

# TÌM KIẾM CÓ ĐỐI THỦ

TS. Nguyễn Quốc Tuấn

# Nội dung

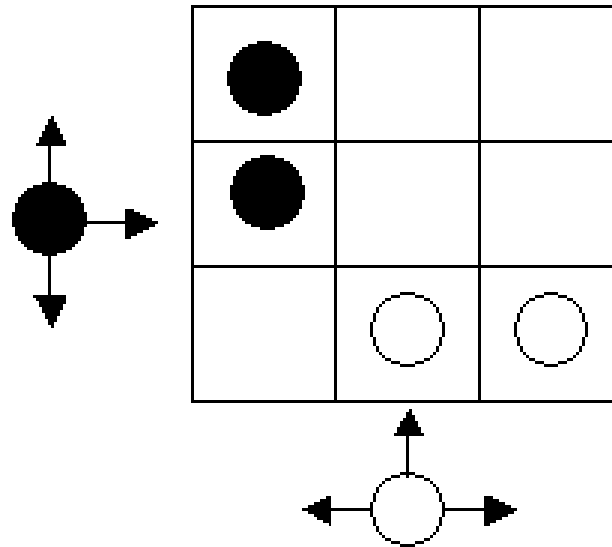
2

- Tổng quan
- Giải thuật Minimax
- Cắt tỉa Alpha/Beta

# Một số ví dụ

3

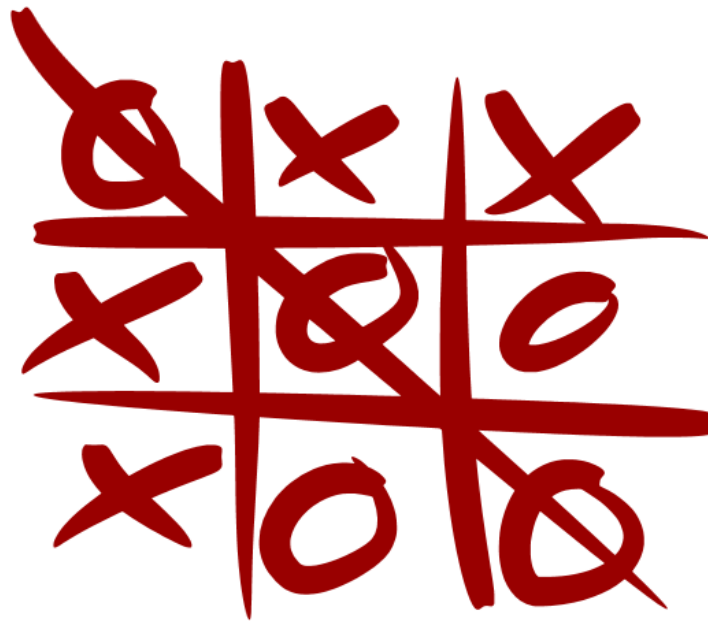
## □ Dodgem



# Một số ví dụ

4

## □ Tic-tac-toe



# Một số ví dụ

5

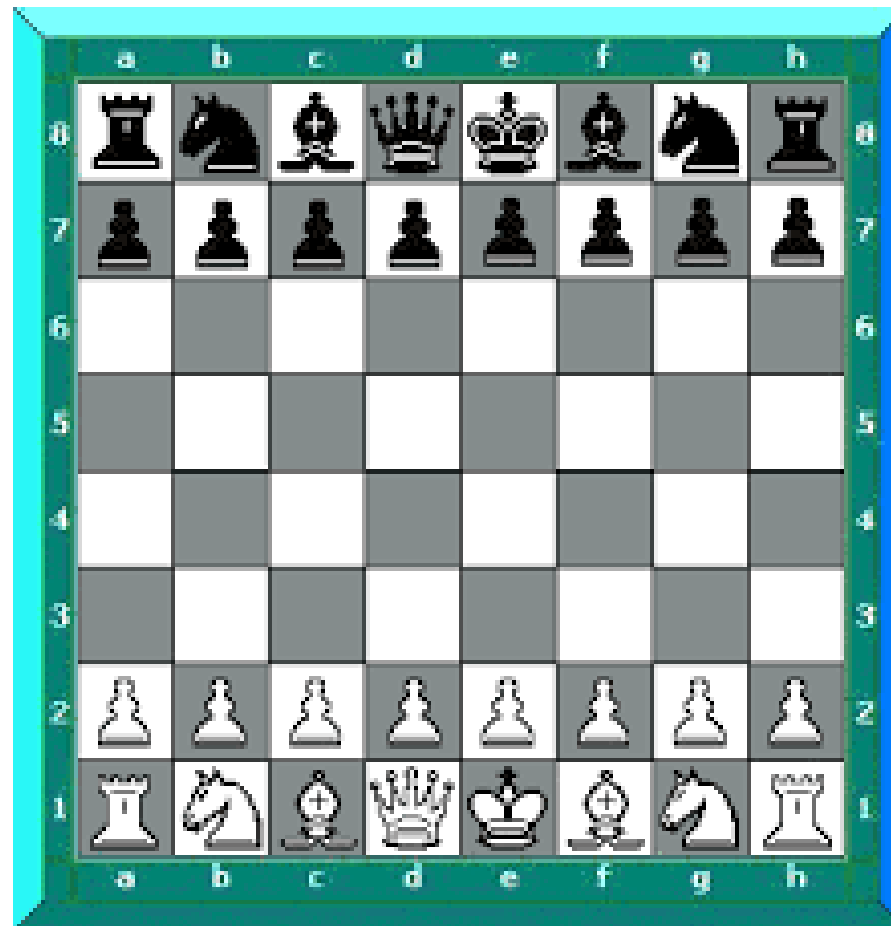
## □ Ô ăn quan



# Một số ví dụ

6

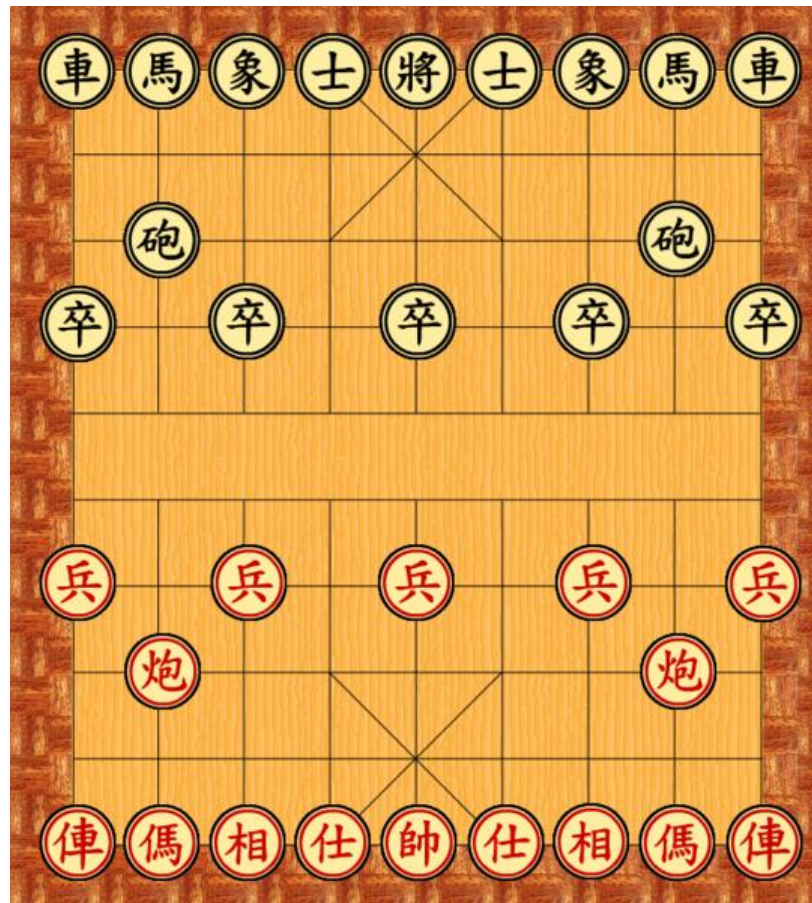
## □ Cờ vua



# Một số ví dụ

7

## □ Cờ tướng



# Tổng quan AI Game Players

- Tic Tac Toe
  - ▣ Tied for best player in world
- Othello
  - ▣ Computer better than any human
  - ▣ Human champions now refuse to play computer
- Backgammon
  - ▣ 1992, Tesauro combines 3-ply search & neural networks (with 160 hidden units) yielding top-3 player
