

# **Brand And Bound**

**GVHD: Nguyễn Thanh Sơn**

**Nhóm 10**

# Danh sách thành viên:

1

Phan Nguyễn Thành Nhân - 19521943

2

Nguyễn Thành Nghĩa - 19521899

3

Tạ Huỳnh Đức Huy - 19521634



# 1. Đặt vấn đề

# Bài toán Ăn khế trả vàng (Knapsack)



Ngày xưa ngày xưa, trong câu chuyện “Ăn khế trả vàng”, chim thần đã chở người anh tới một hòn đảo có chứa các loại đá quý có trọng lượng và giá trị khác nhau



Nhưng người anh chỉ đem theo túi sách chứa tối đa là  $M$   
Với bản tính tham lam, anh ta muốn giàu có nhất.



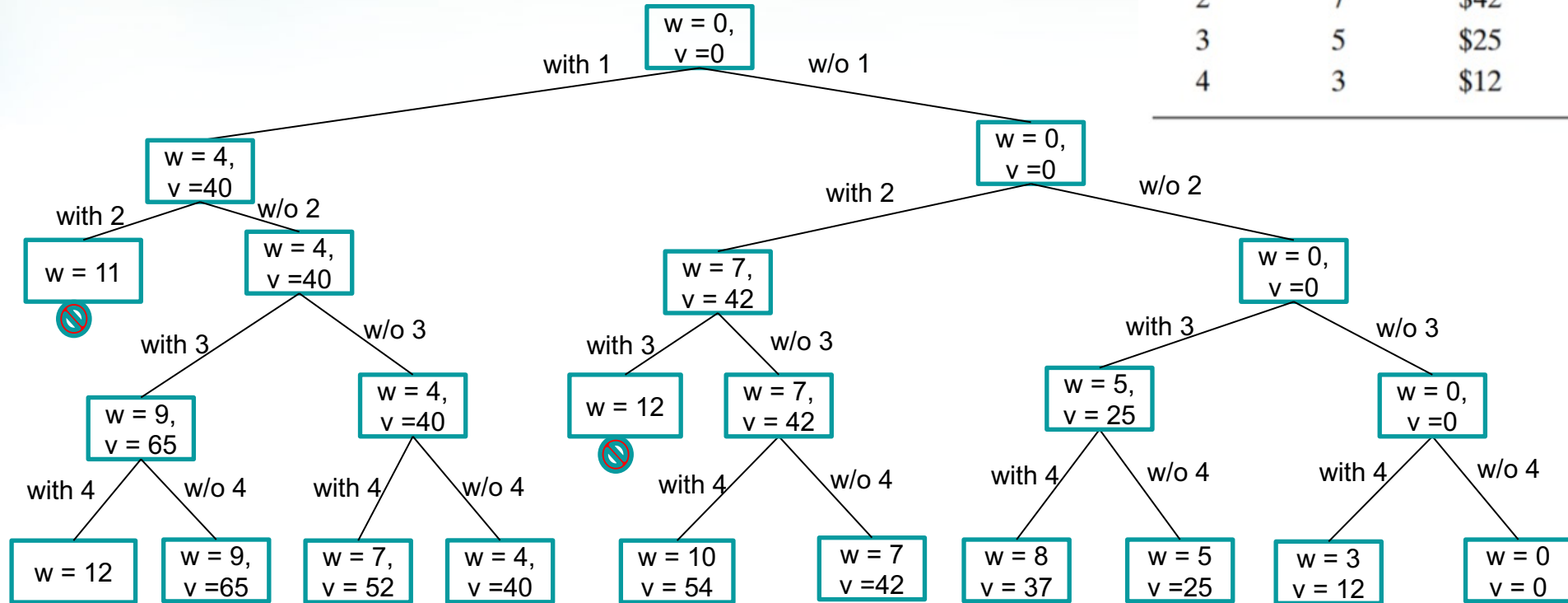
Hãy giúp anh ta lấy những viên đá quý sao cho có giá trị cao nhất mà vẫn chứa vừa cái túi.

