



CANTHO UNIVERSITY

CHƯƠNG 3

KỸ THUẬT THIẾT KẾ THUẬT TOÁN

Bộ môn CÔNG NGHỆ PHẦN MỀM
Khoa Công nghệ thông tin và Truyền thông
Đại học Cần Thơ

Võ Huỳnh Trâm



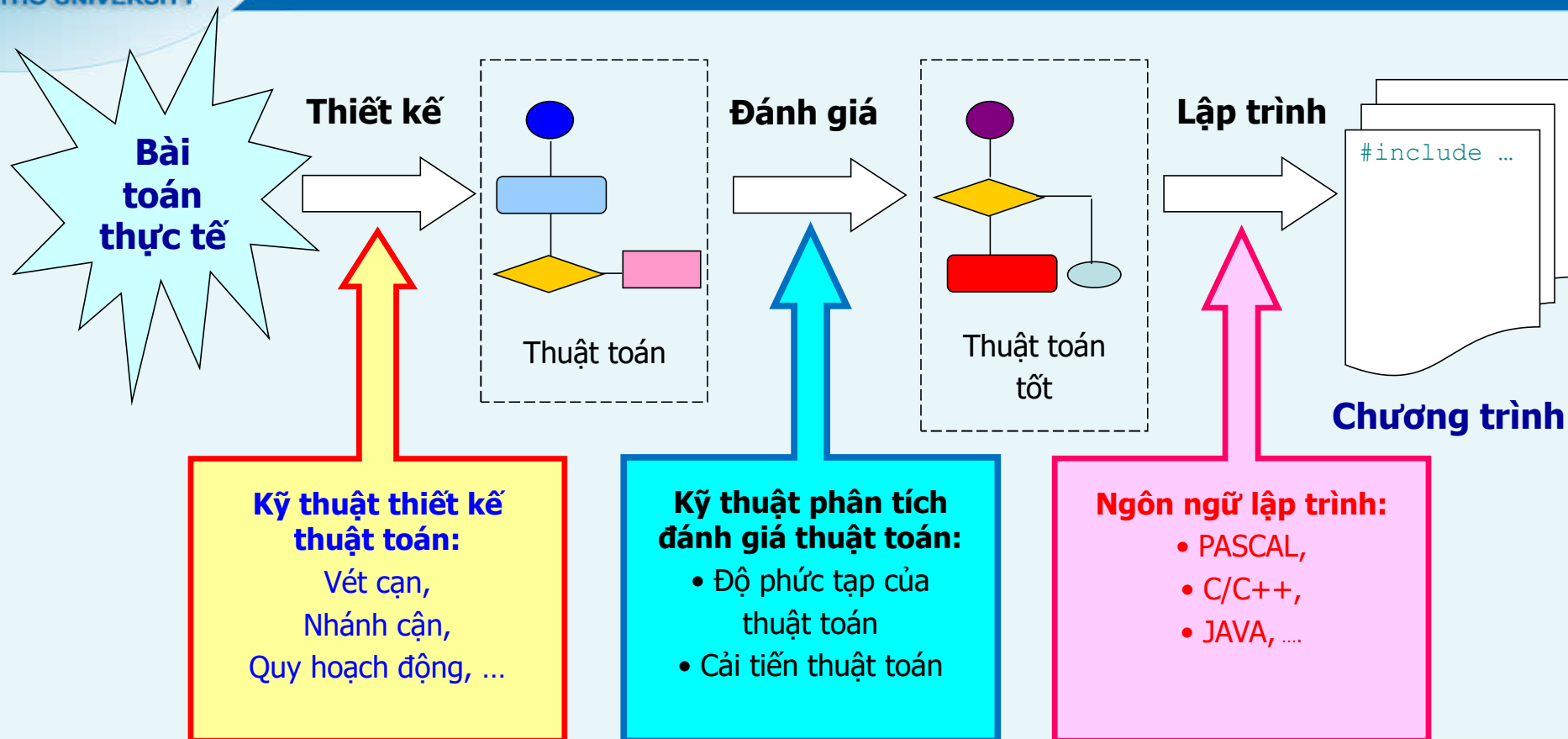
Mục tiêu

- **Biết** các kỹ thuật thiết kế thuật toán: từ ý tưởng cho đến thuật toán chi tiết.
- **Hiểu** rõ nguyên lý của các kỹ thuật phân tích thiết kế thuật toán.
- **Vận dụng** kỹ thuật phân tích thiết kế để giải các bài toán thực tế: các bài toán dạng nào thì có thể áp dụng được kỹ thuật này.



CANTHO UNIVERSITY

Mô hình từ bài toán đến chương trình





Hậu quả của sai sót

- **Lỗi thiết kế** có thể phải **trả giá đắt** nếu không được phát hiện và điều chỉnh sớm trong quá trình phát triển phần mềm.
- Theo báo cáo của **Boehm** và **Papaccio** (1988) ước lượng chi phí cho sửa lỗi phần mềm như sau:

Phân tích yêu cầu (1\$) \Rightarrow Thiết kế (5\$) \Rightarrow Lập trình (10\$)
 \Rightarrow Kiểm thử (20\$) \Rightarrow Triển khai hệ thống (>200\$)



CANTHO UNIVERSITY

Cầu Québec (*Pont de Québec*) Canada



(1903-**29/8/1907** ... 1913-**11/9/1916**-1919)



CANTHO UNIVERSITY

The Iron Ring

– A Canadian Engineering tradition (1922)





CANTHO UNIVERSITY

“The Calling of an Engineer” – Nghi lễ lời thề Kỹ sư (Canada – từ 1925)





CANTHO UNIVERSITY

The Iron Ring

– A Canadian Engineering tradition





CANTHO UNIVERSITY

The Iron Ring – A Canadian Engineering tradition





Một số kỹ thuật thiết kế thuật toán

- Kỹ thuật *Chia để trị* (Divide and Conquer)
- Kỹ thuật **Tham ăn** /*Háu ăn/Tham lam* (Greedy)
- Kỹ thuật **Nhánh cận** (Branch and Bound)
- Kỹ thuật **Quy hoạch động** (Dynamic Programming)
- Kỹ thuật **Quay lui** (Backtracking)
- Kỹ thuật **Cắt tỉa alpha-beta** (Alpha-Beta Pruning) trên Cây trò chơi
- Kỹ thuật *Tìm kiếm địa phương* (Local Search)



CANTHO UNIVERSITY

CHƯƠNG 3

KỸ THUẬT THIẾT KẾ THUẬT TOÁN

KỸ THUẬT CHIA ĐỀ TRỊ

(Đọc tài liệu)



