KĨ THUẬT TRÍ TUỆ NHÂN TẠO

Thuật toán Tìm Kiếm
Monte-Carlo tree search

GIÁO VIÊN HƯỚNG DẪN:
1. Nguyễn Ngọc Thảo

NH ÓM SV TH ỰC HI ỆN:
HBGroup:
1. 1512005 - Nguyễn Hoàng Anh
2. 1512031 - Huỳnh Cao Biên

Khoa Công nghệ Thông tin
Đại học Khoa học Tự nhiên TP HCM

Tháng 3/2018

1 THÔNG TIN THÀNH VIÊN THỰC HIỆN ĐỒ ÁN

MSSV	Họ tên	Email
1512005	Nguyễn Hoàng Anh	1512005@student.hcmus.edu.vn
1512031	Huỳnh Cao Biên	1512209@student.hcmus.edu.vn

2 BẢNG PHÂN CÔNG CÔNG VIỆC

MSSV-Họ Tên	Hỗ trợ review	Phân công	Tài liệu nghiên cứu *(Mục tài liệu tham khảo)*	Thời gian thực hiện		Rút trích thông tin
				Start	End	
1512031-Huỳnh Cao Biên 1512005-Nguyễn Hoàng Anh		Đọc nghiên cứu tài liệu và chọn thuật toán tìm kiếm	[1] [2] [3]	10/3/2018	13/3/2018	• Chỉ chọn thuật toán tree heuristic có mô hình và thuật toán seảch rõ ràng, mới.
1512031-Huỳnh Cao Biên 1512005-Nguyễn Hoàng Anh		Nghiên cứu thuật toán	[3]: Chương 1,2,3,5,6,7	13/3/2018	15/3/2018	• Tìm hiểu sự khác biệt giữa các thuật toán liên quan • Tìm hiểu mô hình, giải thích thuật toán, mô phỏng thuật toán. • Giải thích công thức • Tính ứng dụng
1512031-Huỳnh Cao Biên	1512005-Nguyễn Hoàng Anh	• Viết báo váo, • Thiết kế trình chiếu powerpoint	[3]: Chương 1,2,3,5,6,7	15/3/2018	15/3/2018	• Introduction • Simulation-based search: Two-player games, Simulation, Monte-Carlo simulation, Monte-Carlo tree search, Algorithm
1512005-Nguyễn Hoàng Anh	1512031-Huỳnh Cao Biên	Data & Source code demo	Internet	15/3/2018	15/3/2018	• Tìm kiếm tập dữ liệu test, báo cáo thông kê thực nghiệm và source code liên quan đến đồ án.

3 Nội dung

3.1 Introduction

Các phương pháp Monte Carlo là một lớp các thuật toán để giải quyết nhiều bài toán trên máy tính theo kiểu không tất định, thường bằng cách sử dụng các số ngẫu nhiên (thường là các số giả ngẫu nhiên), ngược lại với các thuật toán tất định. Một ứng dụng cổ điển của phương pháp này là việc tính tích phân xác định, đặc biệt là các tích phân nhiều chiều với các điều kiện biên phức tạp.

Cây tìm kiếm Monte-Carlo là một mô hình mới cho tìm kiếm, đã cách mạng máy tính Go, và nhanh chóng thay thế các thuật toán tìm kiếm truyền thống như là phương pháp được lựa chọn trong các lĩnh vực đầy thử thách như General Game Playing, Amazons, Lines of Action , các trò chơi thẻ nhiều người, và các trò chơi chiến lược thời gian thực. Ý tưởng chính là để mô phỏng hàng ngàn trò chơi ngẫu nhiên từ vị trí hiện tại, sử dụng tự chơi. Vị trí mới được thêm vào cây tìm kiếm, và mỗi nút của cây chứa một giá trị dự đoán ai sẽ thắng từ vị trí đó. Những dự đoán được cập nhật bởi mô phỏng Monte-Carlo: giá trị của một nút đơn giản là kết quả trung bình của tất cả các mô phỏng trò chơi truy cập vào vị trí. Cây tìm kiếm được sử dụng để hướng dẫn mô phỏng theo các đường dẫn đầy hứa hẹn, bằng cách chọn con nút với giá trị tiềm năng cao nhất. Điều này dẫn đến một tìm kiếm có tính chọn lọc cao, rất nhanh chóng xác định được bước đi tốt chuỗi.

