hoặc softmax).
4. Thực hiện hành động và quan sát giá trị r cũng như trạng thái mới s'.
5. Cập nhật giá trị Q cho trạng thái sử dụng giá trị tăng cường được quan sát và giá trị tăng cường lớn nhất có thể cho trạng thái tiếp theo. Việc thực hiện được cập nhật dựa theo công thức mô tả ở trên.
6. Thiết lập trạng thái đến trạng thái mới và lặp lại quá trình này đến tận khi gặp được trạng thái kết thúc.

2.3.4 SARSA

Đây là thuật toán on-policy cho TD Learning, sự khác nhau chính giữa nó và Q-Learning là giá trị tăng cường lớn nhất cho trạng thái tiếp theo không cần thiết được sử dụng để cập nhật giá trị Q. Thay vì đó, một hành động mới, và do đó giá trị tăng cường, được lựa chọn sử dụng cùng chiến lược quyết định hành động ban đầu. Cái tên Sarsa xuất phát từ thực tế là việc cập nhật được thực hiện sử dụng bộ 5 Q(s,a,r,s',a') với s,a là trạng thái và hành động hiện tại, r là giá trị tăng cường được quan sát trong trạng thái tiếp theo và s',a' là cặp trạng thái-hành động tiếp theo. Thuật toán sarsa:

```
Initialize Q(s, a) arbitrarily
Repeat (for each episode):
   Initialize s
   Choose a from s using policy derived from Q
         (e.g., ε-greedy)
   Repeat (for each step of episode):
      Take action a, observe r, s'
      Choose a' from s' using policy derived from Q
         (e.g., ε-greedy)
      Q(s, a) <-- Q(s, a) + α[r + γQ(s', a')- Q(s, a)]
      s <-- s'; a <-- a';
   until s is terminal
```

Như vậy, có hai bước lựa chọn hành động cần thiết để quyết định cặp trạng thái-hành động tiếp theo, tham số α, γ cũng có ý nghĩa giống như trong Q-Learning.

2.4 SO SÁNH CÁC THUẬT TOÁN HỌC TĂNG CƯỜNG ĐIỂN HÌNH

Phương pháp MC thao tác trên kinh nghiệm mẫu do đó có thể sử dụng để học trực tiếp mà không cần mô hình. Tuy nhiên phương pháp MC không có khả năng "tự mồi", nghĩa là chúng không có khả năng tự cập nhật các ước lượng giá trị dựa cơ bản trên các ước lượng giá trị khác (do đó cần phải đợi đến tận kết quả cuối cùng). Như vậy, về ý tưởng, chúng ta cần một phương pháp học từ kinh nghiệm giống MC nhưng cũng phải có khả năng tự cập nhật giống DP. Đó chính là thuật toán học TD. Thuật toán học TD kết hợp ý tưởng của cả hai phương pháp DP và MC. Nó học trực tiếp từ kinh nghiệm đồng thời có khả năng tự cập nhật các ước lượng dựa trên một phần các ước lượng đã được học khác. Phương pháp này liên quan đến các bài toán dự đoán và điều khiển. Phương pháp TD có thể tự cập nhật tại mọi bước thời gian mà không cần đợi đến tận khi kết thúc.

2.5 MỘT SỐ PHƯƠNG PHÁP TIẾN BỘ KHÁC

Có một số kỹ thuật tinh vi và phức tạp hơn nhiều được phát triển bằng sự kết hợp hoặc thay đổi các phương pháp cơ bản đã trình bày ở trên. Nói chung, thời gian tính toán là vấn đề chính được đưa ra trong RL vì rất nhiều bài toán trên thực tế đòi hỏi cập nhật thời gian thực, và các kỹ thuật dựa trên thông tin không được cập nhật sau nhiều bước sẽ gặp khó khăn khi giải quyết những bài toán này. Đó chính là lý do tại sao các mô hình về môi trường đôi khi được thực hiện trong tác tử vì chúng cho phép tác tử sử dụng kỹ thuật lập kế hoạch trên mô hình, trùng khớp với kinh nghiệm trên môi trường thực tế. Giả sử mô hình chính xác, tác tử sẽ có khả năng ra quyết định tối ưu nhanh hơn rất nhiều.

Một kỹ thuật tiến bộ được biết đến đó là lập trình TD-Gammon (Tesauro 1992, 1994, 1995). Đây là một chương trình chơi cờ kết hợp kỹ thuật TD với một mạng nơron hỗ trợ trong việc dự đoán các giá trị tương lai. TD-gammon sau chỉ hai tuần đào tạo (bằng cách tự chơi với nó) đã tăng đến trình độ bằng trình độ của người chơi cờ giỏi nhất thế giới.

Rõ ràng sự phát triển các phương pháp kết hợp và mở rộng các phương pháp RL cơ bản có thể đưa ra kết quả to lớn. Để hiểu rõ hơn các kỹ thuật cơ bản về RL, người ta đã thực hiện một số bài toán khác nhau với các giải thuật khác nhau nhờ đó chúng ta có thể phân tích cách thức làm việc của chúng.

Chương 3 THỬ NGHIỆM

Trong chương này chúng ta sẽ tìm hiểu bài toán thực nghiệm mô phỏng phương pháp học tăng cường. Với bài toán này, bốn giải thuật học tăng cường điển hình bao gồm phương pháp lặp giá trị, phương pháp lặp chiến lược, phương pháp Q-Learning, và phương pháp quét mức độ ưu tiên (prioritized sweeping) được mô phỏng và tính toán để chúng ta có thể quan sát hành vi và phân tích hiệu năng của chúng.

3.1 BÀI TOÁN LỰA CHỌN MÔ PHỎNG

Như chúng ta đã biết phương pháp học tăng cường có ứng dụng quan trọng trong các bài toán điều khiển, đặc biệt là trong ngành khoa học người máy. Ví dụ, một robot được tạo ra để thực hiện nhiệm vụ dọn dẹp phòng một cách tự động, mà không va phải các chướng ngại vật (đồ đạc trong phòng) và không bị rơi vào trạng thái hết ắc quy. Trong trường hợp robot ở trạng thái gần hết ắc quy thì nó cần phải trở lại trạm nạp ắc quy trước khi bị rơi vào trạng thái hết ắc quy và không thể hoạt động.

Một phương pháp giải quyết bài toán này là sử dụng các luật mã hoá điều khiển bằng tay để thăm dò và dọn dẹp phòng, tránh các chướng ngại vật và kiểm tra mức ắc quy so với khoảng cách để đi đến trạm nạp ắc quy, và ra quyết định khi robot cần phải trở về để nạp ắc quy. Tuy nhiên phương pháp tiếp cận này thực hiện khá phức tạp và không dễ mở rộng.

Chúng ta mô hình hoá bài toán theo một cách quan sát khác để giải quyết chúng. Robot sẽ nhận được giá trị tăng cường (+) khi dọn dẹp phòng, giá trị tăng cường (-) khi va vào chướng ngại vật và giá trị tăng cường (-) khi ắc quy bị hết trước khi robot đến được trạm nạp ắc quy. Sau đó, robot có thể học chiến lược

tốt nhất để dọn dẹp phòng bằng cách di chuyển trong phòng và cố gắng cực đại hoá giá trị tăng cường nó nhận được. Chiến lược này dễ dàng thực hiện và mở rộng. Đó chính là phương pháp học tăng cường.

Trong bài toán học tăng cường, người học và người ra quyết định được gọi là tác tử. Tác tử học từ tương tác với môi trường. Có hai dạng bài toán quyết định trong thực tế, đó là tác tử có nhận thức về mô hình của môi trường hoặc không. Các thuật toán trong cả hai trường hợp này đều được mô phỏng trong thực nghiệm.