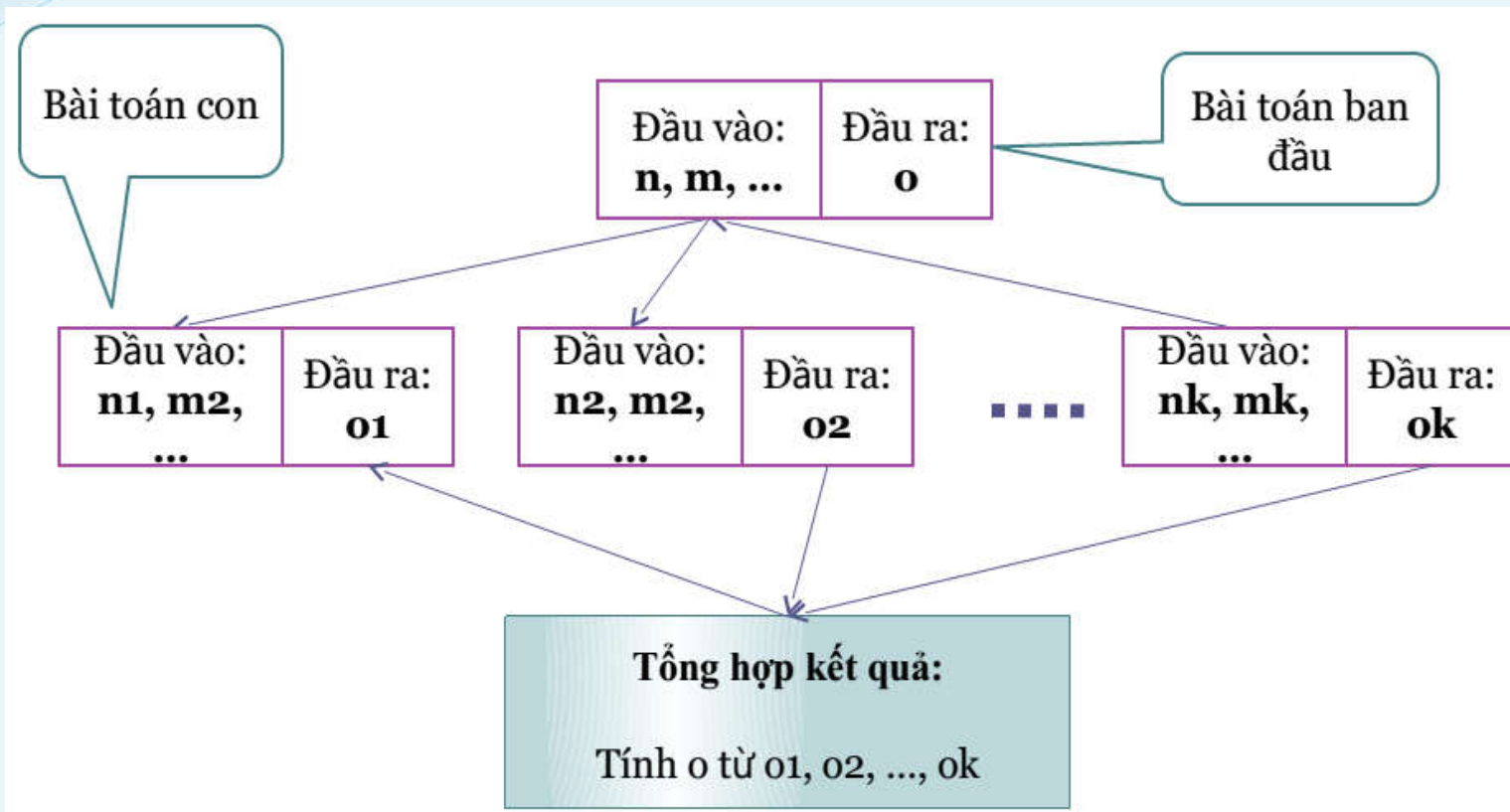
Kỹ thuật Chia để trị

- **Yêu cầu:** Cần phải giải bài toán có kích thước n .
- **Phương pháp:**
 - Chia bài toán ban đầu thành một số bài toán con đồng dạng với bài toán ban đầu có kích thước $< n$.
 - Giải các bài toán con được các lời giải con
 - Tổng hợp lời giải con \rightarrow lời giải bài toán ban đầu.
- **Chú ý :** Đối với từng bài toán con: chia chúng thành các bài toán con nhỏ hơn nữa.

Quá trình phân chia dừng lại khi kích thước bài toán đủ nhỏ để giải dễ dàng: *bài toán cơ sở*.



Kỹ thuật Chia để trị (Phân tích)





Kỹ thuật Chia để trị (thuật toán)

```
solve(n) {  
    if (n đủ nhỏ để có thể giải được)  
        giải bài toán → KQ  
    return KQ;  
    else {  
        Chia bài toán thành các bài toán con  
        kích thước n1, n2, ...  
  
        KQ1 = solve(n1); //giải bài toán con 1  
        KQ2 = solve(n2); //giải bài toán con 2  
        ...  
        Tổng hợp các kết quả KQ1, KQ2, ... → KQ  
    }  
    return KQ;  
}
```



Minh họa Kỹ thuật Chia để trị

- **Kỹ thuật Chia để trị**
(Divide and Conquer)

(1) Quick Sort / Merge Sort

(2) Bài toán nhân hai số nguyên lớn

(3) Bài toán xếp lịch thi đầu thể thao



Ví dụ: QuickSort

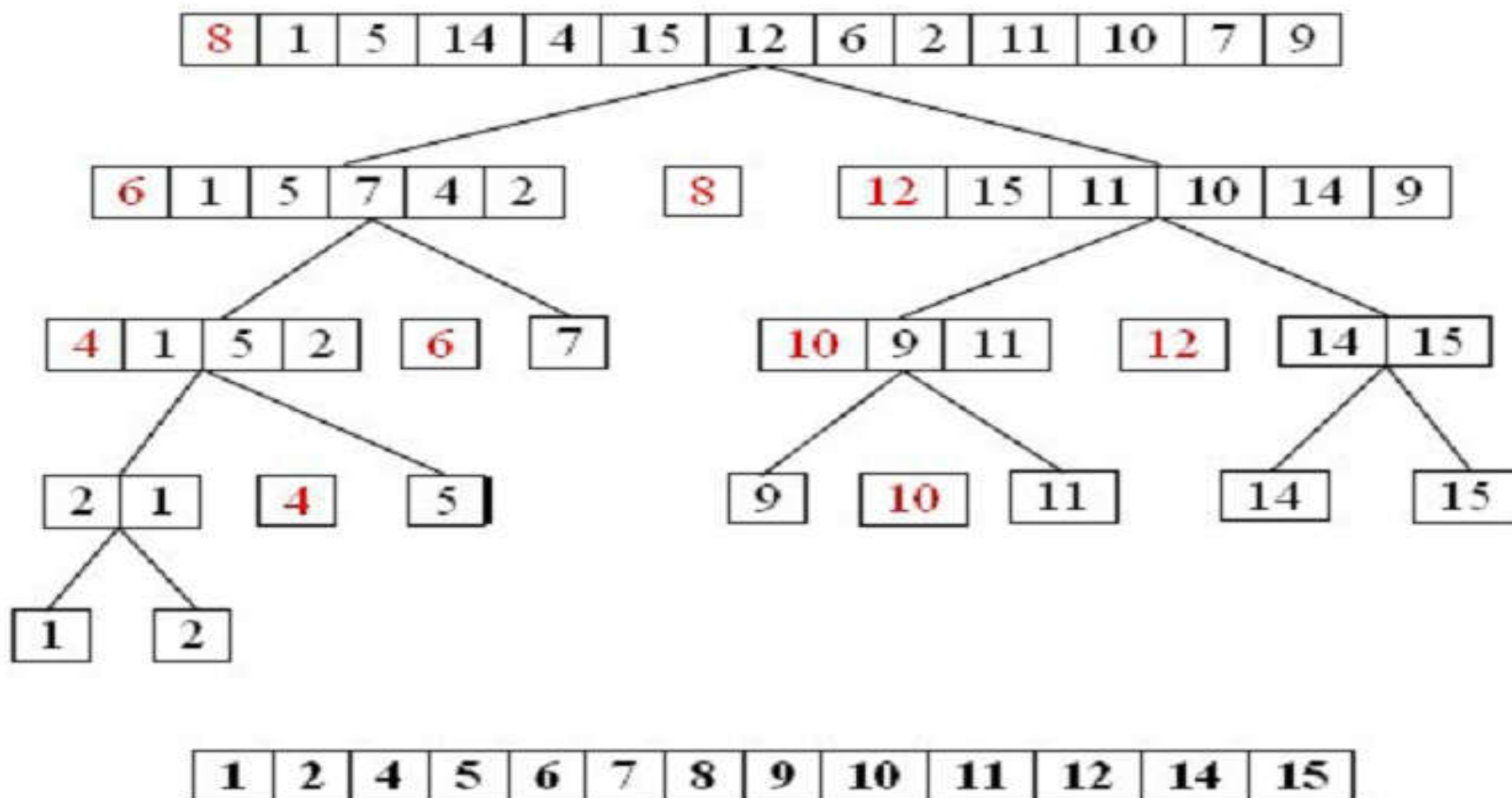
- **QuickSort**: Sắp xếp mảng n số theo thứ tự tăng dần
- Áp dụng kỹ thuật chia để trị:
 - Chia mảng n số thành 2 mảng con:
Trước khi chia phải phân hoạch
 - Giải 2 bài toán con:
Sắp xếp mảng “bên trái”
Sắp xếp mảng “bên phải” .
 - Bài toán cơ sở: Sắp xếp mảng 1 số hoặc nhiều số có giá trị giống nhau.
 - Tổng hợp kết quả : Không cần tổng hợp, đã bao hàm trong giai đoạn phân chia.



Ví dụ: QuickSort

- **QuickSort**: Sắp xếp mảng n số theo thứ tự tăng dần
- Áp dụng kỹ thuật chia để trị:
 - Chia mảng n số thành 2 mảng con: **Trước khi chia phải phân hoạch**
 - Giải 2 bài toán con:
 - Sắp xếp mảng “bên trái”
 - Sắp xếp mảng “bên phải” .
 - Bài toán cơ sở: Sắp xếp mảng 1 số / nhiều số có giá trị giống nhau.
 - Tổng hợp kết quả : Không cần tổng hợp, đã bao hàm trong giai đoạn phân chia.

