

TRÍ TUỆ NHÂN TẠO

(0101001005)

Chương 2:

PHƯƠNG PHÁP TÌM KIẾM

Giảng viên: TS. DƯƠNG VĂN HIẾU
BỘ MÔN CNTT, KHOA KTCN-TRƯỜNG ĐH TIỀN GIANG
ĐT: 0988 987 907, email: duongvanhieu@tgu.edu.vn

1. BÀI TOÁN TÌM KIẾM

1.1. Giới thiệu

□ Thả 1 con chó tại điểm A trên bản đồ, cho trước tất cả con đường đi trên bản đồ, yêu cầu con chó đi đến điểm B (ví dụ ngôi nhà)

□ Câu hỏi:

1. Con chó sẽ đi như thế nào?
2. Có bao nhiêu cách đi từ A đến B?
3. Sau bao lâu thì con chó đến điểm B?
4. Có sử dụng tiêu chuẩn gì để lựa chọn đường đi hay không?

DuongVanHieu@tgu.edu.vn

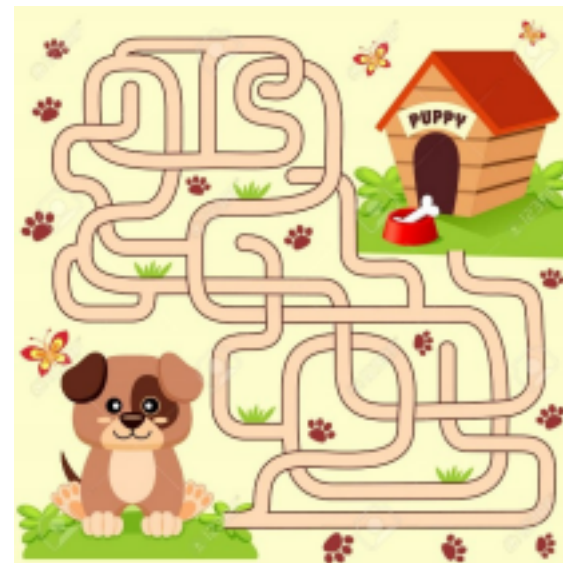


2

1. BÀI TOÁN TÌM KIẾM

1.1. Giới thiệu

- Đối với con chó, mỗi lần di chuyển đến 1 vị trí được đánh dấu bằng “hình dấu chân”
- Trạng thái (vị trí) của con chó có thể là:
 - ✓ Một vị trí nào đó trong số các vị trí dấu chân => **Tất cả vị trí dấu chân là không gian trạng thái**
 - ✓ Vị trí bắt đầu (trạng thái bắt đầu)
 - ✓ Vị trí cần đích (trạng thái đích)
- Mỗi cách đi từ trạng thái đầu đến trạng thái đích là 1 lời giải
- Bài toán có thể có nhiều lời giải



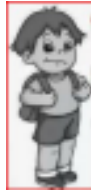
1. BÀI TOÁN TÌM KIẾM

1.1. Giới thiệu

□ Bài toán tìm kiếm là dạng bài toán cho biết trước trạng thái ban đầu, mục tiêu cần đạt hay trạng thái đích; tìm cách đạt được mục tiêu đó với các ràng buộc cho trước

Ví dụ 1:

- Xét đồ thị G có 7 điểm, cho trước chi phí đi từ điểm u đến điểm v trong đồ thị.



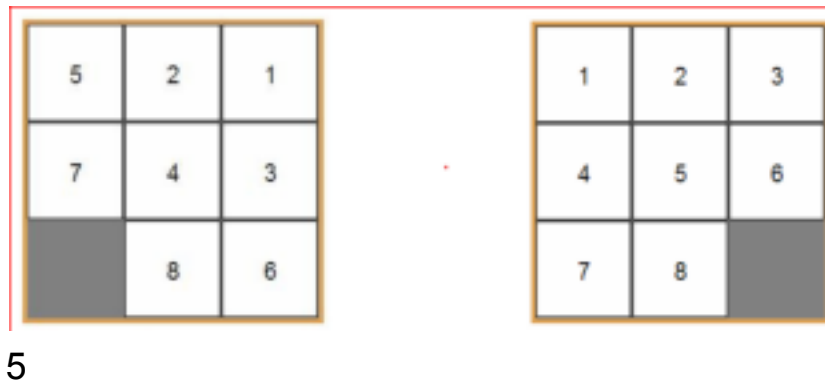
- Tìm cách để người khách đi từ điểm S đến điểm G với chi phí thấp nhất?

1. BÀI TOÁN TÌM KIẾM

1.1. Giới thiệu

Ví dụ 2:

- Cho 1 bàn cờ 9 ô vuông được bố trí sẵn 8 con cờ từ 1 đến 8 (còn 1 ô trống) được gọi là trạng thái ban đầu.
- Mỗi lần di chuyển là hoán vị ô trống với ô kế bên.
- Tìm số lần di chuyển ít nhất để bàn cờ đạt trạng thái đích



DuongVanHieu@tgu.edu.vn

Trạng thái đầu Trạng thái đích

1. BÀI TOÁN TÌM KIẾM

1.2. Định nghĩa

- Bài toán tìm kiếm trong trí tuệ nhân tạo là dạng bài toán có mục tiêu được xác định trước và tìm cách đạt được mục tiêu đó
- Một bài toán tìm kiếm trong trí tuệ nhân tạo được phát biểu gồm 5 phần:
 - ✓ Trạng thái đầu

- ✓ Trạng thái đích
- ✓ Không gian trạng thái bài toán
- ✓ Các phép chuyển trạng thái
- ✓ Chi phí cho các phép chuyển trạng thái

