GIỚI THIỆU

Toán rời rạc là một môn toán dành riêng cho ngành Công nghệ thông tin. Toán rời rạc nghiên cứu lĩnh vực giải quyết các bài toán trên các đối tượng rời rạc, cụ thể là các cấu hình tổ hợp (chỉnh hợp, tổ hợp, hoán vị, …), đại số Boole đồ thị. Các bài toán trên các cấu hình tổ hợp là: bài toán đếm, bài toán tồn tại, bài toán liệt kê. Các bài toán trên đại số Boole là chuẩn tắc hóa và tối thiểu hóa biểu thức Boole, . Các bài toán trên đồ thị là: bài toán tìm đường đi ngắn nhất, bài toán tìm cây phủ nhỏ nhất, bài toán tìm luồng cực đại.

Trên cơ sở kinh nghiệm qua nhiều năm giảng dạy môn học này tại các trường trong Đại học Đà nẵng, chúng tôi soạn giáo trình này để dùng cho hệ kỹ sư, hệ cử nhân, hệ cao đẳng chuyên ngành công nghệ thông tin. Tài liệu được trình bày thành 7 chương:

Chương 1: Bài toán đếm-Bài toán tồn tại
Chương 2: Kỹ thuật đếm nâng cao
Chương 3: Bài toán liệt kê
Chương 4: Đại số Boole
Chương 5: Đại số mệnh đề
Chương 6: Đồ thị
Chương 7: Các bài toán tối ưu trên đồ thị

Đây là môn toán ứng dụng nên phần lý thuyết chúng tôi chỉ trình bày ngắn gọn, không đi sâu vào việc chứng minh các định lý, mà chủ yếu là rút ra các thuật toán. Các thuật toán được trình bày bằng ngôn ngữ giả C, nghĩa là ngôn ngữ C cộng với các từ tiếng Việt nhằm thuật toán được dễ hiểu.

Giáo trình này có thể làm tài liệu giảng dạy tại tất cả các trường trong Đại học Đà nẵng: Đại học Bách khoa, Đại học Sư phạm, Cao đẳng Công nghệ thông tin và ôn thi cao học ngành công nghệ thông tin.

Mặc dù rất cẩn thận trong quá trình biên soạn, tuy nhiên tài liệu không thể tránh khỏi những thiếu sót và hạn chế. Chúng tôi rất mong được sự góp ý quí báu của tất cả bạn đọc và các bạn đồng nghiệp.

Mọi ý kiến đóng góp xin gửi về: Khoa Công nghệ thông tin, trường Đại học Bách khoa, Đại học Đà nẵng hoặc emal: pthanhtao@yahoo.com.

Xin chân thành cảm ơn.

Đà nẵng, tháng 3 năm 2012

****

Chương 1. Bài toán đếm – Bài toán tồn tại

Lý thuyết tổ hợp là một phần quan trọng của toán học rời rạc. Từ một tập nguồn hữu hạn chọn ra một số phần tử có hay không có thứ tự, lặp hay không lặp. Mỗi bộ phần tử như thế gọi là một cấu hình tổ hợp. Đếm các đối tượng có những tính chất nào đó là một bài toán quan trọng của lý thuyết tổ hợp. Giải quyết tốt bài toán đếm giúp ta giải nhiều bài toán trong đánh giá độ phức tạp tính toán của các thuật toán và tìm xác suất rời rạc các biến cố. Phương pháp chung để giải bài toán đếm được dựa trên các nguyên lý đếm cơ bản (nguyên lý cộng, nguyên lý nhân, nguyên lý bù trừ).

Nội dung chính được đề cập trong chương này bao gồm:

- Các nguyên lý cơ bản:
  - Nguyên lý nhân
  - Nguyên lý cộng
  - Nguyên lý bù trừ
  - Nguyên lý Dirichlet
- Các cấu hình tổ hợp cơ bản
  - Chỉnh hợp lặp
  - Chỉnh hợp
  - Hoán vị
  - Tổ hợp
  - Các công thức tổ hợp
- Các cấu hình tổ hợp suy rộng
  - Hoán vị lặp
  - Tổ hợp lặp

1.1. Các nguyên lý cơ bản

1.1.1. Nguyên lý nhân

Một hoạt động được thực hiện bởi **k** *bước: bước* **1** *có* $\mathbf{n_1}$ *cách, bước* **2** *có* $\mathbf{n_2}$ *cách, ..., bước* **k** *có* $\mathbf{n_k}$ *cách. Tổng số hoạt động là*

$$\mathbf{n} = n_1 \times n_2 \times \ldots \times n_k = \prod_{i=1}^{k} n_i$$

Dạng tập hợp

Cho **k** *tập* $\mathbf{A_1, A_2, \ldots, A_k}$. *Số phần tử của tập tích Đề-các*

$$|\mathbf{A_1 \times A_2 \times \ldots \times A_k}| = |\mathbf{A_1}| \times |\mathbf{A_2}| \times \ldots \times |\mathbf{A_k}|$$

$$\left|\prod_{i=1}^{k} A_i\right| = \prod_{i=1}^{k} |A_i|$$

Ví dụ. Đếm số xâu chữ độ dài 3 gồm các chữ cái trong tập X={A, B, C, D, E}:

a) *Bất kỳ*
b) *Không lặp chữ cái*

Giải.

Gọi $S=s_1s_2s_3$ là xâu chữ độ dài 3 gồm các chữ cái trong tập X.

a) $\forall i=1..3$, s_i có 5 cách chọn. Theo nguyên lý nhân, số xâu chữ trên là 5x5x5=125.

b) s_1 có 5 cách chọn,
sau khi có s_1 thì s_2 có 4 cách chọn,
sau khi có s_1s_2 thì s_3 có 3 cách chọn.
Theo nguyên lý nhân, số xâu chữ trên là 5x4x3=60

Ví dụ. Đếm số xâu nhị phân độ dài *n*.