Ví dụ: QuickSort





Độ phức tạp của QuickSort

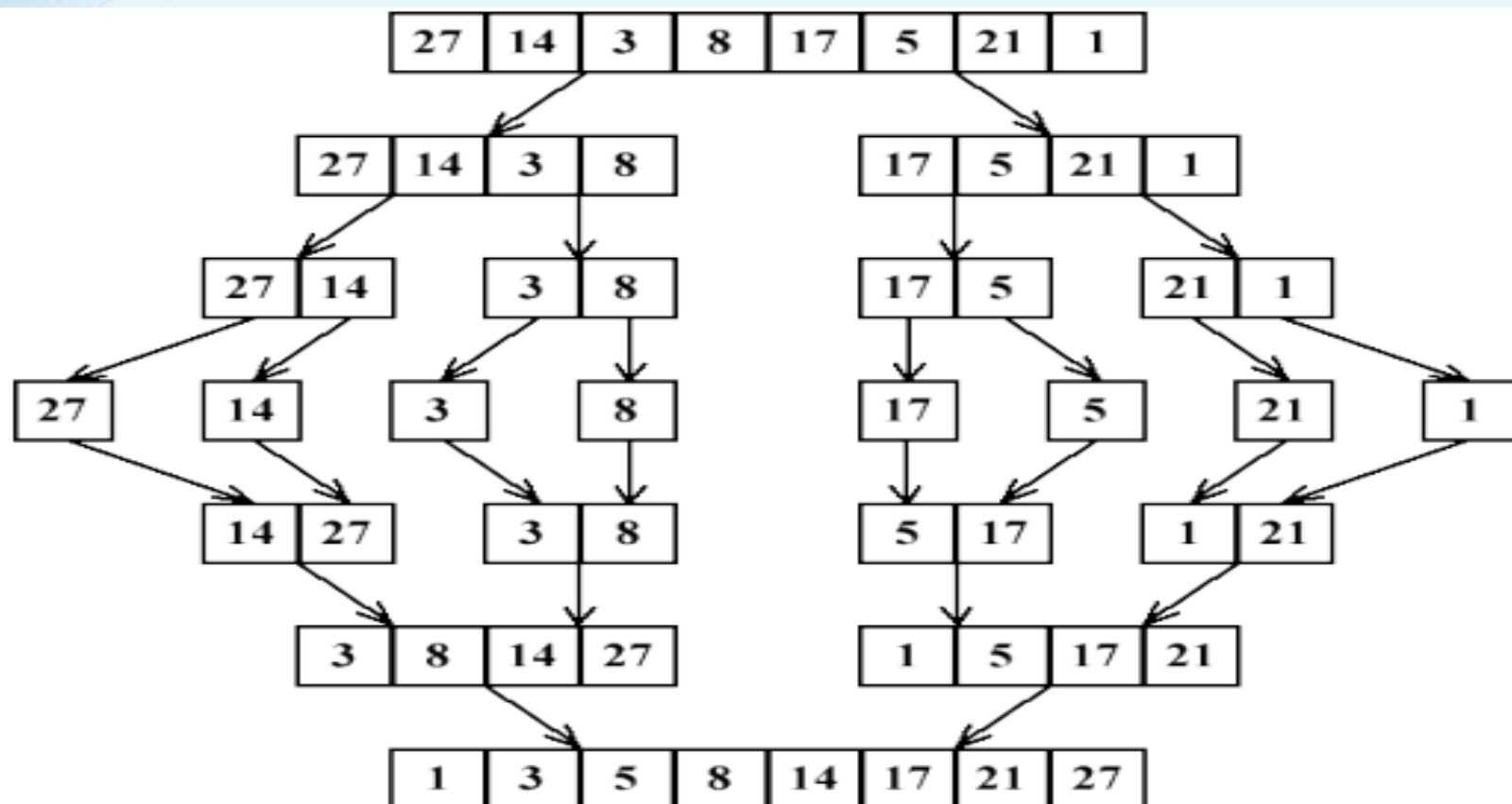
- *Trường hợp xấu nhất:* Mảng n số đúng thứ tự tăng dần
 - Phân hoạch **lệch**: phần tử chốt có giá trị lớn nhất (cần n phép so sánh để biết nó là phần tử cuối cùng)
 - Độ phức tạp : $O(n^2)$
- *Trường hợp tốt nhất:*
 - Phân hoạch **đều**: phần tử chốt là phần tử giữa mảng
 - Độ phức tạp : $O(n \log n)$
- *Trường hợp trung bình :*
 - Độ phức tạp : **$O(n \log n)$**



Ví dụ: MergeSort

- **MergeSort**: Sắp xếp mảng n số theo thứ tự tăng dần
- Áp dụng kỹ thuật chia để trị:
 - Chia mảng n số thành 2 mảng con:
Không cần phân hoạch, cứ cắt mảng ra làm hai ($n/2$)
 - Giải 2 bài toán con:
Sắp xếp mảng “bên trái”
Sắp xếp mảng “bên phải”
 - Bài toán cơ sở: Sắp xếp mảng 1 số.
 - Tổng hợp kết quả: Trộn kết quả (theo thứ tự) của 2 bài toán con.

Ví dụ: MergeSort





Độ phức tạp của MergeSort

- *Sắp xếp mảng n số :*
 - Số lần so sánh: $C(n) = 2 C(n/2) + n$
 - Độ phức tạp : **$O(n \log n)$**
 - Cần thêm n đơn vị bộ nhớ cho lưu trữ



Bài toán nhân 2 số nguyên lớn

- Kiểu dữ liệu **số nguyên** trong các NNLT (*integer* trong Pascal, *int* trong C...) có miền giá trị hạn chế
→ Người lập trình phải tìm một **CTDL thích hợp** để biểu diễn cho *số nguyên lớn*.
(Chẳng hạn, *số nguyên lớn* = chuỗi ký tự, mỗi ký tự lưu trữ một chữ số)
- Để thao tác được → phải xây dựng các phép toán cho số nguyên như *phép cộng*, *phép trừ*, *phép nhân*, ...



Thuật toán nhân 2 số nguyên lớn

- **Bài toán:** Nhân 2 số nguyên lớn X và Y , mỗi số có n chữ số.
- **Phương pháp nhân thông thường:** $X = 1426 \times Y = 3219$ với $n = 4$

$$\begin{array}{r} 1426 \\ \times 3219 \\ \hline 12834 \\ + 1426 \\ 2852 \\ 4278 \\ \hline 4590294 \end{array}$$

- Việc nhân từng chữ số của X và Y tốn n^2 phép nhân.
- Nếu phép nhân 2 chữ số tốn $O(1) \rightarrow$ Độ phức tạp là $O(n^2)$.



Thuật toán Chia để trị cho bài toán nhân 2 số nguyên lớn

- Áp dụng kỹ thuật Chia để trị:

- Để đơn giản cho việc phân tích thuật toán: giả sử n là lũy thừa của 2.
- Về phương diện lập trình: Thuật toán đúng trong mọi trường hợp n bất kỳ.
- Biểu diễn $X = A10^{n/2} + B$ và $Y = C10^{n/2} + D$
Trong đó A, B, C, D là các số có $n/2$ chữ số.
- $X = 1234$ thì $A = 12$ và $B = 34$ vì $12 \cdot 10^2 + 34 = 1234 = X$
 $\Rightarrow XY = AC10^n + (AD + BC)10^{n/2} + BD$



Thuật toán Chia để trị cho bài toán nhân 2 số nguyên lớn

Thuật toán:

(1) Phân tích bài toán ban đầu thành một số bài toán nhân 2 số có $n/2$ chữ số.

(2) Kết hợp các kết quả trung gian theo công thức

$$XY = AC10^n + (AD + BC)10^{n/2} + BD$$

Việc phân chia này sẽ dẫn đến các bài toán nhân 2 số có 1 chữ số: Đây là *bài toán cơ sở*.



Đánh giá thuật toán

- Theo công thức $XY = AC10^n + (AD + BC)10^{n/2} + BD$: Ta thực hiện **4 phép nhân các số nguyên có $n/2$ chữ số**, 3 phép cộng các số lớn hơn n chữ số và 2 phép nhân với 10^n và $10^{n/2}$.
 - Phép cộng các số lớn hơn n chữ số cần $O(n)$.
 - Phép nhân với 10^n tốn $O(n)$ (*dịch trái n lần*).
- Gọi $T(n)$ là thời gian nhân 2 số có n chữ số, ta có phương trình đệ quy sau: $T(1) = 1$

$$T(n) = 4T(n/2) + n$$

$\Rightarrow T(n) = O(n^2)$: không cải tiến gì so với nhân thông thường.



Cải tiến thuật toán

- Biến đổi công thức $XY = AC10^n + (\text{AD} + \text{BC})10^{n/2} + \text{BD}$ (1)
thành $XY = AC10^n + [(\text{A-B})(\text{D-C}) + \text{AC} + \text{BD}]10^{n/2} + \text{BD}$ (2)
- Theo công thức này, ta chỉ cần **3 phép nhân các số nguyên có $n/2$ chữ số**, 6 phép cộng trừ và 2 phép nhân với $10^n, 10^{n/2}$.

