

## AI Overview

### Searching Algorithms

- Tìm kiếm mù, tìm kiếm theo kinh nghiệm (Không sinh ra trạng thái lặp)
- Tìm kiếm tối ưu (Sinh ra trạng thái lặp)

### BFS

start (A) Goal (F)

Graph edges and weights:

- A to B: 3
- A to C: 1
- B to D: 3
- B to E: 1
- C to E: 2
- D to F: 1
- E to F: 3

Table  $h(x)$ :

$x$	$h(x)$
A	0
B	1
C	2
D	1
E	1
F	0

$L = \{A\}$

BFS Table:

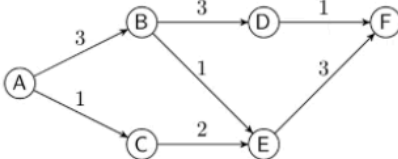
BFS	u	ps kế (v)	L
	A	B, C	B, C
	B	D, E	C, D, E
	C	X	D, E
	D	F	E, F
	E		F
	F	stop	$\phi$

Search Tree:

- A (1)
  - B (2)
    - D (4)
      - F (6)
  - C (3)
    - E (5)

KL: A  $\rightarrow$  B  $\rightarrow$  D  $\rightarrow$  F.

## DFS

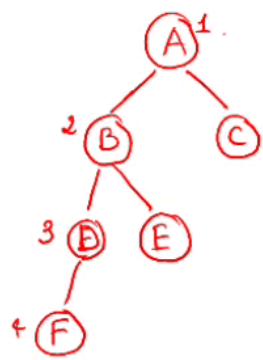


	$h(x)$
A	0
B	1
C	2
D	1
E	1
F	0

$L = \{A\}$

DFS :

U	Ds kế (v)	L.
A	B, C	$\{B, C\}$
B	D, E	$\{D, E, C\}$
D	F	$\{F, E, C\}$
F	stop	

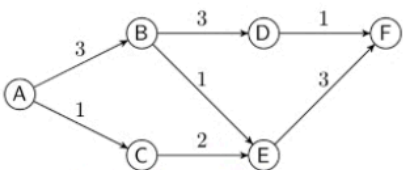


KL:  $A \rightarrow B \rightarrow D \rightarrow F$ .

Bình Nguyễn Thị Hải

=> Không thể khẳng định BFS, DFS tốt hơn về mặt thời gian, mà chỉ ở dạng tổng quát.

## Best First Search

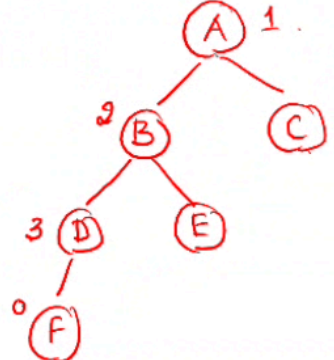


Best First Search (BFS)  
Hill climbing.

	$h(x)$
A	0
B	1
C	2
D	1
E	1
F	0

$L = \{A\}$

u	Đã kiểm tra	L
A	B, C	$\{B_1, C_2\}$
B	D, E	$\{D_1, E_1, C_2\}$
D	F	$\{F_0, E_1, C_2\}$
F	goal.	

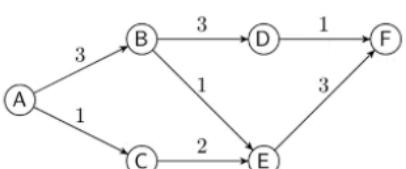


KL:  $A \rightarrow B \rightarrow D \rightarrow F$ .

Bình Nguyễn Thị Hải

Mục đích của thuật toán Best First Search là giảm số lượng trạng thái ta cần phải xét, chứ không đảm bảo đường đi ngắn nhất.

## Hill Climbing



	$h(x)$
A	0
B	1
C	2
D	<del>1</del> 3
E	<del>1</del> 4
F	0

u	Đã kiểm tra	L
A	B, C	$B_1, C_2$
B	<u>D, E</u>	$\{D_3, E_4, C_2\}$
D	F	$\{F, E, C\}$

Bình Nguyễn Thị Hải

Chỉ sắp xếp danh sách kề **không** sắp xếp toàn bộ danh sách L

A\*

Graph structure and heuristic values:

	$h(x)$
A	0
B	1
C	2
D	1
E	1
F	0

Handwritten notes:

$A^*$   
Nhất v. cần.  
 $f(n) = \underline{h(n)} + \underline{g(n)}.$

Partial search tree diagram:

```

graph TD
    A((A)) --- B((B))
    A --- C((C))
    B --- D((D))
    C --- E((E))
    D --- F((F))
    E --- F
  
```

Handwritten table structure:

u	ds kề	L.
A	B	

Handwritten formula for the search tree:

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

Handwritten diagram showing a node  $n$  with a child node  $F$  and a label  $h(n)$  pointing to  $F$ .