Giải

Gọi $b=b_1b_2...b_n$ là xâu nhị phân độ dài *n*.
$\forall i=1..n$, b_i có 2 cách chọn (0 hoặc 1).
Theo nguyên lý nhân, số xâu nhị phân độ dài n là 2 x 2 x ... x 2=2^n.

Ví dụ. Đếm số các số lẻ gồm hai chữ số.

Giải.

Gọi n = $\overline{ab}$ là một số lẻ gồm hai chữ số.
Chữ số $\bar{a}$ có 9 cách chọn (1..9), chữ số $\bar{b}$ có 5 cách chọn (1,3,5,7,9).
Theo nguyên lý nhân, số các số lẻ gồm hai chữ số là 9x5 = 45.

Ví dụ. Đếm số lệnh gán khi thực hiện đoạn lệnh sau:

```
for (i=1; i<=m; i++)
  for (j=1; j<=n; j++)
        A[i][j]=i*n+j;
```

Giải.

Theo nguyên lý nhân, số lệnh gán là $m_x n$.

1.1.2. Nguyên lý cộng

Một hoạt động có thể được thực hiện bởi một trong **k** *bước riêng biệt: bước* **1** *có* $\mathbf{n_1}$ *cách, bước* **2** *có* $\mathbf{n_2}$ *cách, ..., bước* **k** *có* $\mathbf{n_k}$ *cách. Tổng số hoạt động là*

$$\mathbf{n} = n_1 + n_2 + \ldots + n_k = \sum_{i=1}^{k} n_i$$

Dạng tập hợp

Đơn giản: *Cho hai tập* **A, B** *rời nhau. Có*

$$|A \cup B| = |A| + |B|$$

Tổng quát: *Cho k tập rời nhau từng đôi một* $A_1, A_2, ..., A_k$. *Có*

$$|A_1 \cup A_2 \cup \ldots \cup A_k| = |A_1| + |A_2| + \ldots + |A_k|$$

$$\left|\bigcup_{i=1}^{k} A_i\right| = \sum_{i=1}^{k} |A_i|$$

Ví dụ. Đếm số byte có hai bit đầu 00 hoặc 11.

Giải.

Gọi A, B là tập các byte (xâu nhị phân độ dài 8) có hai bit đầu 00 hoặc 11 tương ứng.

Theo ví dụ trên, $|A|=|B|=2^6=64$.
Có $A\cap B=\emptyset$. Theo nguyên lý cộng, số byte có hai bit đầu 00 hoặc 11 là
$|A|+|B|=64+64=128$.

Ví dụ. Một đoàn thi OLP của Đại học Đà Nẵng đi dự thi Tin và Toán gồm 2 trường Đại học Bách khoa và Đại học Sư phạm. Mỗi sinh viên chỉ dự thi một môn. Số sinh viên Bách khoa là 9. Số sinh viên thi môn Tin là 10. Số sinh viên Bách khoa thi Tin bằng số sinh viên Sư phạm thi Toán. Toàn đoàn có mấy sinh viên?

Giải.
Gọi S, B là tập các sinh viên Sư phạm, Bách khoa tương ứng.
S_1, S_2 là là tập các sinh viên Sư phạm thi Tin, Toán tương ứng.
B_1, B_2 là là tập các sinh viên Bách khoa thi Tin, Toán tương ứng.
Có $|B_1|+|B_2| =9$,
$|B_1|+|S_1| =10$,
$|B_1| = |S_2|$
Suy ra $|S_1|+|S_2| = 10$. Hay số sinh viên Sư phạm là 10.
Vậy, toàn đoàn có 10+9 = 19 sinh viên.
Tất cả suy luận trên từ nguyên lý cộng.

Ví dụ. Đếm số lệnh gán khi thực hiện đoạn lệnh sau:

```
for (i=1; i<=m; i++) X[i] =i;
for (j=1; j<=n; j++)  Y[j]=1+2*j;
```

Giải.
Theo nguyên lý cộng, số lệnh gán là m+n.

1.1.3. Nguyên lý bù trừ

Đơn giản:
Cho hai tập A, B. *Có* $|A\cup B|=|A|+|B|-|A\cap B|$
Cho ba tập A, B, C. *Có*
$|A\cup B\cup C| = |A|+|B|+|C| - (|A\cap B| + |A\cap C| + |B\cap C|) + |A\cap B\cap C|$

Tổng quát: *Cho* **n** *tập* $A_1, A_2, \ldots, A_n$.

$$\left|\bigcup_{k=1}^{n} A_i\right| = \sum_{k=1}^{n} (-1)^{k+1} N_k$$

trong đó $\mathbf{N_k}$ *là tổng phần tử của giao* **k** *tập trong* **n** *tập* $A_1, A_2, \ldots, A_n$.

Ví dụ. Đếm số nguyên dương không quá 100 chia hết cho 3 hoặc 7.

Giải.
Gọi S={1..100}, A là tập các số trong S chia hết cho 3 và B là tập các số trong S chia hết cho 7. Thì $|A|=33$ và $|B|=14$. $A\cap B$ là tập các số nguyên như thế chia hết cho 3 và 7, nghĩa là chia hết cho 21, $A\cap B = 4$. Do đó, số nguyên dương không quá 100 chia hết cho 3 hoặc 7 là $|A\cup B| = |A|+|B| - |A\cap B|= 33+14 - 4 = 43$.

Ví dụ. Một lớp gồm 50 sinh viên, có 30 sinh viên nữ, và có 35 sinh viên tóc vàng. Chứng tỏ rằng có ít nhất 15 sinh viên nữ tóc vàng.

Giải.
Gọi A là tập các sinh viên nữ, B là tập các sinh viên tóc vàng. Thì
$|A\cap B| = |A|+|B| - |A\cup B| = 30 + 35 - |A\cup B|$
≥ 15 vì $|A\cup B| \leq 50$

Ví dụ. Đếm số byte có 2 bit đầu 00 hoặc 2 bit cuối 11.