Ta có phương trình đệ quy nhsau:

$$T(1) = 1$$

$$T(n) = 3T(n/2) + n$$

$$\Rightarrow T(n) = \mathbf{O(n^{\log 3})} = \mathbf{O(n^{1.59})} : \text{Cải tiến hơn rất nhiều}$$



Cải tiến thuật toán

- Biến đổi công thức $XY = AC10^n + (AD + BC)10^{n/2} + BD$ (1)
thành $XY = AC10^n + [(A - B)(D - C) + AC + BD]10^{n/2} + BD$ (2)
vì $(A - B)(D - C) + AC + BD = AD - AC - BD + BC + AC + BD$
- Theo công thức này, ta chỉ cần **3 phép nhân các số nguyên có $n/2$ chữ số**, 6 phép cộng trừ và 2 phép nhân với $10^n, 10^{n/2}$.

Ta có phương trình đệ quy nhsau:

$$T(1) = 1$$

$$T(n) = 3T(n/2) + n$$

$\Rightarrow T(n) = O(n^{\log 3}) = O(n^{1.59})$: Cải tiến hơn rất nhiều



Độ phức tạp Bài toán nhân 2 số nguyên lớn

- Thuật toán **Nhân thông thường** : $T(n) = O(n^2)$
- Thuật toán **Chia để trị** dùng Công thức (1)

Công thức (1) : $XY = AC10^n + (AD + BC)10^{n/2} + BD$

$$T(n) = O(n^2)$$

- Thuật toán **Chia để trị cải tiến** dùng Công thức (2)

Công thức (2): $XY = AC10^n + [(A-B)(D-C) + AC + BD]10^{n/2} + BD$

$$T(n) = O(n^{\log 3}) = O(n^{1.59})$$



Thuật toán nhân 2 số nguyên lớn

```
Big_Integer mult (Big_Integer X, Big_Integer Y, int n) {  
    Big_Integer m1, m2, m3, A, B, C, D;  
    int s; /* lưu dấu của tích XY */  
    s = sign(X)*sign(Y); /* sign(X) trả về 1 nếu X dương; -1 nếu X âm; 0 nếu X = 0 */  
    X = ABS(X);  
    Y = ABS(Y);  
    if (n == 1) return X*Y*s;  
    A = left(X, n/2);  
    B = right(X, n/2);  
    C = left(Y, n/2);  
    D = right(Y, n/2);  
    m1 = mult(A, C, n/2);  
    m2 = mult(A - B, D - C, n/2);  
    m3 = mult(B, D, n/2);  
    return s*(m1*10n + (m1 + m2 + m3)*10n/2 + m3);  
}
```



Bài toán xếp lịch thi đấu thể thao

- **Bài toán** : Xếp lịch thi đấu vòng tròn một lượt cho n đấu thủ. Mỗi đấu thủ đấu với $n-1$ đấu thủ còn lại, đấu 1 trận mỗi ngày.
- **Yêu cầu** : Xếp lịch sao cho *số ngày thi đấu là ít nhất*.
- **Phân tích**:
 - Tổng số trận đấu cần xếp là $n(n-1)/2$.
 - Nếu n *chẵn*: cần xếp $n/2$ cặp đấu mỗi ngày \rightarrow số ngày là $n-1$.
 - Nếu n *lẻ*, thì $n-1$ chẵn: xếp $(n-1)/2$ trận mỗi ngày \rightarrow cần n ngày
 - Giả sử $n = 2^k$ thì n chẵn, do đó cần ít nhất $n-1$ ngày.



Thuật toán Chia để trị cho bài toán xếp lịch thi đấu

Áp dụng kỹ thuật Chia để trị:

- Xem lịch thi đấu là 1 bảng gồm n *dòng* (tương ứng với n đấu thủ) và $n-1$ *cột* (tương ứng với $n-1$ ngày): $\hat{o}(i,j)$ biểu diễn đấu thủ i phải đấu trong ngày j .
- **Chia để trị:** thay vì xếp cho n người, ta xếp cho $n/2$ người. Quá trình phân chia sẽ dừng lại khi xếp lịch thi đấu cho 2 đấu thủ = *Bài toán cơ sở*.
- *Bài toán cơ sở* : xếp 2 đấu thủ thi đấu 1 trận trong 1 ngày.

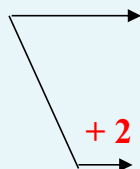


CANTHO UNIVERSITY

Ví dụ Xếp lịch thi đầu

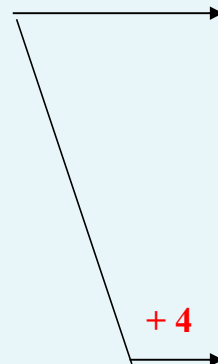
2 đầu thủ

	1
1	2
2	1



4 đầu thủ

	1	2	3
1	2	3	4
2	1	4	3
3	4	1	2
4	3	2	1



8 đầu thủ

	1	2	3	4	5	6	7
1	2	3	4	5	6	7	8
2	1	4	3	6	5	8	7
3	4	1	2	7	8	5	6
4	3	2	1	8	7	6	5
5	6	7	8	1	2	3	4
6	5	8	7	2	1	4	3
7	8	5	6	3	4	1	2
8	7	6	5	4	3	2	1

Bài toán
cơ sở



Hiệu quả của Chia để trị

⇒ Bài toán con cân bằng

- **Ví dụ:** MergeSort chia bài toán thành 2 bài toán con có cùng kích thước $n/2$ và do đó thời gian chỉ là **$O(n \log n)$** . Ngược lại trong trường hợp xấu nhất của QuickSort, khi mảng bị *phân hoạch lệch* thì thời gian thực hiện là **$O(n^2)$** .
- **Nguyên tắc chung:** Chia bài toán thành các bài toán con có *kích thước xấp xỉ bằng nhau* thì hiệu suất sẽ cao hơn.



CANTHO UNIVERSITY

CHƯƠNG 3

KỸ THUẬT THIẾT KẾ THUẬT TOÁN

KỸ THUẬT THAM ĂN



Kỹ thuật “tham ăn” (Greedy)

- Thường dùng giải các bài toán tối ưu tổ hợp.
- **Mục đích:** Tìm một lời giải *tốt* trong thời gian *chấp nhận được* (độ phức tạp đa thức thay vì lũy thừa)
- Áp dụng kỹ thuật này tuy không cho lời giải tối ưu nhưng sẽ cho một lời giải “tốt” (được lợi khá nhiều về mặt thời gian).



Bài toán tối ưu tổ hợp

- **Bài toán**: Cho hàm $f(X)$ xác định trên một tập hữu hạn các phần tử D .
 - **Hàm mục tiêu** : $f(X)$
 - **Phương án** : $X = (x_1, x_2, \dots, x_n), \forall X \in D$
- **Yêu cầu** : Cần tìm *phương án tối ưu* $X^* \in D$ sao cho
$$f(X^*) \rightarrow \text{MIN/MAX}$$

\Rightarrow Dùng phương pháp “**vét cạn**” để tìm phương án tối ưu: cần một *thời gian mũ*.



Minh họa Kỹ thuật Tham ăn

- **Kỹ thuật Tham ăn**
(Greedy)