DuongVanHieu@tgu.edu.vn

6

1. BÀI TOÁN TÌM KIẾM

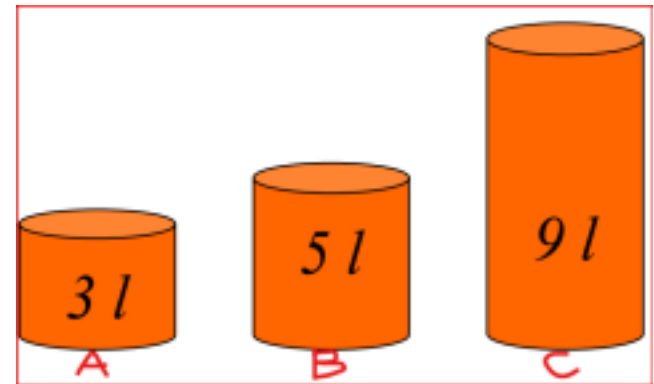
1.2. Định nghĩa

□ Ví dụ 1: Bài toán đong nước:

□ Sử dụng 3 can (3 lít, 5 lít, 9 lít). Làm thế nào để đong được 7 lít nước trong can C?

□ Phát biểu bài toán gồm 5 phần:

- ✓ Trạng thái ban đầu : $(0, 0, 0)$
- ✓ Trạng thái đích: $(x, y, 7)$



- ✓ Không gian trạng thái: bộ 3 số (a, b, c) với a, b, c là số lít nước trong từng can tại sau mỗi lần thay đổi.
- ✓ Phép di chuyển trạng thái: làm rỗng 1 can, chuyển từ can x sang can y cho đến khi hết nước trong can x hoặc đầy can y
- ✓ Chi phí chuyển trạng thái: 1 cho 1 lần di chuyển

DuongVanHieu@tgu.edu.vn

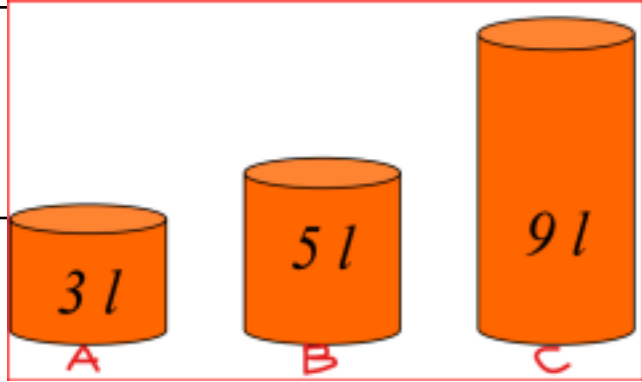
1. BÀI TOÁN TÌM KIẾM

1.2. Định nghĩa

□ Ví dụ 1: Bài toán đóng nước:

□ Ví dụ quá trình tìm lời giải

Trạng thái

7			
Bắt đầu			
d/c 1:	3 l	5 l	9 l
	A	B	C
Đổ 5 lít vào can B			
d/c 2: Đổ 3 lít từ can B vào c			
d/c 3: Đổ 2 lít từ can A vào c			

d/c 4: Đổ 5 lít vào can B⁸ thái đích

d/c 5: Đổ 5 lít từ can B vào C Đạt trạng thái đích (x

1. BÀI TOÁN TÌM KIẾM

1.2. Định nghĩa

□ Ví dụ 2: Bàn cờ 9 ô vuông:

Trạng thái đầu Trạng thái đích

5	2	1
7	4	3
	8	6

1	2	3
4	5	6
7	8	

Trạng

- ✓ Trạng thái đầu: vị trí 8 ô có con cờ lúc ban đầu. ✓ Trạng thái đích: vị trí 8 ô có con cờ cần đạt. ✓ Không gian trạng thái: các cách bố trí 8 con cờ trên bàn cờ
- ✓ Phép di chuyển trạng thái: di chuyển ô trống đi 1 trong các hướng UP, DOWN, LEFT, RIGHT.
- ✓ Chi phí cho mỗi lần di chuyển là 1.

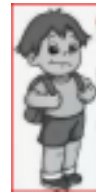
DuongVanHieu@tgu.edu.vn

9

1. BÀI TOÁN TÌM KIẾM

1.2. Định nghĩa

□ Ví dụ 3:



□ Bài toán du lịch



- ✓ Trạng thái đầu: vị trí ban đầu của em bé
- ✓ Trạng thái đích: vị trí cuối cùng mà em bé muốn đến
- ✓ Không gian trạng thái: vị trí của em bé sau mỗi lần di chuyển
- ✓ Phép di chuyển trạng thái: em bé di chuyển từ địa điểm hiện tại đến địa điểm kế tiếp.
- ✓ Chi phí cho mỗi lần di chuyển là x (giá trị nối 2 điểm).

10

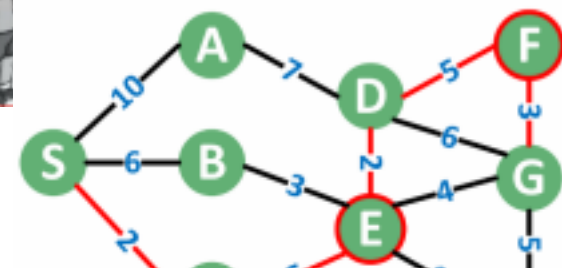
DuongVanHieu@tgu.edu.vn

1. BÀI TOÁN TÌM KIẾM



1.3. Tìm lời giải tối ưu

- Một bài toán có thể có nhiều lời



giải ☐ Phải tìm lời giải tối ưu

- ✓ Dựa trên thông tin của bài toán đã được cho trước
- ✓ Dựa trên thuật toán, tổ chức dữ liệu

✓ Ví dụ:

- ✓ Tìm số lần di chuyển ít nhất (bài toán bàn cờ). ✓
- Tìm cốc lần rót nước ít nhất (bài toán đong nước) ✓
- Tìm đường đi ngắn nhất (bài toán du lịch). ✓
- Tìm đường đi có chi phí thấp nhất (bài toán du lịch)