item	weight	value
1	4	\$40
2	7	\$42
3	5	\$25
4	3	\$12

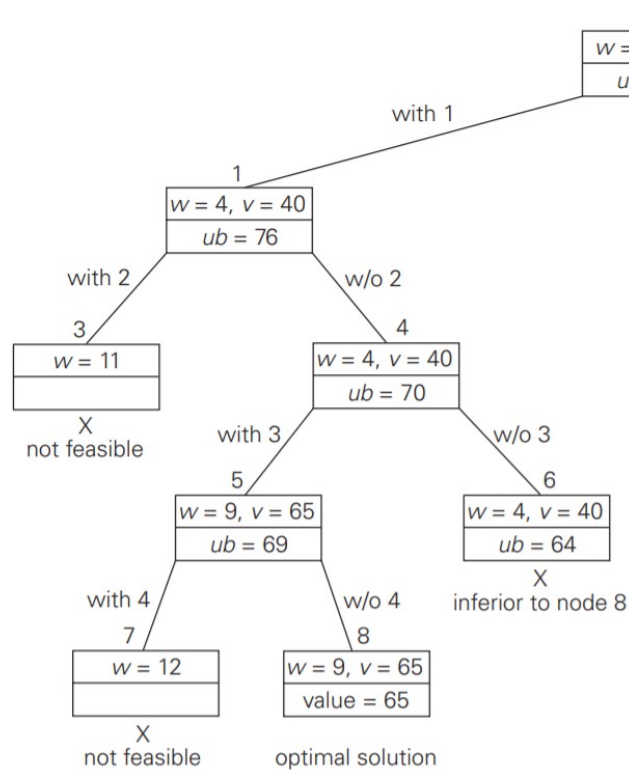
$$M = 10$$

# Back tracking

item	weight	value
1	4	\$40
2	7	\$42
3	5	\$25
4	3	\$12



# Branch And Bound



item	weight	value	<u>value</u> weight
1	4	\$40	10
2	7	\$42	6
3	5	\$25	5
4	3	\$12	4



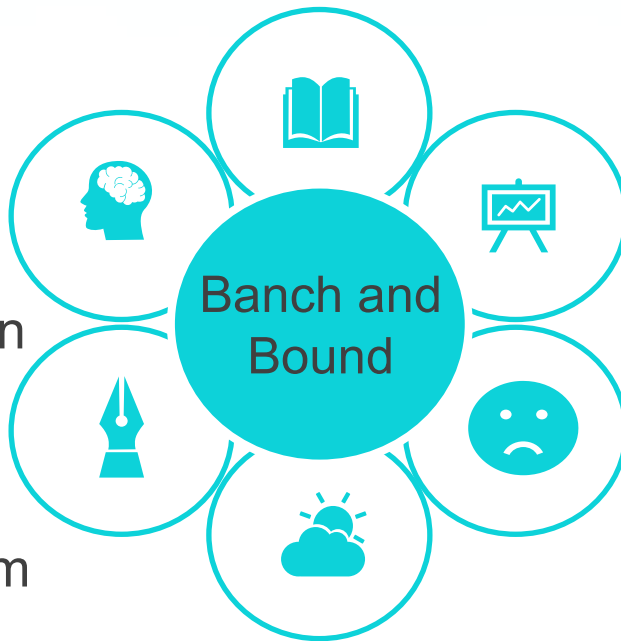
Khái Niệm



Mô hình thuật toán



Ưu và nhược điểm



Đặc điểm bài toán



Ví Dụ



Bài tập về nhà



# 1. Khái niệm



**Branch and bound** là thuật toán được thiết kế để tối ưu hóa bài toán tổ hợp và rời rạc.



Về bản chất **Branch and bound** là cải tiến của Backtracking.



Xây dựng cây tìm kiếm tối ưu, nhưng không xây dựng toàn bộ cây mà sử dụng giá trị cận để hạn chế bớt các nhánh.