Cố gắng làm giảm tg tìm kiếm và cố gắng tìm đường đi ngắn nhất (phụ thuộc vào hàm đánh giá)

	$h(x)$
A	0
B	3
C	2
D	1
E	1
F	0

$A^*$   
Nhánh và cần.  
 $f(n) = h(n) + g(n)$

u	Ds kế	L.
A	$B_4^3, C_3^1$	$\{C_3^1, B_4^3\}$
$C_3^1$	$E_4^3$	$B_4^3, E_4^3(c)$
$B_4^3$	$D_7^6, E_5^4(B)$	$E_4^3(c), E_5^4(B), D_7^6$
$E_4^3(c)$	$F_6^6$	$E_5^4(B), F_6^6, D_7^6$
$E_5^4(B)$	$F_7^7$	$F_6^6, D_7^6, F_7^7$
$F_6^6$	stop.	

## Nhánh và cần

	$h(x)$
A	0
B	3
C	2
D	1
E	1
F	0

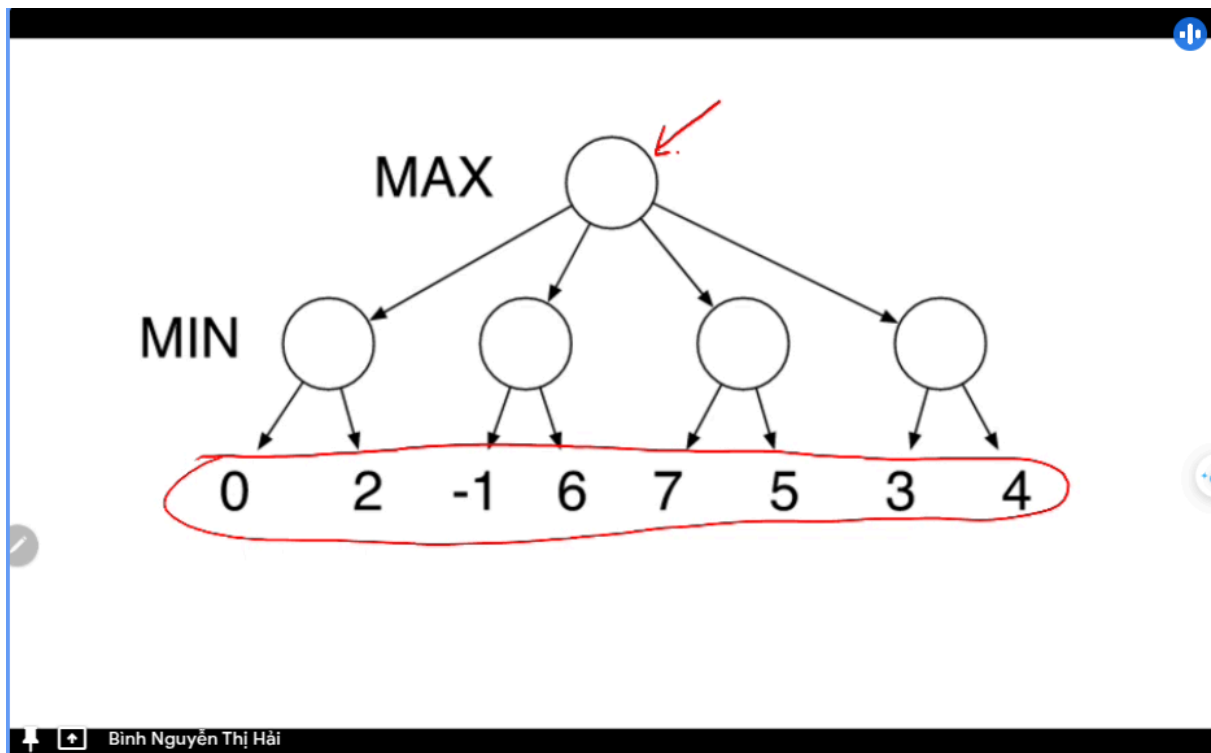
u	Ds kế	L.
A	$B_4^3, C_3^1$	$\{C_3^1, B_4^3\}$
$C_3^1$	$E_4^3$	$\{E_4^3, B_4^3\}$
$E_4^3$	$F_6^6$	$\{F_6^6, B_4^3\}$
$F_6^6$	Goal $\rightarrow cost = 6$	$(A \rightarrow C \rightarrow E \rightarrow F)$
$B_4^3$	$D_7^6, E_5^4$	$\{E_5^4, D_7^6\}$