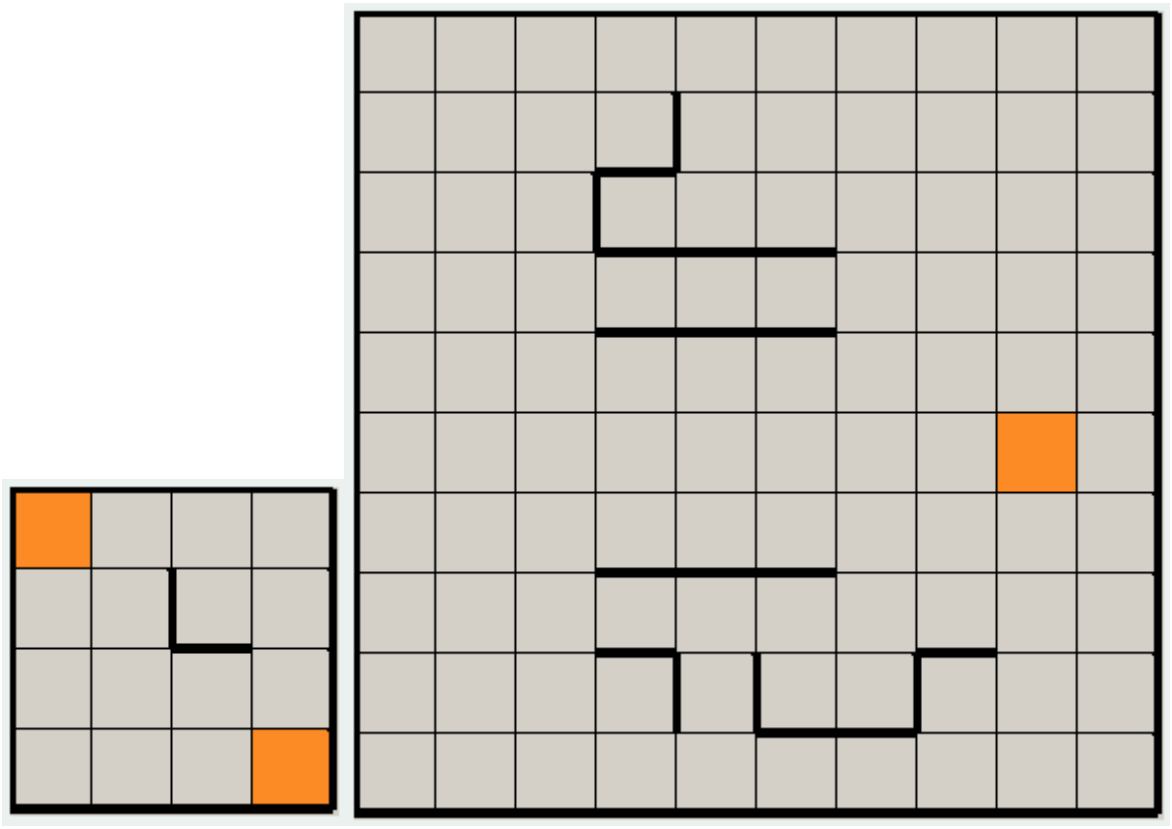
Chiến lược $\pi(s)$ là một ánh xạ từ tập các trạng thái sang tập các hành động. Một chiến lược tối ưu $\pi^*(s)$ là ánh xạ từ tập các trạng thái đến hành động tốt nhất. Việc tìm kiếm chiến lược tối ưu chính là vấn đề chính trong RL.

Bài toán robot ở trên được mô hình hoá trong một dạng bài toán mê lộ để dễ dàng cho việc minh hoạ hoạt hình. Một mê lộ với mỗi ô vuông biểu diễn một trạng thái. Có một hoặc một vài ô đích. Có thể có một số bức tường biểu diễn bằng một đường biên giữa hai ô kề nhau. Tại mỗi thời điểm tác tử chiếm lĩnh một ô, ở đó, nó có thể thực hiện 4 hành động: lên, xuống, trái, phải để di chuyển đến trạng thái gần kề. Tác tử có thể bị va vào tường, khi đó nó sẽ vẫn ở lại trạng thái cũ và nhận một giá trị phạt (biểu diễn là một số dương). Tại mỗi bước di chuyển, tác tử sẽ nhận một giá trị phạt bằng với chi phí đường đi. Chi phí đường đi là 1 đơn vị cho mọi bước tác tử thực hiện chuyển trạng thái từ trạng thái này sang trạng thái khác. Tác tử sẽ nhận giá trị phạt bằng 0 trong bước chuyển trạng thái đến đích. Mục đích của tác tử là tìm ra đường đi đến đích với ít chi phí nhất.

Đưa ra một trạng thái bắt đầu, tác tử có mục tiêu là đi được đến đích với giá trị phạt thấp nhất có thể, khi đó tác tử đã tìm được chiến lược tối ưu cho bài toán mê lộ đã cho. Một chú ý ở đây là để thuận tiện cho việc biểu diễn các giá trị toán

học, bài toán thực nghiệm này thể hiện hơi khác về cách biểu diễn giá trị tăng cường so với các mô hình thường sử dụng trong các bài toán quyết định. Trong các bài toán quyết định như nghiên cứu trong lý thuyết, tác tử nhận giá trị tăng cường, ví dụ khi đến ô đích giá trị tăng cường là 0, đến các ô còn lại giá trị tăng cường (tương ứng với giá trị phạt) là -1. Ở bài toán thực nghiệm thay vì biểu diễn giá trị tăng cường dưới dạng một số (-), chúng ta coi là tác tử nhận giá trị phạt (+), và như vậy, nếu như mô hình RL lý thuyết là cực đại hoá các giá trị tăng cường thu được, thì ở bài toán thực nghiệm, tác tử cố gắng cực tiểu hoá giá trị phạt hay chính là cực tiểu hoá chi phí đường đi. Thực ra bản chất của chú ý này chính là việc lấy max {các giá trị (-)} = lấy min {các giá trị (+)}. Do đó, các biểu thức toán trong các giải thuật áp dụng thực nghiệm trình bày sau đây sẽ vẫn khớp với các giải thuật như trong lý thuyết.

Bài toán biểu thị dạng ô lưới như hình vẽ, trong bài toán ta có thể thay đổi số lượng, kích thước ô lưới cũng như số lượng và vị trí tường chắn, ô đích.



Hình 3.1: Minh hoạ bài toán

3.2 PHƯƠNG PHÁP HỌC TĂNG CƯỜNG LỰA CHỌN MÔ PHỎNG

3.2.1 Phương pháp quy hoạch động (DP)

Trong trường hợp đã biết về mô hình của môi trường, bài toán được biểu diễn dưới dạng bài toán quyết định Markov và sử dụng phương pháp quy hoạch động (DP) để giải quyết dựa trên thuật toán lặp giá trị hoặc thuật toán lặp chiến lược để tìm ra chiến lược tối ưu.

Thuật toán lặp giá trị:

Algorithm 1 Value Iteration

1: initialize $V(s)$ arbitrarily
2: **repeat**
3: **for all** $s \in S$ **do**
4: $V_{t+1}(s) = \min_{a \in A} \left(PCost + \sum_{s' \in S} P(s'|s,a) \cdot x_{ss'} \right)$
5: **end for**
6: **until** $V(s)$ has converged
7: **for all** $s \in S$ **do**
8: $\pi^*(s) = \arg\min_{a \in A} \left(\sum_{s' \in S} (P(s'|s,a) \cdot x_{ss'}) \right)$
9: **end for**

Trong phương pháp lặp giá trị, trước tiên sẽ tính hàm giá trị tối ưu và dựa trên đó để có được chính sách tối ưu. Để tính hàm giá trị tối ưu, trong bài toán thực nghiệm này, ta định nghĩa $V_k(S_i)$ là giá trị cực đại tổng các mục tiêu có thể đạt được trong tương lai bắt đầu từ trạng thái S_i và sau k bước. Như vậy, $V_1(S_i)$ sẽ là mục tiêu tức thời và $V_2(S_i)$ là hàm của $V_1(S_i)$. Mở rộng khái niệm này, ta có thể biểu diễn $V_{t+1}(S_i)$ cho việc thiết lập kinh nghiệm như sau:

$$V_{t+1}(s) = \min_{a \in A} \left(PCost + \sum_{s'=S} (P(s'|s,a) \cdot x_{ss'}) \right)$$

Trong đó:

- PCost = PathCost.
- P(s'|s,a) = xác suất chuyển từ trạng thái s sang trạng thái s' sau hành động a.
- $x_{ss'} = V_t(s')$ nếu phép chuyển đổi từ s sang s' là an toàn.

  $x_{ss'}$ = Penalty + $V_t(s)$, nếu phép chuyển trạng thái từ s sang s' là không an toàn.

Phép lặp được thực hiện đến tận khi hàm giá trị hội tụ và không có sự thay đổi về giá trị nữa. Mục đích trong bài toán thử nghiệm này là cực tiểu hoá giá trị

phạt và chiến lược tối ưu sẽ là: lựa chọn hành động với trạng thái có hàm giá trị nhỏ nhất:

$$\pi^*(s) = \arg\min_{a \in A} \left(\sum_{s' \in S} (P(s'|s,a) \cdot x_{ss'}) \right)$$

Thuật toán lặp chiến lược:

Algorithm 2 Policy Iteration

```
initialise π(s) arbitrarily
repeat
    repeat
        for all s ∈ S do
            V_{t+1}(s) = PCost + Σ_{s'∈S} (P(s'|s, π(s)) · x_{ss'})
        end for
    until V(s) has converged
    for all s ∈ S do
        π_{t+1}(s) = arg min_{a∈A} (Σ_{s'=S} (P(s'|s, a) · x_{ss'}))
    end for
until policy good enough
```

Phương pháp này thao tác trực tiếp trên chiến lược thay vì phải tìm chúng thông qua hàm giá trị tối ưu. Các bước cơ bản của thuật toán như sau:

- Chọn một chiến lược bất kỳ.
- Đánh giá chiến lược.
- Cập nhật chiến lược.
- Nếu chưa phải là chiến lược tối ưu, quay về bước 2.