11

DuongVanHieu@tgu.edu.vn

2. PHƯƠNG PHÁP TÌM KIẾM

2.1. Giới thiệu

- ☐ Có rất nhiều chiến lược tìm kiếm:

- ✓ Được các nhà nghiên cứu đưa ra,
 - ✓ Được các nhà phát triển ứng dụng sử dụng trong cài đặt ứng dụng
- Vấn đề quan trọng là phải lựa chọn chiến lược tìm kiếm phù hợp với đặc điểm của bài toán

12

DuongVanHieu@tgu.edu.vn

2. PHƯƠNG PHÁP TÌM KIẾM

2.2. Tiêu chí lựa chọn chiến lược tìm kiếm □ Lựa chọn chiến lược tìm kiếm phải dựa trên 4 tiêu chí: ✓ Tính

hoàn thành (phải tìm thấy giải pháp)

✓ Độ phức tạp thời gian (thời gian tìm kiếm phải hợp lý) ✓

Độ phức tạp không gian (tiêu bao bộ nhớ hợp lý) ✓ Tính tối ưu (cho kết quả tốt như hoặc hơn chiến lược khác)

13

DuongVanHieu@tgu.edu.vn

2. PHƯƠNG PHÁP TÌM KIẾM

2.3. Giải thuật tìm kiếm tổng quát

□ Đầu vào:

- ✓ **Bài toán (Problem)** với 5 thành phần đã biết là trạng thái đầu, trạng thái đích, không gian trạng thái, cách chuyển trạng thái, chi phí chuyển trạng thái
- ✓ **Chiến lược tìm kiếm**

□ Đầu ra:

- ✓ Lời giải (solution) hoặc thất bại (failure)

□ Thao tác của thuật toán "Sinh ra cây lời giải tìm năng":

- ✓ Có có nút gốc là trạng thái ban đầu của bài toán
- ✓ Mở rộng nút con theo **chiến lược tìm kiếm** đã được chọn

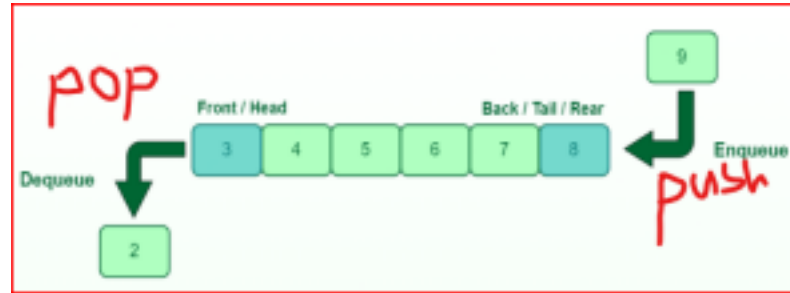
14

DuongVanHieu@tgu.edu.vn

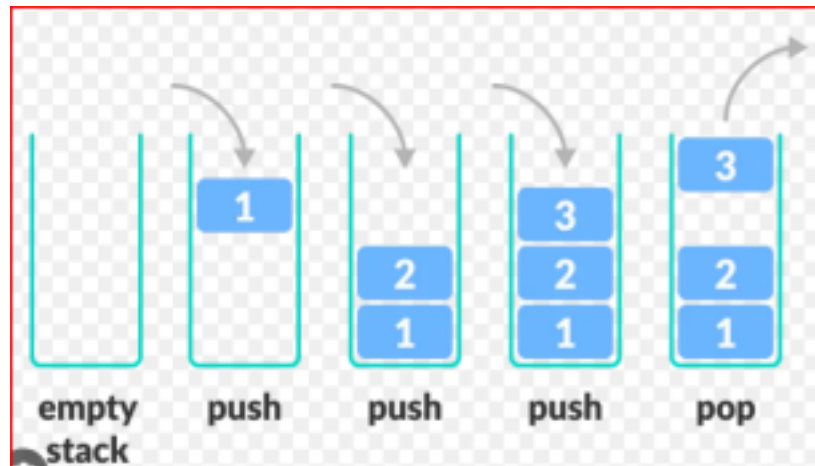
2. PHƯƠNG PHÁP TÌM KIẾM

2.3. Giải thuật tìm kiếm tổng quát

□ **Sử dụng 1 trong 2 cấu trúc dữ liệu cơ bản: ✓ Hàng đợi (queue):** Theo quy tắc vào trước ra trước (FIFO)



✓ **Ngăn xếp (stack):** Theo quy tắc vào trước ra sau (FILO)

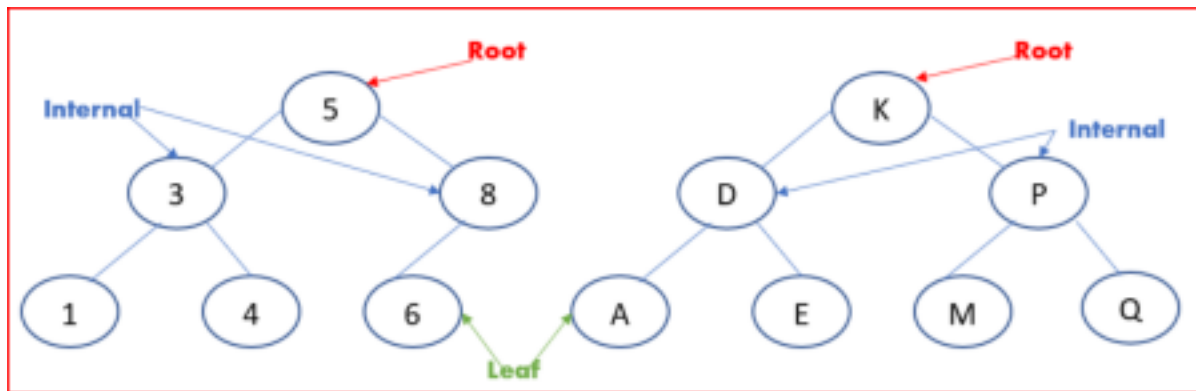


2. PHƯƠNG PHÁP TÌM KIẾM

2.3. Giải thuật tìm kiếm tổng quát

□ Cây tìm kiếm:

- ✓ Trong quá trình thực hiện giải thuật tìm kiếm, 1 tìm kiếm được sinh ra theo chiến lược tìm kiếm đã cho
- ✓ Gốc của cây là trạng thái ban đầu
- ✓ Mở rộng đến khi đi đến trạng thái đích (trạng thái cần tìm)



(Ví dụ cây tìm kiếm nhị phân)

2. PHƯƠNG PHÁP TÌM KIẾM

2.3. Giải thuật tìm kiếm tổng quát

□ Quy ước “tên hành động”:

- ✓ **Make_Queue/make_stack** là tạo hàng đợi hoặc ngăn xếp
- ✓ **Father(i)** là chỉ nút cha của nút i
- ✓ **Path(node, father)** dùng để lần ngược đường đi từ trạng thái node về nút gốc.
- ✓ **Pop(queue/stack)** là lấy ra giá trị ở đầu ngăn xếp hoặc stack
- ✓ **Push(queue/stack)** là đẩy giá trị mới vào cuối hàng đợi hoặc đầu ngăn xếp
- ✓ **Expand-nodes** là mở rộng nút trên cây
- ✓ **Adjacent-nodes** là lấy nút lân cận của nút đang xét

2. PHƯƠNG PHÁP TÌM KIẾM

2.3. Giải thuật tìm kiếm tổng quát

□ Giải thuật tổng quát (tham khảo tài liệu [1])

DuongVanHieu@tgu.edu.vn



18

2. PHƯƠNG PHÁP TÌM KIẾM

2.3. Giải thuật tìm kiếm tổng quát

- ☐ **Giải thuật in đường đi từ nút đích đến nút gốc (tham khảo tài**

liệu [1])



19

DuongVanHieu@tgu.edu.vn

3. TÌM KIẾM MÙ

3.1. Giới thiệu

□ Tìm kiếm mù (blind search) còn có tên gọi khác là tìm

kiểm không đầy đủ thông tin (uninform search) □ Đặc điểm của bài toán là:

- ✓ Không biết trước thông tin về số lần chuyển trạng thái trong quá trình di chuyển từ trạng thái đầu đến trạng thái đích
- ✓ Chỉ dữ dụng 5 thông tin về trạng thái đầu, trạng thái đích, không gian trạng thái, cách di chuyển, chi phí di chuyển ✓
Thông tin có được chỉ giúp phân biệt được trạng thái hiện tại với trạng thái đích (để xác định điều kiện dừng)

20

DuongVanHieu@tgu.edu.vn

3. TÌM KIẾM MÙ

3.2. Tìm theo chiều rộng (Breadth-first Search)

□ Tìm kiếm theo chiều rộng là chiến lược tìm kiếm đơn

giản và được sử dụng nhiều

□ Cách làm việc của chiến lược:

- ✓ Nút gốc (**nút mức 0**) được mở rộng trước
- ✓ Tiếp theo là tất cả các nút được sinh ra từ nút gốc (**nút mức 1**) được mở rộng
- ✓ Tiếp tục là tất cả các **nút mức 2** được mở rộng, tiếp nữa là các nút mức 3, 4, 5, ... được mở rộng cho đến khi tìm được đáp án cho bài toán

21

DuongVanHieu@tgu.edu.vn

3. TÌM KIẾM MÙ

3.2. Tìm theo chiều rộng (Breadth-first Search)

□ Ví dụ cây nhị phân

Chiều cao $d=1$

Chiều cao $d=2$

Mở rộng cây tìm kiếm từ
mức $d=0$ đến $d=1, 2, 3, \dots$
Cho đến khi tìm được lời
giải cho bài toán

Chiều cao $d=3$

22

DuongVanHieu@tgu.edu.vn

3. TÌM KIẾM MÙ

3.2. Tìm theo chiều rộng (Breadth-first Search)

□ Kỹ thuật được sử dụng:

- ✓ Cấu trúc dữ liệu hàng đợi (queue) được sử dụng để lưu trữ trạng thái các nút của cây
- ✓ Các nút được sinh ra trong quá trình thực hiện giải thuật được cập nhật vào hàng đợi và lấy ra theo nguyên tắc vào trước ra trước (FIFO)



DuongVanHieu@tgu.edu.vn 23

3. TÌM KIẾM MÙ

3.2. Tìm theo chiều rộng (Breadth-first Search)

□ Thuật toán:

Cho trước:

1. Không gian trạng thái là “tất cả các điểm trên bản đồ”
2. Trạng thái bắt đầu (điểm đầu), ký hiệu $S = \text{Start}$
3. Trạng thái đích (điểm cuối), ký hiệu là $G = \text{Goal}$
4. Cách di chuyển trên bản đồ (tất cả đường nối 2 điểm trên bản đồ)
5. Chi phí di chuyển giữa 2 điểm trên bản đồ

24

DuongVanHieu@tgu.edu.vn

3. TÌM KIẾM MÙ

3.2. Tìm theo chiều rộng (Breadth-first Search)

 Thuật toán:

 Cách thực hiện:

1. Bắt đầu từ trạng thái đầu (S), thêm nó vào hàng đợi. 2.

Bước lặp:

- ✓ Nếu hàng đợi rỗng thì dừng \Rightarrow Tìm kiếm thất bại
- ✓ Lấy giá trị đỉnh ở đầu của hàng đợi.
- ✓ Nếu giá trị lấy ra = G thì dừng \Rightarrow In ra lời giải
- ✓ Nếu giá trị lấy ra \neq G thì
 - Duyệt qua tất cả các đỉnh kề của đỉnh hiện tại và thêm chúng vào hàng đợi nếu chưa được thăm.
 - Đánh dấu đỉnh hiện tại là đã thăm.