Nhánh cần sắp xếp cục bộ, chứ không phải toàn bộ không gian mà mình sinh ra. Cho nên khi gặp đường đi ngắn nhất thì chưa chắc (chỉ mang tính tạm thời thôi)

=> Nhánh và cận là thuật toán **duy nhất có 1 điểm dừng** khi **danh sách rỗng**

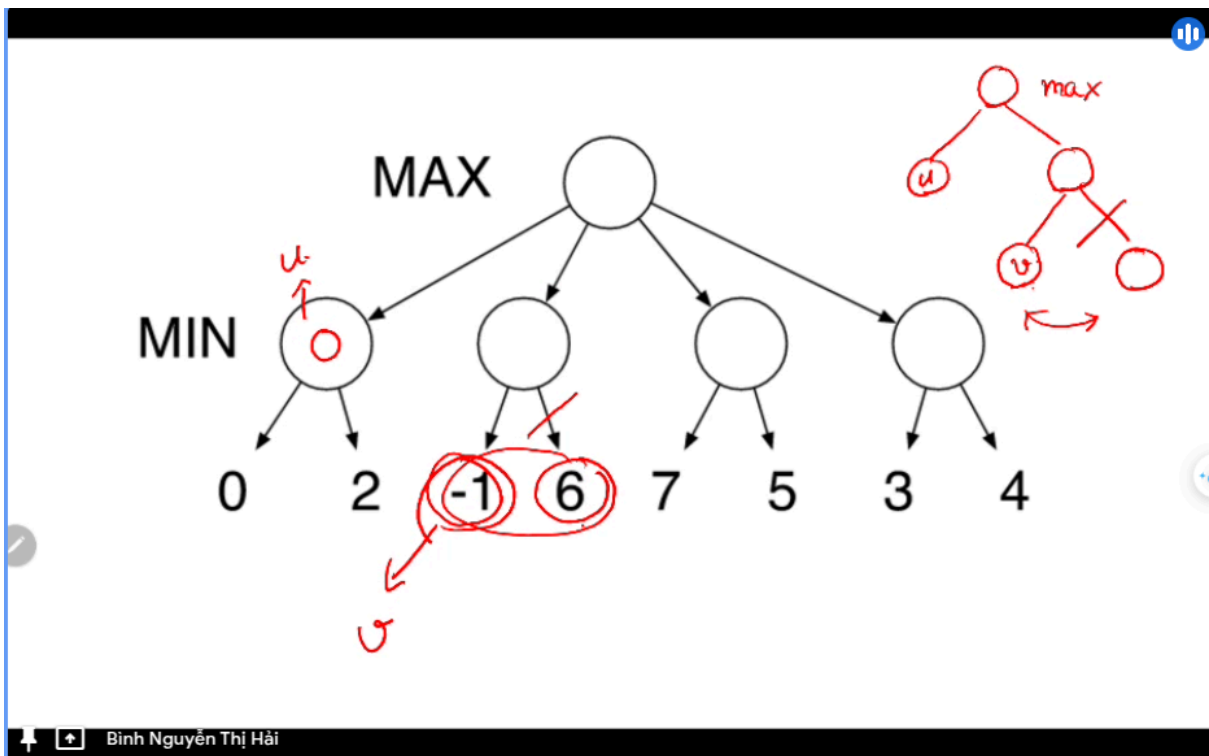
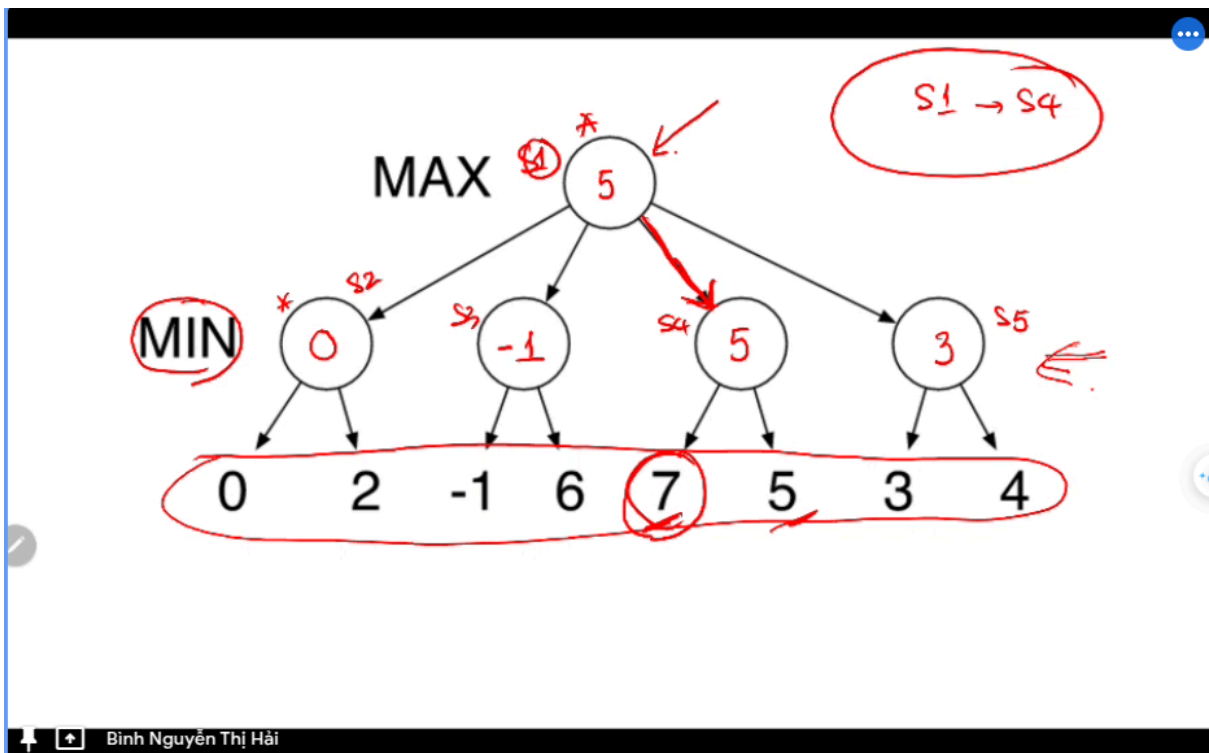
=> Các thuật toán khác **2 điểm dừng** khi **danh sách rỗng** và khi **gặp điểm dừng**