Giải.
Gọi A là tập các số byte có 2 bit đầu 00, B là tập các số byte có hai bit cuối 11.
Có $|A|=|B|=64$.
Số byte có 2 bit đầu 00 và 2 bit cuối 11 là $|A\cap B|=2^4=16$ (4 bit giữa bất kỳ).
Theo nguyên lý nhân, số byte có 2 bit đầu 00 hoặc 2 bit cuối 11 là
$|A\cup B|=|A|+|B|-|A\cap B|= 64+64-16=112$.

1.1.4. Nguyên lý Dirichlet(*chuồng chim*)

<u>Đơn giản</u>
Cho ánh xạ f: X→Y.
Nếu $|X|>|Y|$ *thì tồn tại hai phần tử phân biệt* $x_1, x_2\in X$ *sao cho* $f(x_1)\equiv f(x_2)$.

<u>Tổng quát</u>
Cho ánh xạ f: X→Y.
Đặt $m=|X|$, $n=|Y|$ và $k=\lceil m/n\rceil$.
Tồn tại k *phần tử* $x_1, x_2, ..., x_k\in X$ *sao cho* $f(x_1) \equiv f(x_2) \equiv ... \equiv f(x_k)$.

Ví dụ. Chứng tỏ rằng trong *n+1* số nguyên dương có giá trị không quá *2n* thì có một số là bội của số khác.

Giải.
Gọi *n+1* số nguyên dương $a_1, a_2, \dots, a_n, a_{n+1}$.
$\forall$ i=*1..n+1* đặt $a_i=2^{k_i}q_i$, với q_i lẻ ,$1\leq q_i\leq 2n$. {*1..2n*} có *n* số lẻ. Theo nguyên lý Dirichlet có $i\neq j$ sao cho $q_i =q_j$. Nếu $k_i\geq k_j$ thì a_i là bội a_j; ngược lại a_j là bội a_i.

Ví dụ. Chứng tỏ rằng số hữu tỉ là một số thập phân vô hạn tuần hoàn.

Giải

Gọi $x=\frac{p}{q}$ là số hữu tỉ với $p\in Z$ và $q\in Z^+$.

Gọi $r_1, r_2, \dots$ là dãy số dư khi chia p cho q để lấy dãy chữ số thập phân $d_1, d_2, \dots$.
Có $\forall$ i: $0\leq r_i\leq q-1$. Xét q+1 số $r_1, r_2, \dots r_q, r_{q+1}$. Theo nguyên lý Dirichlet có i<j sao cho $r_i=r_j$. Vậy $r_{i+1}, r_{i+2}, \dots, r_j$ là đoạn tuần hoàn trong dãy số dư $r_1, r_2, \dots$ Tương ứng có dãy chữ số thập phân $d_1, d_2, \dots$. cũng tuần hoàn. Hay số hữu tỉ là một số thập phân vô hạn tuần hoàn.

Ví dụ. Một bữa tiệc có *n* người. Chứng tỏ rằng có hai người cùng số người quen.

Giải.
Gọi $q_1, q_2, ..., q_n$ là số người quen của người *1, 2, ..., n* tương ứng.
Có $\forall$ i =1..n: $0\leq q_i\leq n-1$.

Tuy nhiên, không thể xảy ra đồng thời: có một người không quen ai cả ($q_i=0$)và có một người quen tất cả mọi người ($q_j=n-1$). Vậy chỉ xảy ra một trong hai trường hợp:

- $\forall$ i =1..n: $0\leq q_i\leq n-2$.
- $\forall$ i =1..n: $1\leq q_i\leq n-1$.

Cả hai trường hợp *n* số q_i nhận *n-1* giá trị. Theo nguyên lý Dirichlet có $i\neq j$ sao cho $q_i=q_j$. Hay người *i* và người *j* có cùng số người quen.

Ví dụ. Cho tam giác đều ABC độ dài cạnh bằng 2. Phải chọn trong tam giác ABC ít nhất bao nhiêu điểm để đảm bảo có hai điểm cách nhau không quá 1.

Giải.
Chia tam giác đều ABC thành 4 tam giác đều cạnh 1 bởi trung điểm các cạnh. Theo nguyên lý Dirichlet nếu có 5 điểm trong tam giác đều ABC phải có 2 điểm trong tam giác đều cạnh 1 và chúng cách nhau không quá 1. Với 4 điểm O(tâm), A, B, C thì không đảm bảo, vì: AB = AC = BC = 2>1 và OA = OB = OC = $\frac{2}{3}\frac{\sqrt{3}}{2}2=\frac{2}{\sqrt{3}}$ >1.
Vậy 5 điểm là ít nhất.

Ví dụ. Một võ sĩ thi đấu liên tục trong 10 ngày, mỗi ngày ít nhất 1 trận và tổng số không quá 15 trận. Chứng tỏ rằng có những ngày liên tiếp võ sĩ đã thi đấu đúng 4 trận.

Giải.
Gọi a_i là số trận đấu cho đến hết ngày thứ *i* (*i=1..10*).
Có: $1\leq a_1<a_2< \dots <a_9<a_{10}\leq 15$.
Nên: $5\leq a_1+4<a_2+4< \dots <a_9+4<a_{10}+4\leq 19$.
20 số trong hai dãy nhận giá trị trong {*1..19*}. Theo nguyên lý Dirichlet phải có hai số bằng nhau. Do cả hai dãy đều là dãy tăng ngặt, nên hai số bằng nhau phải thuộc hai dãy khác nhau. Hay, có $a_i+4 = a_j$. Nghĩa là $a_j - a_i = 4$: từ ngày *i+1* đến hết ngày *j* võ sĩ đã đấu đúng *4* trận.