Bước lặp giá trị ở trên được sử dụng để đánh giá chiến lược và tìm hàm giá trị V(s) cho chiến lược π(s). Hàm giá trị được tính để đánh giá chiến lược cũng có thể được sử dụng để cập nhật chiến lược. Chiến lược mới được quyết định tương tự như trong phương pháp lặp giá trị ở trên.

3.2.2 Học không có mô hình (Phương pháp Q-Learning)

Algorithm 3 Q-learning

1: initialise $Q(s,a)$ arbitrarily
2: **loop**
3: initialize s
4: **repeat**
5: select action a from s based on $Q(s,a)$ using ϵ-greedy policy
6: take action a, Observe reward r, next state s'
7: $Q(s,a) = (1-\alpha)Q(s,a) + \alpha[X + min_{a'}Q(s',a')]$
8: **until** s is goal
9: **end loop**

Thuật toán Q-Learning ước lượng giá trị của cặp trạng thái – hành động Q(s,a). Mỗi khi các giá trị này được học, hành động tối ưu từ bất kỳ trạng thái nào sẽ ứng với giá trị Q nhỏ nhất.. Vì thế, nếu Q*(s,a) là giá trị tối ưu thì chiến lược tối ưu được biểu diễn như sau:

$$\pi^*(s) = \arg\min_{a \in A} (Q^*(s,a))$$

Ước lượng các giá trị Q được thực hiện trên cơ sở kinh nghiệm sử dụng luật học sau:

$$Q(s,a) = Q(s,a) + \alpha \left(X + \min_{a' \in A} (Q(s',a') - Q(s,a)) \right)$$

Trong đó:

- s’ là trạng thái mới sau khi thực hiện hành động a trên trạng thái s.
- X là giá trị tăng cường quan sát được.

  X = *PathCost*, nếu bước chuyển trạng thái là an toàn.

  X = R, nếu bước chuyển trạng thái là không an toàn.
- α là hệ số học. Hệ số học quyết định khả năng cập nhật giá trị Q hiện tại. Hệ số học lớn nghĩa là việc học xảy ra nhanh và ngược lại.

Để các giá trị Q hội tụ đến giá trị tối ưu, đòi hỏi mọi cặp trạng thái – hành động phải được thăm dò với đủ số lần cần thiết, nói chung là một số vô hạn lần. Chiến lược thăm dò cần được sử dụng để lựa chọn một hành động với trạng thái cho trước.

Có một số chiến lược lựa chọn ta đã trình bày trong phần lý thuyết chương 1. Ở bài toán thực nghiệm này ta sử dụng một số chiến lược thăm dò cho việc tích luỹ kinh nghiệm đó là chiến lược ε-greedy: lựa chọn hành động tốt nhất với xác suất (1-ε) và lựa chọn hành động ngẫu nhiên với xác suất ε. Mỗi khi đến đích, xác lập lại trạng thái bắt đầu ngẫu nhiên cho bước lặp tiếp theo.

3.2.3 Học dựa trên mô hình (Phương pháp prioritized sweeping)

Đây là một thuật toán thông minh dựa trên mô hình. Thăm dò các phần của không gian trạng thái sử dụng thông tin có được từ kinh nghiệm. Nó sử dụng và cập nhật các giá trị kết hợp với trạng thái thay vì cặp trạng thái – hành động như phương pháp Q-Learning.

Ý tưởng cơ bản của thuật toán này là khi tác tử bắt gặp một bước chuyển trạng thái đáng ngạc nhiên, ví dụ như khi bước chuyển trạng thái làm thay đổi hàm giá trị của trạng thái hiện tại với một lượng đáng kể. Thông tin này sẽ được lưu lại. Khi bước chuyển trạng thái được lặp lại (khi giá trị mới của hàm giá trị bằng với giá trị mong đợi), các tính toán tiếp tục được thực hiện trong các phần tương ứng. Để xây dựng mô hình và tạo ra các lựa chọn xấp xỉ, tác tử phải thực hiện lưu trữ các thông tin sau:

- Các thông tin thống kê cho bước chuyển từ trạng thái s sang trạng thái s’ trên hành động a. Thông tin này được sử dụng để ước lượng xác suất phép chuyển đổi trạng thái P(s’|s,a).

- Các thông tin thống kê cho giá trị tăng cường nhận được để thực hiện hành động a trong trạng thái s.
- Thông tin về mọi trạng thái trước: trạng thái có xác suất chuyển đổi khác 0 đối với một số hành động.

Sử dụng xây dựng mô hình từ kinh nghiệm. Thuật toán ước lượng giá trị trạng thái V(s) sử dụng phép lặp giá trị theo luật cập nhật sau:

$$\hat{V}(s) = \min_{a \in A} \left(\hat{r}_i^a + \sum_{s' \in S} \hat{q}_{ss'}^a \hat{V}(s') \right)$$

Trong đó:

- V(s): ước lượng giá trị tối ưu từ trạng thái bắt đầu s.
- r_i^a: ước lượng giá trị tăng cường tức thì.
- $q^a_{ss'}$: ước lượng xác suất chuyển trạng thái từ trạng thái s sang s' với hành động a.

Giải thuật như mô tả dưới đây:

Algorithm 4 Prioritized Sweeping

loop
  repeat
    Take action a from current state, observe new state and reward.
    Update model with observed information.
    Promote recent state to top of priority queue
    while Number of backups processed is less than allowed and priority queue not empty **do**
      Remove the top state from priority queue. Call it s.
      $V_{old} = \hat{V}(s)$
      $\hat{V}(s) = \min_{a \in A} \left(\hat{r}_i^a + \sum_{s' \in S} \hat{q}_{ss'}^a \hat{V}(s') \right)$
      $\Delta = |V_{old} - \hat{V}(S)|$
      for each $(s', a') \in preds(s)i$ **do**
        $P = \hat{q}_{s's}^{a'} \Delta$
        if ($P > \epsilon$ (a tiny threshold)) and (S' not on queue or P exceeds the current priority of s') **then**
          Promote s' to new priority P
        end if
      end for
    end while
  until s is goal
end loop

3.3 KỊCH BẢN VÀ KẾT QUẢ THỬ NGHIỆM

Từ bài toán thực nghiệm: tìm đường đi trong mê lộ sử dụng các phương pháp học tăng cường ta thực hiện các đo đạc thống kê trên các kịch bản kiểm tra để có được các biểu đồ biểu diễn mối quan hệ, phép so sánh giữa các thuật toán.

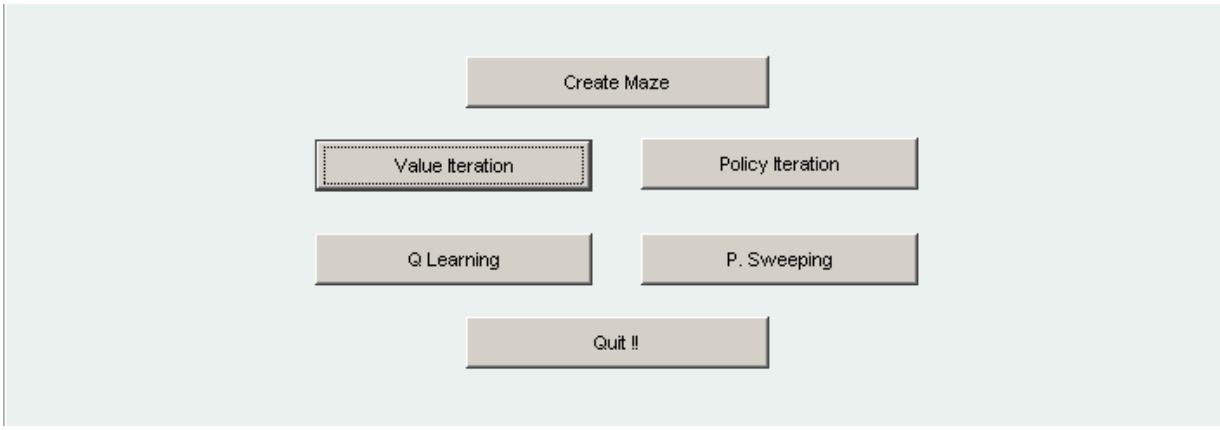
Một số tham số được dùng trong bài toán thực nghiệm như sau:

- ***Tham số 'pjog'***: biểu diễn nhiễu trong môi trường: Mỗi trạng thái trong môi trường có một số hữu hạn các bước thành công N. Nếu trong trạng

thái s, tác tử quyết định thực hiện hành động a thì tác tử sẽ kết thúc thành công cặp trạng thái-hành động (s,a) với xác suất (1-pjog). Kết thúc N-1 bước thành công còn lại của trạng thái đó với hành động a là pjog/(N-1). Với bài toán mê lộ, N = 4 do tác tử có thể thực hiện được 1 trong 4 hành động: lên, xuống, trái, phải.