Phần mở rộng, tìm kiếm cây săn Monte-Carlo, sử dụng một hàm heuristic để khởi tạo các giá trị của các vị trí mới trong cây tìm kiếm

3.2 Simulation-based search

Cây tìm kiếm Monte-Carlo mô phỏng

Cây tìm kiếm Monte-Carlo (MCTS) sử dụng mô phỏng Monte-Carlo để đánh giá các nút của cây tìm kiếm. Các giá trị trong cây tìm kiếm sau đó được sử dụng để chọn hành động tốt nhất trong các mô phỏng tiếp theo. Cây tìm kiếm Monte-Carlo mô phỏng dung phép suyệt tuần tự từ trái qua phải, khi một node được chọn thì node đó sẽ được duyệt đến cuối cùng khi nào chương kết thúc chu kì của node đó thì thôi, khí đó giá trị các node trên đường đi sẽ được cập nhật theo một công thức cho trước.

Cây tìm kiếm T chứa một nút, n(s), tương ứng với mỗi trạng thái đã được nhìn thấy trong quá trình mô phỏng. Mỗi nút chứa tổng số đếm cho trạng thái, N(s), và một giá trị hành động Q(s, a) và đếm N(s, a) cho mỗi hành động a ∈ A.

Mô phỏng bắt đầu từ trạng thái gốc s0, và được chia thành hai giai đoạn. Khi st trạng thái được biểu diễn trong cây tìm kiếm, st ∈ T, chính sách cây được sử dụng để chọn hành động. Nếu không, một chính sách mặc định được sử dụng để mô phỏng các mô phỏng để hoàn thành. Phiên bản đơn giản nhất của thuật toán mà chúng ta gọi là MCTS tham lam, chọn hành động tham lam có giá trị cao nhất trong giai đoạn đầu tiên, argmax(a) Q (st, a); và chọn các hành động thống nhất một cách ngẫu nhiên trong giai đoạn thứ hai.

Mỗi trạng thái và hành động trong cây tìm kiếm được đánh giá bằng kết quả trung bình trong quá trình mô phỏng. Sau mỗi mô phỏng s0, a0, s1, a1, ..., sT với kết quả z, mỗi nút trong cây tìm kiếm, {n(st) | st ∈ T}, cập nhật số của nó, và cập nhật giá trị hành động Q (st, at) đến giá trị MC mới (tương đương (3)). Bản cập nhật này cũng có thể được thực hiện từng bước mà không cần xem xét lại các mô phỏng trước, bằng cách tăng số đếm và cập nhật giá trị đối với kết quả z.

$N(st) \leftarrow N(st) + 1$, (4)

$N(st,at) \leftarrow N(st,at) + 1$, (5)

$Q(st,at) \leftarrow Q(st,at) + (z - Q(st,at))/N(st,at)$. (6)

Monte-carlo cải tiến

Lựa chọn hành động tham lam thường là một cách không hiệu quả để xây dựng một cây tìm kiếm vì nó thường tránh tìm kiếm hành động sau một hoặc nhiều kết quả nghèo nàn, ngay cả khi có sự không chắc chắn đáng kể về giá trị của những hành động đó. Khám phá cây tìm kiếm hiệu quả hơn, có thể áp dụng nguyên tắc lạc quan khi đối mặt với sự không chắc chắn, có lợi cho hành động với giá trị tiềm năng lớn nhất. Để thực hiện nguyên tắc này, mỗi giá trị hành động nhận được tiền thưởng tương ứng với số tiền không chắc chắn trong giá trị hiện tại của nhà nước và hành động.
Thuật toán UCT áp dụng nguyên tắc này để tìm kiếm Monte-Carlo tree, bằng cách xử lý mỗi trạng thái của cây tìm kiếm như là một đa năng kẻ cướp, trong đó mỗi hành động tương ứng với một cánh tay của tên cướp. Chính sách cây lựa chọn hành động bằng cách sử dụng Thuật toán UCB1, tối đa hóa sự tự tin trên, liên quan đến giá trị của các hành động.
Cụ thể giá trị của mỗi node được cập nhật theo công thức sau:

$$Q^{\oplus}(s,a) = Q(s,a) + c\sqrt{\frac{\log N(s)}{N(s,a)}}, \tag{7}$$

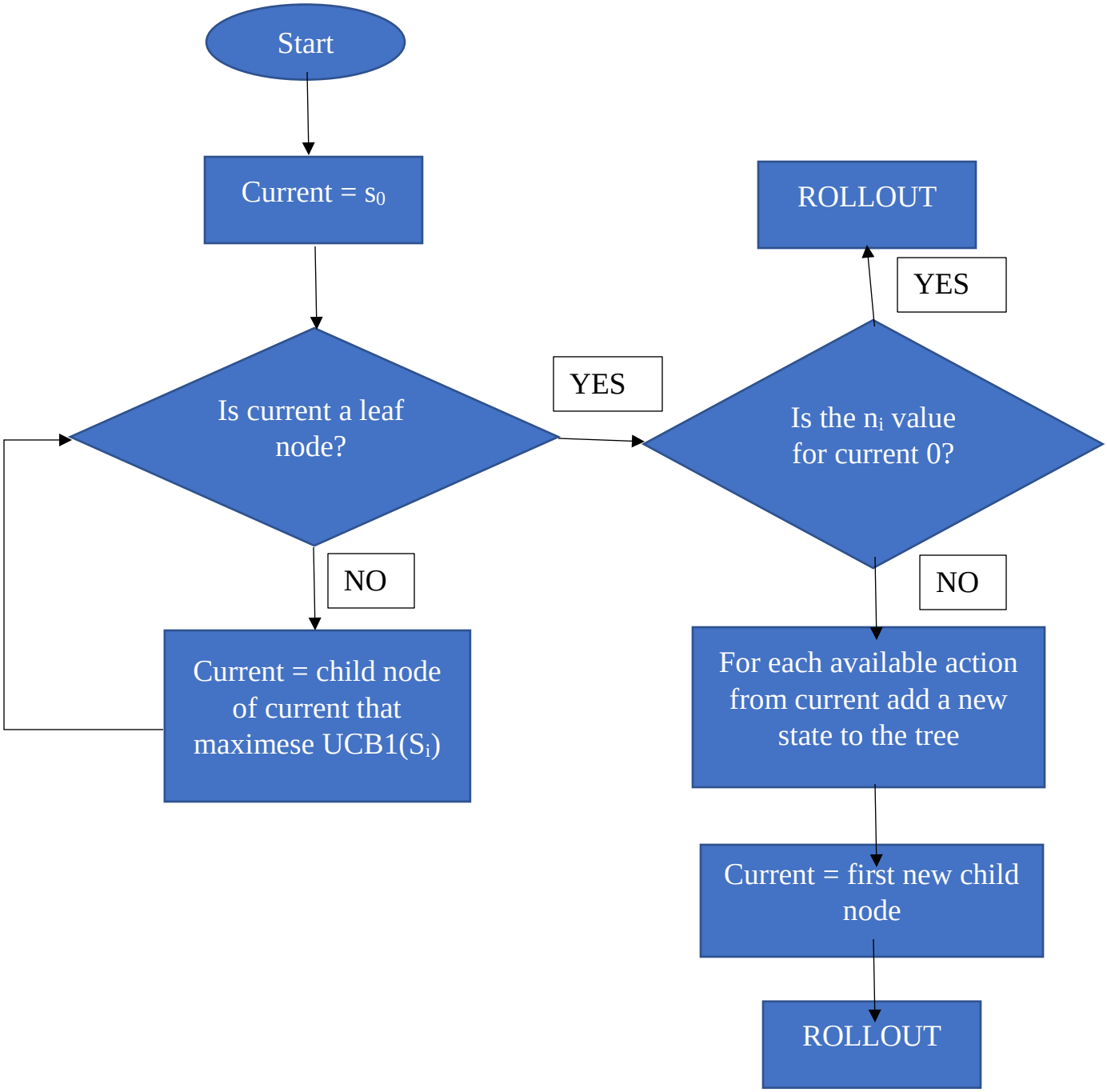
$$a^* = \underset{a}{\operatorname{argmax}}\, Q^{\oplus}(s,a) \tag{8}$$