(1) Bài toán trả tiền của máy ATM: $f(X) \rightarrow \text{MIN}$

(2) Bài toán người giao hàng : $f(X) \rightarrow \text{MIN}$

(3) Bài toán Cái ba lô
 $f(X) \rightarrow \text{MAX}$



(1) Bài toán trả tiền của máy rút tiền tự động ATM

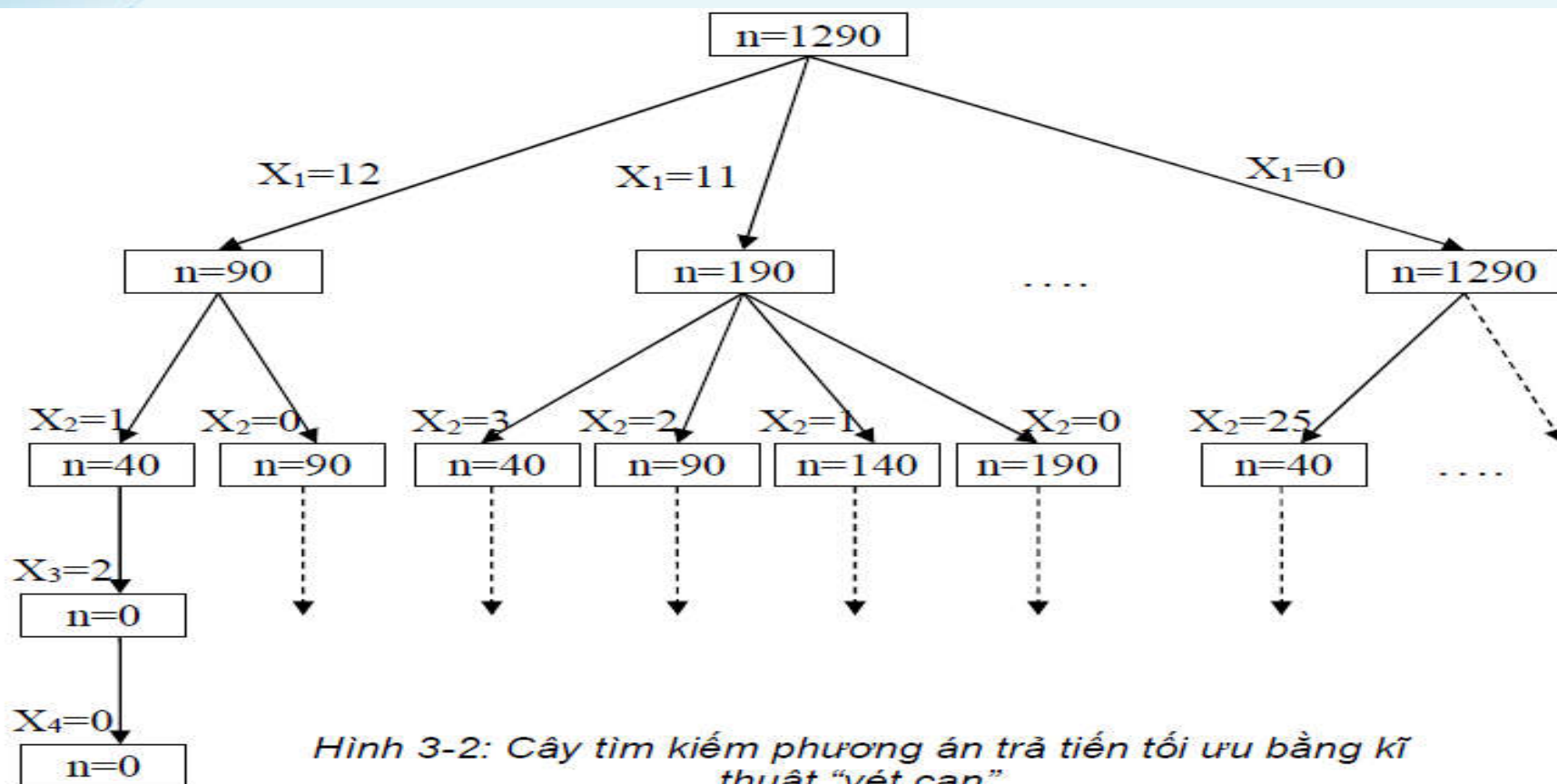
- **Bài toán**: Trong máy ATM, có sẵn các loại tiền mệnh giá **100.000** đồng, **50.000** đồng, **20.000** đồng và **10.000** đồng.
 - Giả sử mỗi loại tiền có số lượng không hạn chế.
 - Khi khách hàng cần rút một số tiền **n** đồng (tính chẵn đến 10.000 đồng, tức là **n** chia hết cho 10.000).
- **Yêu cầu**: Hãy tìm một phương án trả tiền sao cho trả đủ **n** đồng và *số tờ giấy bạc phải trả là ít nhất*.
- **Ví dụ**: $n = 1.290.000$ đồng



(1) Bài toán trả tiền của máy rút tiền tự động ATM

- **Phương án** : Gọi $\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3, \mathbf{X}_4)$ là một *phương án* trả tiền.
Trong đó : \mathbf{X}_1 là số tờ giấy bạc 100.000 đồng,
 \mathbf{X}_2 là số tờ giấy bạc 50.000 đồng,
 \mathbf{X}_3 là số tờ giấy bạc 20.000 đồng
 \mathbf{X}_4 là số tờ giấy bạc 10.000 đồng.
- **Hàm mục tiêu** : $\mathbf{f}(\mathbf{X}) = \mathbf{X}_1 + \mathbf{X}_2 + \mathbf{X}_3 + \mathbf{X}_4 \rightarrow \text{MIN}$
và $\mathbf{X}_1 * 100.000 + \mathbf{X}_2 * 50.000 + \mathbf{X}_3 * 20.000 + \mathbf{X}_4 * 10.000 = \mathbf{n}$

Kỹ thuật vét cạn





Kỹ thuật Tham ăn

- **Ý tưởng**: Để $(X_1 + X_2 + X_3 + X_4)$ nhỏ nhất thì các **tờ giấy bạc mệnh giá lớn phải được chọn nhiều nhất**.
 - Trước hết ta chọn tối đa các tờ giấy bạc 100.000 đồng:
 X_1 là số nguyên lớn nhất sao cho $X_1 * 100.000 \leq n$
hay $X_1 = n \text{ DIV } 100.000$.
 - Xác định số tiền cần rút còn lại là hiệu: $n - X_1 * 100.000$
 - Chuyển sang chọn loại giấy bạc 50.000 đồng, và cứ tiếp tục như thế cho các loại mệnh giá khác.

$$\text{Ví dụ : } 160.000 = 1 * 100.000 + 1 * 50.000 + 0 * 20.000 + 1 * 10.000$$

$$240.000 = 2 * 100.000 + 0 * 50.000 + 2 * 20.000 + 0 * 10.000$$



Bài toán trả tiền của máy rút tiền tự động ATM: Ví dụ

Ví dụ : $n = 1290000$

Phương án trả tiền như sau:

- $X_1 = 1290000 / 100000 = 12$

Số tiền còn lại là :

$$1290000 - 12 * 100000 = 90000$$

- $X_2 = 90000 / 50000 = 1$

Số tiền còn lại là :

$$90000 - 1 * 50000 = 40000$$

- $X_3 = 40000 / 20000 = 2$

Số tiền còn lại là :

$$40000 - 2 * 20000 = 0$$

- $X_4 = 0 / 10000 = 0$

Ta có phương án :

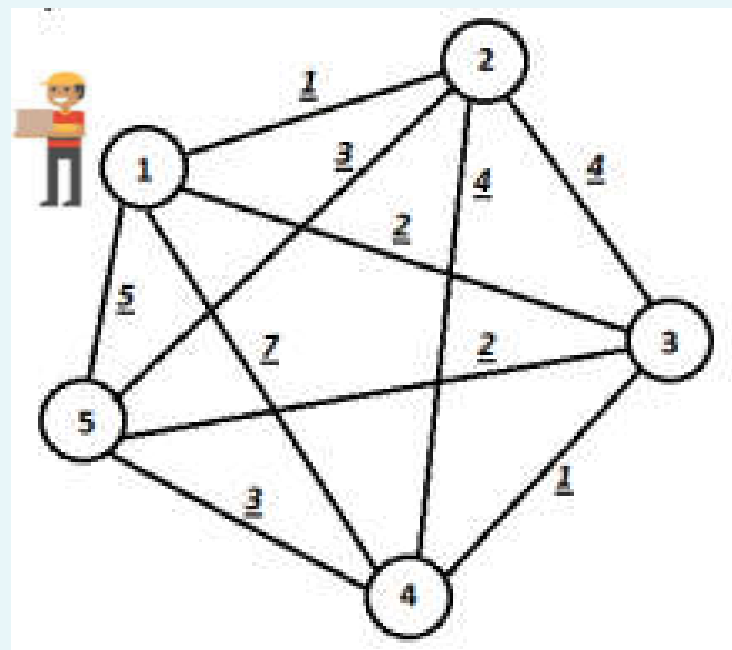
$$\mathbf{X} = (12, 1, 2, 0)$$

Với phương án này thì giá trị hàm mục tiêu là 15 (tổng số tờ = 15 tờ)



(2) Bài toán đường đi của người giao hàng (TSP - Travelling Salesman Problem) : Mô tả

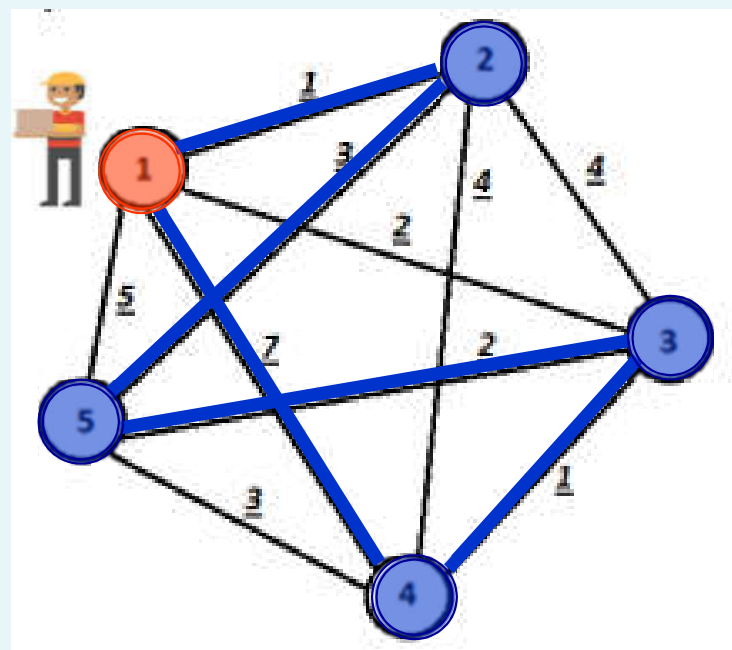
- **Bài toán**: Một người giao hàng cần đi giao hàng tại **n** thành phố. Xuất phát từ một thành phố, đi qua các thành phố khác và trở về thành phố ban đầu.
Mỗi thành phố chỉ đến **1 lần**.
 - Giả thiết mỗi thành phố đều có đường đi đến các thành phố còn lại.