- ***Tham số ε***: xuất hiện trong thuật toán Q-Learning và thuật toán quét độ ưu tiên. Tham số đặc biệt này để đặc trưng cho chiến lược lựa chọn ε-Greedy. Trong chiến lược này, tác tử quyết định hành động tối ưu với xác suất (1-ε) và thực hiện các hành động ngẫu nhiên khác với xác suất ε/(N-1).
- Số các đoạn lặp (***episodes***) sử dụng trong thuật toán Q-Learning.
- Độ chính xác θ.
- Hệ số học α.

Giao diện chương trình mô phỏng bài toán:

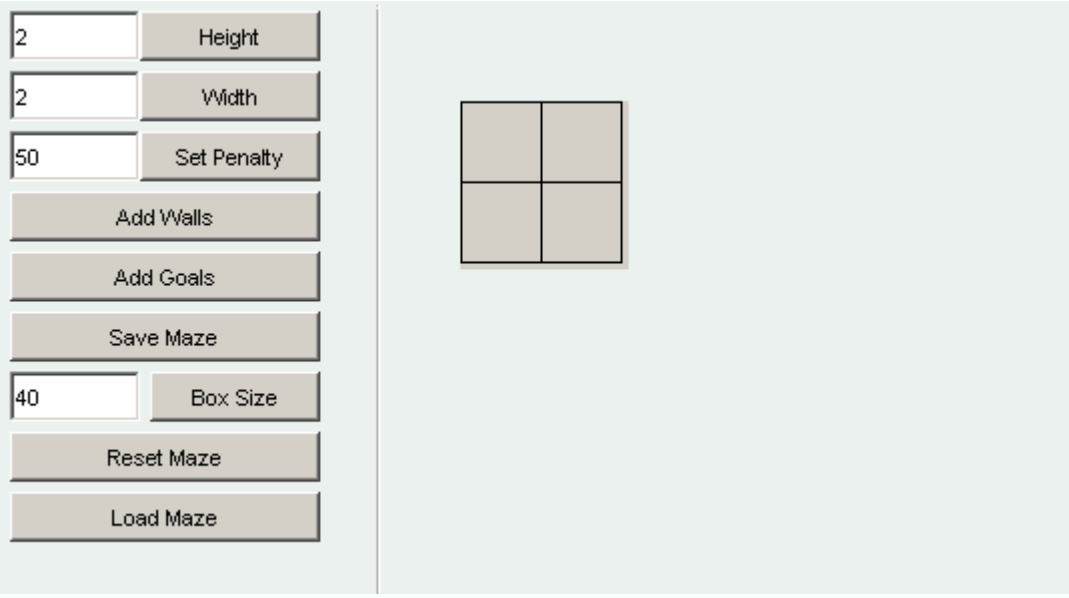


Hình 3.2 Giao diện chương trình mô phỏng

Chức năng *Create Maze*:

Tạo các mẫu bài toán mê lộ với các kích thước khác nhau, số lượng ô trạng thái, số bức tường, số trạng thái đích tuỳ thuộc vào yêu cầu bài toán. Chức năng này cũng cho phép thiết lập giá trị phạt mỗi khi tác tử bị va vào tường bằng cách

nhập giá trị vào ô tương ứng và nhấn nút *Set Penalty*. Sau khi tạo mẫu bài toán mê lộ mong muốn, nhấn nút *Save Maze* lưu mẫu bài toán ra file *.maze để gọi lại bài toán này trong các chức năng khác.

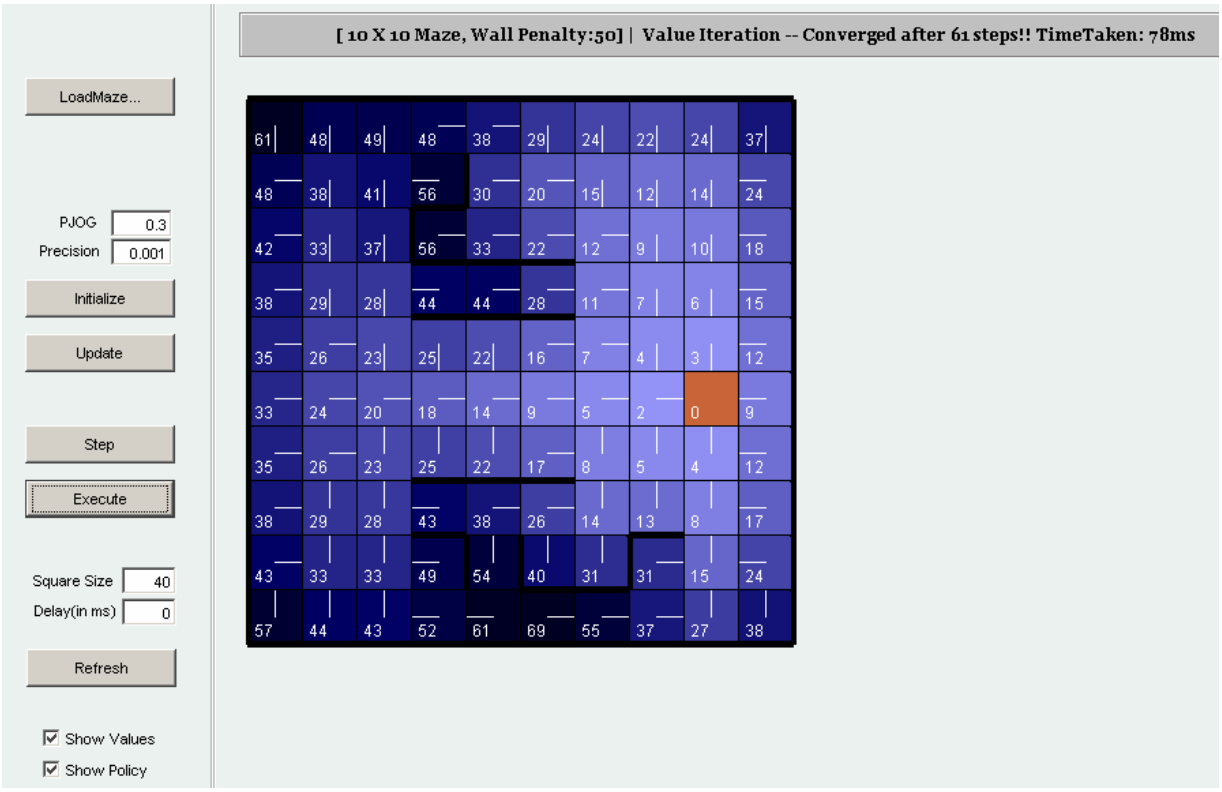


Hình 3.2: Chức năng Create Maze

Các chức năng *Value Iteration*, *Policy Iteration*, *Q-Learning* và *P.Sweeping* tương ứng với bốn loại giải thuật RL đã trình bày trong chương 2, áp dụng vào bài toán mê lộ.