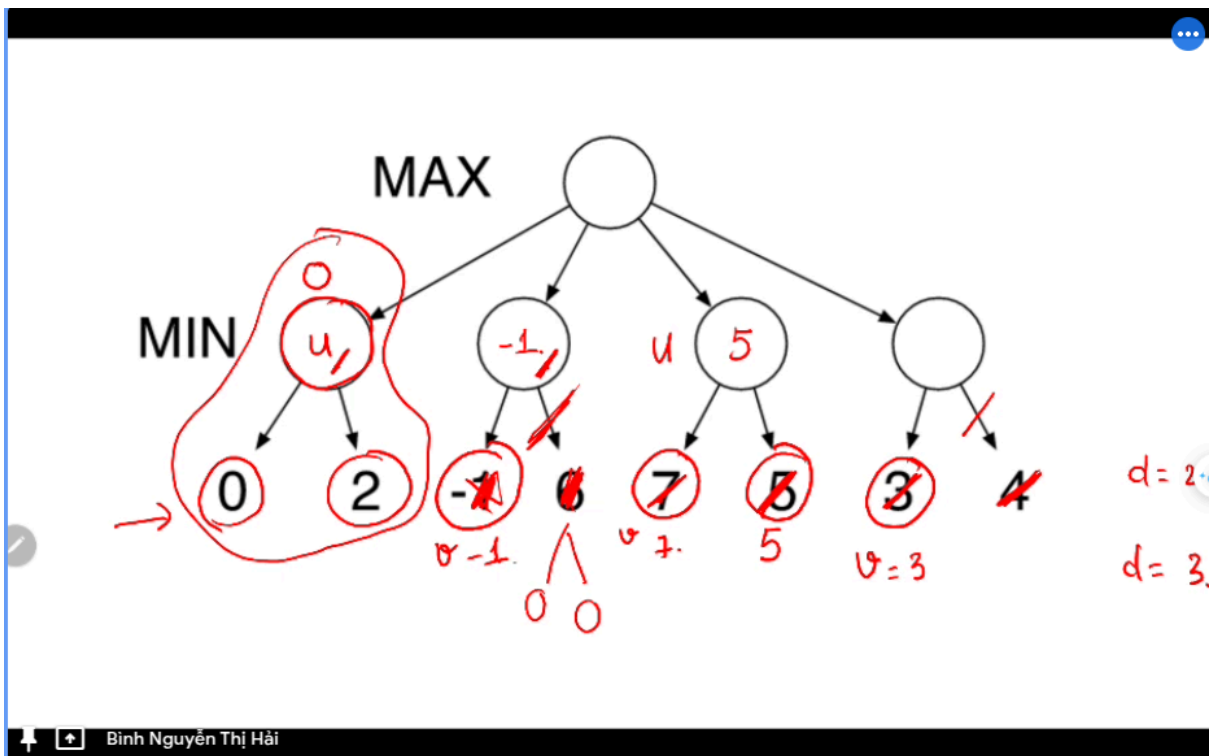
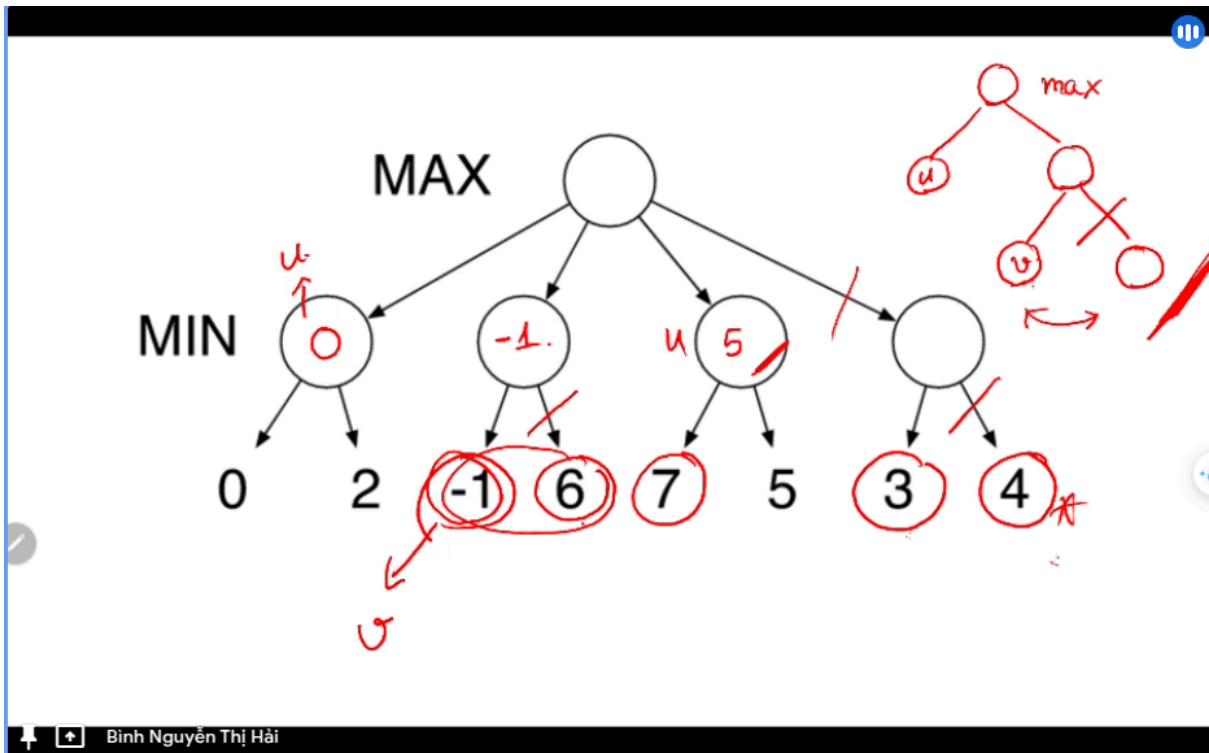
### Tìm kiếm MiniMax