- Checkers
  - ▣ 1994, Chinook ended 40-year reign of human champion Marion Tinsley
- Chess
  - ▣ 1997, Deep Blue beat human champion Gary Kasparov in six-game match
  - ▣ Deep Blue searches 200M positions/second, up to 40 ply
- Go
  - ▣ 2008, MoGo running on 25 nodes (800 cores) beat Myungwan Kim

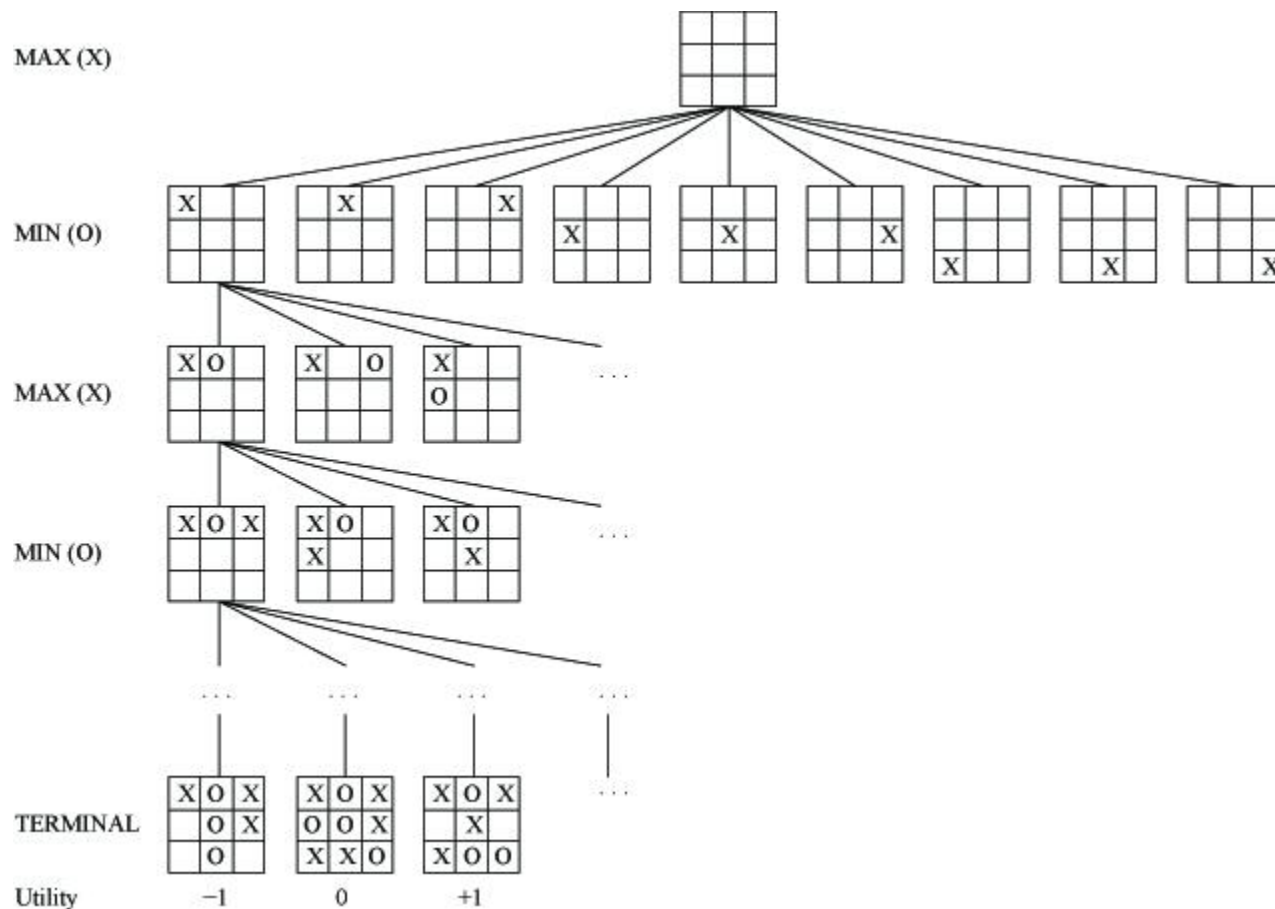


# Kỹ thuật tìm kiếm trong trò chơi

- Trạng thái khởi tạo
  - ▣ Trạng thái ban đầu của trò chơi
- Toán tử
  - ▣ Các nước đi hợp lệ mà người chơi có thể di chuyển
- Trạng thái kết thúc
  - ▣ Lá của cây trò chơi
  - ▣ Trạng thái kết thúc ván chơi
- Hàm đánh giá
  - ▣ Hàm kết cuộc
  - ▣ Giá trị đầu ra của trò chơi
  - ▣ Ví dụ: trò chơi tic tac toe: -1, 0, hoặc 1