DuongVanHieu@tgu.edu.vn

25

3. TÌM KIẾM MÙ

3.2. Tìm theo chiều rộng (Breadth-first Search)

□ Thuật toán



https://en.wikipedia.org/wiki/Breadth-first_search

DuongVanHieu@tgu.edu.vn 26

3. TÌM KIẾM MÙ

3.2. Tìm theo chiều rộng (Breadth-first

Search) □ Thuật toán:



- ✓ G là đồ thị (bản đồ) chứa các nút (điểm)
- ✓ Q là hàng đợi chứa trạng thái các nút trên bản đồ. ✓
- v, w là các nút (điểm) trên bản đồ
- ✓ Q.enqueue(s) là **đưa nút s vào cuối** hàng đợi Q
- ✓ v:=Q.dequeue() là **lấy giá trị từ đầu** hàng đợi ra và

gán vào v https://en.wikipedia.org/wiki/Breadth-first_search

3. TÌM KIẾM MÙ

3.2. Tìm theo chiều rộng (Breadth-first Search)

□ Thuật toán (tham khảo tài liệu [1])

DuongVanHieu@tgu.edu.vn



28

3. TÌM KIẾM MÙ

3.2. Tìm theo chiều rộng (Breadth-first Search)

□ Thuật toán (tham khảo tài liệu [1])

- ✓ S là trạng thái đầu
 - ✓ G là trạng thái đích
 - ✓ Giải thuật bắt đầu xét với hàng đợi chứa trạng thái đầu ✓
- Lấy trạng thái ở đầu hàng đợi ra kiểm tra xem có phải trạng thái đích hay không
- Nếu phải thì in lời giải và dừng
 - Nếu không phải thì lấy tất cả trạng thái con của trạng thái đang xét đưa vào cuối hàng đợi
- ✓ Lặp lại đến khi tìm được trạng thái đích (thành công) hoặc hàng đợi rỗng (thất bại)

29

DuongVanHieu@tgu.edu.vn

3. TÌM KIẾM MÙ

3.2. Tìm theo chiều rộng (Breadth-first

Search) □ Thuật toán (tham khảo tài liệu [1])



Mời SV giải thích kết quả ở trên

DuongVanHieu@tgu.edu.vn 30

3. TÌM KIẾM MÙ

3.2. Tìm theo chiều rộng (Breadth-first Search)

□ Bài tập tại lớp 1

- ✓ Tìm một ví dụ về bài toán tìm kiếm theo chiều rộng
- ✓ Giải thích các bước thực hiện

□ Bài tập tại lớp 2

- ✓ Nhận xét về trường hợp “tốt nhất”, “xấu nhất” khi tìm đường đi từ điểm S đến điểm G trên đồ thị

31

3.3. Tìm theo chiều sâu (Depth-first Search)

- Tìm kiếm theo chiều sâu cũng là chiến lược tìm kiếm đơn giản và được sử dụng nhiều
- Cách làm việc của chiến lược:
 - ✓ Nút gốc (**nút mức 0**) được mở rộng trước
 - ✓ Tiếp theo là tất cả các nút **ở nhánh bên trái nhất** được sinh ra từ nút gốc được mở rộng
 - ✓ Tiếp tục là tất cả các **ở nhánh tiếp theo tính từ trái qua phải** được mở rộng cho đến khi tìm được đáp án cho bài toán

3. TÌM KIẾM MÙ

3.3. Tìm theo chiều sâu (Depth-first Search) □ Kỹ thuật được sử dụng:

- ✓ Cấu trúc dữ liệu ngăn xếp (stack) được sử dụng để lưu trữ trạng thái các nút của cây
- ✓ Các nút được sinh ra trong quá trình thực hiện giải thuật được cập nhật vào ngăn xếp theo nguyên tắc vào sau ra trước hoặc vào trước ra sau

3. TÌM KIẾM MÙ

3.3. Tìm theo chiều sâu (Depth-first Search) □ Thuật toán:

□ Cho trước:

1. Không gian trạng thái là “tất cả các điểm trên bản đồ”
2. Trạng thái bắt đầu (điểm đầu), ký hiệu $S = \text{Start}$
3. Trạng thái đích (điểm cuối), ký hiệu là $G = \text{Goal}$
4. Cách di chuyển trên bản đồ (tất cả đường nối 2 điểm trên bản đồ)
5. Chi phí di chuyển giữa 2 điểm trên bản đồ

34

3.3. Tìm theo chiều sâu (Depth-first Search) □ Thuật toán:

□ Cách thực hiện:

1. Bắt đầu từ trạng thái đầu (S), thêm nó vào ngăn xếp. 2.

Bước lặp:

- ✓ Nếu ngăn xếp rỗng thì dừng \Rightarrow Tìm kiếm thất bại
- ✓ Lấy giá trị đỉnh ở đầu của ngăn xếp.
- ✓ Nếu giá trị lấy ra = G thì dừng \Rightarrow In ra lời giải
- ✓ Nếu giá trị lấy ra \neq G thì
 - Duyệt qua tất cả các đỉnh kề của đỉnh hiện tại và thêm chúng vào ngăn xếp nếu chưa được thăm.
 - Đánh dấu đỉnh hiện tại là đã thăm.

