Ôn tập môn Toán rời rạc

1 Mệnh đề và tập hợp

1.1 Lý thuyết

1.1.1 Bảng giá trị chân lý mệnh đề

p	q	$\bar{p}$	$p \wedge q$	$p \vee q$	$p \oplus q$	$p \rightarrow q$	$p \leftrightarrow q$
T	T	F	T	T	F	T	T
T	F	F	F	T	T	F	F
F	T	T	F	T	T	T	F
F	F	T	F	F	F	T	T

Hình 1: Bảng giá trị chân lý của các phép toán mệnh đề

1.1.2 Bảng mệnh đề tương đương

1.1.3 Dạng chuẩn tắc của mệnh đề (chuẩn hội)

Biến đổi công thức về dạng chuẩn hội là một cách hiểu quả để chứng minh tính tương đương của các mệnh đề và không sử dụng bảng chân trị. Các bước để biến đổi một công thức về dạng chuẩn hội:

1. Bỏ các phép kéo theo ($\rightarrow$) bằng cách thay $(p \rightarrow q) \Leftrightarrow \neg p \vee q$
2. Áp dụng luật De Morgan để chuyển các phép phủ định vào sát các mệnh đề; thay $\neg\bar{p} \Leftrightarrow p$
3. Áp dụng luật phân phối thay các công thức có dạng: $(p \vee (q \wedge r)) \Leftrightarrow (p \vee q) \wedge (p \vee r)$

$p \wedge \mathtt{T} \Leftrightarrow p$	**(Luật đồng nhất)**	$p \vee q \Leftrightarrow q \vee p$	**(Luật giao hoán)**
$p \vee \mathtt{F} \Leftrightarrow p$	**Identity laws**	$p \wedge q \Leftrightarrow q \wedge p$	**Commutative laws**
$p \vee \mathtt{T} \Leftrightarrow \mathtt{T}$	**(Luật nuốt)**	$(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$	**(Luật kết hợp)**
$p \wedge \mathtt{F} \Leftrightarrow \mathtt{F}$	**Domination laws**	$(p \wedge q) \wedge r \Leftrightarrow p \wedge (q \wedge r)$	**Associative laws**
$p \vee p \Leftrightarrow p$	**(Luật lũy đẳng)**	$p \vee (q \wedge r) \Leftrightarrow (p \vee q) \wedge (p \vee r)$	**(Luật phân phối)**
$p \wedge p \Leftrightarrow p$	**Idempotent laws**	$p \wedge (q \vee r) \Leftrightarrow (p \wedge q) \vee (p \wedge r)$	**Distribute laws**
$\neg(\neg p) \Leftrightarrow p$	**(Luật phủ định kép)**	$\neg(p \wedge q) \Leftrightarrow \neg p \vee \neg q$	**(Luật De Morgan)**
	Double negation law	$\neg(p \vee q) \Leftrightarrow \neg p \wedge \neg q$	**De Morgan's laws**

Hình 2: Bảng mệnh đề tương đương (1)

1.1.4 Vị từ và lượng từ

Vị từ Tổng quát, giả sử **M** là một tập hợp các phần tử nào đó. **M** thường được gọi là trường hay miền xác định của các phẩn tử thuộc **M**. Khi đó, biểu thức **P(x)** gọi là vị từ xác định trên trường **M** nếu khi thay x bởi một phần tử bất kỳ của trường **M** thì **P(x)** sẽ trở thành một mệnh đề trên trường **M**.

Lượng từ

- Lượng từ với mọi của **P(x)**, ký hiệu $\forall x P(x)$: *"P(x) đúng với mọi phần tử x thuộc trường đang xét"*
- Lượng từ tồn tại của **P(x)**, ký hiệu $\exists x P(x)$: *"Tồn tại ít nhất 1 phần tử x trong trường đang xét sao cho P(x) đúng"*.

1.1.5 Tổ hợp

Các phép toán trên tập hợp

- Phép hợp: $A \cup B = \{x | x \in A \vee x \in B\}$
- Phép giao: $A \cap B = \{x | x \in A \wedge x \in B\}$
- Phép hiệu: $A - B = \{x | x \in A \wedge x \notin B\}$
- Phép bù: $\bar{A} = \{x | x \notin A\}$

TABLE 7 Logical Equivalences Involving Conditional Statements.

$p \to q \equiv \neg p \vee q$
$p \to q \equiv \neg q \to \neg p$
$p \vee q \equiv \neg p \to q$
$p \wedge q \equiv \neg(p \to \neg q)$
$\neg(p \to q) \equiv p \wedge \neg q$
$(p \to q) \wedge (p \to r) \equiv p \to (q \wedge r)$
$(p \to r) \wedge (q \to r) \equiv (p \vee q) \to r$
$(p \to q) \vee (p \to r) \equiv p \to (q \vee r)$
$(p \to r) \vee (q \to r) \equiv (p \wedge q) \to r$

TABLE 8 Logical Equivalences Involving Biconditional Statements.

$p \leftrightarrow q \equiv (p \to q) \wedge (q \to p)$
$p \leftrightarrow q \equiv \neg p \leftrightarrow \neg q$
$p \leftrightarrow q \equiv (p \wedge q) \vee (\neg p \wedge \neg q)$
$\neg(p \leftrightarrow q) \equiv p \leftrightarrow \neg q$

Hình 3: Bảng mệnh đề tương đương (2)

- Hợp các tập hợp: $\cup_{i=1}^{n} A_i = A_1 \cup A_2 \cup ... \cup A_n$
- Giao các tập hợp: $\cap_{i=1}^{n} A_i = A_1 \cap A_2 \cap ... \cap A_n$

Các hằng đẳng thức trên tập hợp

Hằng đẳng thức	**Tên gọi**
$A \cup \emptyset = A$ $A \cap U = A$ (U là tập vô hạn)	Luật đồng nhất
$A \cup U = U$ $A \cup \emptyset = A$	Luật nuốt
$A \cup A = A$ $A \cap A = A$	Luật lũy đẳng
$\bar{A} = A$	Luật bù
$A \cup B = B \cup A$ $A \cap B = B \cap A$	Luật giao hoán
$A \cup (B \cup C) = (A \cup B) \cup C$ $A \cap (B \cap C) = (A \cap B) \cap C$	Luật kết hợp
$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$ $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$	Luật phân phối
$\neg(A \cup B) = \neg A \cap \neg B$ $\neg(A \cap B) = \neg A \cup \neg B$	Luật De Morgan

1.2 Bài tập

Bài 1. Sử dụng các phép biến đổi tương đương và các mệnh đề tương đương cơ bản, chứng minh sự tương đương logic sau:

1. $\neg p \to (q \to r) \Leftrightarrow q \to (p \vee v)$
2. $(p \to q) \leftrightarrow \neg q \to \neg p$
3. $p \vee q \vee (\neg p \wedge \neg q \wedge r) \Leftrightarrow p \vee q \vee r$

Bài 2. Sử dụng phương pháp lập bảng chân trị, chứng minh sự tương đương logic sau:

1. $\neg(p \leftrightarrow q) = \neg p \leftrightarrow q$
2. $(p \to q) = (\neg p \vee q)$
3. $\neg(p \vee (\neg p \wedge q)) = \neg p \wedge \neg q$
4. $(p \wedge q) \to (p \vee q) = T$
5. $\neg p \to (q \to r) = q \to (p \vee r)$

Bài 3. Kiểm tra các suy luận sau:

1. $[(p \to q) \wedge \neg q \wedge \neg r] \to \neg(p \vee r)$
2. $[(p \to (q \to r)) \wedge (\neg q \to \neg p) \wedge p] \to r$
3. $[(p \vee q) \wedge (\neg q \vee r) \wedge \neg r] \to q]$

Bài 4. Ba sinh viên A, B, C bị nghi ngờ đã gian lận trong bài thi. Khi được hỏi đã trả lời như sau:

- A: "B dã chép bài còn C vô tội"
- B: "Nếu A có tội thì C cũng có tội"
- C: "Tôi không gian lận"

1. Nếu cả 3 đều đúng thì ai gian lận?
2. Nếu cả 3 không gian lận thì ai nói dối?
3. A nói thật, B nói dối thì ai gian lận?
4. Nếu cả 3 đều nói dối thì sao?

Bài 5. Cho các mệnh đề sau:

- p = "Tam giác ABC là tam giác cân"
- q = "Tam giác ABC là tam giác đều"
- r = "Tam giác ABC có 2 cạnh BA = BC"
- s = "Tam giác ABC có góc A bằng 60 độ"

Viết các mệnh đề sau bằng ngôn ngữ tự nhiên và cho biết chận trị mệnh đề"

1. $q \to p$
2. $r \wedge \neg s \to q$
3. $r \vee s \to p$

Bài 6. Cho các vị từ P(x) = "$x \leq 5$", Q(x) = "$(x+3) \bmod 2 = 0$", R(x) = "$x > 0$". Tìm chân trị các mệnh đề sau:

1. $P(2) \vee (Q(2) \vee R(2))$
2. $P(2) \wedge (\neg Q(2) \vee \neg R(2))$
3. $P(3) \rightarrow (Q(3) \rightarrow R(3))$
4. $P(3) \wedge Q(3)) \rightarrow R(3)$

2 Bài toán đếm

2.1 Cấu hình tổ hợp

Tên gọi	Khái niệm	Công thức
Chỉnh hợp lặp	Số cách lấy ra k phần tử có thứ tự từ tập hợp gồm n phần tử	n^k
Chỉnh hợp (không lặp)	Số cách lấy ra k phần tử có thứ tự từ tập hợp gồm n phần tử *sao cho không có phần tử nào trùng nhau*	$A_n^k = \frac{n!}{(n-k)!}$
Hoán vị	Số cách xếp thứ tự n phần tử	$n!$
Tổ hợp	Số cách chọn ra k phần tử không có thứ tự tập n phần tử sao cho không có phần tử nào trùng nhau	$C_n^k = \frac{n!}{k!\,(n-k)!}$
Tổ hợp lặp	Số cách phân n phần tử vào k nhóm Số nghiệm nguyên không âm của phương trình: x1 + x2 + … + xk = n	C_{n+k-1}^{k-1}

Hình 4: Tổng hợp kiến thức cấu hình tổ hợp

2.2 Tập hợp

Nguyên lý	Khái niệm
Nguyên lý cộng	Nếu A, B, C không giao nhau thì: N(A∪B∪C) = N(A) + N(B) + N(C)
Nguyên lý nhân	N (A1× A2× .. Am) = N (A1) N (A2) . . . N (Am).
Nguyên lý bao hàm loại trừ	N(A∪B∪C)= N(A)+N(B)+N(C)–N(A∩B)–N(A∩C)– N(B∩C)+N(A∩B∩C)

Hình 5: Các nguyên lý trong tập hợp

2.3 Hệ thức truy hồi

Hệ thức truy hồi đối với dãy số $\{a_n\}$ là công thức biểu diễn a_n qua một hay nhiều số hạng đi trước của dãy, cụ thể là $a_1, a_2, .., a_{n-1}, \forall n \geq 1$. Nghiệm của hệ thức truy hồi a_n một công thức tính trực tiếp dựa vào n và các hằng số.

2.3.1 Hệ thức truy hồi bậc 2

$$a_n = c_1 \times a_{n-1} + c_2 \times a_{n-2}, c_2 \neq 0, n \geq 2$$

Phương trình đặc trưng của hệ thức truy hồi:

$$r^2 - c_1 \times r - c_2 = 0$$

Nghiệm phương trình đặc trưng	**Nghiệm hệ thức truy hồi**
2 nghiệm phân biệt $\boldsymbol{r_1, r_2}$	$a_n = \alpha_1 r_1^n + \alpha_2 r_2^n$
Nghiệm kép $\boldsymbol{r_0}$	$a_n = \alpha_1 r_0^n + \alpha_2 n r_2^n$
2 nghiệm phức liên hợp Dạng đại số: $\boldsymbol{r_1 = a + b.i}$ $\boldsymbol{r_2 = a - b.i}$ Dạng lượng giác $r_1 = r(\cos(\theta) + i.\sin(\theta))$ $r_2 = r(\cos(\theta) - i.\sin(\theta))$ Với $r = \sqrt{a^2 + b^2}$ $\theta = \tan^{-1}\frac{b}{a}$	$a_n = r^n(\alpha_1 \cos(n\theta) + \alpha_2 \sin(n\theta))$

Hình 6: Nghiệm của hệ thức truy hồi bậc 2

2.4 Hệ thức truy hồi bậc k

$$a_n = c_1 \times a_{n-1} + c_2 \times a_{n-2} + ... + c_k \times a_{n-k}, c_k \neq 0, n \geq k$$

Nghiệm phương trình đặc trưng	**Nghiệm hệ thức truy hồi**
k nghiệm phân biệt $r_1, r_2, \ldots, r_k$	$a_n = \alpha_1 r_1^n + \alpha_1 r_2^n + \cdots + \alpha_k r_k^n$

Hình 7: Nghiệm của hệ thức truy hồi bậc k

ÔN TẬP TOÁN RỜI RẠC

Câu 1:

a. Giải hệ thức truy hồi sau:
   $a_0 = 2$, $a_1 = 6$, $a_n = 3a_{n-1} - 2a_{n-2}$ với $n \geq 2$
b. Tìm hệ thức truy hồi để tính số sâu nhị phân có độ dài n và chứa 3 số 0 liên tiếp
c. Tính số xâu nhị phân thỏa mãn điều kiện ở câu b với n = 6

Câu 2:

a. Giải hệ thức truy hồi sau:
   $a_0 = 6$, $a_1 = 9$, $a_n = 7a_{n-1} - 12a_{n-2}$ với $n \geq 2$
b. Tìm hệ thức truy hồi để tính số sâu nhị phân có độ dài n, bắt đầu bằng số 0 và có chứa 2 số 0 liên tiếp
c. Tính số xâu nhị phân thỏa mãn điều kiện ở câu b với n = 6