1.2. Các cấu hình tổ hợp cơ bản

1.2.1. Chỉnh hợp lặp

Cho tập X *gồm* **n** *phần tử,* |X|=n. Chỉnh hợp lặp chập **k** của X *là một bộ có thứ tự gồm* **k** *phần tử của* X*, trong đó các phần tử có thể lặp.*

Gọi $S=s_1s_2..s_k$ là một chỉnh hợp lặp X chập *k*.
$\forall$i=1..k, s_i có *n* cách chọn.
Theo ngyên lý nhân, số chỉnh hợp lặp X chập *k* là

$$F_n^k=n^k$$

Với bài toán đếm có thể gọi số chỉnh hợp lặp X chập *k* là ***n chập k***.

1.2.2. Chỉnh hợp

Cho tập X *gồm* **n** *phần tử,* |X|=n. Chỉnh hợp (không lặp) chập **k** của X *là một bộ có thứ tự gồm* **k** *phần tử của* X*, trong đó các phần tử không được lặp.*

Gọi $S=s_1s_2..s_k$ là một chỉnh hợp X chập *k*.
s_1 có *n* cách chọn.

Sau khi có s_1 thì s_2 có *n-1* cách chọn.
Sau khi có s_1s_2 thì s_3 có *n-2* cách chọn.
…
Sau khi có $s_1s_2…s_{k-1}$ thì s_k có *n-k+1* cách chọn.
Theo ngyên lý nhân, số chỉnh hợp không lặp *n* chập *k* là

$$A_n^k = n(n-1)\dots(n-k+1)$$

$$A_n^k = \frac{n!}{(n-k)!}$$

Lưu ý k≤n, theo nguyên lý chuồng chim.

1.2.3. Hoán vị

Cho tập X *gồm* **n** *phần tử,* |X|=n. Hoán vị của X *là một cách sắp xếp* **n** *phần tử của X.*

Mỗi hoán vị của *n* phần tử là một chỉnh hợp không lặp *n* chập *n*.
Vậy, số hoán vị của *n* phần tử là

$$P_n = A_n^n$$

$$P_n = n!$$

1.2.4. Tổ hợp

Cho tập X *gồm* **n** *phần tử,* |X|=n. Tổ hợp chập ***k*** của X *là một tập con gồm* k *phần tử của* X. *Nói cách khác, là một bộ không lặp và không thứ tự gồm* k *phần tử của* X.

Mỗi tổ hợp *n* chập *k* phát sinh được $p_k=k!$ hoán vị của nó là các chỉnh hợp không lặp *n* chập *k*; ngược lại, chỉnh hợp không lặp *n* chập *k* bỏ đi thứ tự là tổ hợp *n* chập *k.*
Vậy, số tổ hợp *n* chập *k* là

$$C_n^k = A_n^k / k!$$

$$C_n^k = \frac{n!}{(n-k)!\,k!}$$

Có thể dùng ký hiệu C(n,k) hay $\binom{n}{k}$.

Ví dụ.

Cho ánh xạ f: X→ Y, |X|=k, |Y|=n. Đếm số ánh xạ f

a) Bất kỳ.
b) Đơn ánh.
c) Song ánh.

Giải

a) Bất kỳ. Bằng số chỉnh hợp lặp: n^k
b) Đơn ánh. Bằng số chỉnh hợp: $\frac{n!}{(n-k)!}$

c) Song ánh(n=k). Bằng số hoán vị: n!

1.2.4. Các công thức tổ hợp

a. $C_n^0 = C_n^n = 1$

b. $C_n^{n-k} = C_n^k$

c. $C_n^k = C_{n-1}^{k-1} + C_{n-1}^k$

d. Nhị thức Newton

$$(x+y)^n = \sum_{k=0}^{n} C_n^k x^{n-k} y^k$$

Ví dụ. Cho tập X, |X|=n.

a. Đếm số tập con của tập X, |X|=n.
b. Chứng tỏ tập con có số lẻ và số chẵn phần tử bằng nhau.

Giải.

a. Thay x = y = 1 vào nhị thức Newton có số tập con của tập X là 2^n.
Cách khác, gọi X={x_1, x_2, …, x_n}, mỗi tập con A của X đặt tương ứng với xâu bit b= $b_1b_2\ldots b_n$ với ý nghĩa: b_i=1 khi và chỉ khi $x_i \in$ A. Có 2^n xâu bit nên có 2^n tập con A của X.
b. Thay x = 1, y =-1 vào nhị thức Newton, có kết quả.

Ví dụ. Cho lưới $m_x n$. Đếm số đường đi từ góc dưới bên trái đến góc trên bên phải dọc theo các cạnh. Biết rằng hướng đi chỉ là qua phải hoặc lên trên.

Giải.

Mỗi đường đi là một bộ gồm m+n cạnh, trong đó có m cạnh lên trên và n cạnh qua phải. Mỗi đường đi ứng với một cách chọn m vị trí cho m cạnh lên trên trong m+n vị trí, tất nhiên n vị trí còn lại là qua phải. Vậy số đường đi bằng số tổ hợp m+n chập m và bằng

C(m+n, m)= $\frac{(m+n)!}{m!n!}$

Các ứng dụng của nguyên lý bù trừ tổng quát

Với nguyên lý bù trừ tổng quát: *Cho* **n** *tập* A_1, A_2, …, A_n.

$$\left|\bigcup_{k=1}^{n} A_i\right| = \sum_{k=1}^{n} (-1)^{k+1} N_k$$

trong đó $\mathbf{N_k}$ *là tổng phần tử của giao* **k** *tập trong* **n** *tập* A_1, A_2, …, A_n.

Tuy nhiên, có khi cần đếm số phần tử không thuộc tập nào ở trên cả.
Gọi U là tập vũ trụ hữu hạn, chứa tất cả các tập trên và chứa tất cả các phần tử không thuộc tập nào cả.