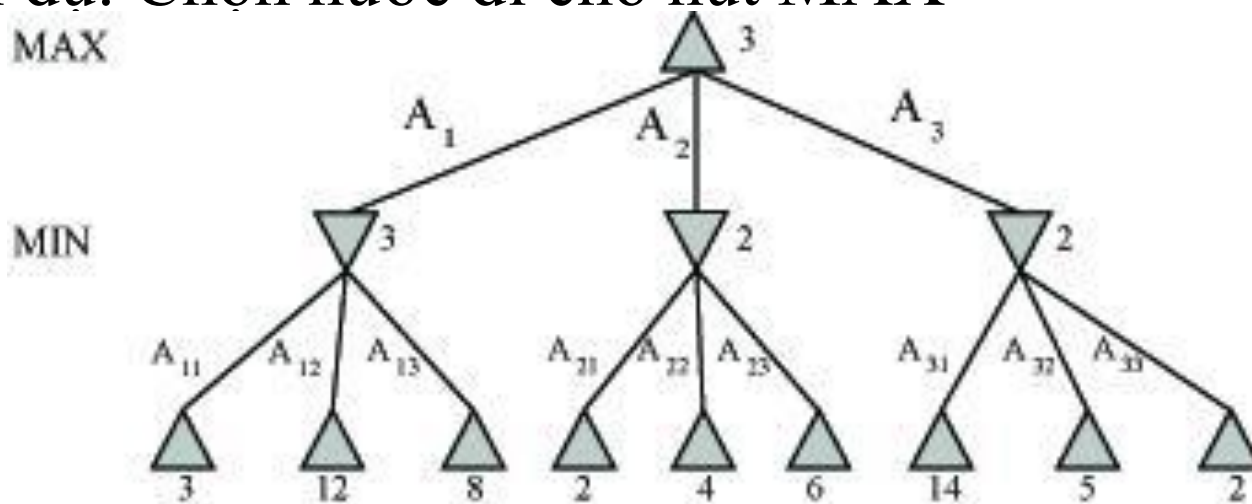
# Cây trò chơi

- Tic tac toe
- Hai người chơi, MAX (X) và MIN (O)
- Hai người lần lượt đi



# Giải thuật Minimax

- Tìm kiếm trên cây trò chơi tới nút lá
- Gán giá trị cho nút lá (kết thúc trò chơi)
- Tìm nước đi tốt nhất cho MAX :
  - MAX chọn nước đi có giá trị lớn nhất trong các nút con
  - MIN chọn nước đi có giá trị nhỏ nhất trong các nút con
- Ví dụ: Chọn nước đi cho nút MAX



# Giải thuật Minimax

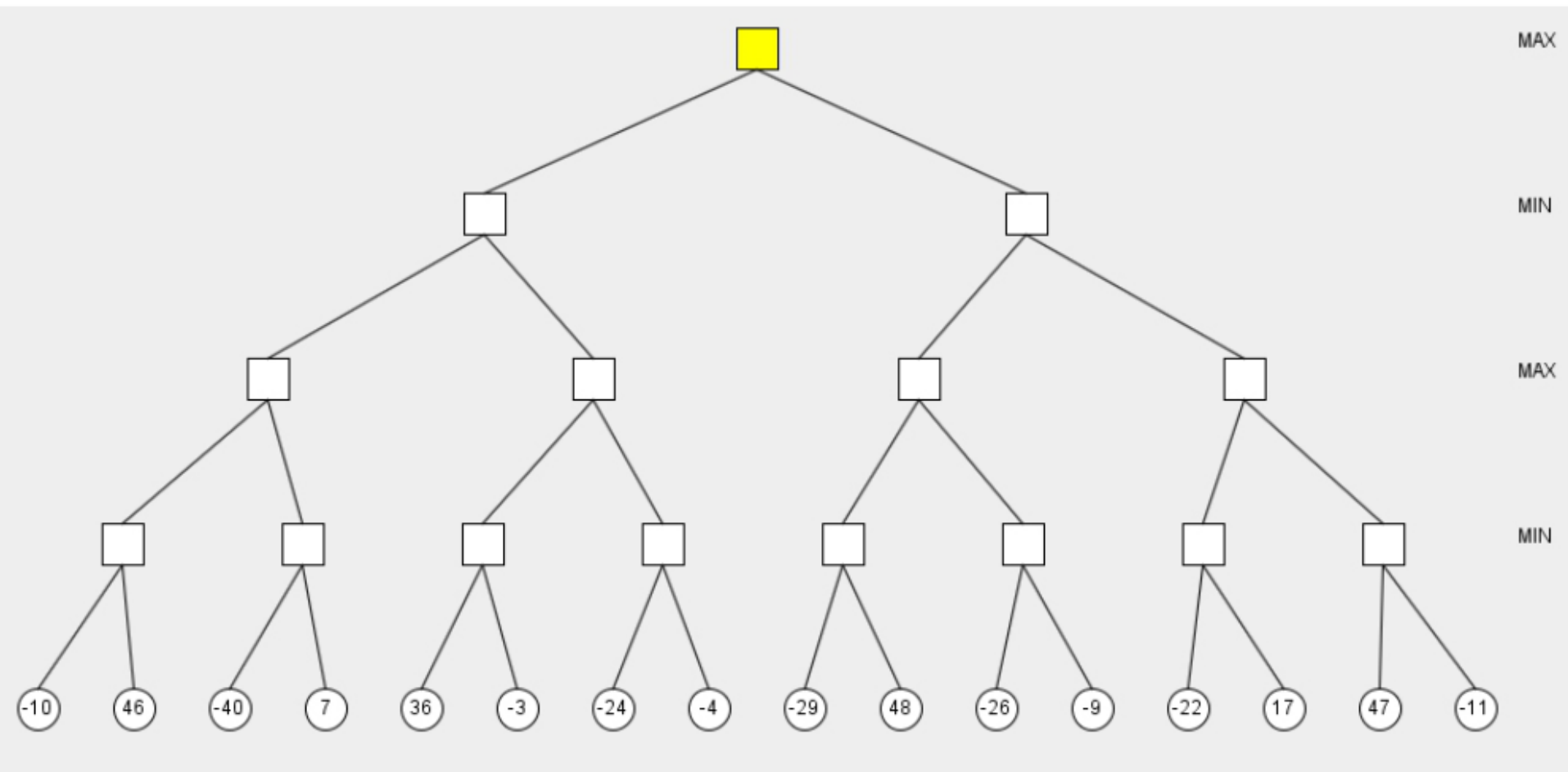
**function** MINIMAX-DECISION(*state*) *returns an action*  
**inputs:** *state*, current state in game  
 $v \leftarrow \text{MAX-VALUE}(\textit{state})$   
**return** the *action* in SUCCESSORS(*state*) with value  $v$