Chức năng *Value Iteration*:

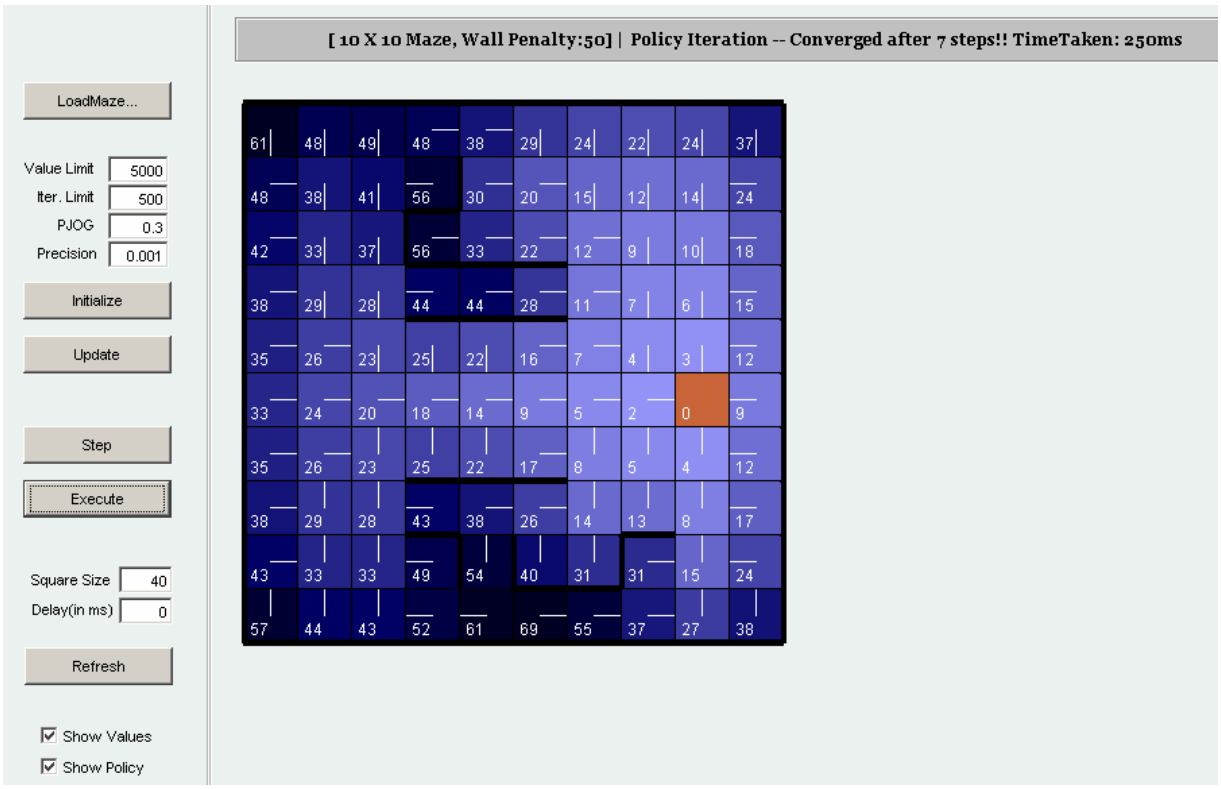
Sử dụng thuật toán lặp giá trị để giải bài toán, với tham số 'pjog' và độ chính xác cho trước, nhấn nút *Initialize* để khởi tạo. Sau đó nhấn nút thực hiện *Execute*, thuật toán lặp giá trị được cài đặt sẵn sẽ cho chúng ta kết quả các hàm giá trị tại mỗi ô trạng thái, cũng như chiến lược tối ưu tại từng trạng thái. Chúng ta cũng biết được số bước cũng như thời gian cần thực hiện để thuật toán hội tụ.



Hình 3.3: Value Iteration

Chức năng *Policy Iteration*:

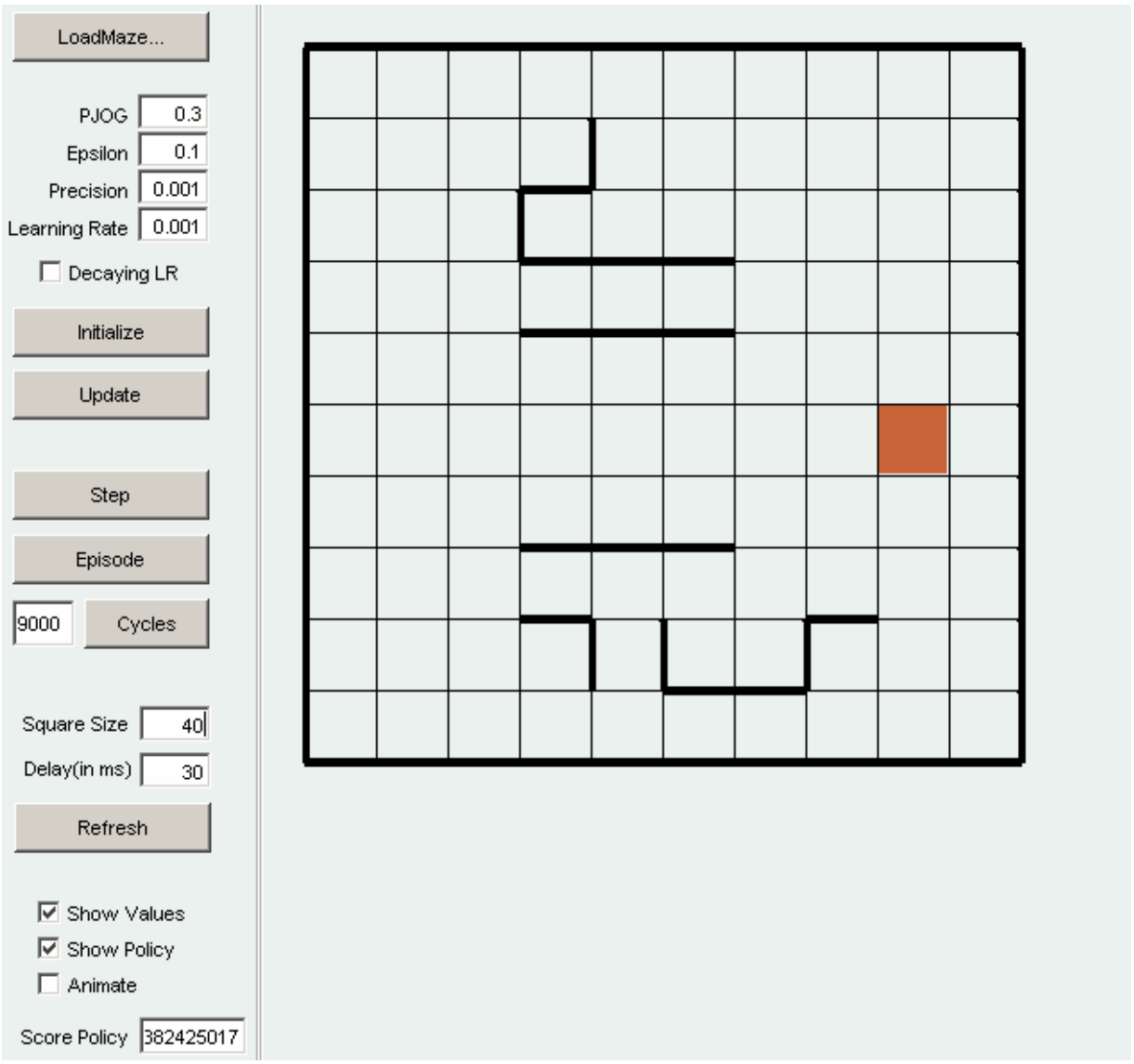
Sử dụng thuật toán lặp chiến lược để giải bài toán, với tham số ‘pjog’ và độ chính xác cho trước, giới hạn về giá trị, giới hạn về bước lặp, nhấn nút *Initialize* để khởi tạo. Sau đó nhấn nút thực hiện *Execute*, thuật toán lặp chiến lược được cài đặt sẵn sẽ cho chúng ta kết quả các hàm giá trị tại mỗi ô trạng thái, cũng như chiến lược tối ưu tại từng trạng thái. Chúng ta cũng biết được số bước cũng như thời gian cần thực hiện để thuật toán hội tụ.



Hình 3.4: Policy Iteration

Chức năng *Q-Learning*:

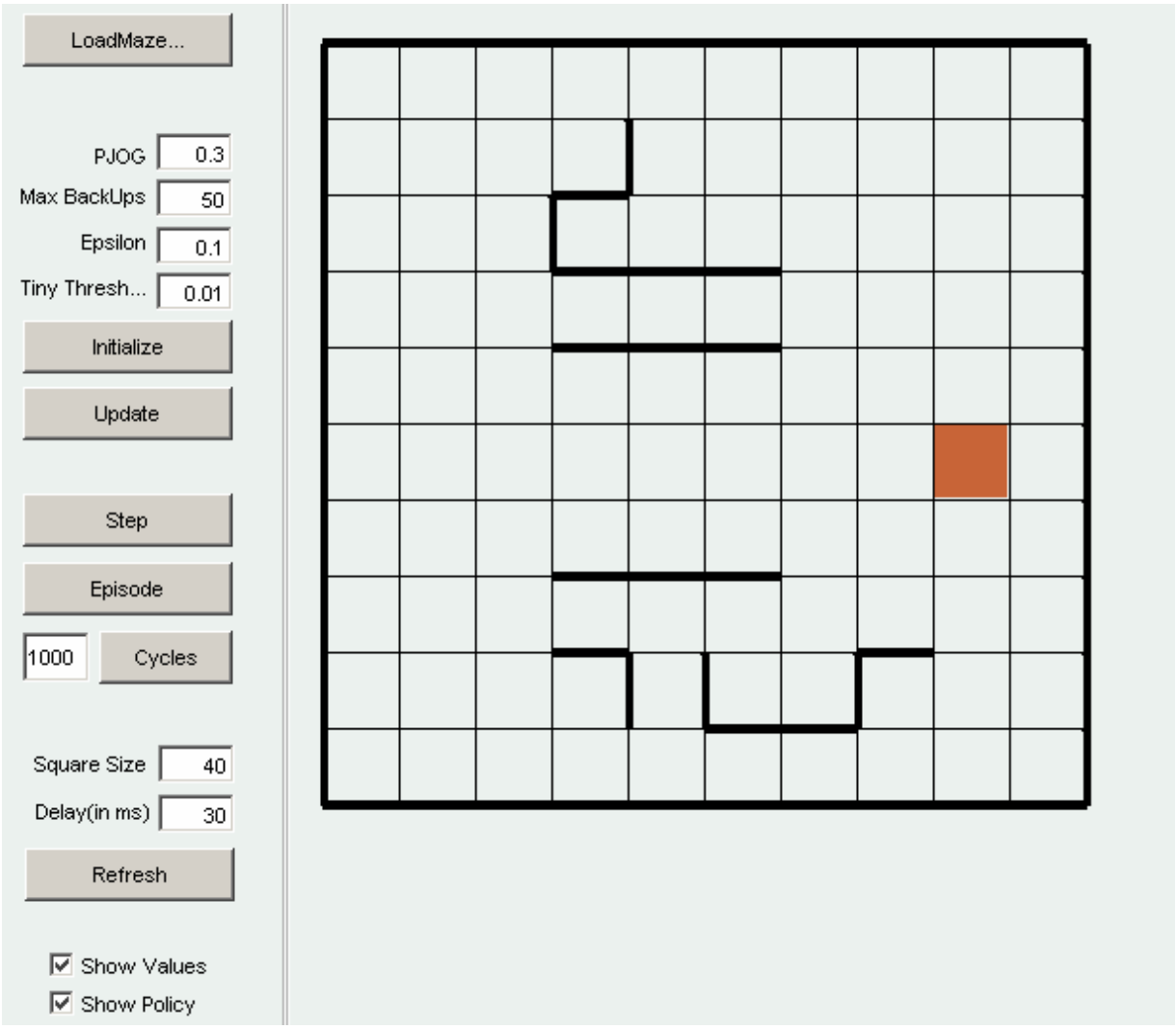
Sử dụng thuật toán Q-Learning để giải bài toán, với các tham số 'pjog', ε, độ chính xác, hệ số học α, số các đoạn lặp *episode* được thực hiện, nhấn nút *Initialize* để khởi tạo. Sau đó nhấn nút *episode*, thuật toán Q-Learning được cài đặt sẵn sẽ cho chúng ta kết quả các giá trị tại mỗi ô trạng thái, cũng như chiến lược tối ưu tại từng trạng thái. Chúng ta cũng biết được giá trị của chiến lược tối ưu *Score Policy*.



Hình 3.5: Q-Learning

Chức năng *P.Sweeping*:

Sử dụng thuật toán Prioritized Sweeping để giải bài toán.



Hình 3.6: P.Sweeping

Trên cơ sở cài đặt bài toán mô phỏng, ta tiến hành kiểm tra theo các kịch bản sau đây để có được các đánh giá nhận xét kết luận về các giải thuật RL.

3.3.1 Kịch bản 1: Thay đổi kích thước không gian trạng thái

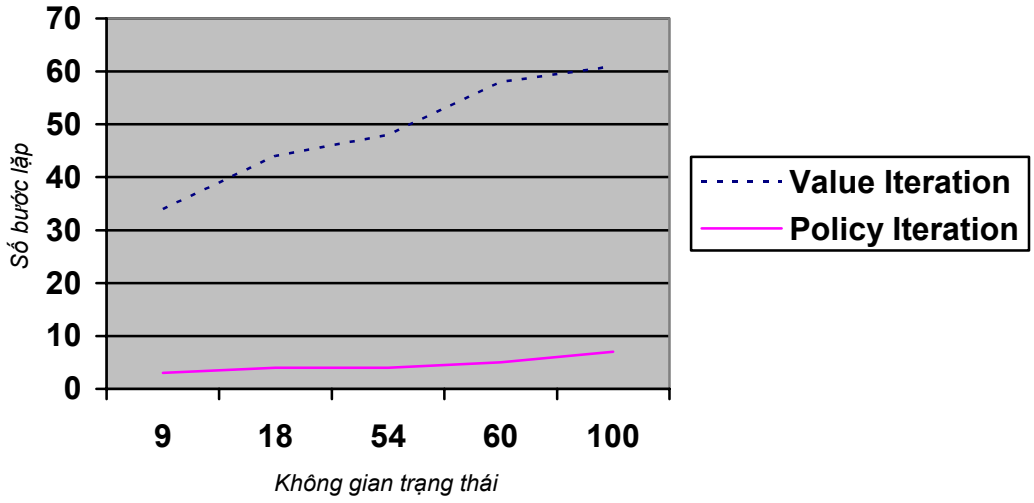
Tiến hành thay đổi kích thước không gian trạng thái của bài toán để kiểm tra độ hội tụ cũng như thời gian thực hiện thuật toán lặp giá trị và thuật toán lặp chiến lược.

Bằng cách chạy các thuật toán lặp giá trị và lặp chiến lược trên các mẫu bài toán mê lộ với số lượng ô khác nhau (mỗi ô tương đương với 1 trạng thái). Ta có các số liệu thống kê như sau:

3.3.1.1 Số bước hội tụ

Không gian trạng thái (Số trạng thái)	*Value Iteration (Số bước hội tụ)*	*Policy Iteration (Số bước hội tụ)*
9	34	3
18	44	4
54	48	4
60	58	5
100	61	7

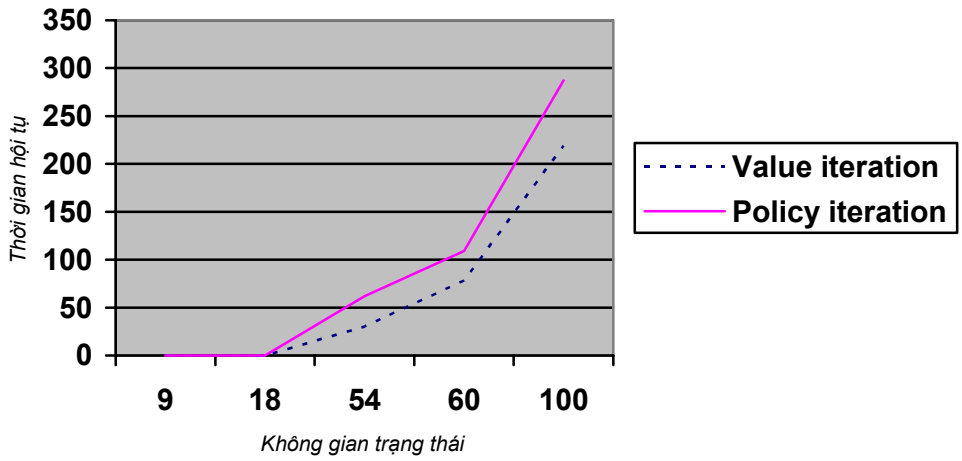
Từ các số liệu thực nghiệm trên ta có biểu đồ biểu diễn mối quan hệ giữa không gian trạng thái và số bước lặp trong thuật toán lặp giá trị và lặp chiến lược như sau:



3.3.1.2 Thời gian hội tụ

Không gian trạng thái (Số trạng thái)	*Value Iteration (Thời gian hội tụ: ms)*	*Policy Iteration (Thời gian hội tụ: ms)*
9	0	0
18	0	0
54	30	62
60	78	109
100	219	287

Từ các số liệu thực nghiệm trên ta có biểu đồ biểu diễn mối quan hệ giữa không gian trạng thái và thời gian hội tụ trong thuật toán lặp giá trị và lặp chiến lược như sau:



3.3.1.3 Phân tích kết quả

Thuật toán lặp chiến lược cần ít số bước lặp đến khi hội tụ hơn so với thuật toán lặp giá trị cho cùng một bài toán mê lộ, nhưng thời gian cần thực hiện thuật toán lặp chiến lược lại lớn hơn so với thuật toán lặp giá trị. Hiệu năng của cả hai thuật toán này phụ thuộc vào tỷ lệ số các hành động đối với số các trạng thái. Tỷ lệ càng cao thì hiệu năng của thuật toán càng cao.