Đặt N=|U| và N* là số phần tử không thuộc tập nào cả.

Có N* $= N - |A_1 \cup A_2 \cup ... \cup A_k|$

$= N - N_1 + N_2 - N_3 + ... + (-1)^k N_k$

Ví dụ. Đếm số toàn ánh từ X vào Y, |X|=k, |Y|=n.

Giải.

Gọi U là tập tất cả các ánh xạ f: X→Y. Có $N=n^k$.

Giả sử $Y=\{y_1, y_2, \ldots, y_n\}$. ∀ *j=1..n*, gọi A_j là tập ánh xạ với y_j không có tạo ảnh.

∀ i = *1..n-1*, N_i là số ánh xạ với *i* phần tử không có tạo ảnh.

Cách lấy ra *i* phần tử để không có tạo ảnh là C(n, i) và các phần tử còn lại bất kỳ. Vậy N_i bằng C(n, i) nhân với số ánh xạ từ X vào tập con gồm *n-i* phần tử còn lại.

$N_i = C(n, i)(n-i)^k$

Theo nguyên lý bù trừ, số toàn ánh là

$$N^* = n^k + \sum_{i=1}^{n-1} (-1)^i (n-i)^k C_n^i$$

Các giá trị cụ thể:

a) k=4, n=2: $N^* = 2^4 - 1^4 C(2,1) = 16-2 = 14$

b) k=4, n=3: $N^* = 3^4 - 2^4 C(3,1) + 1^4 C(3,2) = 81 - 48 + 3 = 36$

c) k=4, n=4: $N^* = 4^4 - 3^4 C(4,1) + 2^4 C(4,2) - 1^4 C(4,3)$

$= 256 - 324 + 96 - 4 = 24$ (= 4!)

d) k=5, n=3: $N^* = 3^5 - 2^5 C(3,1) + 1^5 C(3,2) = 150$

Ví dụ. Có *n* người dự tiệc và để mũ nơi móc. Do cúp điện nên mọi người có thể lấy lại nhầm mũ của mình. Đếm số khả năng mà không một ai nhận đúng mũ của mình cả.

Giải.

Gọi U là tập tất cả các cách phân *n* mũ cho *n* người. Có N=*n!*

∀ j=1..n, gọi A_j là tập các hoán vị mà người *j* nhận đúng mũ.

∀ i=1..n, N_i là số hoán vị mà có i người nhận đúng mũ.

Số cách chọn ra *i* người là C(n,i); số cách phân mũ bất kỳ cho n-i người còn lại là (n-i)!

Hay $N_i = (n-i)!C(n,i)$

Vậy, theo nguyên lý bù trừ

$$N^* = n! - \sum_{i=1}^{n} (-1)^i (n-i)! C_n^i$$

$$N^* = n! + \sum_{i=1}^{n} (-1)^i \frac{n!}{i!}$$

$$N^* = n!\left(1 - \frac{1}{1!} + \frac{1}{2!} - \frac{1}{3!} + ... + (-1)^n \frac{1}{n!}\right)$$

Xác suất để không một ai nhận đúng mũ của mình cả là

N*/n! = $1 - \frac{1}{1!} + \frac{1}{2!} - \frac{1}{3!} + ... + (-1)^n \frac{1}{n!} \approx e^{-1}$

Ví dụ Giả sử $n=6s$, trong đó s là số nguyên dương. Gọi a_n là số các bộ có thứ tự (i, j, k) gồm ba số nguyên không âm khác nhau từng đôi ($i \geq 0$, $j \geq 0$, $k \geq 0$, $i \neq j$, $i \neq k$, $j \neq k$) thỏa mãn $i + j + k \leq n$. Tìm công thức dưới dạng hiện để tính giá trị của a_n.

Giải

Số nghiệm với i, j, k không âm là N=C(n+3,3).

Có 3 khả năng hai số trùng nhau là: i=j, j=k, k=i.

Xét i=j thì $0 \leq i \leq 3s$:

i = j = 3s => k = 0: có 1 nghiệm

i = j = 3s-1 => k = 0..2: có 3 nghiệm

………

i = j = 0 => k = 0..n: có (n+1)=(6s+1) nghiệm

i = j: có 1+3+…+(6s+1)=$(3s+1)^2$ nghiệm.

Vậy tổng số nghiệm cho 3 khả năng là $(3s+1)^2$

Trong đó số nghiệm i = j = k bị trừ 3 lần. Để chỉ loại 1 lần, phải cộng vào 2 lần.

Số nghiệm với i = j = k thì $0 \leq i \leq 2s$ có 2s+1 nghiệm.

Vậy,

- Số nghiệm với hai phần tử trùng nhau là $3(3s+1)^2$
- Số nghiệm với ba phần tử trùng nhau là 2(2s+1)

Nên số nghiệm theo đề bài là

$N^* = C(n+3,3) - 3(3s+1)^2 + 2(2s+1)$

1.3. Các cấu hình tổ hợp suy rộng

1.3.1. Hoán vị lặp

Cho **n** *phần tử với* **k** *loại: loại* **1** *có* $\mathbf{n_1}$ *phần tử, loại* **2** *có* $\mathbf{n_2}$ *phần tử, ..., loại* **k** *có* $\mathbf{n_k}$ *phần tử. Một cách sắp xếp* **n** *phần tử này gọi là một* hoán vị lặp.

Gọi $S = s_1s_2...s_n$ là một hoán vị lặp của *n* phần tử trên.

Có $C(n, n_1)$ cách chọn n_1 vị trí để đặt các phần tử loại *1*.

Sau khi đặt các phần tử loại *1*, có $C(n-n_1, n_2)$ cách chọn n_2 vị trí để đặt các phần tử loại *2*.

...