Câu 3: Hãy tìm nghiệm của công thức truy hồi với điều kiện đầu dưới đây:

a. $a_n = a_{n-1} + 2^n$ với $a_0 = 4$
b. $a_n = -6a_{n-1} - 9a_{n-2}$ với $n \geq 2$ và $a_0 = 3$ và $a_1 = -3$
c. $a_n = 2a_{n-1} + 5a_{n-2} - 6a_{n-3}$ với $n \geq 3$ và $a_0 = 7$ và $a_1 = -4$, $a_2 = 8$

Câu 4:

a. $a_n = -4a_{n-1} - 4a_{n-2}$ với $n \geq 2$, $a_0 = 0$, $a_1 = 15$
b. $a_n = 2a_{n-1} + 5a_{n-2} - 2a_{n-3}$ với $n \geq 3$ và $a_0 = 3$ và $a_1 = 6$, $a_2 = 0$
c. $a_n = 7a_{n-2} + 6a_{n-3}$ với $n \geq 3$ và $a_0 = 9$ và $a_1 = 10$, $a_2 = 32$

Câu 5:

a. Một từ mã máy tính là một xâu có độ dài 11 gồm 4 chữ cái (Lấy tùy ý trong tập các chữ viết hoa tù A đến Z) và 7 chứ số (lấy tùy ý từ tập các chữ số từ 0 đến 9). Đếm số lượng từ mã máy tính như vậy biết rằng các chữ cái và số có thể đứng ở vị trí bất kỳ trong xâu?
b. Tìm nghiệm của hệ truy hồi: $a_n = 10a_{n-1} - 25a_{n-2}$ với $n \geq 2$ và $a_0 = 3$, $a_1 = -3$

Câu 6:

a. Một từ mã máy tính là một xâu có độ dài n gồm một số lẻ chữ số 0, Tìm hệ thức truy hồi và điều kiện đầu cho a_n?
b. Tìm nghiệm của hệ truy hồi sau: $a_n = 7a_{n-2} - 25a_{n-3}$ với $n \geq 3$ và $a_0 = 9$, $a_1 = 10$, $a_2 = 32$
c.

Câu 7:

a. Có bao nhiêu số nguyên dương có 9 chữ số, là số thuận nghịch (đối xứng) và tổng các chữ số bằng 7?
b. Có bao nhiêu số nguyên n trong đoạn từ 0 đến 120 (hay $0 \le n \le 120$) chia hết cho ít nhất 1 trong 3 số 4, 5, 6?

Câu 8:

a. Có bao nhiêu số nguyên dương có 9 chữ số, có 7 chữ số tạo thành số thuận nghịch (đối xứng)?
b. Có bao nhiêu số nguyên dương có 9 chữ số, có 7 chữ số tạo thành số thuận nghịch (đối xứng) và các chữ số đều khác 0?

Câu 9:

a. Phương trình $x_1 + x_2 + x_3 + x_4 + x_5 + x_6 = 30$ có bao nhiêu nghiệm nguyên không âm thỏa mãn $8 \ge x_2 \ge 3$ và $6 \ge x_4 \ge 2$?
b. Trình bày phương pháp liệt kê các tổ hợp chập k của tập {1, 2, …., n} sử dụng phương pháp quy lui.

Câu 10: Phương trình $x_1 + x_2 + x_3 + x_4 + x_5 + x_6 = 28$ có bao nhiêu nghiệm nguyên không âm thỏa mãn:

a. $x_1 \ge 1$, $x_2 \ge 2$, $x_3 \ge 3$, $x_4 \ge 4$, $x_5 \ge 5$, $x_6 \ge 6$?
b. $6 \ge x_1 \ge 1$ và $9 \ge x_2 \ge 4$, $x_6 \ge 6$?

3 Bài toán liệt kê

3.1 Phương pháp sinh

Bài toán	Quy tắc sinh

Sinh xâu nhị phân kế tiếp	Duyệt từ vị trí bên phải nhất của sâu nhị phân S độ dài n, tìm vị trí i đầu tiên thoả mãn $S_i = 0$
	• Nếu tìm được i thì:
	  ○ Gán $S_i = 1$
	  ○ Gán $S_j = 0$ với $i + 1 \leq j \leq n$
	• Ngược lại không tồn tại sâu nhị phân kế tiếp sâu S

Sinh hoán vị kế tiếp	Duyệt từ vị trí bên phải nhất của hoán vị p của n phần tử, tìm vị trí i đầu tiên thoả mãn $p_i < p_{i+1}$
	• Nếu tìm được i thì:
	  ○ Duyệt từ vị trí bên phải nhất của p cho tới $i + 1$, tìm vị trí j đầu tiên thoả mãn $p_j > p_i$
	  ○ Đổi chỗ p_i và p_j
	  ○ Lật ngược vị trí các phần tử $p[i + 1..n]$
	• Ngược lại không tồn tại hoán vị kế tiếp của hoán vị p

Sinh tổ hợp chập k kế tiếp	Duyệt từ vị trí bên phải nhất của c là tổ hợp chập k của n phần tử (hoặc tập con k phần tử của n phần tử), tìm vị trí i đầu tiên thoả mãn $c_i < n - k + i$
	• Nếu tìm được i thì:
	  ○ $c_i = c_i + 1$
	  ○ $c_j = c_i + j - i \; \forall j \in [i + 1, n]$
	• Ngược lại không tồn tại tổ hợp chập k của n phần tử kế tiếp c

Hình 8: Giải một số bài toán thường gặp bằng phương pháp sinh

3.2 Thuật toán quay lui

Thuật toán quay lui dùng để giải bài toán liệt kê các cấu hình. Mỗi cấu hình được xây dựng bằng cách xây dựng từng phần tử, mỗi phần tử được chọn bằng cách thử tất cả các khả năng.

Giả sử cấu hình cần liệt kê có dạng x[1..n], khi đó thuật toán quay lui thực hiện qua các bước:

1. Xét tất cả các giá trị x[1] có thể nhận, thử cho x[1] nhận lần lượt các giá trị đó. Với mỗi giá trị thử gán cho x[1] ta sẽ:
2. Xét tất cả các giá trị x[2] có thể nhận, lại thử cho x[2] nhận lần lượt các giá trị đó. Với mỗi giá trị thử gán cho x[2] lại xét tiếp các khả năng chọn x[3] ... cứ tiếp tục như vậy đến bước:
3. ...
4. Xét tất cả các giá trị x[n] có thể nhận, thử cho x[n] nhận lần lượt các giá trị đó, thông báo cấu hình tìm được ⟨x[1], x[2], ..., x[n]⟩.

Trên phương diện quy nạp, có thể nói rằng thuật toán quay lui liệt kê các cấu hình n phần tử dạng x[1..n] bằng cách thử cho x[1] nhận lần lượt các giá trị có thể. Với mỗi giá trị thử gán cho x[1] bài toán trở thành liệt kê tiếp cấu hình n - 1 phần tử x[2..n].