---

**function** MAX-VALUE(*state*) *returns a utility value*  
**if** TERMINAL-TEST(*state*) **then return** UTILITY(*state*)  
 $v \leftarrow -\infty$   
**for**  $a, s$  in SUCCESSORS(*state*) **do**  
 $v \leftarrow \text{MAX}(v, \text{MIN-VALUE}(s))$   
**return**  $v$

---

**function** MIN-VALUE(*state*) *returns a utility value*  
**if** TERMINAL-TEST(*state*) **then return** UTILITY(*state*)  
 $v \leftarrow +\infty$   
**for**  $a, s$  in SUCCESSORS(*state*) **do**  
 $v \leftarrow \text{MIN}(v, \text{MAX-VALUE}(s))$   
**return**  $v$



# Đặc điểm của Minimax

- Thuật toán Minimax là thuật toán tìm kiếm theo độ sâu.
- Về mặt lý thuyết chiến lược Minimax cho phép ta tìm ra nước đi tối ưu cho Trắng.
- Thực tế chúng ta không có đủ thời gian để xác định được nước đi tối ưu vì phải xem xét đến toàn bộ các đỉnh của cây trò chơi.
  - ▣ Xét đến độ sâu  $h$
  - ▣ Xây dựng hàm lượng giá

# Xây dựng hàm lượng giá

15

u

●		
●	○	○

30	35	40
15	20	25
0	5	10

Value for White

v

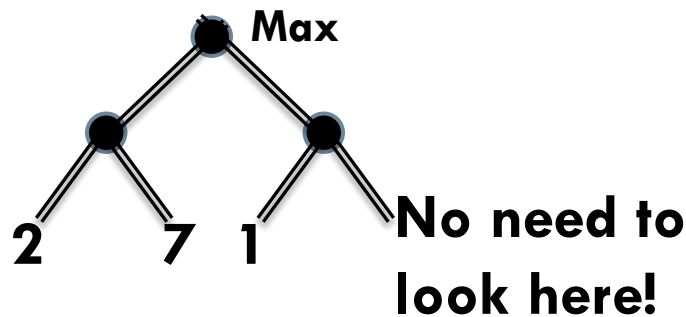
	●	
	○	
●		○

-10	-25	-40
-5	-20	-35
0	-15	-30

Value for Black

# Cắt tỉa Alpha-Beta

- ❑ Trò chơi cờ vua xét đến độ sâu 3-4
- ❑ Cắt tỉa Alpha-beta giảm bớt không gian tìm kiếm mà không ảnh hưởng tới kết quả



Sử dụng cặp giá trị  $[\alpha, \beta]$  tại mỗi nút tìm kiếm

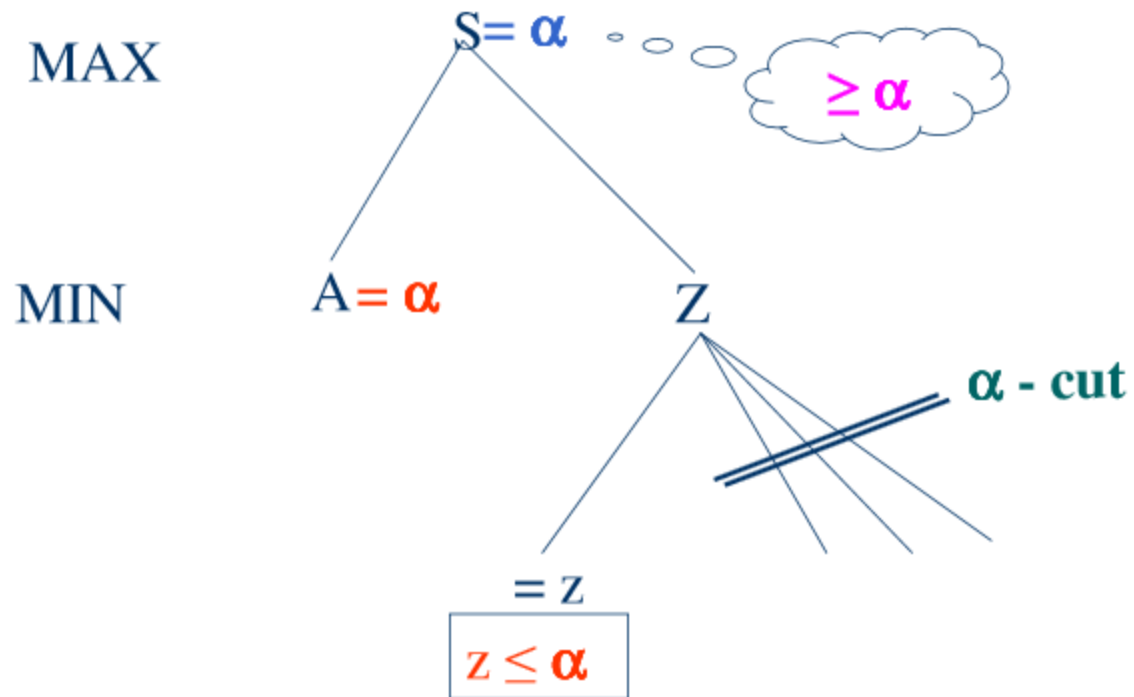
$\alpha$  = giá trị lớn nhất trong các giá trị của các đỉnh con đã đánh giá của một đỉnh Max