3.3.1.4 Giải pháp cải thiện

Phép lặp chiến lược hội tụ với ít số bước lặp hơn nhưng thời gian thực hiện một bước lặp thì nhiều hơn so với phép lặp giá trị. Lý do chính của vấn đề này chính là bước đánh giá chiến lược trong phép lặp chiến lược. Trong bước đó, việc phải giải quyết để tìm ra giá trị chính xác của V(S) cho chiến lược đưa ra mất chi phí khá nhiều so với việc tính thay đổi trong $V_{t+1}(s)$ rất nhỏ khi t tăng. Như vậy, thay vì tìm chính xác giá trị của V(S) cho chiến lược đưa ra, một vài bước lặp giá trị có thể thực hiện như là thay đổi trong hàm giá trị không được nhận biết (nhỏ hơn một ngưỡng nào đó). Do đó, bước đánh giá chiến lược của phương pháp lặp chiến lược được thay đổi như sau để làm tăng hiệu năng của nó.

repeat
  for all $s \in S$ **do**

$$V_{t+1}(s) = PCost + \sum_{s' \in S} (P(s'|s, \pi(s)) \cdot x_{ss'})$$

  end for
until change in $V(s)$ is not significant

3.3.1.5 Kết luận

Phép lặp chiến lược lý tưởng đối với các bài toán có không gian hành động lớn bởi vì nó làm giảm việc xem xét không gian hành động trong ít bước lặp hơn. Trong khi phép lặp giá trị lý tưởng cho những bài toán có không gian trạng thái lớn.

3.3.2 Kịch bản 2: Thay đổi hệ số học

Ta phân tích tác động của hệ số học trong phương pháp Q-Learning và đề xuất hệ số học phù hợp cho môi trường không ổn định.

3.3.2.1 Phân rã hệ số học theo số đoạn lặp

Ở đây ta sử dụng thử nghiệm phương pháp phân rã hệ số học trong thuật toán Q-Learning theo công thức:

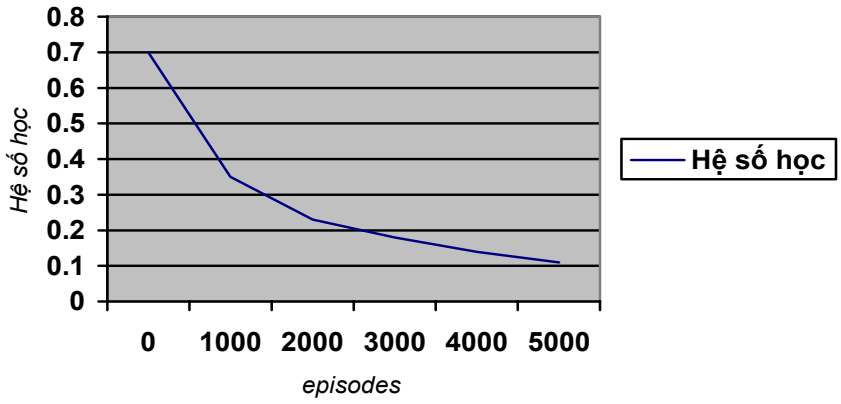
*Hệ số học = (1000 * max Hệ số học)/(1000 + số đoạn lặp)*

Lấy: *max Hệ số học = 0.7*

Ta có các số liệu thống kê thực nghiệm như sau:

Số đoạn lặp (Episodes)	*Hệ số học (α)*
0	0.7
1000	0.35
2000	0.23
3000	0.18
4000	0.14
5000	0.11

Từ các số liệu thực nghiệm trên ta có biểu đồ biểu diễn mối quan hệ giữa hệ số học với số đoạn lặp trong thuật toán Q-Learning như sau:



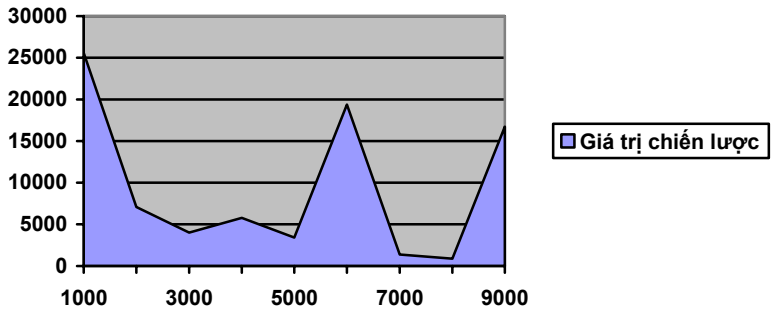
3.3.2.2 Mối quan hệ giữa giá trị chiến lược và hệ số học

Thử nghiệm hệ số học = 0.1

Bảng các số liệu thống kê:

Số đoạn lặp (Episodes)	*Giá trị chiến lược*
1000	25581
2000	7092
3000	4005
4000	5768
5000	3428
6000	19398
7000	1394
8000	902
9000	16735

Từ các số liệu thực nghiệm ở trên ta vẽ biểu đồ biểu diễn giá trị chiến lược khi số bước lặp tăng trong thuật toán Q-Learning như sau:



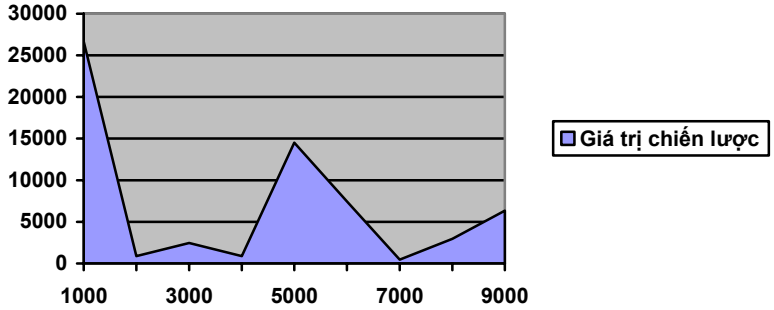
Thử nghiệm hệ số học = 0.001

Bảng các số liệu thống kê:

Số các đoạn lặp (Episodes)	*Giá trị chiến lược*
1000	26645
2000	870
3000	2468
4000	875
5000	14495
6000	7472
7000	453

8000	2955
9000	6348

Từ các số liệu thực nghiệm ở trên ta vẽ biểu đồ biểu diễn giá trị chiến lược khi số bước lặp tăng trong thuật toán Q-Learning như sau:



3.3.2.3 Phân tích kết quả

Với hệ số học = 0.1, tác tử cải thiện chiến lược của nó nhanh hơn nhưng không ổn định ngay đến chiến lược tối ưu mà giữ sự dao động giữa vị trí gần một chiến lược tối ưu và một chiến lược tồi. Trong khi với hệ số học = 0.001, tác tử tiến tới chiến lược tối ưu chậm nhưng đều đặn và chắc chắn.

3.3.2.4 Giải pháp cải thiện

Qua kết quả thử nghiệm ta thấy rằng trong thực tế, tác tử nhạy cảm đối với giá trị tăng cường nó nhận từ môi trường hơn nếu hệ số học cao hơn. Trong môi trường nhiễu, tác tử thực hiện một hành động nhưng có thể không phải là chuyển sang trạng thái tiếp theo được kỳ vọng mà nó bị đẩy sang trạng thái kề có xác suất chắc chắn. Trong môi trường nhiễu, tác tử có thể thực hiện hành động tối ưu nhưng kết quả lại là bước di chuyển sang một trạng thái tồi hoặc nhận một giá trị phạt. Trong tình huống như vậy, nếu hệ số học lớn, hàm giá trị tính cho cặp trạng

thái-hành động thay đổi đáng kể, dẫn đến tác tử thay đổi độ tin cậy về khả năng tối ưu của hành động và tác tử thay đổi chiến lược của nó. Tuy nhiên, khi hệ số học lớn, tác tử lại nhạy cảm hơn với nhiễu môi trường tạo nên bất lợi.

Như vậy, để đạt được tối ưu, ta cần sử dụng hệ số học lớn trong những giai đoạn đầu của tương tác giữa tác tử và môi trường, giúp cho tác tử có thể học được tính động của môi trường nhanh chóng. Sau đó phải thực hiện giảm hệ số học để tiến dần đến chiến lược tối ưu. Đây chính là việc sử dụng phép phân rã hệ số học trong suốt thời gian thực hiện thuật toán.

3.3.2.5 Kết luận

Hệ số học có giá trị lớn hoạt động tốt hơn hệ số học có giá trị nhỏ trong những giai đoạn đầu của tương tác tác tử với môi trường trong khi hệ số học có giá trị nhỏ hoạt động tốt nhất trong các giai đoạn sau của tương tác tác tử với môi trường.