(2) Bài toán đường đi của người giao hàng (TSP - Travelling Salesman Problem) : Mô tả

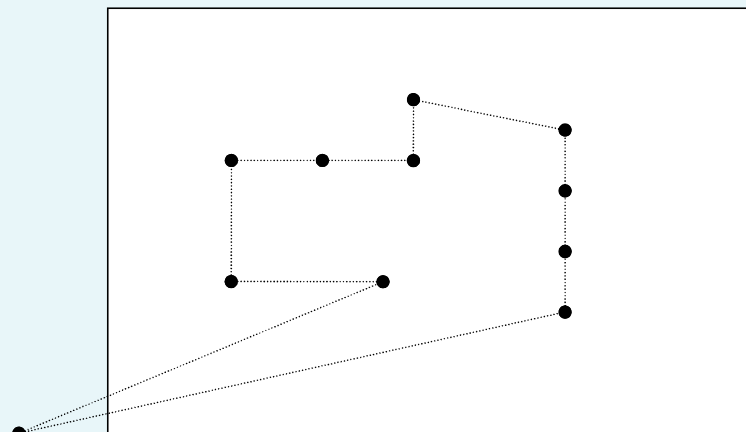
- Khoảng cách giữa các thành phố là được xác định, có thể là *khoảng cách địa lý, cước phí di chuyển* hoặc *thời gian di chuyển* (gọi chung là **độ dài**)
- **Yêu cầu**: Tìm một chu trình sao cho tổng độ dài các cạnh là nhỏ nhất hay một **phương án có giá nhỏ nhất** ?





CANTHO UNIVERSITY

TSP: Một ứng dụng



Vị trí hàn

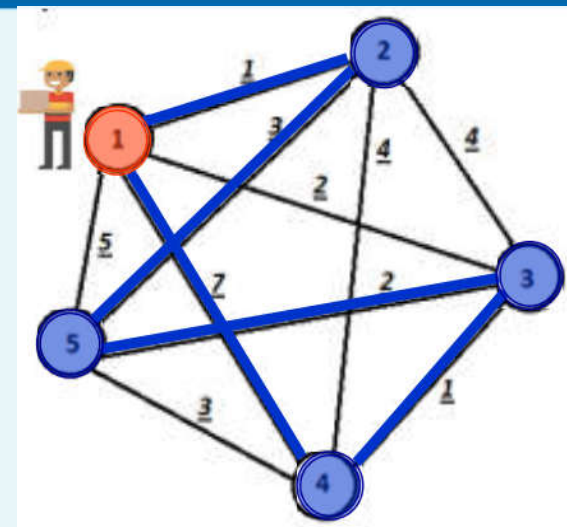
Tấm kim loại

Bài toán hàn các điểm trên một tấm kim loại



(2) Bài toán đường đi của người giao hàng (TSP): Biểu diễn

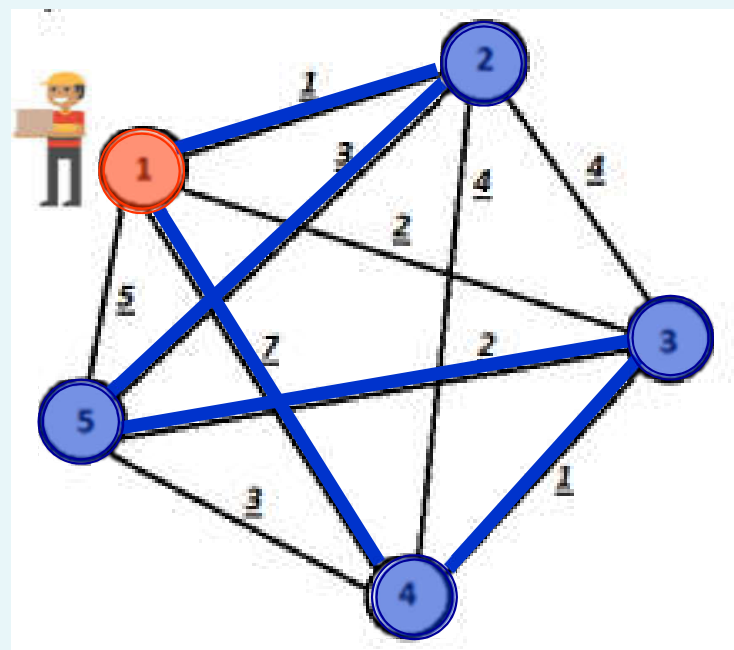
- **Biểu diễn TSP** : Đồ thị vô hướng $G=(V,E)$
 - Tập đỉnh (V) : thành phố
 - Tập cạnh (E) : đường nối các thành phố
 - Trong số cạnh : Khoảng cách giữa 2 thành phố
- **Chu trình Hamilton** = Chu trình đi qua tất cả các đỉnh của G , mỗi đỉnh một lần duy nhất.
- **Vấn đề** : Tìm chu trình Hamilton tối tiểu (tổng độ dài các cạnh là nhỏ nhất) ?





TSP: Phương pháp vét cạn

- **Ý tưởng:** Xét tất cả các chu trình, tính tổng độ dài các cạnh của nó, rồi chọn chu trình nhỏ nhất.
- Với n đỉnh (thành phố): cần xét $((n-1)(n-2)...1)/2 = (n-1)!/2$ chu trình
VD: $n = 5 \rightarrow$ xét $(5-1)!/2 = 12$ chu trình
 \Rightarrow Thuật toán có độ phức tạp thời gian là *hàm mũ* $T(n) = O(n!)$





TSP: Kỹ thuật tham ăn

- Phương pháp Tham ăn (Greedy) đưa ra quyết định dựa ngay vào thông tin đang có, và **trong tương lai sẽ không xem xét lại tác động của các quyết định trong quá khứ.**
- Chính vì thế các thuật toán dạng này *rất dễ đề xuất*, và thông thường chúng *không đòi hỏi nhiều thời gian tính.*
- Tuy nhiên, các thuật toán dạng này **thường không cho kết quả tối ưu.**



TSP: Kỹ thuật tham ăn

Áp dụng kỹ thuật Tham ăn:

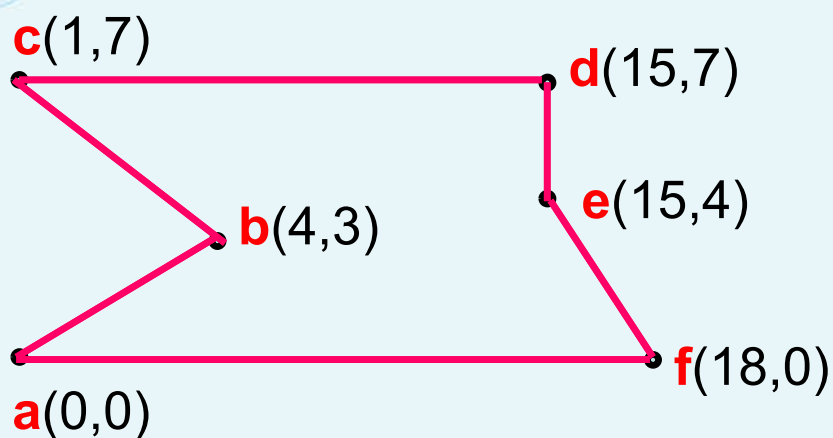
- (1) Sắp xếp các cạnh theo **thứ tự tăng** của *độ dài*.
- (2) Xét các cạnh có *độ dài* từ nhỏ đến lớn để đưa vào chu trình.
- (3) Một cạnh sẽ được đưa vào chu trình nếu:
 - **Không tạo thành một chu trình thiếu**
 - **Không tạo thành một đỉnh có cấp ≥ 3**
- (4) Lặp lại bước (3) cho đến khi xây dựng được một chu trình.

Với kỹ thuật này, chỉ cần $n(n-1)/2$ phép chọn nên ta có một thuật toán cần $T(n) = O(n^2)$ thời gian.