3.3. Tìm theo chiều sâu (Depth-first Search) □ Thuật toán

https://en.wikipedia.org/wiki/Depth-first_search³⁶

DuongVanHieu@tgu.edu.vn

3. TÌM KIẾM MÙ

3.3. Tìm theo chiều sâu (Depth-first Search) □ Thuật toán (tham khảo tài liệu [1])

DuongVanHieu@tgu.edu.vn



3. TÌM KIẾM MÙ

3.3. Tìm theo chiều sâu (Depth-first Search) □ Thuật toán (tham khảo tài liệu [1])

- ✓ S là trạng thái đầu
- ✓ G là trạng thái đích
- ✓ Giải thuật bắt đầu xét với ngăn xếp chứa trạng thái đầu ✓
Lấy trạng thái ở đầu ngăn xếp ra kiểm tra xem có phải trạng thái đích hay không
 - Nếu phải thì in lời giải và dừng
 - Nếu không phải thì trạng thái con của trạng thái đang xét vào đầu ngăn xếp
- ✓ Lặp lại đến khi tìm được trạng thái đích (thành công) hoặc ngăn xếp rỗng (thất bại)

3. TÌM KIẾM MÙ

3.3. Tìm theo chiều sâu (Depth-first Search) □ Thuật toán (tham khảo tài liệu [1])

Mời SV giải thích kết quả ở trên

3. TÌM KIẾM MÙ

3.3. Tìm theo chiều sâu (Depth-first Search)

□ Bài tập tại lớp 1

- ✓ Dừng lại bài toán tìm kiếm theo chiều rộng lúc này
- ✓ Giải thích các bước thực hiện tìm theo chiều sâu

□ Bài tập tại lớp 2

- ✓ Nhận xét về trường hợp “tốt nhất”, “xấu nhất” khi tìm đường đi từ điểm S đến điểm G trên đồ thị

40

3. TÌM KIẾM MÙ

3.4. Tìm theo chiều sâu có giới hạn (Depth-limited Search)

□ Lý do phải giới hạn độ sâu:

- ✓ Không gian trạng thái quá lớn
- ✓ Tốn quá nhiều thời gian và không gian trước khi tìm ra được lời

□ Giải pháp khắc phục:

- ✓ Tìm theo chiều sâu đến mức sâu **maxDepth** thì dừng ✓
- Nếu chưa tìm được lời giải thì chuyển sang chiến lược khác

3. TÌM KIẾM MÙ

3.4. Tìm theo chiều sâu có giới hạn

□ Giải thuật (tham khảo tài liệu [1])



42

3. TÌM KIẾM MÙ

3.4. Tìm theo chiều sâu có giới hạn (Depth-limited Search)

□ Bài tập tại lớp 1 (Nếu độ sâu của bài toán vừa giải >2)

- ✓ Chọn mức giới hạn **maxDepth=2**

- ✓ Giải thích các bước thực hiện tìm theo chiều sâu có giới hạn

□ Bài tập tại lớp 2

- ✓ Tìm 1 bài toán tìm kiếm có độ sâu đầy đủ lớn hơn hơn 4

- ✓ Chọn mức giới hạn **maxDepth=3**

- ✓ Giải thích các bước thực hiện tìm theo chiều sâu có giới hạn

43

DuongVanHieu@tgu.edu.vn

3. TÌM KIẾM MÙ

3.5. Tìm theo chiều sâu tăng dần

□ Lý do phải tăng dần độ sâu:

- ✓ Nếu chọn giá trị giới hạn độ sâu **maxDepth** không phù hợp thì dẫn đến tình huống:
 - Giải pháp nằm ở độ sâu **maxDepth+1** nhưng phải dừng và chuyển sang chiến lược tìm kiếm khác
 - Giá trị **maxDepth** quá lớn nên tốn thời gian và không gian triển khai các nhánh quá sâu mà không cần thiết.

□ Giải pháp khắc phục:

- ✓ Xác định giá trị **maxDepth**
- ✓ Sau đó tăng dần giá trị **maxDepth** nếu chưa tìm thấy lời giải

44

DuongVanHieu@tgu.edu.vn

3. TÌM KIẾM MÙ

3.5. Tìm theo chiều sâu tăng

dần □ Giải thuật (tham khảo tài liệu
[1])

45

DuongVanHieu@tgu.edu.vn

3. TÌM KIẾM MÙ

3.5. Tìm theo chiều sâu tăng dần

□ Bài tập tại lớp 1

✓ Tìm 1 bài toán tìm kiếm có độ sâu đầy đủ lớn hơn hơn 4 ✓

Giải thích các bước thực hiện tìm theo chiều sâu có giới hạn với **maxDepth=0, maxDepth=1,**

maxDepth=2, maxDepth=3, maxDepth=4,...

46

4.1. Giới thiệu

□ Chiến lược tìm kiếm có thông tin có nghĩa là trước khi tìm kiếm chúng ta có thêm sự hiểu biết về bài toán □ Kết quả tìm kiếm sẽ tốt hơn so với tìm kiếm mù □ Các thuật toán dạng này còn có tên gọi là thuật toán tìm kiếm heuristic

47

DuongVanHieu@tgu.edu.vn

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất trước (best first search)

□ Để hạn chế không gian cây các lời giải tiềm năng, chúng ta đưa ra một hàm định hướng việc mở rộng cây tìm kiếm, gọi là hàm $h(n)$

❖ **$h(n)$ là hàm ước lượng chi phí đi từ nút hiện tại (nút n) đến nút đích (nút G)**

- Theo cách này, chúng ta sẽ mở rộng cây theo các nút lá có nhiều tiềm năng chứa trạng thái đích hơn các nút lá khác
- Sử dụng danh sách (List) có sắp xếp theo giá trị của hàm h

4.2. Tìm kiếm tốt nhất trước (best first search)

- Giải thuật giống giải thuật thuật tìm kiếm theo chiều rộng, chiều sâu
- Chỉ cách cấu trúc dữ liệu được dùng là danh sách (list) có thứ tự theo giá trị của hàm h
- Do giải thuật tìm kiếm tốt nhất trước tính chi phí từ nút hiện tại đến nút đích (gọi là hàm $h(n)$) và chọn hướng đi cho chi phí thấp nhất nên được gọi là thuật toán tham ăn/háo ăn (greedy search)