3.3.3 Kịch bản 3: Thay đổi số đoạn lặp

3.3.3.1 Mối quan hệ giữa giá trị chiến lược và số đoạn lặp

Ta phân tích tác động của số đoạn lặp đến giá trị chiến lược thu được trong phương pháp Q-Learning trong cả hai trường hợp không có sự phân rã hệ số học và có sự phân rã hệ số học.

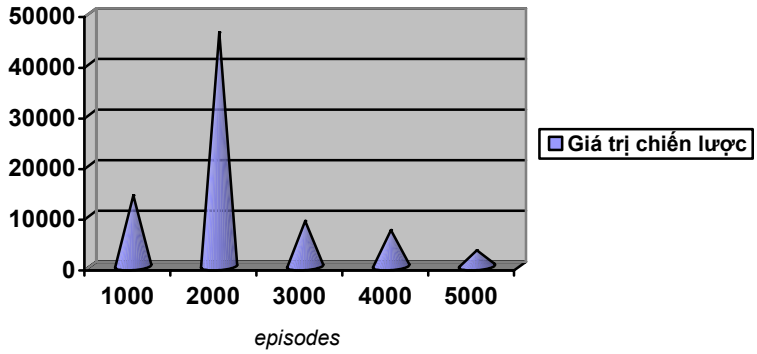
Thay đổi số đoạn lặp khi có phân rã hệ số học

Ta thực nghiệm bài toán với thuật toán Q-Learning trong đó hệ số học không ổn định mà thay đổi theo công thức phân rã trong suốt quá trình chạy thuật toán. Thông tin thực nghiệm thu được như sau:

Số đoạn lặp (Episodes)	*Giá trị chiến lược*
1000	13952

2000	46139
3000	8863
4000	7131
5000	3134

Từ các số liệu thực nghiệm ở trên ta vẽ biểu đồ biểu diễn giá trị chiến lược khi số bước lặp tăng trong thuật toán Q-Learning như sau:

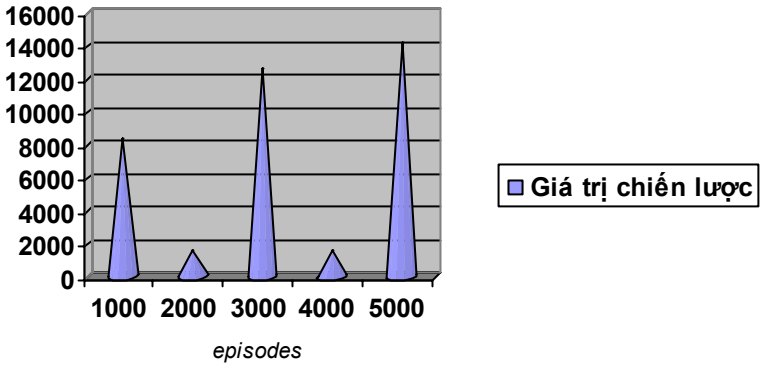


Thay đổi số đoạn episode khi không có phân rã hệ số học

Ta thực nghiệm bài toán với thuật toán Q-Learning trong đó hệ số học ổn định không đổi trong suốt quá trình chạy thuật toán. Thông tin thực nghiệm thu được như sau:

Số đoạn lặp (Episodes)	Giá trị chiến lược
1000	8405
2000	1550
3000	12558
4000	1634
5000	14108

Từ các số liệu thực nghiệm ở trên ta vẽ biểu đồ biểu diễn giá trị chiến lược khi số bước lặp tăng trong thuật toán Q-Learning như sau:



3.3.3.2 Phân tích đánh giá kết quả

Từ các kết quả thực nghiệm ta thấy rằng khi số các đoạn lặp tăng (trong trường hợp có sự phân rã hệ số học thì hệ số học trở nên rất nhỏ), các thay đổi trong môi trường không làm thay đổi nhiều giá trị Q. Do đó, tác tử sẽ duy trì việc thực hiện các chiến lược tối ưu. Khi số đoạn lặp thay đổi, giá trị chiến lược mà tác tử thực hiện cần nhiều thời gian hơn để tiến đến giá trị tối ưu. Có nghĩa là tác tử cần nhiều thời gian hơn để tìm ra được chiến lược tối ưu.

3.3.4 Kịch bản 4: Thay đổi chiến lược lựa chọn

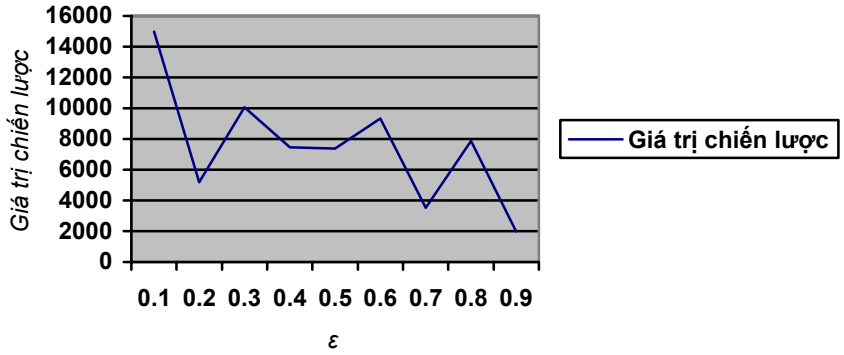
3.3.4.1 Mối quan hệ giữa giá trị chiến lược và tham số chiến lược

Thay đổi giá trị ε của chiến lược lựa chọn ε-Greedy. Quan sát các thông tin về sự thay đổi giá trị chiến lược như sau:

Tham số lựa chọn (ε)	Giá trị chiến lược
0.1	14975
0.2	5193
0.3	10064
0.4	7461
0.5	7375
0.6	9327

0.7	3531
0.8	7885
0.9	1984

Từ các số liệu thực nghiệm ở trên ta vẽ biểu đồ biểu diễn giá trị chiến lược khi tham số lựa chọn chiến lược thay đổi trong thuật toán Q-Learning như sau:



3.3.4.2 Phân tích đánh giá kết quả

Khi thay đổi tham số ε và giữ nguyên giá trị các tham số khác, có thể thấy giá trị chiến lược nhận được giảm theo chiều tăng tham số ε.

ĐÁNH GIÁ KẾT LUẬN

Trong suốt quá trình học hỏi và nghiên cứu làm luận văn, em đã nắm bắt được các vấn đề liên quan đến phương pháp học tăng cường, những ứng dụng thiết thực của nó vào các bài toán thực tế hiện nay. Hiểu rõ ý tưởng, cơ chế hoạt động của các thuật toán học tăng cường phổ biến, cách thức áp dụng chúng trong các bài toán cụ thể.

Em cũng đã tìm hiểu một số bộ công cụ phát triển RL đã có, trên cơ sở đó cài đặt chương trình thử nghiệm mô phỏng bài toán. Đưa ra các kịch bản thử nghiệm để đánh giá các thuật toán. Hướng nghiên cứu trong tương lai là ứng dụng góp phần giải quyết những bài toán quan trọng thiết thực trong bối cảnh xã hội ngày càng hiện đại và phát triển.

TÀI LIỆU THAM KHẢO

Tiếng Anh

1. Bellman, R. (1957). *Applied Dynamic Programming*. Princeton University Press, Princeton, New Jersey.
2. Bertsekas, D. P. (1995). *Dynamic Programming and Optimal Control*, Athena Scientific, Belmont, Massachusetts.
3. Coulom R. (2000): *Reinforcement Learning using Neural Networks*. PhD thesis.
4. Doya K. (1999). *Reinforcement Learning in continuous time and space*.
5. Christ Gaskett (2002). *Q-Learning for Robot Control*, RMIT University.
6. Carlos Henrique Costa Ribeiro. *A Tutorial on Reinforcement Learning Techniques*.
7. Kaelbling L. P. and Littman M. L. *Reinforcement Learning: a Survey*.
8. Puterman, M. L. (1994). *Markovian Decision Problems*.
9. Robinson A. (May 7, 2002). *CS 242 FinalProject: ReinforcementLearning*.
10. Singh, S. P. (1994). *Learning to Solve Markovian Decision Processes*. PhD thesis, University of Massachusetts.
11. Sutton R. and Barto A. (1998). *Reinforcement Learning*: *An Introduction*, MIT Press.
12. V. Gullapalli V. (1992): *Reinforcement Learning and its application to control*.
13. William D. Smart and Leslie Pack Kaelbling (2002). *Effective Reinforcement Learning for Mobile Robots*.
14. Whitehead, S. D. and Lin, L.-J. (1995), *Reinforcement learning of non-markov decision processes*.
15. *Java Reinforcement Learning Framework*