CANTHO UNIVERSITY

TSP: Ví dụ



Ví dụ: Bài toán TSP với $n = 6$ đỉnh

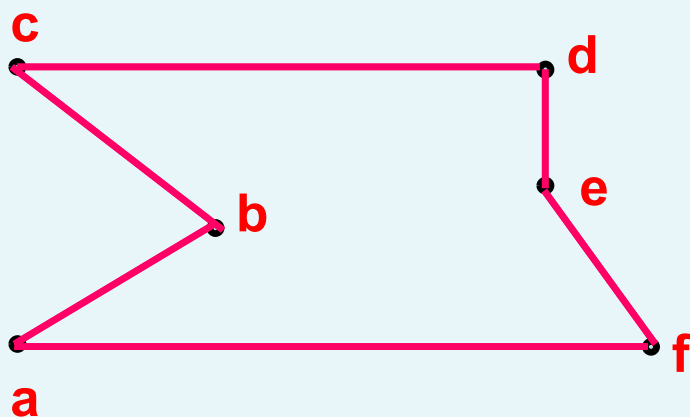
Độ dài cạnh = khoảng cách hình học giữa 2 điểm $A(X_1, Y_1)$ và $B(X_2, Y_2)$ khoảng cách Euclide: $\sqrt{(X_1 - X_2)^2 + (Y_1 - Y_2)^2}$

TT	Cạnh	X_1	Y_1	X_2	Y_2	Độ dài
1	de	15	7	15	4	3.00
2	ab	0	0	4	3	5.00
3	bc	4	3	1	7	5.00
4	ef	15	4	18	0	5.00
5	ac	0	0	1	7	7.07
6	df	15	7	18	0	7.62
7	be	4	3	15	4	11.05
8	bd	4	3	15	7	11.70
9	cd	1	7	15	7	14.00
10	bf	4	3	18	0	14.32
11	ce	1	7	15	4	14.32
12	ae	0	0	15	4	15.52
13	ad	0	0	15	7	16.55
14	af	0	0	18	0	18.06
15	cf	1	7	18	0	18.38

www.ctu.edu.vn



TSP: Kết quả

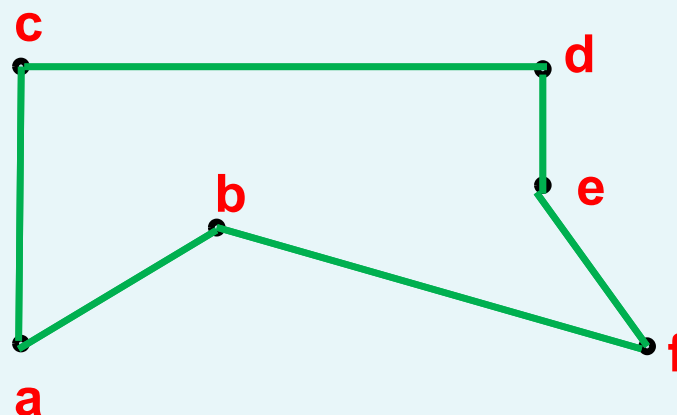


Phương án tốt (Greedy)

$$a-b-c-d-e-f-a = 50$$

- Số phép chọn: $n(n-1)/2$

$$\Rightarrow T(n) = O(n^2)$$



Phương án tối ưu (Vết cặn)

$$a-c-d-e-f-b-a = 48.39$$

- Số chu trình: $(n-1)!/2$

$$\Rightarrow T(n) = O(n!)$$



TSP: Thuật toán tham ăn

void TSP() {

*/*E là tập hợp các cạnh, Chu_trình là tập hợp các cạnh được chọn để đưa vào chu trình, khởi đầu Chu_trình rỗng*/*

*/*Sắp xếp các cạnh trong E theo thứ tự tăng của độ dài*/*

Chu_Trình = Φ ;

Gia = 0.0;

while (E $\neq \Phi$) {

if (cạnh e có thể chọn) {

Chu_Trình = Chu_Trình + [e];

Gia = Gia + độ dài của e; }

E := E-[e];

}



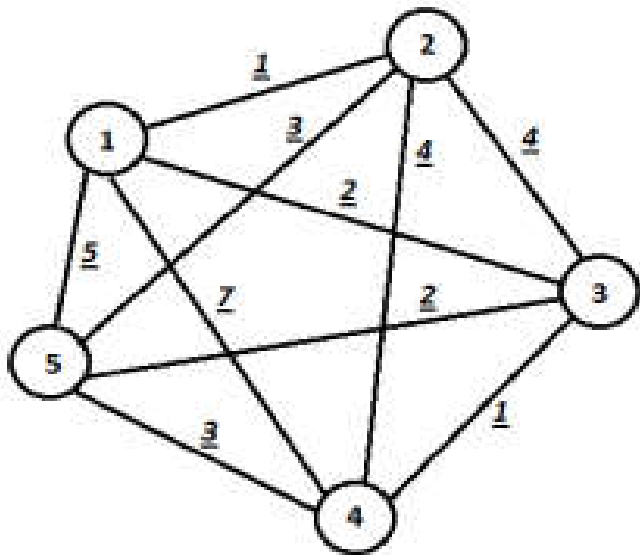
TSP: Một cách tiếp cận khác

- (1). Xuất phát từ 1 đỉnh, chọn cạnh có **độ dài nhỏ nhất** trong tất cả các cạnh đi ra từ đỉnh đó để đến đỉnh kế tiếp.
- (2). Từ đỉnh kế tiếp, chọn cạnh có **độ dài nhỏ nhất** đi ra từ đỉnh này thoả mãn 2 điều kiện nói trên để đi đến đỉnh kế tiếp.
- (3). Lặp lại bước (2) cho đến khi đi tới đỉnh n thì quay trở về đỉnh xuất phát.