Mô hình của thuật toán quay lui có thể mô tả như sau:

```
{Thủ tục này thử cho x[i] nhận lần lượt các giá trị mà nó có thể nhận}
procedure Try(i: Integer);
begin
  for ⟨mọi giá trị V có thể gán cho x[i]⟩ do
    begin
      ⟨Thử cho x[i] := V⟩;
      if ⟨x[i] là phần tử cuối cùng trong cấu hình⟩ then
        ⟨Thông báo cấu hình tìm được⟩
      else
        begin
          ⟨Ghi nhận việc cho x[i] nhận giá trị V (nếu cần)⟩;
          Try(i + 1); {Gọi đệ quy để chọn tiếp x[i+1]}
          ⟨Nếu cần, bỏ ghi nhận việc thử x[i] := V để thử giá trị khác⟩;
        end;
    end;
end;
```

Hình 9: Mô hình của thuật toán quay lui

3.2.1 Bài toán sinh nhị phân

```python
# Function to generate all binary strings
def generateAllBinaryStrings(n, arr, i):

  if i == n:
    print(arr)
    return

  arr[i] = 0
  generateAllBinaryStrings(n, arr, i + 1)

  arr[i] = 1
  generateAllBinaryStrings(n, arr, i + 1)
```

3.2.2 Bài toán sinh hoán vị

```python
def permutations(nums):
    """ Generate a list of permutations from the initial list of numbers
    """
    results = []
    def swap(array, i, j):
        if i != j:
            array[i], array[j] = array[j], array[i]

    def permutate(array, start):
        if start == len(array):
            results.append(list(array))

        for i in range(start, len(array)):
            # change the candidate list for the following permutations
            swap(array, i, start)
            permutate(array, start+1)
            # backtrack so that we could try out other options
            swap(array, i, start)
    # run the permutations
    permutate(nums, 0)
    return results
```

3.2.3 Bài toán sinh tổ hợp

```python
def generate_combinations(nums, k, start=0, current=[], result=[]):
    if len(current) == k:
```

```
        result.append(current.copy())
        return

    for i in range(start, len(nums)):
        current.append(nums[i])
        generate_combinations(nums, k, i + 1, current, result)
        current.pop()

# Example usage:
nums = [1, 2, 3, 4]
k = 2
combinations = []
generate_combinations(nums, k, result=combinations)
print(combinations)
```

3.3 Bài tập

1. Cho tập A = 1, 2, 3, 4, 5, 6, 7, 8, 9. Sử dụng phương pháp sinh hoán vị theo thứ tự từ điển, tìm 4 hoán vị kiền kề tiếp theo của hoán vị 568397421.

2. Cho tập A = 1, 2, 3, 4, 5, 6, 7, 8, 9. Sử dụng phương pháp sinh hoán vị theo thứ tự từ điển, tìm 4 hoán vị kiền kề tiếp theo của hoán vị 458796321.

3. Cho tập A = 1, 2, 3, 4, 5, 6, 7, 8, 9. Sử dụng phương pháp sinh hoán vị theo thứ tự từ điển, tìm 4 hoán vị kiền kề tiếp theo của hoán vị 236897541.

4. Cho tập A = 1, 2, 3, 4, 5, 6, 7, 8, 9. Sử dụng phương pháp sinh tổ hợp chập k của một tập hợp theo thứ tự từ điển, hãy tạo 4 tổ hợp chập 4 liền kề tiếp theo của tổ hợp 2, 6, 8, 9.

5. Cho tập A = 1, 2, 3, 4, 5, 6, 7, 8, 9. Sử dụng phương pháp sinh tổ hợp chập k của một tập hợp theo thứ tự từ điển, hãy tạo 4 tổ hợp chập 4 liền kề tiếp theo của tổ hợp 3, 5, 6, 9.

6. Cho tập A = 1, 2, 3, 4, 5, 6, 7, 8, 9. Sử dụng phương pháp sinh tổ hợp chập k của một tập hợp theo thứ tự từ điển, hãy tạo 4 tổ hợp chập 4 trước liền kề của tổ hợp 3, 5, 6, 9.

7. Cho dãy $A = (a_1, a_2, ..., a_n)$. Viết chương trình trong C/C++/Python liệt kê các dãy con k phần tử giảm dần của dãy số A? Ví dụ: Cho $A = (1, 5, 3, 4, 2, 0)$, k=3, các dãy con thỏa mãn yêu cầu đề bài: (5,3,2); (5,2,0);(5,4,2)... Lưu ý: Không sử dụng các thư viện có sẵn để sinh hoán vị, tổ hợp.

8. Viết chương trình C/C++/Python liệt kê các hoán vị của tập $\{1, 2, ..., n\}$ sử dụng phương pháp sinh theo thứ tự từ điển.

9. Trình bày phương pháp liệt kê các tổ hợp chập k của tập $\{1, 2, ...n\}$ sử dụng phương pháp quay lui.

10. Viết hàm trong C/C++/Python sử dụng phương pháp quay lui liệt kê các xâu nhị phân độ dài n. Sau đó trình bày cây biểu diễn quá trình hoạt động của hàm khi sinh các xâu nhị phân độ dài 3.

11. Viết hàm trong C/C++/Python sử dụng phương pháp sinh theo thứ tự từ điển liệt kê các tổ hợp chập k của tập n phần tử $\{1, 2, ...n\}$.

4 Bài toán tối ưu

Bài toán đếm thực hiện đếm các cấu hình tổ hợp thỏa mãn một số tính chất nào đó. Bài toán liệt kê xem xét từng cấu hình tổ hợp thỏa mãn các tính chất đặt ra. Bài toán tối ưu chỉ quan tâm đến nghiệm tốt nhất (xấu nhất) theo một nghĩa nào đó đặt ra của bài toán. Nội dung chính của chương này là giới thiệu các phương pháp giải quyết bài toán tối ưu đồng thời giải quyết một số bài toán có vai trò quan trọng của lý thuyết tổ hợp.

4.1 Giới thiệu bài toán tổng quát

Bài toán tối ưu tổ hợp: *Hãy lựa chọn trong tất cả các cấu hình tổ hợp chấp nhận được cấu hình có giá trị sử dụng tốt nhất.* Tổng quát, bài toán có thể phát biểu như sau:

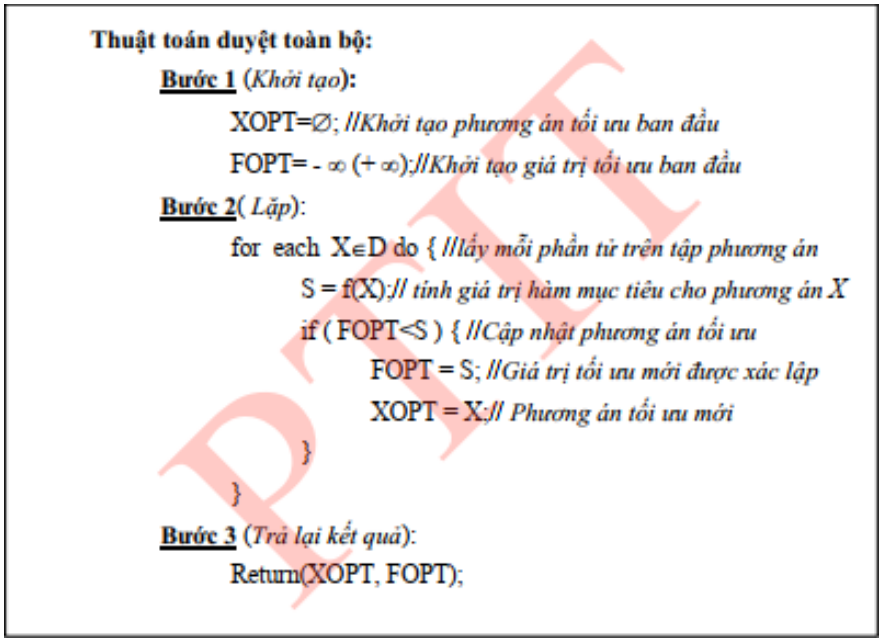
Tìm cực tiểu (hay cực đại) của phiếm hàm $f(x) = min(max)$ với điều kiện $x \in D$, trong đó D là tập hữu hạn các phần tử. Các thành phần chính sau:

- Tập D gọi là tập các phương án của bài toán.
- $x \in D$ được gọi là một phương án.
- $f(x)$ được gọi là hàm mục tiêu của bài toán.
- Phương án x^* đem lại giá trị nhỏ nhất (lớn nhất) cho hàm mục tiêu được gọi là phương án tối ưu.
- $x^* \to f^* = f(x^*)$ được gọi là giá trị tối ưu của bài toán.

4.1.1 Phương pháp duyệt toàn bộ

Một trong những phương pháp hiển nhiên có thể giải bài toán tối ưu tổ hợp là duyệt tất cả các phương án của bài toán, gồm các bước chính sau:

- Liệt kê tập D gồm tất cả các phương án $x \in D$ của bài toán.
- Tính hàm mục tiêu với mỗi phương án $\forall x \in D \rightarrow f(x)$
- Trả về phương án $x^* \in D$ sao cho giá trị mục tiêu $f(x^*)$ là nhỏ nhất (lớn nhất).



Hình 10: Thuật toán duyệt toàn bộ giải bài toán tối ưu tổ hợp

Ghi chú: Trong bước 1 của thuật toán, ta khởi tạo $FOPT = -\infty$ với bài toán cực đại hàm mục tiêu và ngược lại.

4.1.2 Phương pháp nhánh cận

Giả sử bài toán tối ưu tổ hợp có mô hình tổng quát như sau: Tìm $min\{(fx) : x \in D\}$. Với D là tập hữu hạn các phương án x. $D = \{x = (x_1, x_2, ..., x_n) \in A_1 \times A_2 \times ... \times A_n\}$, với $A_1 \times A_2 \times ... \times A_n$ là các tập hữu hạn.

Với tập D như trên, ta có thể sử dụng thuật toán quay lui để liệt kê các phương án của bài toán. Trong quá trình liệt kê theo thuật toán quay lui, ta sẽ xây dựng dần các thành phần của phương án. Ta gọi $(a_1, a_2, ..., a_k)$ là một bộ phận gồm k thành phần - phương án bộ phận cấp k.
Thuật toán nhánh cận có thể áp dụng nếu tìm được hàm g thỏa mãn:

$$g(a_1, a_2, ..., a_k) \leq min\{f(x) : x \in D, x_i = a_i, i = 1, 2, ..., k\}$$

Hàm g sẽ được sử dụng để hạn chế khối lượng duyệt trong quá trình duyệt tất cả các phương án theo thuật toán quay lui. Trong quá trình liệt kê các phương án có thể đã thu được một số phương án của bài toán. Gọi $\bar{f} = f(\bar{x})$ và $\bar{x}$ lần lượt là kỷ lục hiện có và phương án tốt nhất hiện có. Khi đó, nếu $\bar{f} < g(a_1, a_2, ..., a_k) \leq min\{f(x) : x \in D, x_i = a_i, i = 1, 2, ..., k\}$, thì phương án của bài toán $D(a_1, a_2, ..., a_k)$ chắc chắn không chứa phương án tối ưu và không cần phát triển phương án bộ phận $(a_1, a_2, ..., a_k)$ và loại bỏ các phương án trong tập $D(a_1, a_2, .., a_k, .., a_n)$ khỏi quá trình tìm kiếm.

```
procedure Branch(k);
(* Phát triển phương án bộ phận (x1, x2, ..., xk-1) *)
begin
  for ak∈Ak do
    if ak∈Sk then
    begin
      xk := ak;
      if (k = n) then < Cập nhật kỷ lục>
      else
      if g(x1,..., xk) ≤ f̄ then Branch(k+1)
    end;
end;

procedure BranchAndBound;
begin
  f̄ := +∞;
  (* Nếu biết p/án x̄ nào đó thì đặt f̄ = f( x̄ ) *)
  Branch(1);
  if f̄ < +∞ then
      < f̄ là giá trị tối ưu, x̄ là p/án tối ưu >
  else < bài toán không có phương án >;
end;
```

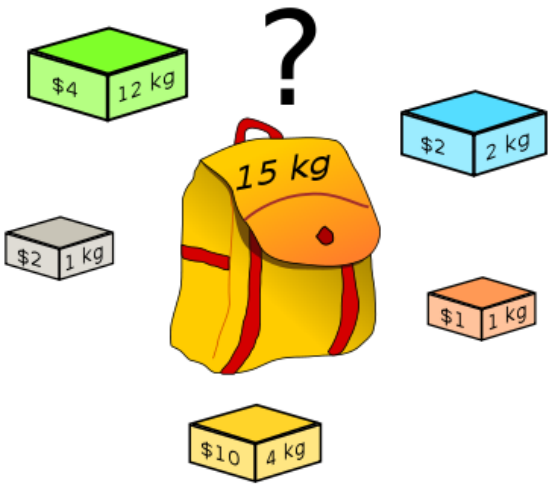
Hình 11: Thuật toán nhánh cận giải bài toán tối ưu tổ hợp

4.2 Bài toán cái túi

4.2.1 Tổng quát bài toán

Bài toán cái túi (Knapsack) là một bài toán tối ưu hóa tổ hợp trong đó ta cần phải lựa chọn một số đồ vật để bỏ vào một chiếc túi (mỗi đồ vật được chọn tối đa 1 lần) sao cho:

- có tổng trọng lượng đồ vật không vượt quá trọng lượng cho phép của túi;
- có tổng giá trị đồ vật là lớn nhất có thể.



Tổng quát hóa

Giả sử trọng lượng tối đa của túi là b, có n đồ vật có thể mang theo. Đồ vật thứ j có trọng lượng a_j và giá trị sử dụng c_j (j=1,2,...,n). Gọi $A = (a_1, a_2, ..., a_n)$, $C = (c_1, c_2, ..., c_n)$ tương ứng với vector trọng lượng và giá trị sử dụng các đồ vật.

Phát biểu bài toán

Một phương án chọn đồ vật có thể biểu diễn như một vector nhị phân có độ dài n: $X = (x_1, x_2, ..., x_n), x_i \in \{0, 1\}, i \in \{1, 2, ..., n\}$:

- $x_i = 0$ có nghĩa đồ vật thứ i không được chọn;
- $x_i = 1$ có nghĩa đồ vật thứ i được chọn.