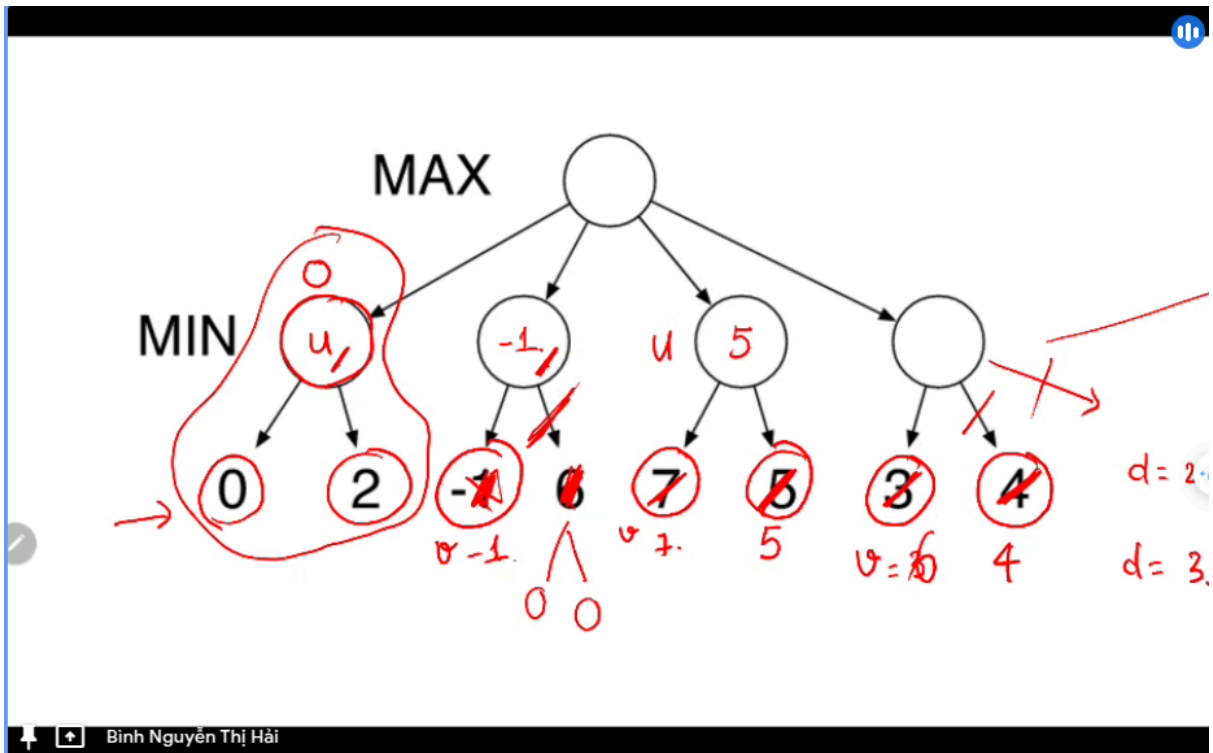
Hàm kết cuộc: Đánh giá trạng thái kết thúc