Sau khi đặt các phần tử với các loại *1, 2, ..., k-1* có $C(n - n_1 - n_2 ... - n_{k-1}, n_k)$ cách chọn n_k vị trí để đặt các phần tử loại *k*.

Theo nguyên lý nhân, số hoán vị lặp trên là

$P(n; n_1,n_2,...,n_k) = C(n, n_1)C(n-n_1, n_2)...C(n - n_1 -...- n_k-1, n_k)$

$$\mathbf{P(n; n_1,n_2,...,n_k)} = \frac{n!}{n_1!n_2!...n_k!}$$

<u>*Lưu ý*</u> Hoán vị không lặp là trường hợp đặc biệt của hoán vị lặp với mỗi loại một phần tử.

Ví dụ.

Tính số cách sắp xếp các xâu chữ

a) MISSISSIPPI. ĐS: $\frac{11!}{1!4!4!2!}$

b) SUCCESS. ĐS: $\frac{7!}{3!1!2!1!}$

1.3.2. Tổ hợp lặp

Cho **n** *loại phần tử, mỗi loại có không ít hơn* **k** *phần tử. Một cách chọn ra* **k** *phần tử (có thể lặp) từ* **n** *loại phần tử này gọi là một* tổ hợp lặp.
Nói các khác: *Cho* |X|=n*, tổ hợp lặp chập* **k** *của* X *là một bộ không thứ tự gồm* **k** *phần tử của* X, *trong đó các phần tử có thể lặp.*

Giả sử mỗi tổ hợp lặp S = $s_1s_2 \ldots s_k$ được sắp xếp như sau: đầu tiên là tất cả các phần tử loại *1*, đến các phần tử loại *2*, ..., cuối cùng là các phần tử loại *n*. Dùng *k* dấu "x" để biểu diễn cho *k* phần tửvà dùng *n-1* dấu"|" để ngăn cho *n* loại.

Vậy mỗi tổ hợp lặp tương ứng với một cách chọn *n-1* vị trí trong *k+n-1* vị trí để đặt *n-1* dấu "|". Có C(k+n-1, n-1) cách chọn. Nên số tổ hợp lặp trên là

C(k+n-1, n-1) = C(k+n-1, k) = (k+n-1)/ (n-1)! k!

$$C_{k+n-1}^{n-1} = C_{k+n-1}^{k} = \frac{(k+n-1)!}{(n-1)!\,k!}$$

Ví dụ. Đếm số cách mua 10 trái cây với 3 loại: *cam, quít, xoài.*

ĐS: C(10+2, 2)=C(12, 2)=66

Ví dụ. Đếm số nghiệm nguyên của phương trình
x + y + z = 12 với x≥0, y≥0, z≥0.

Giải.
Dùng 12 dấu "x" để biểu diễn cho 12 đơn vị và dùng 2 dấu "|" để ngăn các đơn vị cho x, y, z. Mỗi nghiệm ứng với một tổ hợp lặp 3 chập 12.
ĐS: C(12+2, 2)=C(14, 2)=91

Ví dụ. Đếm số nghiệm nguyên của phương trình
x+y+z = 12 với x≥1, y≥-2, z≥3.

Giải
Đặt x'=x-1, y'=y+2, z'=z-3
Tương đương
x'+y'+z' = 10 với x'≥0, y'≥0, z'≥0.
ĐS: C(10+2, 2) = C(12, 2) = 66

Ví dụ. Đếm số nghiệm nguyên của bất phương trình

x + y + z ≤ 12 với x≥0, y≥0, z≥0.

Giải
Đặt t=12-(x + y + z)≥0. Bất phương trình tương đương phương trình
x + y + z + t = 12 với x≥0, y≥0, z≥0, t≥0.
ĐS: C(12+3, 3)=C(15, 3)=455

Ví dụ. Đếm số nghiệm nguyên của phương trình
x+y+z=11 với 0≤ x≤3, 0≤ y≤4, 0≤ z≤6

Giải

Gọi U là tập tất cả các nghiệm không âm của phương trình. Có N=|U|=C(11+2,2)=78.

Gọi A_1 là tập nghiệm với $x\geq4$, $y\geq0$, $z\geq0$

A_2 là tập nghiệm với $x\geq0$, $y\geq5$, $z\geq0$

A_3 là tập nghiệm với $x\geq0$, $y\geq0$, $z\geq7$

Theo nguyên lý bù trừ, số nghiệm nguyên của phương trình là

$N^*=N - |A_1| - |A_2| - |A_3| + |A_1\cap A_2| + |A_1\cap A_3| + |A_2\cap A_3| - |A_1\cap A_2\cap A_3|$

Các giá trị ở vế phải tương ứng là:

$N = 78$, $|A_1| + |A_2| + |A_3| =79$

$|A_1\cap A_2| +|A_1\cap A_3| +|A_2\cap A_3| = 7$

$|A_1\cap A_2\cap A_3| = 0$. Vậy $N^* = 6$

6 nghiệm đó là: (1,4,6), (2,3,6), (2,4,5),(3,2,6),(3,3,5),(3,4,4).