$\beta$  = giá trị nhỏ nhất trong các giá trị của các đỉnh con đã đánh giá của một đỉnh Min



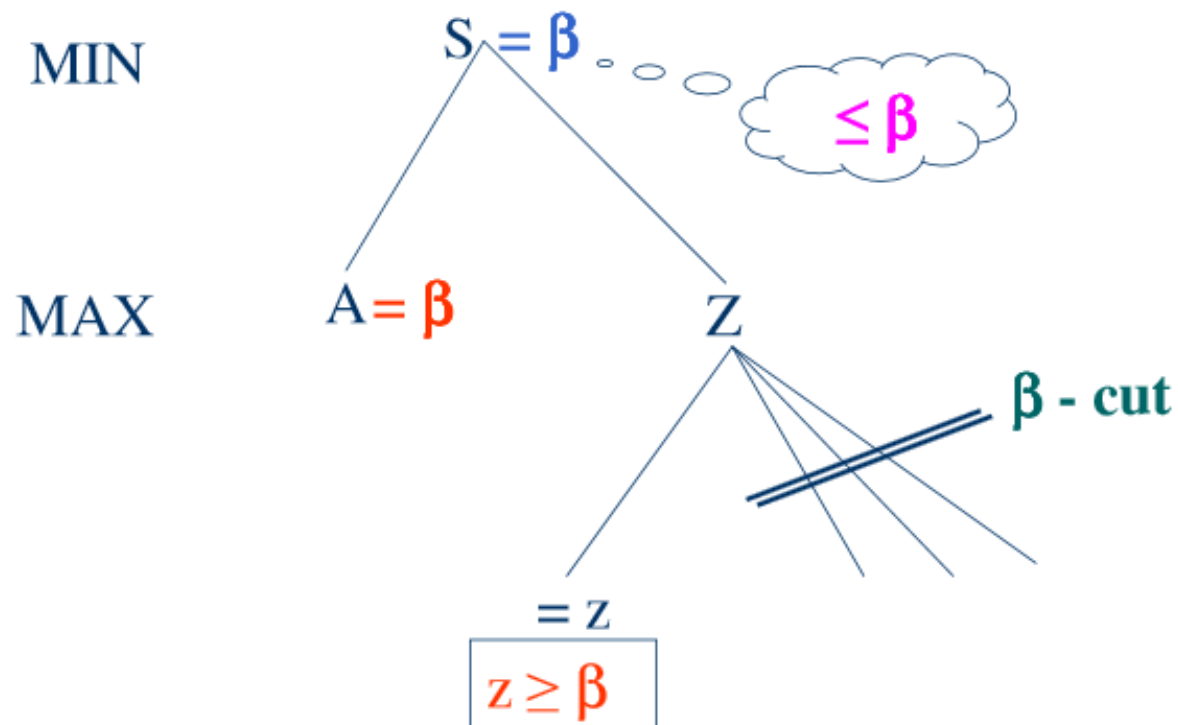
# $\alpha$ -cut

17



# $\beta$ -cut

18



# Giải thuật tìm kiếm alpha-beta

**function** ALPHA-BETA-SEARCH(*state*) *returns an action*

**inputs:** *state*, current state in game

$v \leftarrow \text{MAX-VALUE}(\text{state}, -\infty, +\infty)$

**return** the action in SUCCESSORS(*state*) with value *v*

---

**function** MAX-VALUE(*state*,  $\alpha$ ,  $\beta$ ) *returns a utility value*

**if** TERMINAL-TEST(*state*) **then return** UTILITY(*state*)

$v \leftarrow -\infty$

**for** *a, s* in SUCCESSORS(*state*) **do**

$v \leftarrow \text{MAX}(v, \text{MIN-VALUE}(s, \alpha, \beta))$

**if**  $v \geq \beta$  **then return** *v*

$\alpha \leftarrow \text{MAX}(\alpha, v)$

**return** *v*

---

**function** MIN-VALUE(*state*,  $\alpha$ ,  $\beta$ ) *returns a utility value*

**if** TERMINAL-TEST(*state*) **then return** UTILITY(*state*)

$v \leftarrow +\infty$

**for** *a, s* in SUCCESSORS(*state*) **do**

$v \leftarrow \text{MIN}(v, \text{MAX-VALUE}(s, \alpha, \beta))$

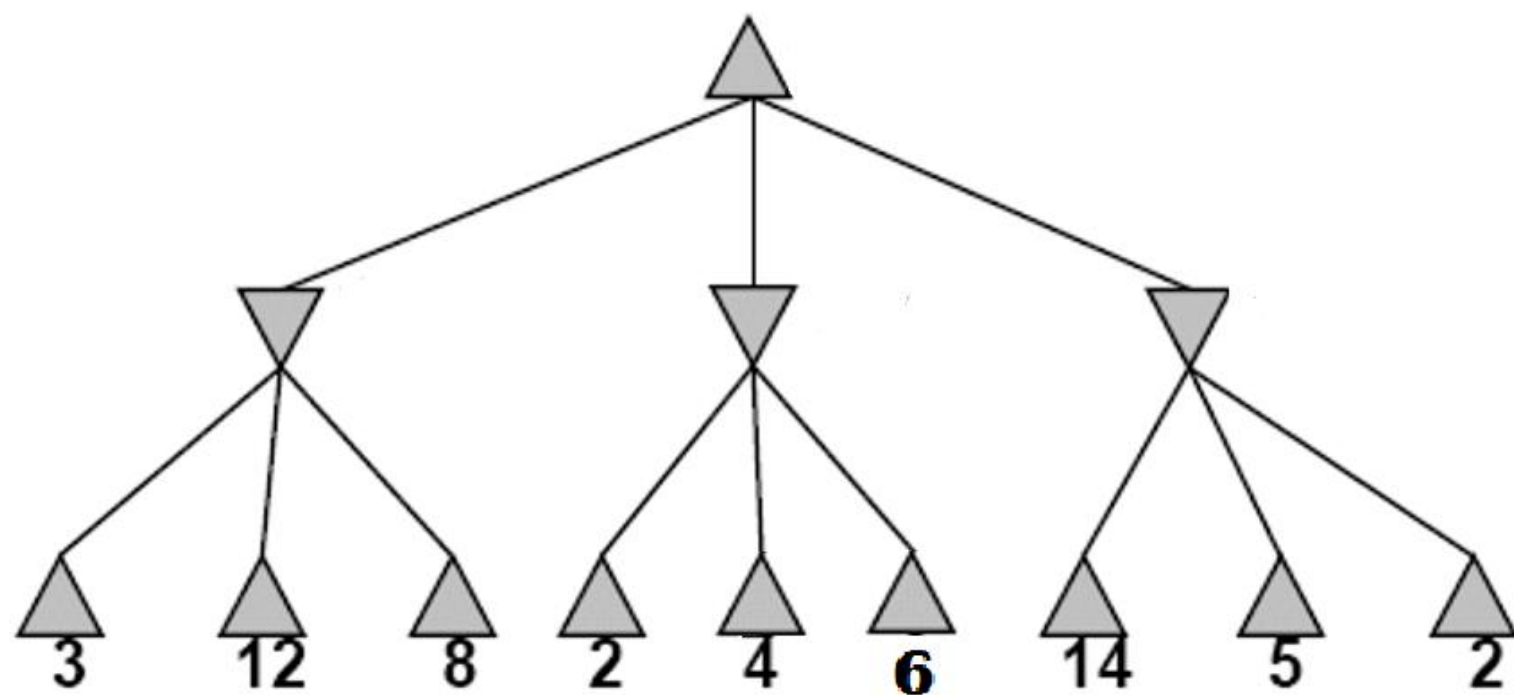
**if**  $v \leq \alpha$  **then return** *v*