Hàm đánh giá thể hiện lợi thế cho bên Max









## Tìm x để cắt

9:48 Thu, Mar 20

Notes  
Tap to add notes

a. Cây trò chơi 1

b. Cây trò chơi 2

c. Cây trò chơi 3

d. Cây trò chơi 4

42 43 44 45 46 47 48 49

Bình Nguyễn Thị Hải

9:50 Thu, Mar 20

Notes  
Tap to add notes

max

min

3

3 12 9 2 4 6 15 6 0

3 4 5 6 7 8

Bình Nguyễn Thị Hải

9:52 Thu, Mar 20

Notes  
Tap to add notes

3 12 9 2 4 6 15 6 0

0 15

3 4 5 6 7 8

Bình Nguyễn Thị Hết

9:54 Thu, Mar 20

Notes  
Tap to add notes

Max:

Min:

3 12 8 2 7 5 14 5 2

4 5 6 7 8 9

Bình Nguyễn Thị Hết

## Leetcode

### 1. Thuật toán BFS và DFS:

- **[Number of Islands](#)**: Bài toán yêu cầu đếm số lượng đảo trong một lưới 2D, nơi '1' đại diện cho đất liền và '0' cho nước. Cả BFS và DFS đều có thể được sử dụng để duyệt qua các ô liên thông.
  - **Word Ladder**: Bài toán tìm đường đi ngắn nhất để biến đổi một từ bắt đầu thành từ kết thúc bằng cách thay đổi từng chữ cái một lần, với mỗi từ trung gian phải có trong danh sách từ cho trước. BFS thường được sử dụng để tìm đường đi ngắn nhất trong trường hợp này.
    - [Word Ladder I](#)
    - [Word Ladder II](#)
  - **[Clone Graph](#)**: Yêu cầu sao chép một đồ thị vô hướng. Cả BFS và DFS đều có thể được sử dụng để duyệt và sao chép các node trong đồ thị.
- 

### 2. Thuật toán Best First Search và A\*:

- **[Path With Minimum Effort](#)**: Tìm đường đi từ góc trên trái đến góc dưới phải của ma trận sao cho sự khác biệt lớn nhất giữa hai ô liên tiếp trên đường đi là nhỏ nhất. Thuật toán Best First Search có thể được áp dụng để giải quyết bài toán này.
- **[Sliding Puzzle](#)**: Giải quyết bài toán trượt ô số bằng cách tìm số bước ít nhất để đạt được trạng thái mục tiêu. A\* là

một thuật toán hiệu quả cho bài toán này.

---

### 3. Thuật toán Hill Climbing:

Mặc dù Hill Climbing không thường được sử dụng trực tiếp trong các bài toán trên LeetCode, nhưng bạn có thể áp dụng nó trong các bài toán tối ưu hóa hoặc tìm kiếm cục bộ. Ví dụ:

- **Maximum Number of Points with Cost**: Tìm số điểm tối đa có thể thu thập được với chi phí di chuyển giữa các hàng trong ma trận. Hill Climbing có thể được sử dụng để tìm nghiệm gần đúng trong các bài toán tối ưu hóa như thế này.
- 

### 4. Thuật toán Nhánh Cận (Branch and Bound):

- **Combination Sum**: Tìm tất cả các tổ hợp của các số mà tổng của chúng bằng một giá trị mục tiêu. Phương pháp Nhánh Cận có thể được áp dụng để cắt bỏ các nhánh không cần thiết trong quá trình tìm kiếm.
  - <https://leetcode.com/problems/combination-sum>
  - <https://leetcode.com/problems/combination-sum-ii>
  - <https://leetcode.com/problems/combination-sum-iii>
  - <https://leetcode.com/problems/combination-sum-iv>
- **N-Queens**: Đặt N quân hậu trên bàn cờ NxN sao cho không quân hậu nào có thể tấn công nhau. Thuật toán Nhánh Cận giúp giảm thiểu số lượng trường hợp cần xem xét bằng cách loại bỏ sớm các cấu hình không hợp lệ.

