1 Search Algorithms (20 pts)

1. Đường đi từ S->G của BFS: S-D-G

Tổng chi phí: total\_cost = 2+8 = 10

1. Đường đi từ S->G của DFS: S-C-F-E-G

Tổng chi phí: total\_cost = 2+1+0+2 = 5

1. Đường đi từ S->G của Iterative deepening (giả sử depth = 2): S-D-G

Tổng chi phí: total\_cost = 2+8 = 10

2 Tower of Hanoi

1. Một trạng thái có thể được biểu diễn bằng vị trí của mỗi đĩa. Mỗi đĩa có thể nằm ở một trong 3 cọc (A, B, C), nên trạng thái là một vector độ dài n, ví dụ (A, A, B).
2. Kích thước không gian trạng thái là3n.
3. Trạng thái bắt đầu: tất cả n đĩa ở cọc đầu tiên (A), sắp theo thứ tự tăng dần kích thước từ trên xuống.
4. Hành động hợp lệ: chọn đĩa trên cùng của một cọc và di chuyển sang cọc khác, miễn là cọc đích rỗng hoặc có đĩa trên cùng lớn hơn đĩa được di chuyển.
5. Kiểm tra mục tiêu (goal test) là kiểm tra xem **toàn bộ n đĩa có đang nằm trên cọc cuối cùng** hay không, và **được sắp xếp theo thứ tự hợp lệ** (tức là đĩa lớn hơn nằm dưới đĩa nhỏ hơn).

3 Designing & Understanding Heuristic

|  |  |
| --- | --- |
| State | h(s) |
| S | 13 |
| A | 10 |
| B | 6 |
| C | 7 |
| D | 4 |
| G | 0 |

1. Path: S-B-D-G

Cost: f(S) = g(S)+h(S) = 0+13 = 13

f(B) = g(B)+h(B) = 3+6 = 9

f(D) = g(D)+h(D) = 4+4 = 8

f(G) = g(G)+h(G) = 8+0 = 8

|  |  |
| --- | --- |
| State | h(s) |
| S | 8 |
| A | 10 |
| B | 5 |
| C | 3 |
| D | 4 |
| G | 0 |

1. Một heuristic nhất quán khi mọi cặp (n,n’) thỏa: h(n) <= c(n,n') + h(n')

h(n) là chi phí ước lượng từ n đến G

h(n') là chi phí ước lượng từ n' đến G

c(n,n') là chi phí thật từ n đến n'.

Giả sử:

n là 1 trạng thái bất kỳ

n' là trạng thái đích

Lúc này h(n) <= c(n,G) + h(G) mà h(G) mặc định = 0 nên suy ra h(n) <= c(n,G). Như ta đã nói ở trên c(n,G) là chi phí thật từ n đến G nên c(n,G) <=> h\*(n) mà một heuristic khả chấp (admissible) khi h(n)<= h\*(n).

* Heuristic nhất quán thì luôn luôn khả chấp

4 Treasure Hunting

1. Explored set: A, C, F, B, G

Path returned: A → C → G

1. Explored set: A, C, D, E

Path returned: A → E

1. Explored set: A, D, C, G (UCS mở D trước vì g(D)=1 là nhỏ nhất; C và G đều có g=2 → chữ cái quyết định pop C trước G)

Path returned: A → D → G

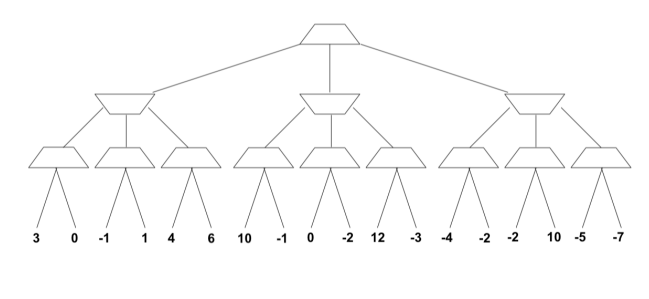
1. Explored set: A, E (từ A, các con có h: E(0), C(1), D(1) → pop E đầu vì h nhỏ nhất)

Path returned: A → E

1. Explored set: A, D, G

Path returned: A → D → G

5 Adversarial Search



X

X

6 True/False Section

1. Sai, A\* với heuristic khả chấp có thể mở rộng ít nút hơn hoặc nhiều hơn, tùy vào heuristic. DFS không đảm bảo số nút mở rộng nhiều hơn hay ít hơn A\*.
2. Đúng. Giả sử ô A(3,5) và B(5,6) thì h(n) = |3-5| + |5-6| = 3 đúng với thực tế là đi 3 ô (1 ô ngang và 2 ô dọc)
3. Đúng, Euclidean ≤ số bước thật (admissible)
4. Sai, Tổng heuristic có thể vượt chi phí thật
5. Sai, Chỉ khi heuristic nhất quán (consistent) thì mới khả chấp (admissible)
6. Sai, A\* với graph search chỉ tối ưu nếu consistent
7. Đúng, Minimax (Vₘ): giả định đối thủ luôn chọn nước tệ nhất cho ta (bi quan).

Expectimax (Vₑ): giả định đối thủ là ngẫu nhiên → giá trị trung bình.

→ Giá trị trung bình luôn ≥ giá trị tệ nhất có thể xảy ra, nên VM ≤VE

1. Sai, Chính sách minimax được thiết kế cho đối thủ tệ nhất, không phải cho ngẫu nhiên.

→Nếu xác suất ngẫu nhiên khác, kỳ vọng điểm có thể thấp hơn VE.

VE chỉ đạt được nếu ta dùng chính sách expectimax tối ưu, chứ không phải minimax policy.