$\beta \leftarrow \text{MIN}(\beta, v)$

**return** *v*

MAX

MIN



MAX



A

B

C

3

5

F

G

4

I

5

7

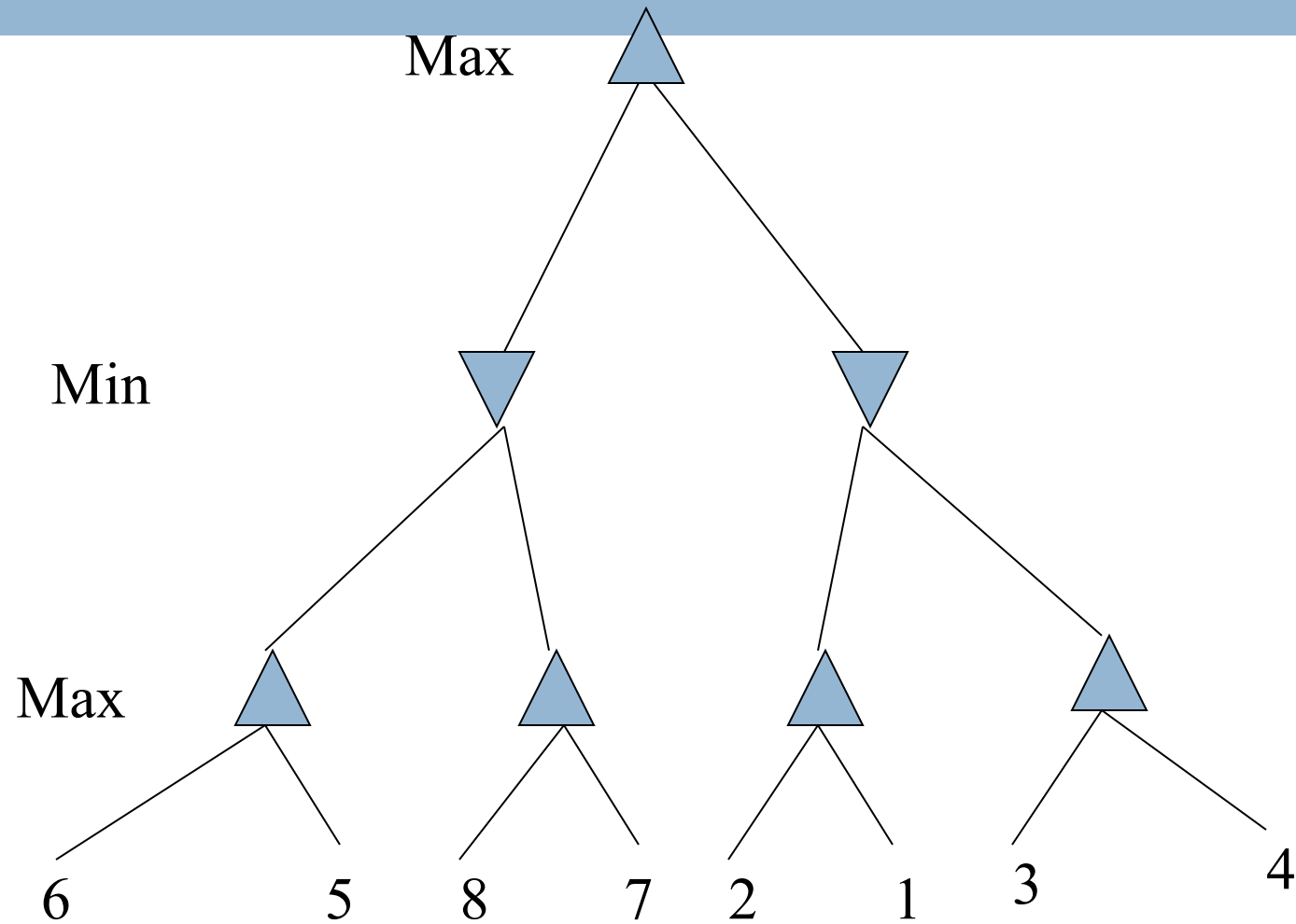