## 2. Đặc điểm nhận dạng



Backtracking có thể giải các bài toán tối ưu bằng cách lựa chọn phương án tối ưu nhất trong tất cả các lời giải tìm được



Tuy nhiên, nhiều bài toán không gian các lời giải là quá lớn nên Backtracking khó đảm bảo về thời gian cũng như kỹ thuật

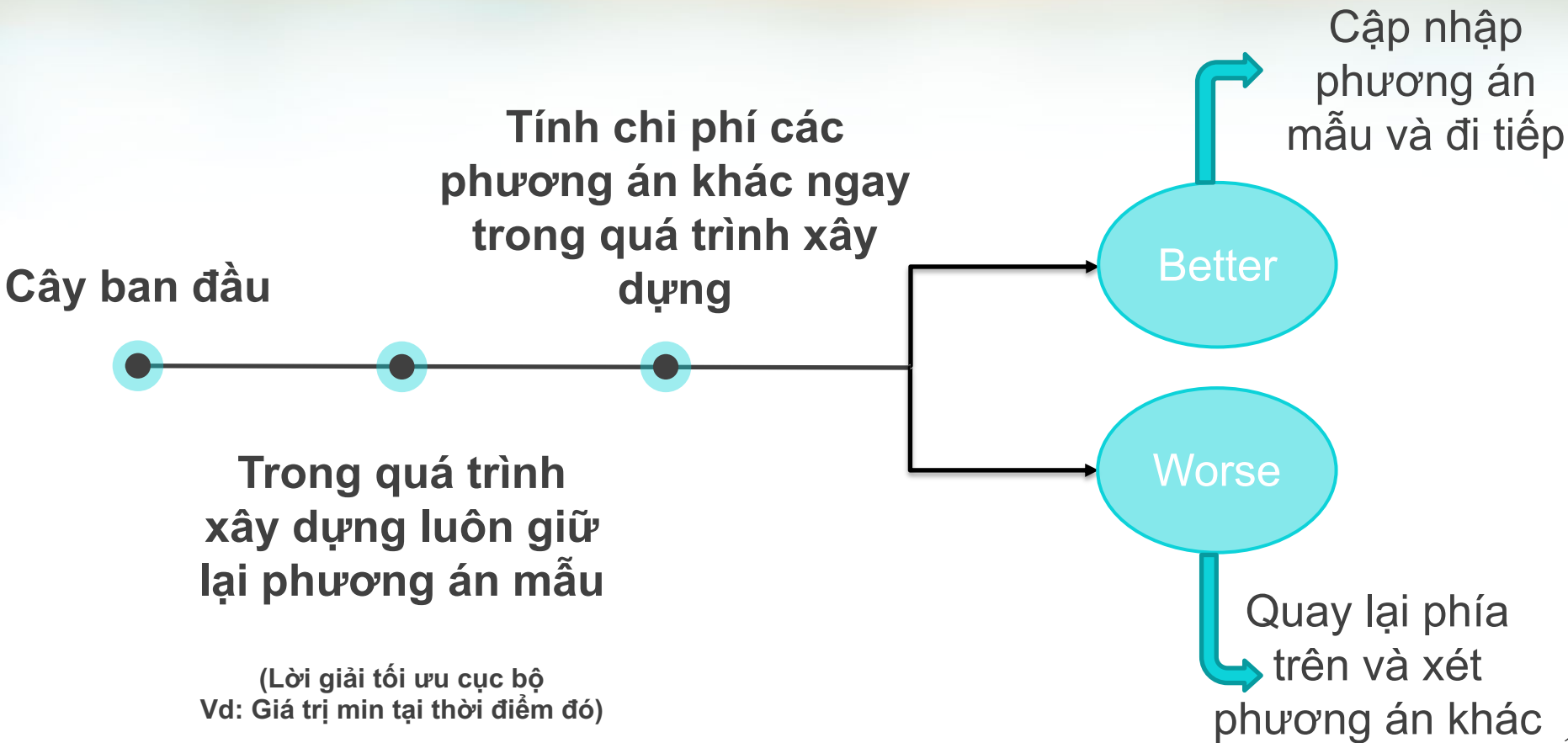


Vì thế **Brand and bound** là một trong những phương pháp cải tiến.