****

Bài tập

Các nguyên lý đếm cơ bản

1. Trên giá sách có 6 quyển sách khác nhau tiếng Anh, 8 quyển sách khác nhau tiếng Pháp, và 10 quyển sách khác nhau tiếng Đức.
   a) Có bao nhiêu cách chọn 3 quyển sách cùng ngôn ngữ
   b) Có bao nhiêu cách chọn 3 quyển sách, mỗi thứ một quyển
   c) Có bao nhiêu cách chọn 2 quyển sách mà 2 ngôn ngữ
2. Đếm số n gồm 2 chữ số, nếu:
   a) n chẵn
   b) n lẻ
   c) n lẻ gồm 2 chữ số khác nhau
   d) n chẵn gồm 2 chữ số khác nhau
3. a) Mật khẩu máy tính gồm 1 chữ cái và 3 hoặc 4 chữ số. Tính số mật khẩu tối đa có thể có.
   b) Như trên nhưng không lặp chữ số
4. Đếm số byte:
   a) Bất kỳ
   b) Có đúng 2 bit 0.
   c) Có ít nhất 2 bit 0.
   d) Bắt đầu 00 và kết thúc 00
   e) Bắt đầu 11 và kết thúc không phải là 11
5. Cho hai tập A, B với |A|=k và |B|=n. Đếm số ánh xạ từ A vào B:
   a) Bất kỳ
   b) Đơn ánh
6. Trong một bữa tiệc có n cặp vợ chồng. Mỗi người bắt tay với một người khác không phải là vợ hay chồng mình. Tính số lần bắt tay
7. Đếm số cách sắp xếp các chữ cái A..E:
   a) Có chứa xâu con AB.
   b) A và B kề nhau.
   c) A và B không kề nhau.
8. Có bao nhiêu cách sắp n cô gái và n con trai ngồi vào một dãy 2n ghế nếu xen kẻ giới tính.
9. Có bao nhiêu số 1 byte bắt đầu bằng 3 bit 0 hoặc kết thúc bằng 2 bit 0
10. Có bao nhiêu số 1 byte có 4 bit 0 liền nhau hoặc 4 bit 1 liền nhau
11. Có bao nhiêu số 1 byte có 3 bit 0 liền nhau hoặc 4 bit 1 liền nhau
12. Gọi D_n là số hoán vị $S=s_1s_2..s_n$ của $X=\{1, 2, \ldots, n\}$ sao cho $s_i \neq i \;\; \forall i = 1..n$
    Tìm công thức tính D_n. D_n gọi là *số mất thứ tự*, hay là *số xáo trộn*.

Nguyên lý Dirichlet (bài toán tồn tại)

13. Trong mặt phẳng xOy lấy ngẫu nhiên 5 điểm toạ độ nguyên. Chứng tỏ rằng có ít nhất một trung điểm của các đoạn nối chúng có toạ độ nguyên.

14. Trong mặt phẳng cho 6 điểm phân biệt nối nhau từng đôi một bởi các đoạn thẳng màu *xanh* hoặc *đỏ*. Chứng tỏ rằng có 3 điểm nối nhau bởi các đoạn thẳng cùng màu.

15. Trong mặt phẳng cho 17 điểm phân biệt nối nhau từng đôi một bởi các đoạn thẳng màu *xanh*, hoặc *đỏ*, hoặc *vàng*. Chứng tỏ rằng có 3 điểm nối nhau bởi các đoạn thẳng cùng màu.

16. Chứng minh rằng trong 9 điểm toạ độ nguyên bất kỳ trong không gian Oxyz thí có ít nhất một trung điểm của các đoạn thẳng nối chúng có toạ độ nguyên.

17. Một thùng chứa 10 quả bóng màu xanh và 10 quả bóng màu đỏ. Phải lấy ngẫu nhiên ít nhất bao nhiêu quả bóng để đảm bảo có 3 quả bóng cùng màu.

18. Cho X={0..10}. Chứng tỏ rằng nếu S là một tập con gồm 7 phần tử của X thì có 2 phần tử của S có tổng bằng 10

19. Chứng tỏ rằng: Trong một mạng máy tính cục bộ phải có hai máy có cùng số máy nối trực tiếp với chúng.

20. Xét một trận đấu n người, mỗi người đấu với mỗi người khác và mỗi người thắng ít nhất một lần. Chứng tỏ rằng có ít nhất 2 người có cùng số lần thắng

21. Chứng tỏ rằng mọi dãy n^2+1 số thực phân biệt đều có chứa một dãy con ít nhất n+1 số hạng hoặc tăng hoặc giảm.

22. Cho S là tập gồm n+1 phần tử trong 2n số tự nhiên đầu tiên. Chứng tỏ rằng S chứa 2 số nguyên mà một số là bội của số kia.

23. Chứng tỏ rằng trong n+1 số nguyên có ít nhất hai số đồng dư n

24. Cần ít nhất bao nhiêu cặp số nguyên (a, b) để chắc chắn có hai cặp (a_1, b_1) và (a_2, b_2) mà $a_1 \equiv a_2$ (mod 5) và $b_1 \equiv b_2$ (mod 5)

25. Một võ sĩ quyền anh thi đấu giành chức vô địch trong 75 giờ. Mỗi giờ đấu ít nhất một trận, nhưng toàn bộ không quá 125 trận. Chứng tỏ rằng có những giờ liên tiếp đã đấu 24 trận

26. Chứng tỏ rằng trong dãy *n* số nguyên thì có một hay nhiều số liên tiếp có tổng chia hết *n*.

Các cấu hình tổ hợp

27. Có bao nhiêu cách sắp *n* người ngồi vào bàn tròn xoay.

28. Cho lưới chữ nhật $m_x n$. Đếm số cách đi từ góc dưới bên trái lên góc trên bên phải, biết rằng chỉ đi qua phải và lên trên dọc theo các cạnh.

29. Chứng minh:

a) $\sum_{r=0}^{n} C(n,r)=2^n$ b) $\sum_{r=0}^{n} (-1)^r C(n,r)=0$

c) $\sum_{r \text{ chan}}^{n} C(n,r))=\sum_{r \text{ le}}^{n} C(n,r) =2^{n-1}$

Các cấu hình tổ hợp suy rộng

30. Đếm số cách phân 30 sinh viên cho 5 thầy giáo hướng dẫn đồ án.

31. Có bao nhiêu cách đặt n vật vào m hộp sao cho không có hộp nào trống

32. Chứng tỏ rằng số chuỗi n-bit có đúng k bit 0 và không có 2 bit 0 kề nhau là C(n-k+1, k).

33. Có n người sắp thành hàng. Đếm số cách:
a) Chọn ra k người bất kỳ.
b) Chọn ra k người sao cho không có 2 người kề nhau được chọn.