49

4.2. Tìm kiếm tốt nhất trước (best first search)



4. TÌM KIẾM CÓ THÔNG TIN

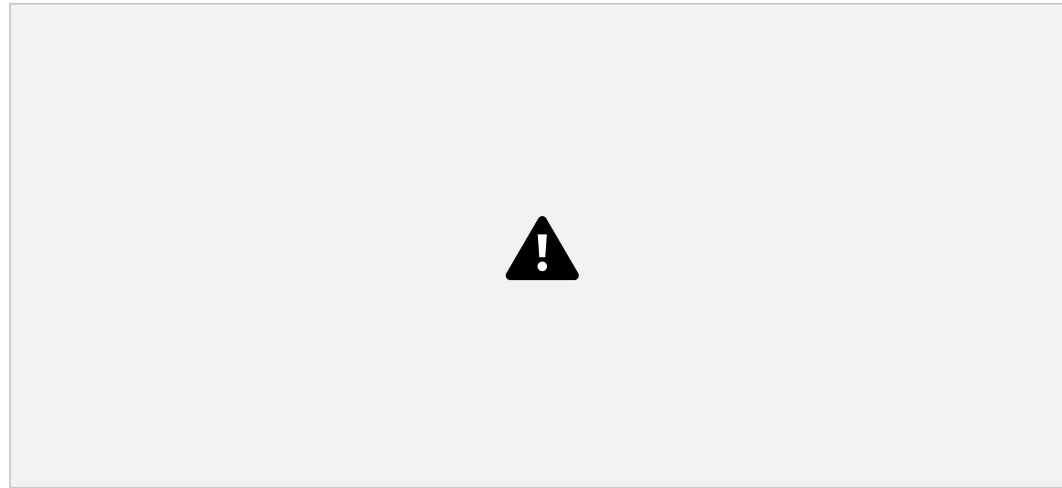
4.2. Tìm kiếm tốt nhất trước (best first search)

51

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất trước (best first search) □ Ví dụ bài toán 9 ô vuông

- Cho trước trạng thái đầu và trạng thái đích



- Cách di chuyển ô trống
- $h(n)$ = tổng khoảng cách các vị trí của từng quân cờ trên bàn cờ n (trạng thái n) so với trạng thái đích

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất trước (best first search) ☐ Ví dụ bài



=> $h(\text{trạng thái đầu})$
= tổng số lần di chuyển = 8

DuongVanHieu@tgu.edu.vn

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất trước (best first search) □ Ví dụ bài toán 9 ô vuông.

✓ Cây tìm kiếm
(từ trạng thái đầu,
có 2 cách di chuyển²
=>sinh 2 nút con)



DuongVanHieu@tgu.edu.vn

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất trước
(best first search) □ Ví dụ bài
toán 9 ô vuông

✓ Cách tính $h(A)$





55

$\Rightarrow h(A) = \text{tổng số lần di chuyển} = 7$

DuongVanHieu@tgu.edu.vn

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất trước (best first search) □ Ví dụ bài toán 9 ô vuông





56

$\Rightarrow h(B) = \text{tổng số lần di chuyển} = 9 > h(A)$

DuongVanHieu@tgu.edu.vn

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất trước (best first search) □ Ví dụ bài toán 9 ô vuông. Kết quả cây

bậc 1



57

DuongVanHieu@tgu.edu.vn

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất trước (best first search) □ Ví dụ bài toán 9 ô vuông.

✓ Vì $h(A) < h(B)$
=> mở rộng cây theo
hướng nút A
=> Có 3 cách di
Chuyển ô trống
=> Nút A có 3 nút
con là C, D, E

58

DuongVanHieu@tgu.edu.vn

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất trước (best first search) □ Ví dụ bài toán 9 ô



vuông

✓ Cách tính $h(C)$

$\Rightarrow h(C) = \text{tổng số lần di chuyển} = 6$

59

DuongVanHieu@tgu.edu.vn

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất

trước (best first search) □ Ví dụ bài toán 9 ô vuông

✓ Cách tính $h(D)$

$\Rightarrow h(D) = \text{tổng số lần di chuyển} = 6 = h(D)$

60

DuongVanHieu@tgu.edu.vn

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất trước (best first search)

□ Ví dụ bài toán 9 ô vuông



Cách tính $h(E)$

$\Rightarrow h(E) = \text{tổng số lần di chuyển} = 8 > h(C), h(D)$

\Rightarrow **Bỏ nút E vì**

trùng với nút S

DuongVanHieu@tgu.edu.vn 61

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất trước (best first search) □ Ví dụ bài toán 9 ô vuông. Kết quả cây



bậc 2

DuongVanHieu@tgu.edu.vn 62

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất trước (best first search) □ Ví dụ bài toán 9 ô vuông.

✓ Vì $h(C)=h(D)=\min$
 \Rightarrow mở rộng cây theo
hướng nút C và nút D

* Nút C có 1 con trùng A \Rightarrow bỏ nút con trùng nút
A \Rightarrow Nút C còn 1 con được gán nhãn là F * Nút D
có 1 con trùng A \Rightarrow bỏ nút con trùng nút A \Rightarrow Nút
D còn 3 con được gán nhãn là G, H, I

63

DuongVanHieu@tgu.edu.vn

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất trước (best first
search) □ Ví dụ bài toán 9 ô vuông.