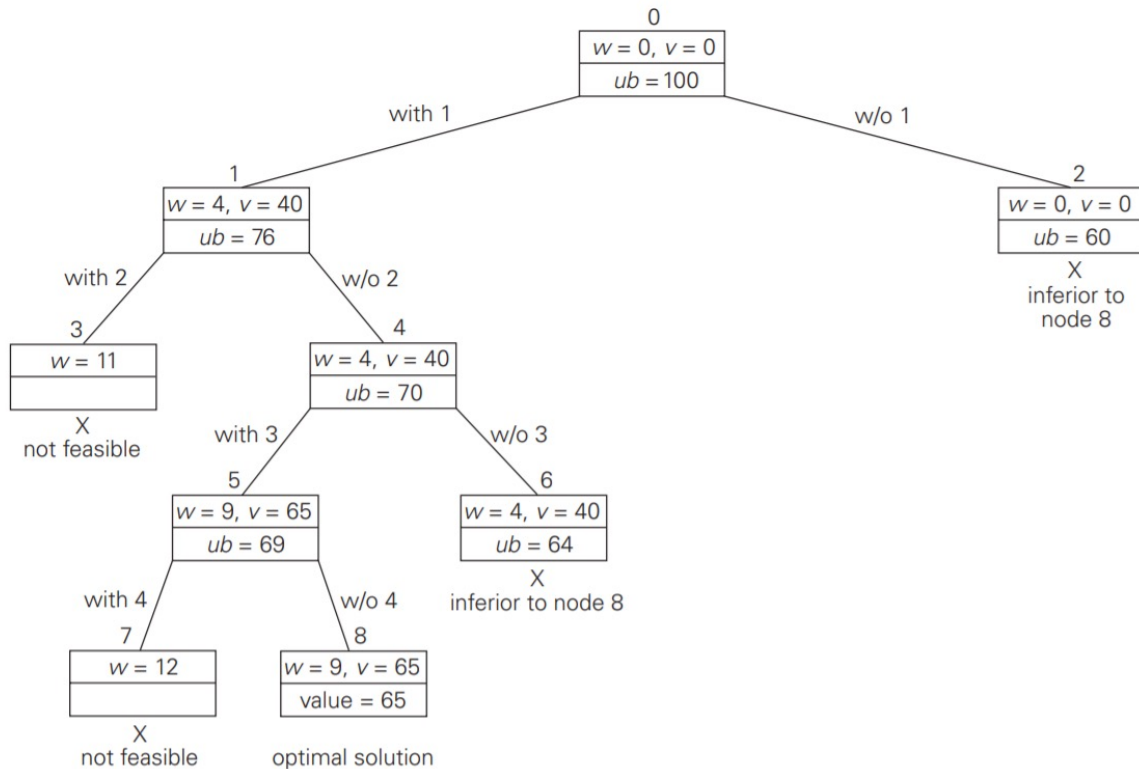
**Brand and bound** được áp dụng để tìm giải pháp tối ưu cho các bài toán tối ưu tổ hợp.

