AI\_Introductory\_Assignment 1 Nguyen\_Tran\_Le\_Tuan\_CS3

Source code: <https://github.com/LeTuanCS150901/AI_Introductory/blob/master/Source_Code/AI_Introductory_Assignment_Le_Tuan_CS3.ipynb>

**Bài 1:**

**Step 1**: Đầu tiên ta sẽ khởi tạo Initial State c ho 3 can nước (3L, 8L, 12L) tương đương với (x, y, z) với điều kiện ban đầu:

0 ≤ x ≤ 3, 0 ≤ y ≤ 8, 0 ≤ z ≤ 12

**Step 2**: Ta khởi tạo Goal State để đong được 1 lít nước tương đương với

(1, y, z) với điều kiện:

0 ≤ y ≤ 8, 0 ≤ z ≤ 12

**Step 3**: Ta thực hiện tìm kiếm từ Initial State đến Goal State trong Space Search (không gian trạng thái) dưới đây với mỗi thao tác là một trạng thái khác nhau

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Số Thao tác | Thao tác | X: Ca 3L | Y: Ca 8L | Z: Ca 12L |
| 1 | Initial State | 0 | 0 | 0 |
| 2 | Đổ đầy ca Z | 0 | 0 | 12 |
| 3 | Đổ đầy ca X từ ca Z | 3 | 0 | 9 |
| 4 | Đổ đầy ca Y từ ca Z | 3 | 8 | 1 |

* Kết quả: Sau 3 bước ta đã tìm được kết quả là (3, 8, 1) hay ca X đong được 3L nước, ca Y đong được 8L nước, ca Z đong được 1L nước.

**Bài 2:**

**Step 1**: Ta gọi nhà truyền giáo = ‘A’ và kẻ ăn thịt = ‘B’, sau đó ta khởi tạo Initial State ban đầu là

[ (A A A B B B), ( ) ] – với list bên trong đầu tiên là 6 người ở 1 bên bờ sông và list bên trong thứ hai là không có ai bên kia bờ sông.

**Step 2**: Ta khởi tạo Goal State là [ ( ), (A A A B B B) ] – trạng thái chuyển hết 6 người qua bờ sông bên kia với điều kiện số phần tử B <= A ở mỗi bờ sông hoặc trên thuyền.

**Step 3**: Ta thực hiện tìm kiếm từ Initial State đến Goal State trong Space Search (không gian trạng thái) dưới đây với mỗi thao tác là một trạng thái khác nhau.

|  |  |  |  |
| --- | --- | --- | --- |
| Số Thao tác | Thao tác | Bờ sông trái | Bờ sông phải |
| 1 | Initial State | A, A, A, B, B, B |  |
| 2 | Chuyển A và B qua sông phải | A, A, B, B | A, B |
| 3 | A quay lại bờ sông trái | A, A, A, B, B | B |
| 4 | Chuyển 2B qua sông phải | A, A, A | B, B, B |
| 5 | B quay lại sông trái | A, A, A, B | B, B |
| 6 | Chuyển 2A qua sông phải | A, B | A, A, B, B |
| 7 | A và B quay lại sông trái | A, A, B, B | A, B |
| 8 | Chuyển 2A qua sông phải | B, B | A, A, A, B |
| 9 | B quay lại sông trái | B, B, B | A, A ,A |
| 10 | Chuyển 2B qua sông phải | B | A, A, A, B ,B |
| 11 | B quay lại sông trái | B, B | A, A, A, B |
| 12 | Goal State |  | A, A, A, B, B, B |

* Kết quả: Sau 11 bước ta đã chuyển được hết người truyền giáo và kẻ ăn thịt sang bờ sông bên phải với Goal State là:

[ ( ), (A, A, A, B, B, B) ]

**Bài 3:**

1. Sai. Vì:

* Thuật toán A\* với hàm heuristic chấp nhận được sẽ luôn mở rộng nhiều node nhất có thể để có tính được đi ngắn nhất từ Start đến Goal, trong khi tìm kiếm chiều sâu nếu chỉ được Goal thì sẽ chỉ mở rộng một số lượng node nhất định khi duyệt.