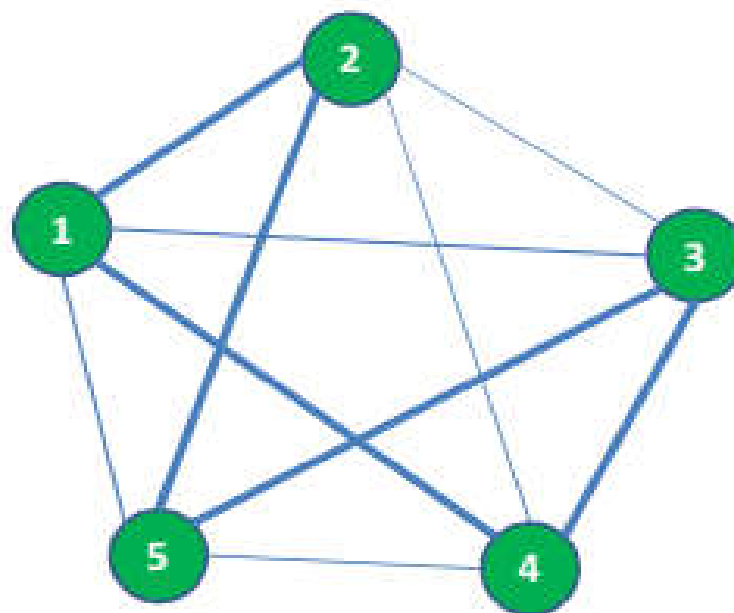
TSP: Một cách tiếp cận khác

- Trở về đỉnh đầu



TOUR={ $(1,2), (2,5), (5,3), (3,4), (4,1)$ }

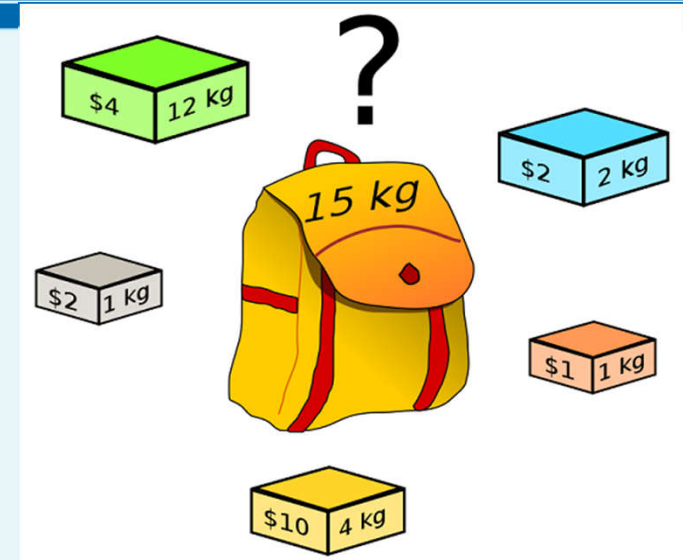
COST=1+3+2+1=7+7





(3) Bài toán cái ba lô

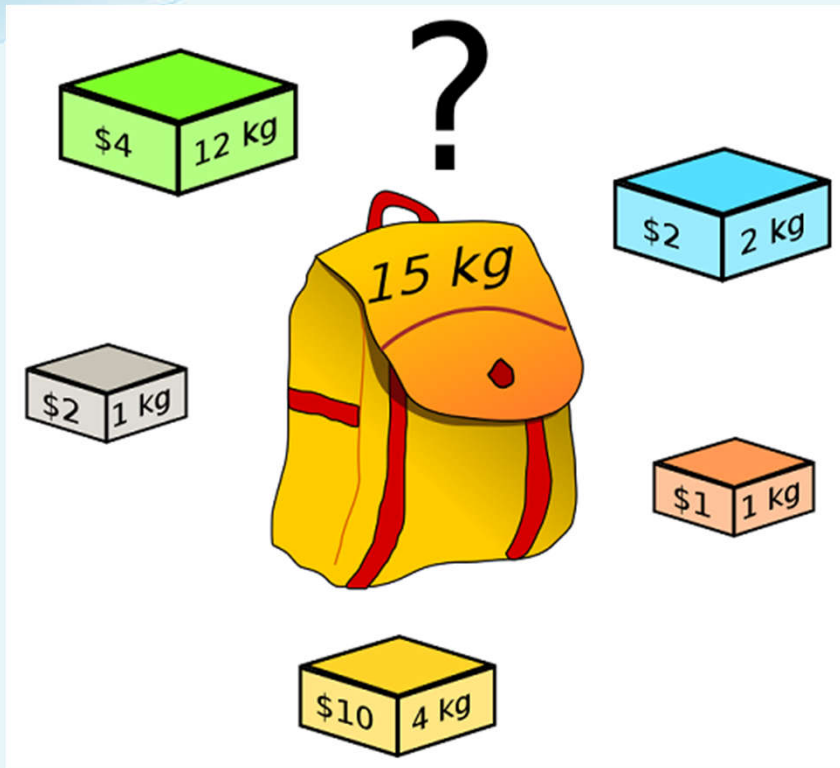
- **Bài toán:** Cho một cái ba lô có thể đựng trọng lượng W và n loại đồ vật, mỗi đồ vật i có một **trọng lượng g_i** và một **giá trị v_i** . Tất cả đồ vật đều có **số lượng không hạn chế** \rightarrow Bài toán CBL1



- **Yêu cầu:** Tìm cách chọn đồ vật đựng vào ba lô (đồ vật nào, số lượng bao nhiêu) sao cho tổng trọng lượng không vượt quá W và **tổng giá trị là lớn nhất?**



(3) Bài toán cái ba lô



Trọng lượng = W

CBL 1

Số đồ vật = n

Đồ vật i

TL g_i

GT v_i

SL $--$



(3) Bài toán cái ba lô: Mô hình hóa

- *Phương án*: Gọi $\mathbf{X} = (X_1, X_2, \dots, X_n)$ với X_i là số nguyên không âm (X_i là số lượng đồ vật thứ i)
- *Hàm mục tiêu*: Cần tìm phương án \mathbf{X} sao cho:
 - $X_1g_1 + X_2g_2 + \dots + X_ng_n \leq W$
 - $f(\mathbf{X}) = X_1v_1 + X_2v_2 + \dots + X_nv_n \rightarrow \text{MAX}$



(3) Bài toán cái ba lô: Ý tưởng

Ý tưởng: Đồ vật có **đơn giá** lớn nhất còn lại được lấy trước (nếu có thể).

– Sắp xếp các đồ vật theo thứ tự *không tăng* của **đơn giá** (giá trị một đơn vị trọng lượng của đồ vật)

$$\text{ĐG} = \text{GT} : \text{TL}$$

– Xét chọn đồ vật từ trên xuống



(3) Bài toán cái ba lô: Thuật toán tham ăn

Áp dụng kỹ thuật Tham ăn:

- (1). Tính **đơn giá** cho các loại đồ vật.
- (2). Xét các loại đồ vật theo **thứ tự đơn giá từ lớn đến nhỏ**.
- (3). Với mỗi đồ vật được xét sẽ lấy một **số lượng tối đa** mà trọng lượng còn lại của ba lô cho phép.
- (4). Xác định **trọng lượng còn lại** của ba lô và quay lại bước (3) cho đến khi không còn có thể chọn được đồ vật nào nữa.



Bài toán cái ba lô: Ví dụ

Ví dụ : Có một ba lô có trọng lượng là 37 và 4 loại đồ vật với trọng lượng và giá trị tương ứng được cho trong bảng bên dưới:

Loại đồ vật	Trọng lượng	Giá trị
A	15	30
B	10	25
C	2	2
D	4	6

CBL 1 $W = 37$
 $n = 4$

ĐG = GT : TL



Bài toán cái ba lô: Thuật toán Tham ăn

CBL 1 $W = 37$

ĐV	TL	GT	ĐG
B	10	25	2.5
A	15	30	2.0
D	4	6	1.5
C	2	2	1.0

$$\text{ĐG} = \text{GT} : \text{TL}$$

- Phương án $X = (X_A, X_B, X_C, X_D)$

- $X_B = 37/10 = 3$

$$W = 37 - 3 \cdot 10 = 7.$$

- $X_A = 7/15 = 0$

- $X_D = 7/4 = 1$

$$W = 7 - 4 \cdot 1 = 3.$$

- $X_C = 3/2 = 1.$

$$W = 3 - 2 = 1$$

→ Phương án là $X = (0, 3, 1, 1)$

- Tổng TL: $3 \cdot 10 + 1 \cdot 4 + 1 \cdot 2 = 36$

- Tổng GT: $3 \cdot 25 + 1 \cdot 6 + 1 \cdot 2 = 83.$



Bài toán cái ba lô: Tổ chức dữ liệu

- Mỗi đồ vật được biểu diễn bởi một mẫu tin có các trường:
 - **Ten**: Lưu trữ tên đồ vật.
 - **Trong_luong**: Lưu trữ trọng lượng của đồ vật.
 - **Gia_tri**: Lưu trữ giá trị của đồ vật
 - **Don_gia**: Lưu trữ đơn giá của đồ vật
 - **Phuong_an**: Lưu trữ số lượng đồ vật được chọn theo phương án.
- Danh sách các đồ vật được biểu diễn bởi một mảng các đồ vật.



Bài toán cái ba lô: Chương trình

```
#define MAX_SIZE 100
typedef struct {
    char Ten [20];
    float Trong_luong, Gia_tri, Don_gia;
    int Phuong_an;
} Do_vat;

typedef Do_vat Danh_sach_do_vat[MAX_SIZE];

void Greedy (Danh_sach_do_vat dsdv, float W) {
    int i;
    /*Sắp xếp mảng dsdv theo thứ tự giảm của don_gia*/
    for (i = 0; i < n; i++) {
        dsdv[i].Phuong_an = Chon(dsdv[i].Trong_luong, W);
        W = W - dsdv[i].phuong_an * dsdv[i].Trong_luong
    }
}
```



Biến thể của bài toán cái ba lô

- Có một số biến thể của bài toán cái ba lô như sau:
 - (1) *Mỗi đồ vật i chỉ có một số lượng s_i* : Với bài toán này, khi lựa chọn vật i ta không được lấy một số lượng vượt quá s_i → **Bài toán CBL 2**
 - (2) *Mỗi đồ vật chỉ có một cái*: Với bài toán này, với mỗi đồ vật ta chỉ có thể **chọn hoặc không chọn** → **Bài toán CBL 3**

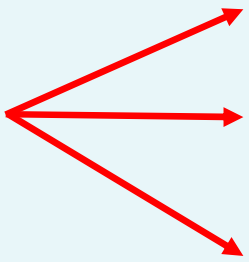


CANTHO UNIVERSITY

Biến thể của bài toán cái ba lô


Trọng lượng = W

Số đồ vật = n

Đồ vật i 

TL g_i

GT v_i

SL 

CBL 1

CBL 2

CBL 3

SL 

SL 



Bài tập 9

BT 9.1: Cho bài toán cái ba lô với trọng lượng của ba lô **$W = 30$** và **5 loại đồ vật** được cho trong bảng bên dưới. Tất cả các đồ vật đều *chỉ có 1 cái*. Giải bài toán bằng **kỹ thuật tham ăn** (Greedy) ?

Loại đồ vật	Trọng lượng	Giá trị
A	15	30
B	10	25
C	2	2
D	4	6
E	8	24



Bài tập 11

BT 11.1: Bài toán mua hàng với số tiền **$M = 40$** và **5 loại hàng** hóa được cho trong bảng bên dưới.

Giải bài toán bằng **kỹ thuật tham ăn** (Greedy) ?

Loại hàng	Giá tiền	Giá trị dinh dưỡng	Số lượng
A	10	25	2
B	15	30	1
C	4	6	1
D	2	2	2
E	8	24	1