### 3. Mô hình chung thuật toán



# 3. Mô hình chung thuật toán

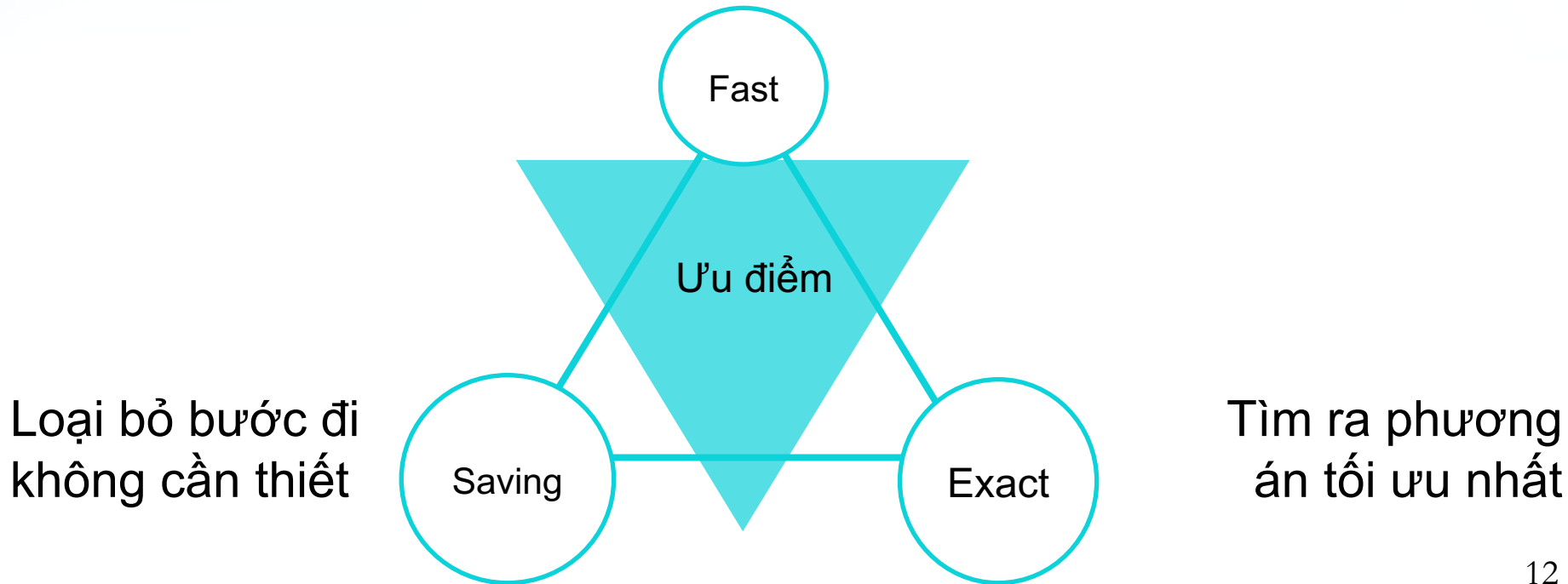
- Cây tìm kiếm
- Mỗi nút lá tương ứng với một phương án
- Với mỗi nút t, ta có thể tính được giá trị của hàm mục tiêu (với bài toán tối đa) hoặc giá trị của hàm mục tiêu (với bài toán tối thiểu)
- Với bài toán tối đa, ta có thể tính được giá trị cận trên (ub) của hàm mục tiêu tại mỗi nút t
- Với bài toán tối thiểu, ta có thể tính được giá trị cận dưới (lb) của hàm mục tiêu tại mỗi nút t



phương án có thể  
tính được giá trị của hàm mục tiêu tại mỗi nút con  
tính được giá trị cận trên (ub) của hàm mục tiêu tại mỗi nút con  
tính được giá trị cận dưới (lb) của hàm mục tiêu tại mỗi nút con  
tính được giá trị của hàm mục tiêu tại mỗi nút con

## 4. Ưu và nhược điểm

Có thể không cần duyệt hết nút trên cây



## 4. Ưu và nhược điểm



Cần xây dựng hàm tính  
cận phụ thuộc vào từng  
bài toán cụ thể

Bản chất vẫn là GT vét cạn  
Tốn nhiều thời gian

# Bài toán Ăn khế trả vàng (Knapsack)



Ngày xưa ngày xưa, trong câu chuyện “Ăn khế trả vàng”, chim thần đã chở người anh tới một hòn đảo có chứa các loại đá quý có trọng lượng và giá trị khác nhau

item	weight	value	$\frac{\text{value}}{\text{weight}}$
1	4	\$40	10
2	7	\$42	6
3	5	\$25	5
4	3	\$12	4



Nhưng người anh chỉ đem theo túi sách chứa tối đa là  $M$   
Với bản tính tham lam, anh ta muốn giàu có nhất.