1. Đúng. Vì:

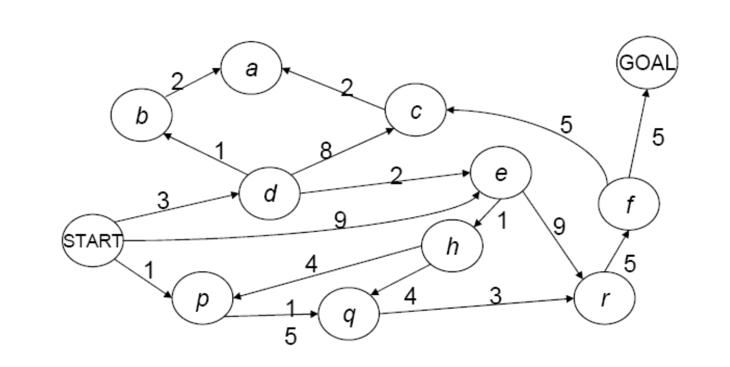
* Khi h(n) = 0, tương đương với việc đã tìm ra lời giải (hay đã đến đích)
* Chấp nhận được

1. Đúng. Vì

* Tìm kiếm theo chiều rộng sẽ duyệt tất cả các node kề với nó

**Bài 4**:

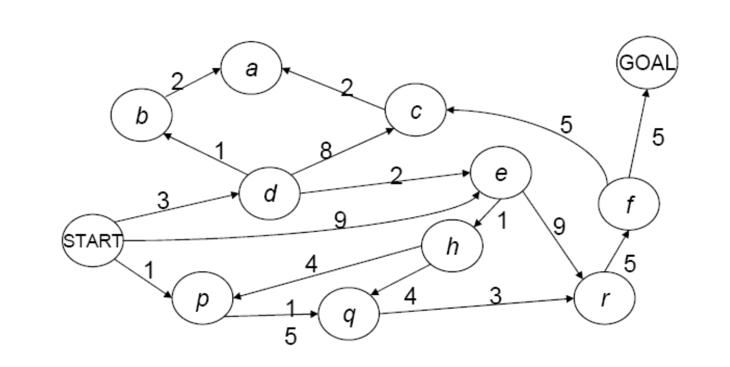
**a. Tìm thiếu theo chiều rộng**



|  |  |
| --- | --- |
| Expand Node | Queue |
|  | S |
| S | DS, ES, PS |
| DS | ES, PS, BD, CD |
| ES | PS, BD, CD, HE, RE |
| PS | BD, CD, HE, RE, QP |
| BD | CD, HE, RE, QP, AB |
| CD | HE, RE, QP, AB |
| HE | RE, QP, AB |
| RE | QP, AB, FR |
| QP | AB, FR |
| AB | FR |
| FR | GF, CF |
| GF | Goal |

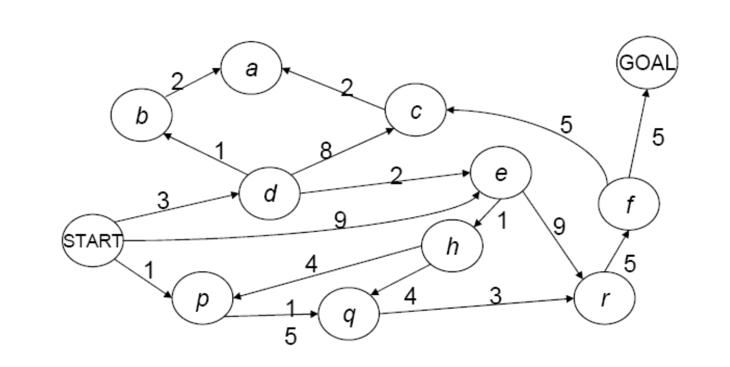
* Path = G🡨F🡨R🡨E🡨S với cost = 28

**b. Tìm kiếm theo chiều sâu**



|  |  |
| --- | --- |
| Expand Node | Stack |
|  | S |
| S | DS, ES, PS |
| DS | ED, ES, PS |
| ED | HE, ES, PS |
| HE | QH, ES, PS |
| QH | RQ, ES, PS |
| RQ | FR, ES, PS |
| FR | GF, ES, PS |
| GF | Goal |