8

0

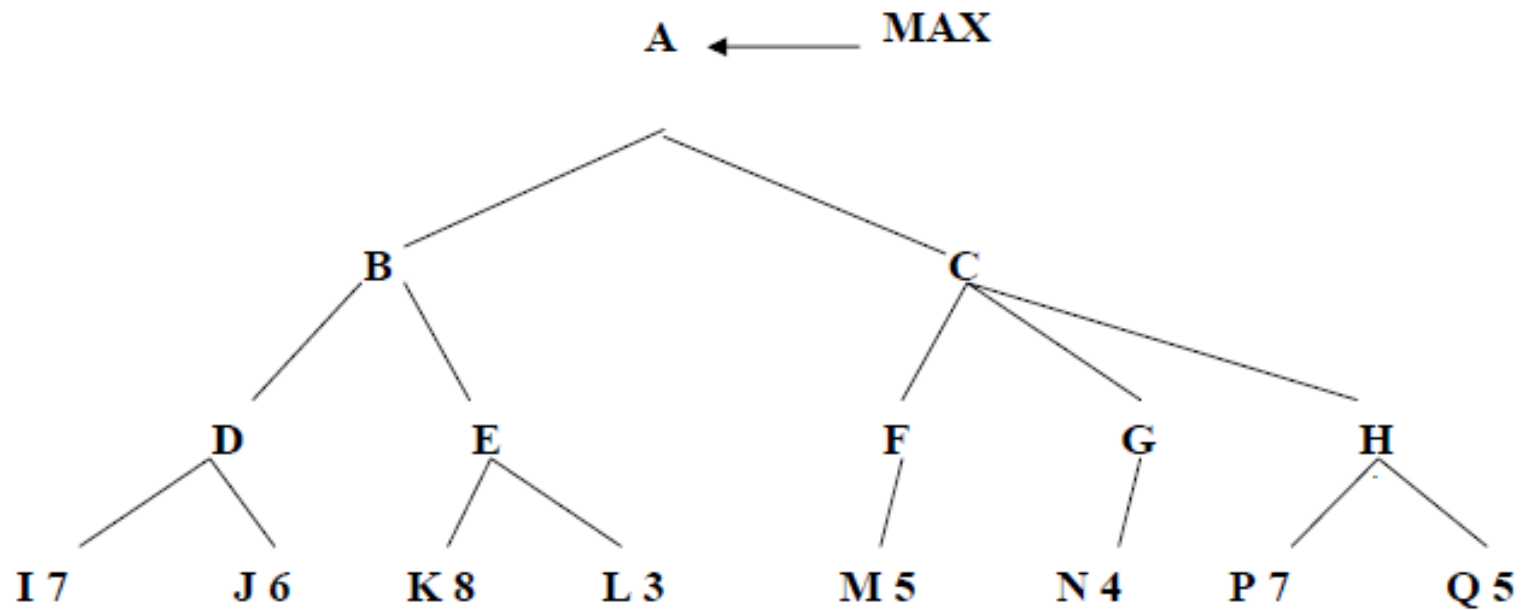
7



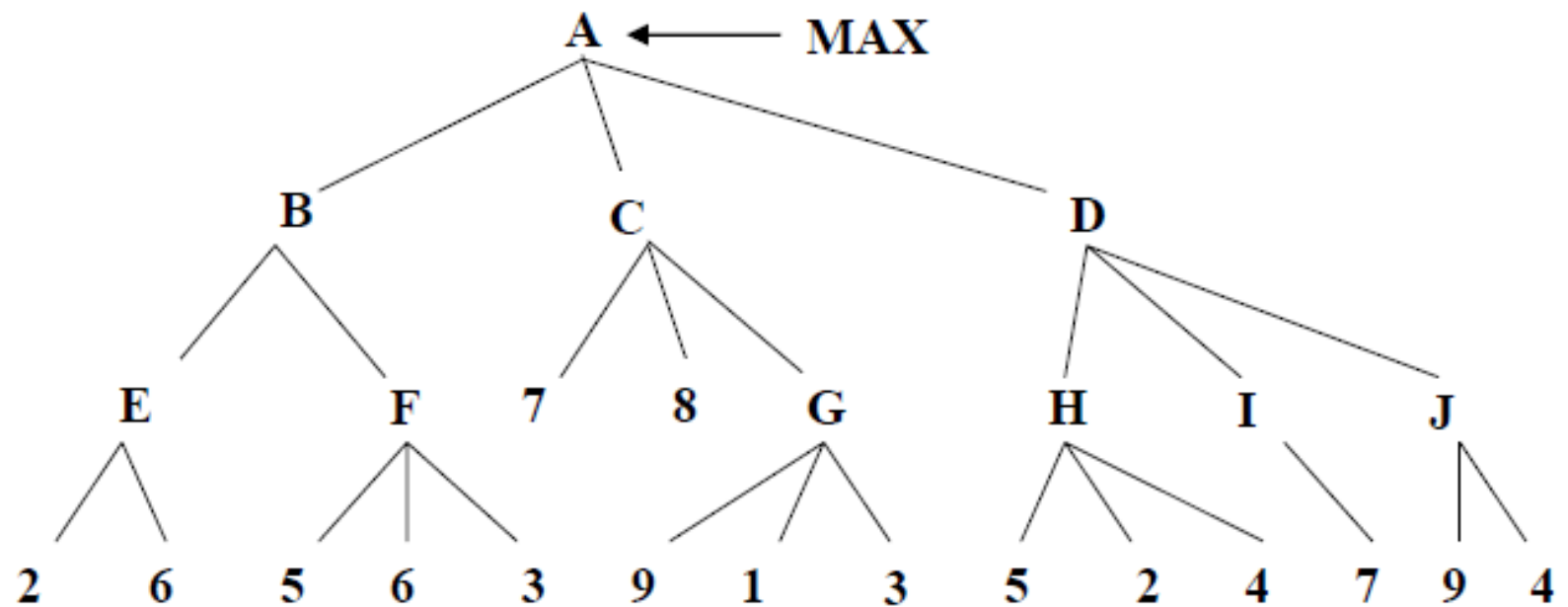
# Ví dụ



# Alpha-Beta