$$M = 10$$



Hãy giúp anh ta lấy những viên đá quý sao cho có giá trị cao nhất mà vẫn chứa vừa cái túi

# 5. Ví dụ



## Knapsack problem



Tiếp cận bằng phương pháp tham lam

**B1:** Sắp xếp lại các món đồ theo chiều giảm dần tỷ lệ giá trị/khối lượng (value/weight)

**B2:** Tìm giới hạn trên  $ub$  ở mỗi tập hợp con bằng công thức:

$$ub = v + (W-w)(v_{i+1}/w_{i+1})$$

Với  $v$ ,  $w$  là tổng giá trị và khối lượng của tập hợp con

(khởi tạo ban đầu  $v = 0$ ,  $w = 0$ )

**B3:** Với mỗi tập hợp con đã tìm được, chọn ra tập hợp con có giá trị giới hạn trên lớn nhất để tiếp tục tìm kiếm

**B4:** Lặp lại bước 2 và bước 3 cho đến khi khối lượng vượt quá sức chứa của túi hoặc tìm được lời giải tối ưu đầu tiên.

# 5. Ví dụ



## Knapsack problem

item	weight	value	$\frac{\text{value}}{\text{weight}}$
1	4	\$40	10
2	7	\$42	6
3	5	\$25	5
4	3	\$12	4

The knapsack's capacity  $W$  is 10.