* Path = G🡨F🡨R🡨Q🡨H🡨E🡨D🡨S với cost = 23

**c. Tìm kiếm theo giá thành thống nhất**

|  |  |
| --- | --- |
| Expand Node | Priority Queue |
|  | S |
| S | DS (3), ES (9), PS (1) |
| PS | DS (3), ES (9), QP (6) |
| DS | ED (5), QP (6), BD (4), CB (11) |
| BD | ED (5), QP (6), CB (11), AB (6) |
| ED | QP (6), CB (11), AB (6), HE (6), RE (14) |
| AB | CB (11), QP (6), HE (6), RE (14) |
| HE | CB (11), QP (6), RE (14) |
| QP | CB (11), RQ (9) |
| RQ | CB (11), FR (14) |
| CB | AC (13), FR (14) |
| AC | FR (14) |
| FR | GF (19), CF (19) |
| CF | GF (19), AC (21) |
| GF | Đích |

* Path = G🡨F🡨R🡨Q🡨P🡨S với cost = 19

**d.Tìm kiếm sâu dần**

|  |  |
| --- | --- |
| Expand Node | Stack |
|  | C=0 |
|  | S |
| S |  |
|  | C=1 |
|  | S |
| S | DS, ES, PS |
| DS | ES, PS |
| ES | PS |
| PS |  |
|  | C=2 |
|  | S |
| S | DS, ES, PS |
| DS | BD, CD, ED, ES, PS |
| BD | CD, ED, ES, PS |
| CD | ED, ES, PS |
| ED | ES, PS |
| ES | HE, RE, PS |
| HE | RE, PS |
| RE | PS |
| PS | QP |
| QP |  |
|  | C=3 |
|  | S |
| S | DS, ES, PS |
| DS | BD, CD, ED, ES, PS |
| BD | AB, CD, ED, ES, PS |
| AB | CD, ED, ES, PS |
| CD | AC, ED, ES, PS |
| AC | ED, ES, PS |
| ED | HE, RE, ES, PS |
| HE | RE, ES, PS |
| RE | ES, PS |
| ES | HE, RE, PS |
| HE | QH, RE, PS |
| QH | RE, PS |
| RE | FR, PS |
| FR | PS |
| PS | QP |
| QP | RQ |
| RQ |  |
|  | C=4 |
|  | S |
| S | DS, ES, PS |
| DS | BD, CD, ED, ES, PS |
| BD | AB, CD, ED, ES, PS |
| AB | CD, ED, ES, PS |
| CD | AC, ED, ES, PS |
| AC | ED, ES, PS |
| ED | HE, RE, ES, PS |
| HE | QH, RE, ES, PS |
| QH | RE, ES, PS |
| RE | FR, ES, PS |
| FR | ES, PS |
| ES | HE, RE, PS |
| HE | QH, RE, PS |
| QH | RQ, RE, PS |
| RQ | RE, PS |
| RE | FR, PS |
| FR | CF, GF, PS |
| CF | GF, PS |
| GF | Goal |