Trong đó c là hằng số thuộc miền số thực R, log là logarit của số tự nhiên.
Dưới đây là đoạn mã giả mô phỏng giải thuật cải tiến:

Algorithm 1 Two-player UCT

```
procedure UctSearch(s_0)
    while time available do
        Simulate(board, s_0)
    end while
    board.SetPosition(s_0)
    return SelectMove(board, s_0, 0)
end procedure

procedure Simulate(board, s_0)
    board.SetPosition(s_0)
    [s_0, ..., s_T] = SimTree(board)
    z = SimDefault(board)
    Backup([s_0, ..., s_T], z)
end procedure

procedure SimTree(board)
    c = exploration constant
    t = 0
    while not board.GameOver() do
        s_t = board.GetPosition()
        if s_t ∉ tree then
            NewNode(s_t)
            return [s_0, ..., s_t]
        end if
        a = SelectMove(board, s_t, c)
        board.Play(a)
        t = t + 1
    end while
    return [s_0, ..., s_{t-1}]
end procedure

procedure SimDefault(board)
    while not board.GameOver() do
        a = DefaultPolicy(board)
        board.Play(a)
    end while
    return board.BlackWins()
end procedure

procedure SelectMove(board, s, c)
    legal = board.Legal()
    if board.BlackToPlay() then
        a* = argmax_{a∈legal}(Q(s,a) + c*sqrt(log N(s) / N(s,a)))
    else
        a* = argmin_{a∈legal}(Q(s,a) - c*sqrt(log N(s) / N(s,a)))
    end if
    return a*
end procedure

procedure Backup([s_0, ..., s_T], z)
    for t = 0 to T do
        N(s_t) = N(s_t) + 1
        N(s_t, a_t) += 1
        Q(s_t, a_t) += (z - Q(s_t, a_t)) / N(s_t, a_t)
    end for
end procedure

procedure NewNode(s)
    tree.Insert(s)
    N(s) = 0
    for all a ∈ A do
        N(s,a) = 0
        Q(s,a) = 0
    end for
end procedure
```

Sơ đồ thuật toán:



Rollout(S_i):

Loop forever:

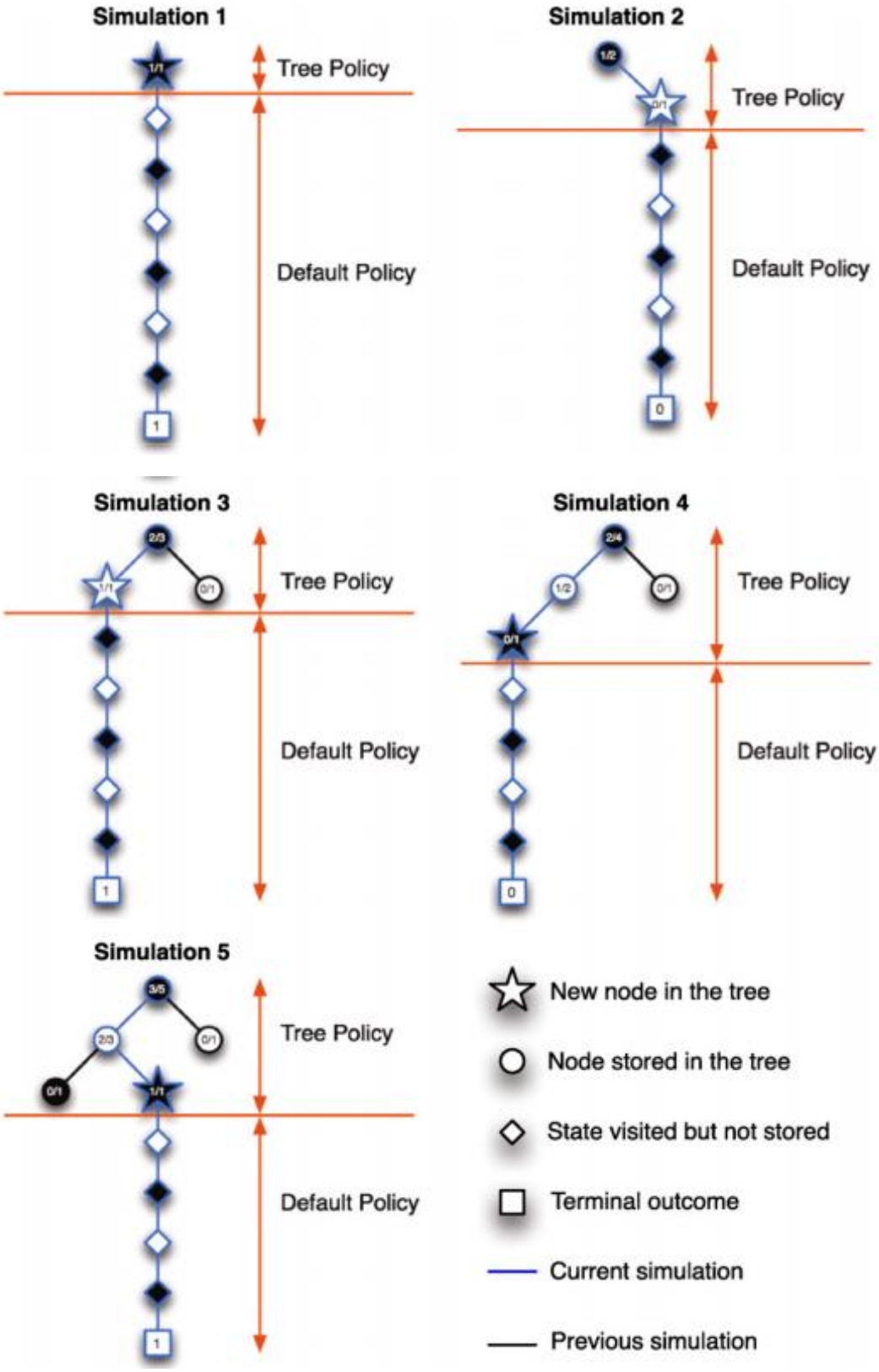
If S_i is a terminal state

Return value(S_i)

Ai = random(available action)

S_i = simulation(S_i, a_i)

End Loop


Simulation 1
Tree Policy
Default Policy
1/1
1
Simulation 2
Tree Policy
Default Policy
1/2
0/1
0
Simulation 3
Tree Policy
Default Policy
2/3
1/1
0/1
1
Simulation 4
Tree Policy
Default Policy
2/4
1/2
0/1
0/1
0
Simulation 5
Tree Policy
Default Policy
3/5
2/3
0/1
0/1
1/1
1
New node in the tree
Node stored in the tree
State visited but not stored
Terminal outcome
Current simulation
Previous simulation

3.3 Tính ứng dụng

Phương pháp Monte Carlo có một vị trí hết sức quan trọng trong vật lý tính toán và nhiều ngành khác, có ứng dụng bao trùm nhiều lĩnh vực, từ tính toán trong sắc động lực học lượng tử, mô phỏng hệ spin có tương tác mạnh, đến thiết kế vỏ bọc nhiệt hay hình dáng khí động lực học. Các phương pháp này đặc biệt hiệu quả khi giải quyết các phương trình vi-tích phân; ví dụ như trong mô tả trường bức xạ hay trường ánh sáng trong mô phỏng hình ảnh 3 chiều trên máy tính, có ứng dụng trong trò chơi điện tử, kiến trúc, thiết kế, phim tạo từ máy tính, các hiệu ứng đặc biệt trong điện ảnh, hay trong nghiên cứu khí quyển, và các ứng dụng nghiên cứu vật liệu bằng laser...

4 Tài liệu tham khảo

1. *https://www.sciencedirect.com/science/article/pii/0004370290900544*
2. https://www.sciencedirect.com/science/article/pii/S000437021100052X
3. Sylvain Gelly Université Paris Sud, LRI, CNRS, INRIA, France, David Silver University College London, UK: "*Monte-Carlo tree search and rapid action value estimation in computer Go*", Artificial Intelligence 175 (2011) 1856–1875.