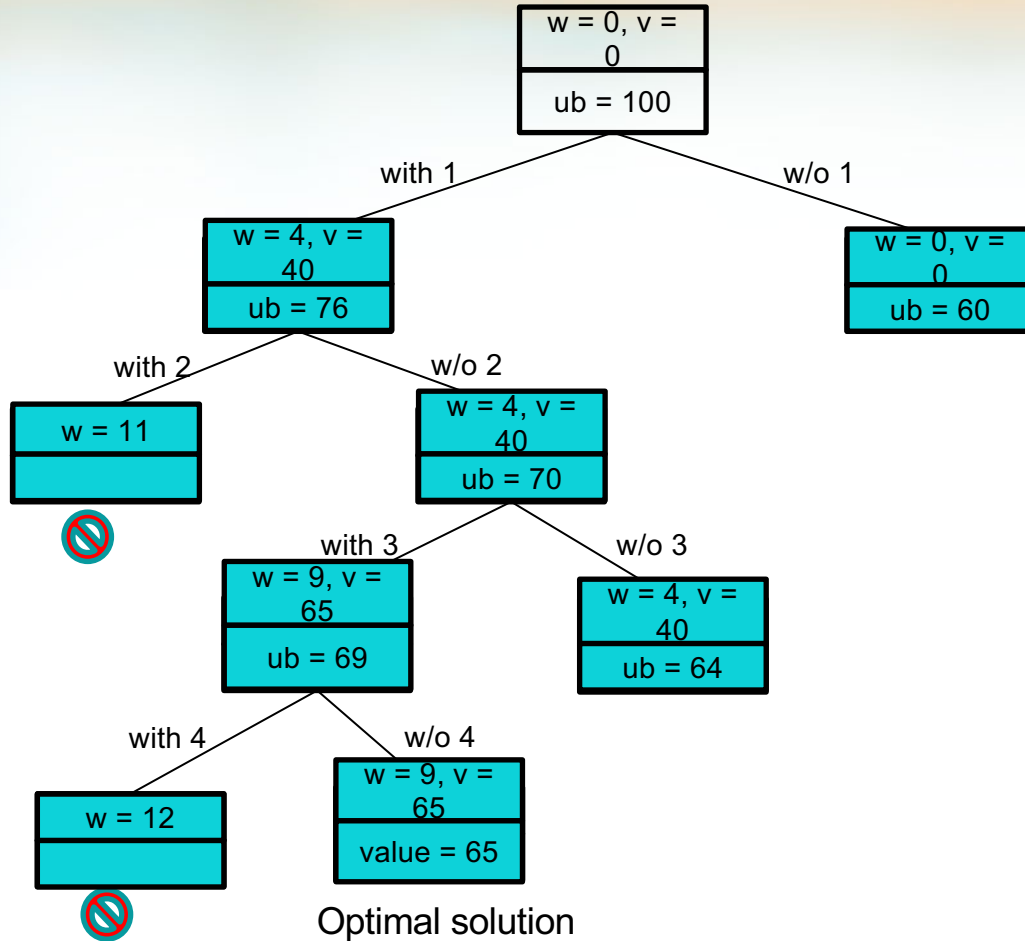


$$\begin{aligned}ub &= v + (W-w)(v_{i+1}/w_{i+1}) \\&= 0 + (10 - 0)(40/4) \\&= 100\end{aligned}$$



# 5. Ví dụ

## Branch and Bound



item	weight	value	$\frac{\text{value}}{\text{weight}}$
1	4	\$40	10
2	7	\$42	6
3	5	\$25	5
4	3	\$12	4

$$ub = v + (W - w)(v_{i+1}/w_{i+1}).$$

# 5. Ví dụ



## 8 puzzle problem



Cho một bảng  $3 \times 3$  với 8 ô (mỗi ô có một số từ 1 đến 8) và một ô trống. Các bạn hãy di chuyển làm sao cho giống với mẫu cuối cùng và dùng ít bước nhất. Ô trống có thể trượt trái, phải, trên, xuống)

Initial  
configuration

1	2	3
5	6	
7	8	4

Final  
configuration

1	2	3
5	8	6
	7	4

$$C = 0 + 4 = 4$$

1	2	3
5	6	
7	8	4

$$C = 1 + 5 = 6$$

1	2	
5	6	3
7	8	4

$$C = 1 + 5 = 6$$

1	2	3
5	6	4
7	8	

$$C = 1 + 3 = 4$$

1	2	3
5		6
7	8	4

$$C = 2 + 2 = 4$$

1	2	3
5	8	6
7		4

$$C = 2 + 4 = 6$$

1		3
5	2	6
7	8	4

$$C = 2 + 4 = 6$$

1	2	3
	5	6
7	8	4

$$C = 2 + 4 = 6$$

1	2	3
5	6	
7	8	4

$$C = 3 + 0 = 3$$

1	2	3
5	8	6
	7	4

$$C = 3 + 2 = 5$$

1	2	3
5	8	6
7	4	

$C(X) = g(X) + h(X)$  where

$g(X)$  = cost of reaching the current node from the root

$h(X)$  = cost of reaching an answer node from  $X$ .

Final configuration

1	2	3
5	8	6
	7	4

# 6. Bài Tập về nhà



Nộp bài qua email: 19521943@gm.uit.edu.vn

Một nhà máy, có  $N$  công nhân và  $N$  công việc. Mỗi công nhân làm từng việc trong thời gian khác nhau, tuy nhiên mỗi người chỉ làm một việc và không ai làm trùng với người khác. Bạn hãy phân chia sao cho tổng thời gian làm việc của các công nhân là ít nhất.

$$C = \begin{matrix} & \begin{matrix} \text{job 1} & \text{job 2} & \text{job 3} & \text{job 4} \end{matrix} \\ \begin{bmatrix} 9 & 2 & 7 & 8 \\ 6 & 4 & 3 & 7 \\ 5 & 8 & 1 & 8 \\ 7 & 6 & 9 & 4 \end{bmatrix} & \begin{matrix} \text{person } a \\ \text{person } b \\ \text{person } c \\ \text{person } d \end{matrix} \end{matrix}$$

Input: đầu vào là 1 số nguyên  $n$ , mỗi  $n$  dòng tiếp theo là thời gian làm việc của từng công nhân trong từng việc.

Output: Tổng thời gian làm việc ít nhất của các công nhân.

Input	Output
4 9 2 7 8 6 4 3 7 5 8 1 8 7 6 9 4	13

# Tài liệu tham khảo



Introduction to the Design and Analysis of Algorithms



<https://www.geeksforgeeks.org/branch-and-bound-algorithm/>



<https://www.baeldung.com/cs/branch-and-bound>



Thank you