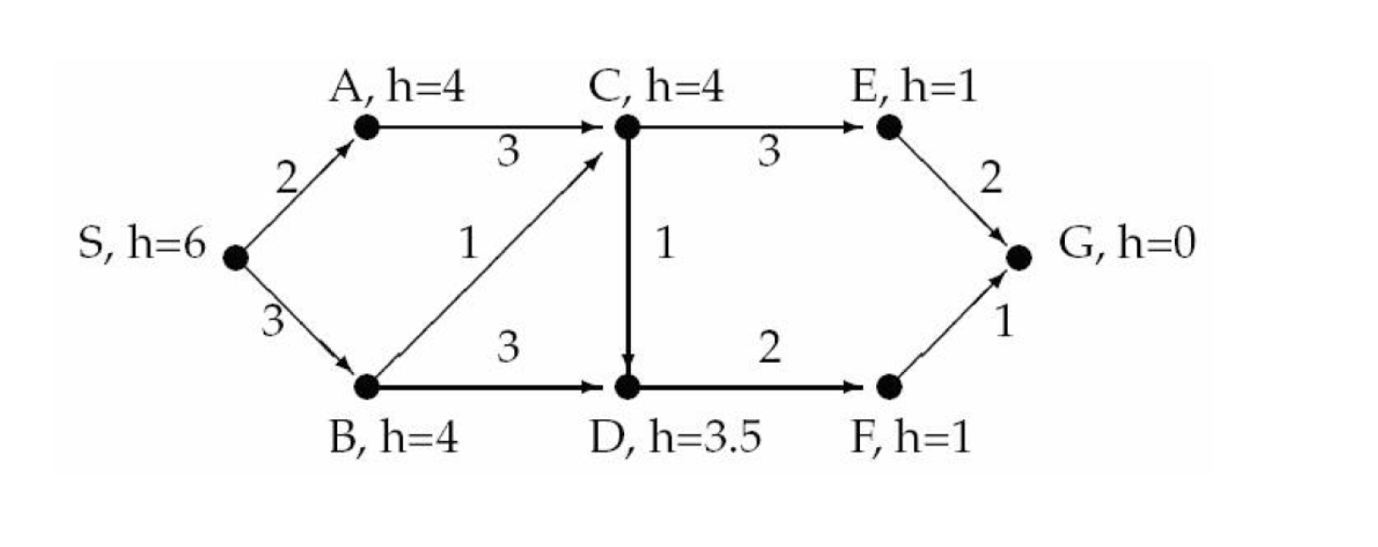
* Path = G🡨F🡨R🡨E🡨S với cost = 28

**Tổng kết**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tìm kiếm | Chiều sâu | Chiều rộng | Giá thành thống nhất | Sâu dần |
| Giá | 23 | 28 | 19 | 28 |

* Trong trường hợp này giá thành thống nhất tìm được đường đi là ngắn nhất

**Bài 5:**



Hàm heuristic chấp nhận được vì:

* Với h(S) = 6 (Tại vị trí Start) ta có đường đi tối ưu nhất từ S đến G sẽ là: h\*(S) = 3+1+1+2+1 =8
* Mà Điều kiện để hàm heuristic chấp nhận được: 0<= h(N) <= h\*(N)
* Hàm heuristic h (N) chấp nhận được

**b. Greedy Best-First Search**

|  |  |
| --- | --- |
| Expand Node | H(n) |
|  | S (6) |
| S | AS (4), BS (4) |
| BS | AS (4), DB (3.5) |
| DB | AS (4), FD (1) |
| FD | AS (4), GF (0) |
| GF | Đích |

=> Path = G 🡨 F 🡨D 🡨 B 🡨 S với cost = 9

**c. Thuật toán A\* với h là hàm heuristic**

|  |  |
| --- | --- |
| Expand Node | H(n) |
|  | S |
| S | AS (4+2), BS (4+3) |
| AS | BS (4+3), CA (4+2+3) |
| CA | BS (4+3), EC (1+2+3+3), DC (3+2+3+1) |
| DC | BS (4+3), EC (1+2+3+3),  FD (1+2+3+1+2) |
| FD | BS (4+3), EC (1+2+3+3),  GF (0+2+3+1+2+1) |
| GF | Goal |

* Đường đi từ S đến G theo thuật toán A\* là: G🡨 F🡨D🡨C🡨A🡨S với cost = 9

Tuy nhiên, đây không phải là đường đi ngắn nhất. Đường đi ngắn nhất sẽ là: G🡨F🡨D🡨C🡨B🡨S với cost = 8.