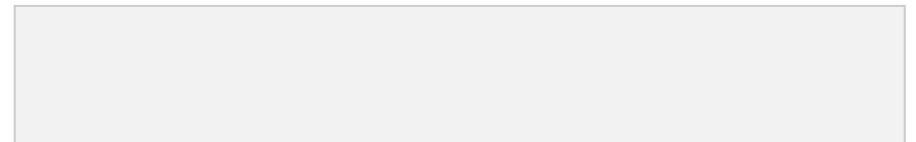
✓ Kết quả mở rộng cây theo 2 nút C, D



DuongVanHieu@tgu.edu.vn 64

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất



trước (best first search) □ Ví dụ bài toán 9 ô



$\Rightarrow h(F) = \text{tổng số lần di chuyển} = 7$

65

DuongVanHieu@tgu.edu.vn

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất trước (best first search)

□ Ví dụ bài toán 9 ô vuông

✓ Cách tính $h(G)$



$\Rightarrow h(G) = \text{tổng số lần di chuyển} = 7$

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất trước (best first search)



ng



$\Rightarrow h(H)$ = tổng số lần di chuyển = 7

DuongVanHieu@tgu.edu.vn

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất trước (best first search)



ng



$\Rightarrow h(I) = \text{tổng số lần đi}$

chuyển = 7

Ta có $h(F)=h(G)=h(H)=h(I)=\min=7$

DuongVanHieu@tgu.edu.vn 68

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất trước (best first search) □ Ví dụ bài toán 9 ô vuông. Kết quả cây



bậc 2

DuongVanHieu@tgu.edu.vn 69

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất trước (best first

search) □ Ví dụ bài toán 9 ô vuông.

✓ Mở rộng cây theo 4 nút F, G, H, I được cây bậc 4

* Nút F có 1 con trùng C \Rightarrow bỏ nút con trùng nút C

\Rightarrow Nút F còn 2 con được gán nhãn là J, K * Nút G

có 1 con trùng D \Rightarrow bỏ nút con trùng nút D \Rightarrow Nút

G còn 2 con được gán nhãn là L, M

70

DuongVanHieu@tgu.edu.vn

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất trước (best first

search) □ Ví dụ bài toán 9 ô vuông.

✓ Mở rộng cây theo 4 nút F, G, H, I được cây bậc 4

* Nút H có 1 con trùng D \Rightarrow bỏ nút con trùng nút

D \Rightarrow Nút H còn 2 con được gán nhãn là N, O * Nút

I có 1 con trùng D \Rightarrow bỏ nút con trùng nút D \Rightarrow

Nút I còn 2 con được gán nhãn là P, Q

71

DuongVanHieu@tgu.edu.vn

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất trước (best first

search) □ Ví dụ bài toán 9 ô vuông.

✓ Kết quả mở rộng cây theo 4 nút F, G, H, I

72

DuongVanHieu@tgu.edu.vn

4. TÌM KIẾM CÓ THÔNG TIN

4.2. Tìm kiếm tốt nhất trước (best first



**Thực
hiện
tiếp để
tìm lời
giải**

73

DuongVanHieu@tgu.edu.vn

4. TÌM KIẾM CÓ THÔNG TIN

4.3. Các biến thể của giải thuật tìm kiếm tốt

nhất trước (best first search)

□ Biến thể của thuật toán tìm kiếm tốt nhất: 1)

Khi hàm $h(n)$ là chi phí của dãy phép chuyển từ trạng thái đầu đến trạng thái n thì giải thuật best first-search có tên gọi khác là giải thuật tìm kiếm đều (uniform search).

=> Cây tìm kiếm sẽ mở rộng đều về tất cả các hướng theo vết dầu loang từ trạng thái đầu => cho lời giải với chi phí nhỏ nhất, tuy nhiên cây tìm kiếm sinh ra trong giải thuật này thường có kích thước rất lớn.

DuongVanHieu@tgu.edu.vn 74

4. TÌM KIẾM CÓ THÔNG TIN

4.3. Các biến thể của giải thuật tìm kiếm tốt

nhất trước (best first search)

□ Biến thể của thuật toán tìm kiếm tốt nhất: 2)

Khi $h(n)$ là ước lượng chi phí/khoảng cách từ n đến đích thì giải thuật best-first-search được gọi là giải thuật tham ăn (greedy search)

=> Cây tìm kiếm sẽ mở rộng theo nút lá gần đến nút đích nhất => có xu hướng cho ra kết quả trong thời gian nhanh nhất, nhưng không phải lúc nào cũng là lời giải ngắn nhất

75

DuongVanHieu@tgu.edu.vn

4. TÌM KIẾM CÓ THÔNG TIN

4.3. Các biến thể của giải thuật tìm kiếm tốt nhất

trước (best first search)

□ Gọi:

- $f(n)$ là chi phí từ trạng thái đầu đến trạng thái n
- $g(n)$ là chi phí từ trạng thái n đến trạng thái đích
- $h(n)=f(n)+g(n)$ là chi phí từ trạng thái đầu đến trạng thái đích

➤ Giải thuật A^* triển khai đường đi theo hàm $h(n)=f(n)+g(n)$

76

DuongVanHieu@tgu.edu.vn

4. TÌM KIẾM CÓ THÔNG TIN

4.3. Các biến thể của giải thuật tìm kiếm tốt nhất

trước (best first search)

➤ Giải thuật A* triển khai **S** $f(H)$ **H**

đường đi theo hàm

$$h(n)=f(n)+g(n)$$

$$g(H) \quad h(H)=f(H)+g(H)$$

G

➤ Tham khảo tài liệu [1], trang 30, 31

77

DuongVanHieu@tgu.edu.vn

4. TÌM KIẾM CÓ THÔNG TIN

4.4. Giải thuật leo đồi

- Ý tưởng: Tìm kiếm theo chiều sâu kết hợp với hàm đánh giá. Mở rộng trạng thái hiện tại và đánh giá các trạng thái con của nó bằng hàm đánh giá heuristic.
- Tại mỗi bước, nút lá “tốt nhất” sẽ được chọn để đi tiếp

78

4.4. Giải thuật leo đồi

□ Giải thuật (theo tài liệu [1])

DuongVanHieu@tgu.edu.vn



4. TÌM KIẾM CÓ THÔNG TIN

4.4. Giải thuật leo đồi

□ Ví dụ (theo tài liệu [1])



DuongVanHieu@tgu.edu.vn

80