- <https://leetcode.com/problems/n-queens>
- <https://leetcode.com/problems/n-queens-ii>

## 5. Các bài toán LeetCode áp dụng Adversarial Search

### (1) [Guess the Word](#)

- **Mô tả:** Người chơi cố gắng đoán một từ bí mật trong số các từ được cung cấp bằng cách đặt câu hỏi và nhận phản hồi về mức độ giống nhau giữa từ đoán và từ bí mật.
- **Cách giải:** Sử dụng thuật toán **MiniMax** để chọn từ tối ưu nhất, giảm thiểu số lần đoán cần thiết.

### (2) [Predict the Winner](#)

- **Mô tả:** Hai người chơi luân phiên chọn số từ mảng, ai có tổng điểm cao hơn thì thắng. Hãy xác định xem người chơi đầu tiên có thể thắng không nếu cả hai đều chơi tối ưu.
- **Cách giải:**
  - Sử dụng thuật toán **MiniMax** để đánh giá từng lựa chọn.
  - Có thể tối ưu bằng **Alpha-Beta Pruning** hoặc **Dynamic Programming**.

### (3) [Stone Game](#)

- **Mô tả:** Hai người chơi chơi trò chơi với các viên đá có giá trị khác nhau, mỗi lượt lấy một viên từ đầu hoặc cuối dãy.
- **Cách giải:**
  - Dùng **MiniMax** để chọn nước đi tối ưu.
  - Cải thiện bằng quy tắc trò chơi (có chứng minh toán học rằng người chơi đầu tiên luôn có chiến thuật thắng).

## Represent & Process Knowledge

Knowledge Base: Cơ sở tri thức là tập hợp của tất cả các mệnh đề, tất cả giá trị đều nhận giá trị true

Mô hình - Model: Là những trường hợp, giá trị cụ thể mà làm cho các phát biểu trong KB là đúng

Q = "x là số nguyên tố"

P = "x lớn hơn 10"

→ Mô hình của KB x = 11, 13, 17,....

Xây dựng cơ sở tri thức -> Biểu diễn cơ sở tri thức

### Logic Mệnh Đề

- Mệnh đề đơn
- Mệnh đề phức hợp

# Semantics of the Propositional Logic



- Semantics defines the meaning of the sentences of a language.

$P$	$Q$	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
False	False	True	False	False	True	True
False	True	True	False	True	True	False
True	False	False	False	True	False	False
True	True	False	True	True	True	True

## Example 1:

- Complete the truth table of  $(P \Rightarrow Q) \wedge S$
- Find a model of the given proposition.



1. Double-negation law:  
 $\neg(\neg P) \equiv P$
2. Complement law:  
a.  $P \vee \neg P \equiv \text{True}$   
b.  $P \wedge \neg P \equiv \text{False}$
3. Identity law:  
a.  $P \vee \text{False} \equiv P$   
b.  $P \wedge \text{True} \equiv P$
4. Conditional identities:  
a.  $P \Rightarrow Q \equiv \neg P \vee Q$   
b.  $P \Rightarrow Q \equiv \neg Q \Rightarrow \neg P$
9. Associative law:  
a.  $(P \wedge Q) \wedge R \equiv P \wedge (Q \wedge R)$   
b.  $(P \vee Q) \vee R \equiv P \vee (Q \vee R)$
10. Idempotent law:  
a.  $P \wedge P \equiv P$   
b.  $P \vee P \equiv P$
11. Domination law:  
a.  $P \wedge \text{False} \equiv \text{False}$   
b.  $P \vee \text{True} \equiv \text{True}$
5. Conditional identities:  
 $P \Leftrightarrow Q \equiv (P \Rightarrow Q) \wedge (Q \Rightarrow P)$
6. De Morgan's law:  
a.  $\neg(P \vee Q) \equiv \neg P \wedge \neg Q$   
b.  $\neg(P \wedge Q) \equiv \neg P \vee \neg Q$
7. Distributive law:  
a.  $R \vee (P \wedge Q) \equiv (R \vee P) \wedge (R \vee Q)$   
b.  $R \wedge (P \vee Q) \equiv (R \wedge P) \vee (R \wedge Q)$
8. Commutative law:  
a.  $P \wedge Q \equiv Q \wedge P$   
b.  $P \vee Q \equiv Q \vee P$
12. Absorption law:  
a.  $P \vee (P \wedge Q) \equiv P$   
b.  $P \wedge (P \vee Q) \equiv P$