Ràng buộc và mục tiêu

Mỗi phương án chọn đồ vật $X = (x_1, x_2, ..., x_n)$ phải có tổng trọng lượng không vượt quá b. Nghĩa là tập phương án D được xác định như sau:

$$D = \{X = (x_1, x_2, ..., x_n) : \sum_{i=1}^{n} a_i \times x_i \leq b\}$$

Với mỗi phương án chọn đồ vật $X \in D$, ta có:

- tổng trọng lượng của đồ vật: $g(X) = \sum_{i=1}^{n} a_i \times x_i$
- tổng giá trị đồ vật: $f(x) = \sum_{i=1}^{n} c_i \times x_i$

4.2.2 Phương pháp duyệt toàn bộ

Ví dụ 1. Giải bài toán cái túi dưới đây bằng phương pháp duyệt toàn bộ.

$$\begin{cases} f(X) = 10x_1 + 5x_2 + 3x_3 + 6x_4 \to max \\ 5x_1 + 3x_2 + 2x_3 + 4x_4 \leq 8 \\ x_j \in \{0, 1\}, j = 1, 2, 3, 4 \end{cases}$$

Lời giải: Gọi $A = (a_1, a_2, a_3, a_4) = (5, 3, 2, 4), C = (c_1, c_2, c_3, c_4) = (10, 5, 3, 6)$ tương ứng với vector trọng lượng và giá trị sử dụng các đồ vật.
Tập các phương án của bài toán $D = \{X = (x_1, x_2, x_3, x_4)\}$:

$$g(X) = \sum_{i=1}^{4} a_i \times x_i = 5x_1 + 3x_2 + 2x_3 + 4x_4 \leq 8$$

Hàm mục tiêu của bài toán:

$$F(X) = \sum_{i=1}^{4} c_i \times x_i = 10x_1 + 5x_2 + 3x_3 + 6x_4$$

Kết quả từng bước theo thuật toán duyệt toàn bộ như sau:

$X = (x_1, x_2, x_3, x_4)$	$g(X) \leq 8$	$F(X) = \sum_{i=1}^{4} c_i \times x_i$
$0, 0, 0, 0$	$0 \leq 8 : X \in D$	0
$0, 0, 0, 1$	$4 \leq 8 : X \in D$	6
$0, 0, 1, 0$	$2 \leq 8 : X \in D$	3
$0, 0, 1, 1$	$6 \leq 8 : X \in D$	9
$0, 1, 0, 0$	$3 \leq 8 : X \in D$	5
$0, 1, 0, 1$	$7 \leq 8 : X \in D$	11
$0, 1, 1, 0$	$5 \leq 8 : X \in D$	8
$0, 1, 1, 1$	$9 > 8 : X \notin D$	$\emptyset$
$1, 0, 0, 0$	$5 \leq 8 : X \in D$	10
$1, 0, 0, 1$	$9 > 8 : X \notin D$	$\emptyset$
$1, 0, 1, 0$	$7 \leq 8 : X \in D$	13
$1, 0, 1, 1$	$11 > 8 : X \notin D$	$\emptyset$
$1, 1, 0, 0$ **(XOPT)**	$8 \leq 8 : X \in D$	**15 (FOPT)**
$1, 1, 0, 1$	$12 > 8 : X \notin D$	$\emptyset$
$1, 1, 1, 0$	$10 > 8 : X \notin D$	$\emptyset$
$1, 1, 1, 1$	$14 > 8 : X \notin D$	$\emptyset$

Vậy, phương án tối ưu là $(1, 1, 0, 0)$ với giá trị tối ưu là 15.

4.2.3 Phương pháp nhánh cận

Mô hình toán học của bài toán với b là khối lượng tối đa của túi có dạng sau:

$$f^* = max\{f(x) = \sum_{i=1}^{n} c_i \times x_i : \sum_{i=1}^{n} a_i \times x_i \leq b, x_i \in Z^+, j = 1, 2, ..., n\} \quad (1)$$

Tập phương án của bài toán D:

$$D = \{x = (x_1, x_2, ..., x_n) : \sum_{i=1}^{n} a_i \times x_i \leq b\}$$

Đầu tiên, tiến hành sắp xếp đồ vật theo thứ tự ưu tiên về tỉ lệ giá trị trên đơn vị khối lượng, thỏa mãn:

$$\frac{c_1}{a_1} \geq \frac{c_2}{a_2} \geq ... \geq \frac{c_n}{a_n}$$

Trong quá trình liệt kê các phương án có thể đã thu được một số phương án của bài toán. Gọi $\bar{f} = f(\bar{x})$ và $\bar{x}$ lần lượt là kỷ lục hiện có và phương án tốt nhất hiện có.
Với mỗi phương án bộ phận cấp k: $(x_1, x_2, ..., x_k)$ ta có:

- Giá trị hiện tại của túi: $\delta_k = \sum_{i=1}^{k} c_i \times x_i$;
- Trọng lượng hiện tại của túi: $\sum_{i=1}^{k} a_i \times_i$
- Trọng lượng còn lại của túi: $w_k = b - \sum_{i=1}^{k} a_i \times_i$;
- Giá trị tốt nhất (cận trên) cho phương án bộ phận: $g(x_1, x_2, ..., x_k) = \delta_k + \frac{c_{k+1} \times w_k}{a_{k+1}}$.

Khi đó, nếu $\bar{f} > g(x_1, x_2, ..., x_k)$, thì phương án của bài toán $D(x_1, x_2, ..., x_k)$ chắc chắn không chứa phương án tối ưu và không cần phát triển tiếp tục. Thuật toán nhánh cận giải bài toán cái túi được mô tả dưới đây.

```
Thuật toán Brach_And_Bound (i) {
    t = ( b - b_i)/A[i]; //Khởi tạo số lượng đồ vật thứ i
    for j = t; j ≥0; j--){
        x[i] = j; //Lựa chọn x[i] là j;
        b_i = b_i + a_i x_i; // Trọng lượng túi cho bài toán bộ phận thứ i.
        σ_i = σ_i + c_i x_i;// Giá trị sử dụng cho bài toán bộ phận thứ i
        If (k==n) <Cập nhật kỷ lục>;
        else if (δ_k + (c_{k+1}*b_k)/a_{k+1} >FOPT) //Nhánh cận được triển khai tiếp theo
            Branch_And_Bound(k+1);
        b_i = b_i − a_i x_i;
        σ_i = σ_i − c_i x_i;
    }
}
```

Hình 12: Thuật toán nhánh cận giải bài toán cái túi

Ví dụ: Giải bài toán cái túi dưới đây bằng phương pháp nhánh cận.

$$\begin{cases} f(X) = 10x_1 + 5x_2 + 3x_3 + 6x_4 \rightarrow max \\ 5x_1 + 3x_2 + 2x_3 + 4x_4 \leq 8 \\ x_j \in \{0, 1\}, j = 1, 2, 3, 4 \end{cases}$$