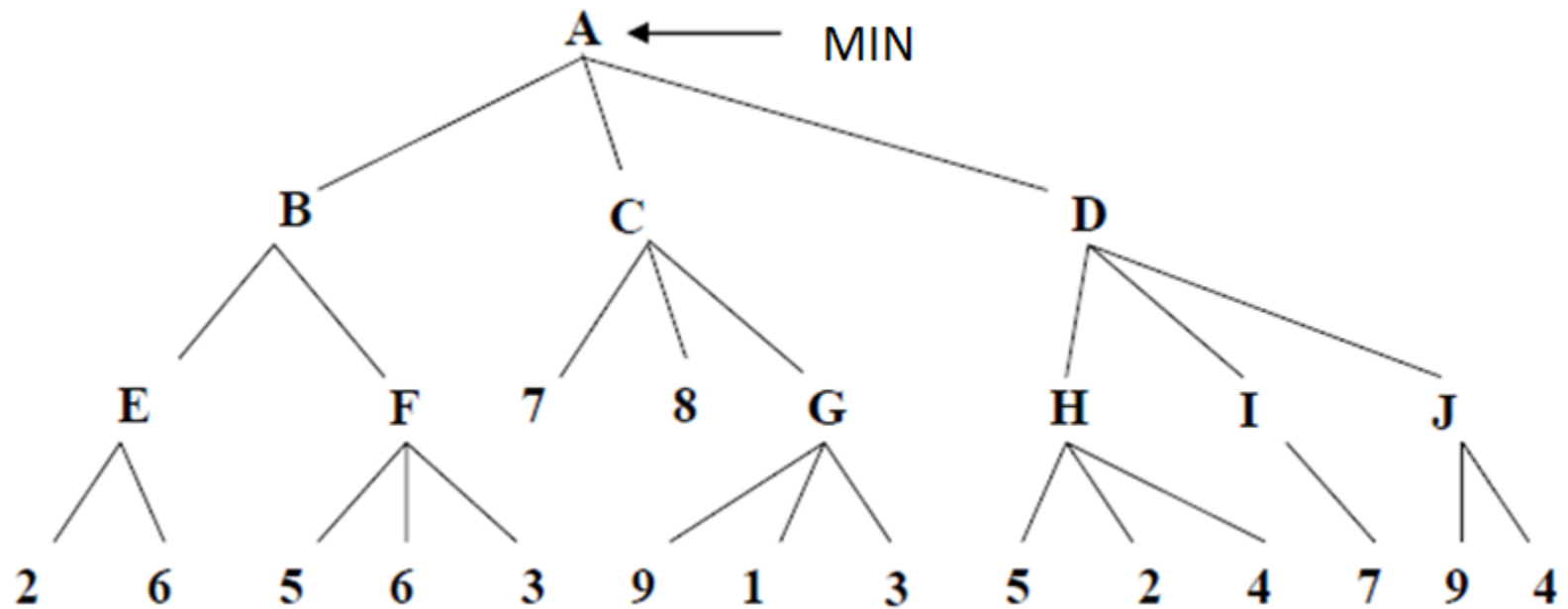


# Alpha-Beta





# Alpha-Beta

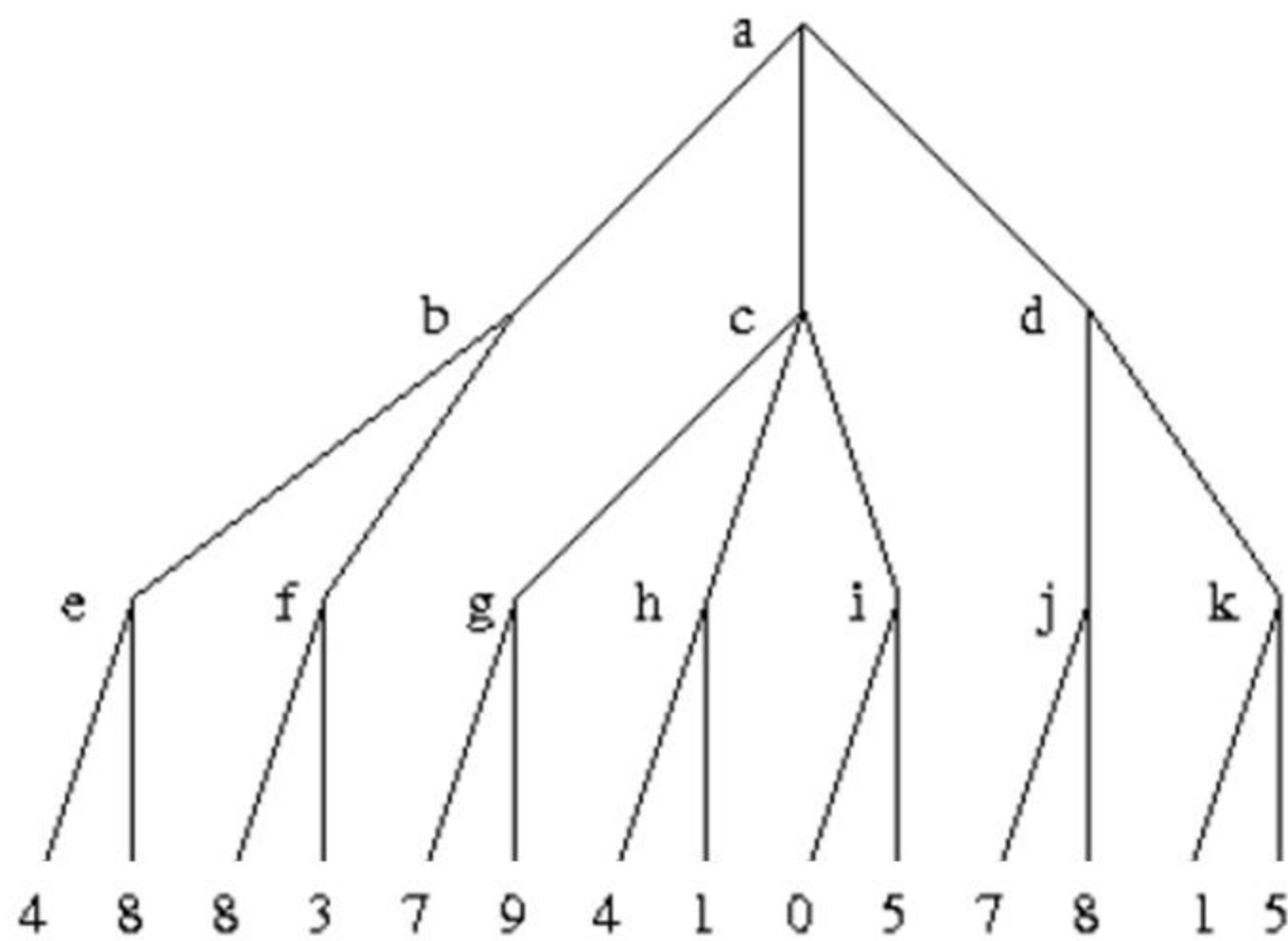


max

min

max

min



MAX

MIN

MAX