21

## Exercise

Using laws of propositional logic to proof:

④  $P \Rightarrow (\neg P \Rightarrow P) \equiv \text{True}$

- $(P \wedge Q) \Rightarrow P \equiv \text{True}$
- $P \Rightarrow (Q \Rightarrow (P \wedge Q)) \equiv \text{True}$
- $\neg(Q \Rightarrow P) \vee (P \wedge Q) \equiv Q$
- $P \vee ((P \wedge Q) \vee (P \wedge \neg R)) \equiv P \wedge ((\neg Q \Rightarrow R) \vee \neg(Q \vee (R \wedge S) \vee (R \wedge \neg S)))$

20



Aladdin finds two trunks A and B in a cave. He knows that each of them either contains a treasure or a fatal trap.

On trunk A is written: "At least one of these two trunks contains a treasure."

On trunk B is written: "In A there's a fatal trap."

Aladdin knows that either both the inscriptions are true, or they are both false.

Can Aladdin choose a trunk being sure that he will find a treasure?

If this is the case, which trunk should he open?

24

- *The guardian of the gold street:* "This road will bring you straight to the center. Moreover, if the stones take you to the center, then also the marble takes you to the center."
- *The guardian of the marble street:* "Neither the gold nor the stones will take you to the center."
- *The guardian of the stone street:* "Follow the gold and you'll reach the center, follow the marble and you will be lost."

*Given that you know that all the guardians are liars, can you choose a road being sure that it will lead you to the center of the labyrinth? If this is the case, which road you choose?*

28

$\neg G \vee (S \wedge \neg M)$        $G \vee S$        $\neg G \vee M$

$g$	$m$	$s$	2.7	2.8	2.9	$2.7 \wedge 2.8 \wedge 2.9$
1	1	1	0	1	1	0
1	1	0	0	1	1	0
1	0	1	1	1	0	0
1	0	0	0	1	0	0
0	1	1	1	1	1	1
0	1	0	1	0	1	0
0	0	1	1	1	1	1
0	0	0	1	0	1	0

*True* (pointing to the first row)

$\neg G \vee (S \wedge \neg M)$        $G \vee S$        $\neg G \vee M$

$g$	$m$	$s$	2.7	2.8	2.9	$2.7 \wedge 2.8 \wedge 2.9$
1	1	1	0	1	1	0
1	1	0	0	1	1	0
1	0	1	1	1	0	0
1	0	0	0	1	0	0
0	1	1	1	1	1	1
0	1	0	1	0	1	0
0	0	1	1	1	1	1
0	0	0	1	0	1	0

*True* (pointing to the first row)

*False* (pointing to the fifth row)

$\{ G \leftarrow \text{False}, M \leftarrow \text{True}, S \leftarrow \underline{\text{True}} \}$   
 $\{ G \leftarrow \text{False}, M \leftarrow \text{False}, S \leftarrow \underline{\text{True}} \}$

Chọn đi theo con đường bằng đá.

#### QUESTION 1

Three boxes are presented to you. One contains gold, the other two are empty. Each box has imprinted on it a clue as to its contents; the clues are:

- Box 1 "The gold is in Box 2"
- Box 2 "The gold is not here"
- Box 3 "The gold is not here"

Only one message is true; the other two are false. Which box has the gold?

#### QUESTION 2

Kyle, Neal, and Grant find themselves trapped in a dark and cold dungeon (HOW they arrived there is another story). After a quick search the boys find three doors, the first one red, the second one blue, and the third one green.

Behind one of the doors is a path to freedom. Behind the other two doors, however, is an evil fire-breathing dragon. Opening a door to the dragon means almost certain death.

On each door there is an inscription:

On green door: freedom is behind this door

On blue door: freedom is not behind this door

On red door: freedom is not behind the blue door

Given the fact that at LEAST ONE of the three statements on the three doors is true and at LEAST ONE of them is false, which door would lead the boys to safety?

25

## Logic Vị Từ

### ĐỀ THI:

- 2 thuật toán không thi BFS, DFS. Tập trung vào Heuristic và Optimal. Tìm kiếm có đối thủ.

Bài tập hình như trên lớp + bài toán thực tế

- Biểu diễn logic: Logic mệnh đề. Tập trung vào cho sẵn mệnh đề logic (sử dụng các phép thuật giải, các phép biến đổi tương đương), bài toán thực tế