Lời giải:
Gọi $A = (a_1, a_2, a_3, a_4) = (5, 3, 2, 4), C = (c_1, c_2, c_3, c_4) = (10, 5, 3, 6)$ tương ứng với vector trọng lượng và giá trị sử dụng các đồ vật.
Tập các phương án của bài toán $D = \{X = (x_1, x_2, x_3, x_4)\}$:

$$g(X) = \sum_{i=1}^{4} a_i \times x_i = 5x_1 + 3x_2 + 2x_3 + 4x_4 \leq 8$$

Hàm mục tiêu của bài toán:

$$F(X) = \sum_{i=1}^{4} c_i \times x_i = 10x_1 + 5x_2 + 3x_3 + 6x_4$$

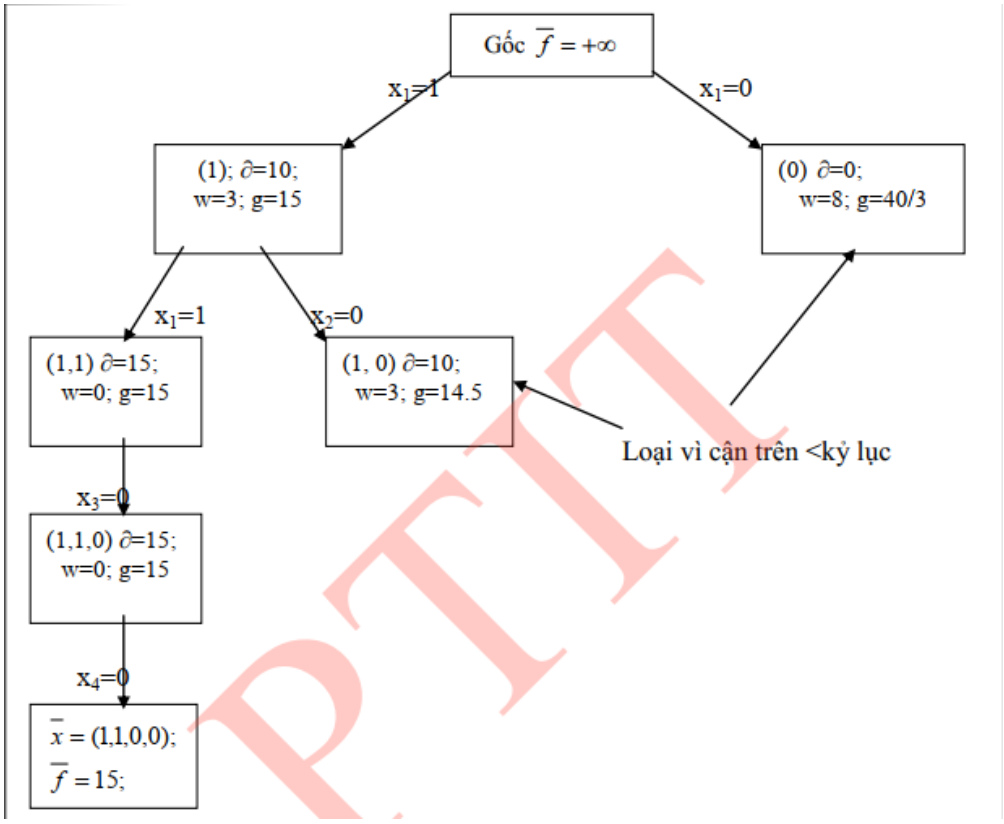
Sắp xếp đồ vật theo thứ tự ưu tiên về tỉ lệ giá trị trên đơn vị khối lượng:

$$\frac{10}{5} \geq \frac{5}{3} \geq \frac{3}{2} \frac{6}{4}$$

Với mỗi phương án bộ phận thứ $i : X = (x_1, x_2, ..., x_i)$:

- δ là giá trị của các đồ vật hiện tại trong túi;
- w là trọng lượng còn lại của túi;
- g là giá trị tốt nhất (cận trên) cho phương án bộ phận;
- $\bar{f}$ là kỷ lục hiện có.

Kết quả của bài toán theo thuật toán nhánh cận được thể hiện như sau:



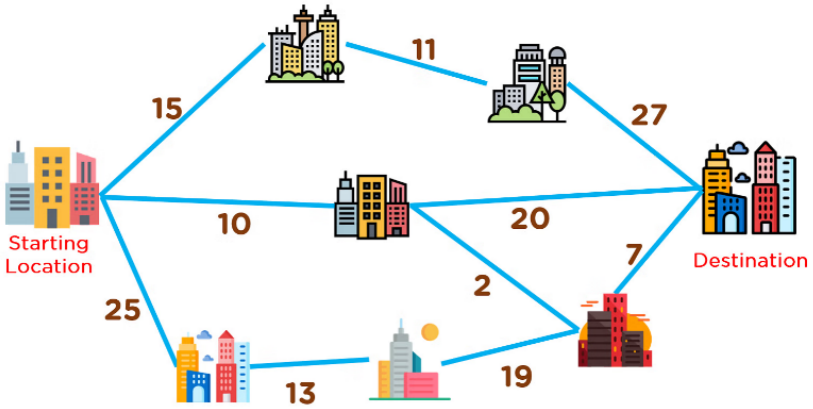
Hình 13: Lời giải bài toán cái túi theo thuật toán nhánh cận

Vậy, phương án tối ưu là $(1, 1, 0, 0)$ với giá trị tối ưu là 15.

4.3 Bài toán người du lịch

4.3.1 Tổng quát bài toán

Một người du lịch trên hệ thống n thành phố. Giữa các thành phố có thể có hoặc không các đường nối, mỗi đường nối có chi phí xác định từ trước. Người du lịch xuất phát từ một thành phố, đi tới tất cả các thành phố khác và mỗi thành phố đi qua một lần và quay trở lại thành phố ban đầu. Hãy xác định một hành trình sao cho tổng chi phí trên đường đi là nhỏ nhất.



Tổng quát hóa

Giả sử có n thành phố cần tham quan $T_1, T_2, ..., T_n$, $c[i][j]$ là chi phí đi từ thành phố T_i đến thành phố $T_j (i, j = 1, 2, ..., n)$. Không hạn chế tính tổng quát, ta giả sử người du lịch luôn xuất phát tại một thành phố cố định (giả sử là thành phố số 1).

Phát biểu bài toán

Một phương án hành trình của người du lịch có dạng $X = (x_1, x_2, ..., x_n, x_1)$ là hoán vị của 1,2,...,n và $x_1 = 1$ ở vị trí đầu tiên và cuối cùng vì ta giả sửu người du lịch xuất phát từ thành phố số 1 và kết thúc tại điểm xuất phát. Như vậy, ta có $(n-1)!$ hành trình.

Ràng buộc và mục tiêu

Tập phương án của bài toán là:

$$D = \{X = (x_2, x_2, ..., x_n) : x_1 = 1; x_i \neq x_j \, \forall i \neq j; i, j = 2, 3, ..., n\}$$

Với mỗi phương án hành trình $X \in D$, ta có tổng chi phí là:

$$f(X) = \sum_{i=1}^{n-1} c[x_i, x_{i+1}] + c[x_n, 1]$$

Bài toán người du lịch là bài toán tối ưu tổ hợp tìm $min\{f(X) : X \in D\}$

4.3.2 Phương pháp duyệt toàn bộ

Tham khảo bài toán cái túi 4.2.2.

4.3.3 Phương pháp nhánh cận

Bài toán người du lịch là bài toán tối ưu tổ hợp $f(X) = \sum_{i=1}^{n-1} c[x_i, x_{i+1}] + c[x_n, 1] \to min$.
Đầu tiên, xác định $c_{min} = min\{c[i,j], i, j = 1, 2, ..., n; i \neq j\}$ là chi phí nhỏ nhất để đi lại giữa 2 thành phố.
Trong quá trình liệt kê các phương án có thể đã thu được một số phương án của bài toán. Gọi $\bar{f} = f(\bar{x})$ và $\bar{x}$ lần lượt là kỷ lục hiện có và phương án tốt nhất hiện có.
Giả sử ta đang có phương án bộ phận $(x_1, x_2, ..., x_k)$. Phương án tương ứng với hành trình bộ phận qua k thành phố:

$$T_{x_1} \to T_{x_2} \to ... \to T_{x_{k-1}} \to T_{x_k}$$

Chi phí phải trả theo hành trình bộ phận này: $\delta = \sum_{i=1}^{k} c[x_i, x_{i+1}]$.
Ta còn phải đi qua $n-k$ thành phố còn lại rồi quay trở về thành phố Tx_1, tức đi qua $n-k+1$ đoạn đường. Cận dưới cho phương án với bộ phận $(x_1, x_2, ..., x_k)$ được tính:

$$g(x_1, x_2, ..., x_k) = \delta + (n - k + 1) \times c_{min}$$

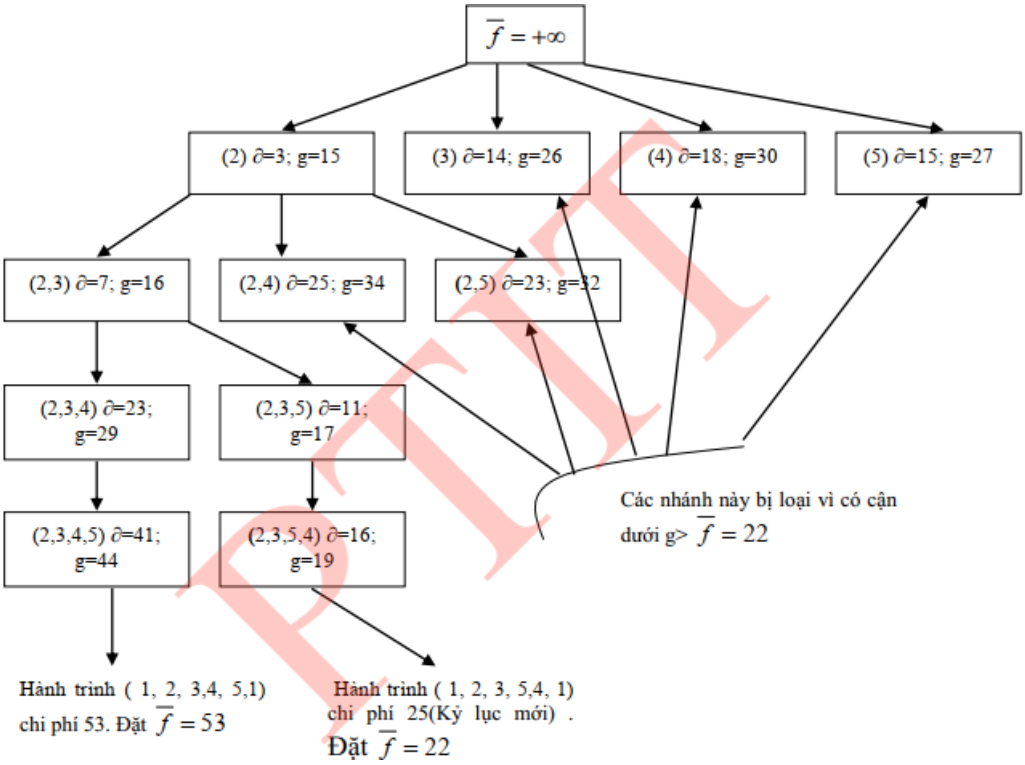
Khi đó, nếu $\bar{f} < g(x_1, x_2, ..., x_k)$, thì phương án của bài toán $D(x_1, x_2, ..., x_k)$ chắc chắn không chứa phương án tối ưu và không cần phát triển tiếp tục.
Ví dụ: Giải bài toán người du lịch với ma trận chi phí:

$$C = \begin{vmatrix} 0 & 3 & 14 & 18 & 15 \\ 3 & 0 & 4 & 22 & 20 \\ 17 & 9 & 0 & 16 & 4 \\ 6 & 2 & 7 & 0 & 12 \\ 9 & 15 & 11 & 5 & 0 \end{vmatrix}$$

Lời giải: Ta có:

- $c_{min} = 2$.
- δ là chi phí hiện tại theo hành trình bộ phận.
- g là cận dưới của hành trình chứa hành trình bộ phận.

Quá trình thực hiện như sau:



Hình 14: Cây tìm kiếm lời giải bài toán người du lịch

Vậy ta thu được phương án tối ưu $(1, 2, 3, 5, 4, 1)$ với chi phí nhỏ nhất là 22.

4.4 Bài tập

1. Trình bày thuật toán nhánh cận giải bài toán cái túi?
2. Trình bày thuật toán duyệt toàn bộ giải bài toán cái túi.
3. Giải bài toán cái túi sau bằng phương pháp duyệt toàn bộ và nhánh cận:

$$\begin{cases} 5x_1 + x_2 + 9x_3 + 3x_4 \to max \\ 4x_1 + 2x_2 + 7x_3 + 3x_4 \leq 10 \\ x_j \in \{0, 1\}, j = 1, 2, 3, 4 \end{cases}$$

4. Giải bài toán cái túi sau bằng phương pháp duyệt toàn bộ và nhánh cận:

$$\begin{cases} 7x_1 + 3x_2 + 2x_3 + x_4 \to max \\ 5x_1 + 3x_2 + 6x_3 + 4x_4 \leq 12 \\ x_j \in \{0, 1\}, j = 1, 2, 3, 4 \end{cases}$$

5. Giải bài toán người du lịch với ma trận chi phí sau:

$$\begin{vmatrix} \infty & 31 & 15 & 23 & 10 & 17 \\ 16 & \infty & 24 & 07 & 12 & 12 \\ 34 & 03 & \infty & 25 & 54 & 25 \\ 15 & 20 & 33 & \infty & 50 & 40 \\ 16 & 10 & 32 & 03 & \infty & 23 \\ 18 & 20 & 13 & 28 & 21 & \infty \end{vmatrix}$$

6. Giải bài toán người du lịch với ma trận chi phí sau:

$$\begin{vmatrix} \infty & 03 & 93 & 13 & 33 & 09 \\ 04 & \infty & 77 & 42 & 21 & 16 \\ 45 & 17 & \infty & 36 & 16 & 28 \\ 39 & 90 & 80 & \infty & 56 & 07 \\ 28 & 46 & 88 & 33 & \infty & 25 \\ 03 & 88 & 18 & 46 & 92 & \infty \end{vmatrix}$$