34. Đếm số cách sắp xếp 9 viên bi giống nhau vào 3 túi A, B, C.

35. Đếm số cách sắp xếp 9 viên bi gồm 2 bi xanh, 3 bi đỏ và 4 bi vàng thành một hàng.

36. Đếm số cách sắp xếp 9 viên bi gồm 2 bi xanh, 3 bi đỏ và 4 bi vàng vào 3 túi A, B, C.

37. Đếm số nghiệm nguyên của:
a) $x + y + z = 10$ với $x\geq 0$, $y\geq 0$, $z\geq 0$.
b) $x + y \leq 10$ với $x\geq 0$, $y\geq 0$.
c) $x + y + z = 13$ với $x\geq 0$, $y\geq 1$, $z\geq 2$.
d) $x + y + z = 10$ với $0\leq x\leq 3$, $0\leq y\leq 4$, $0\leq z\leq 5$.

38. Đếm từ gồm các chữ cái trong từ sau, yêu cầu phải dùng tất cả các chữ cái:
a) COMPUTER
b) SUCCESS
c) MISSISSIPPI

****

Chương 2. Kỹ thuật đếm nâng cao

Với các bài toán đếm, nếu không thể dùng các nguyên lý cơ bản để đếm, khi đó lập hệ thức truy hồi rồi giải.

2.1. Hệ thức truy hồi

Hệ thức truy hồi bậc **k** *của dãy số* $\{a_n\}$ *là công thức tính* a_n *qua* **k** *phần tử trước nó.*
Dạng: $a_n = f(a_{n-1}, a_{n-2}, \ldots, a_{n-k})$
Hệ thức truy hồi tuyến tính thuần nhất bậc **k** hệ số hằng *có dạng:*

$$a_n = c_1a_{n-1} + c_2a_{n-2} + \ldots + c_ka_{n-k} \quad (*)$$

trong đó $c_k \neq 0$.

Lưu ý rằng *với một hệ thức truy hồi bậc* **k**, *nếu có* **k** *giá trị đầu:*
$a_0 = I_0, a_1 = I_1, \ldots, a_{k-1} = I_{k-1}$
thì xác định duy nhất một dãy $\{a_n\}$

Ví dụ

a) Đếm số lần chuyển đĩa của bài toán tháp Hanoi (tháp Brahma: Bà-la-môn).
b) Đếm số xâu nhị phân độ dài *n* không có hai bit 0 kề nhau.

Giải

a) Gọi C_n là số lần chuyển đĩa. Có hệ thức truy hồi: $C_n = 2C_{n-1} +1$. Đây là hệ thức truy hồi bậc *1* nên cần *1* giá trị đầu là: $C_1=1$. Giải sau.
b) Gọi a_n là số xâu nhị phân độ dài *n* không có hai bit 0 kề nhau.
   Gọi $b = b_1b_2\ldots b_n$ là xâu nhị phân độ dài *n* không có hai bit 0 kề nhau.
   Xét 2 trường hợp:
   - Nếu $b_n=1$ thì số xâu b bằng số xâu $b_1b_2\ldots b_{n-1}$ không có hai bit 0 kề nhau và bằng a_{n-1}.
   - Nếu $b_n=0$ thì phải có $b_{n-1}=1$ và số xâu b bằng số xâu $b_1b_2\ldots b_{n-2}$ không có hai bit 0 kề nhau và bằng a_{n-2}.

   Theo nguyên lý cộng, có hệ thức truy hồi: $a_n = a_{n-1} + a_{n-2}$. Đây là hệ thức truy hồi bậc 2 nên cần 2 giá trị đầu là: $a_1=2$ và $a_2=3$. Giải sau.

2.2. Giải hệ thức truy hồi

Giải HTTH là tìm một công thức hiện(tường minh) cho số hạng tổng quát a_n mà không phải tính qua k phần tử trước nó. Không có phương pháp chung để giải hệ thức truy hồi tổng quát. Hai phương pháp sau để giải hệ thức truy hồi dạng đơn giản.

- **Phương pháp thế**
- **Phương pháp phương trình đặc trưng**

2.2.1. Phương pháp thế

Phương pháp thế để giải hệ thức truy hồi bậc 1 bằng cách thay a_n bởi a_{n-1}, a_{n-1} bởi a_{n-2}, …, cho đến khi gặp giá trị đầu $a_0=I_0$ thì có được một công thức rõ ràng cho a_n.

Ví dụ. Gọi C_n là số lần chuyển *n* đĩa của bài toán tháp Hanoi. Có hệ thức truy hồi
$C_n = 2C^{n-1} + 1$ *và* $C_1=1$.

$C_n = 2C_{n-1} + 1$
$C_n = 2^2C_{n-2}+2+1$
$C_n = 2^3C_{n-3}+2^2+2+1$
...
$= 2^{n-1}C_1+2^{n-2}+...+2^2+2+1$
$= 1+2+2^2+...+2^{n-2}+2^{n-1}= 2^n-1$ *(tổng cấp số nhân)*

Vậy $C_n = 2^n-1$

Tuy nhiên, cần phải kiểm tra lại kết quả bằng cách dùng nguyên lý quy nạp. Với giới hạn của giáo trình, có thể công nhận một cách tự nhiên.

2.2.2. Phương pháp phương trình đặc trưng

Phương pháp phương trình đặc trưng để giải hệ thức truy hồi *bậc 2 tuyến tính thuần nhất hệ số hằng*

$a_n = c_1a_{n-1} + c_2a_{n-2}$ (1) với $a_0=I_0$, $a_1=I_1$.

trong đó $c_2 \neq 0$.

Với phương trình đặc trưng

$x^2 = c_1x + c_2$ (2)

Định lý 1.

Nếu α_1 và α_2 là hai nghiệm phân biệt của (2) thì tồn tại duy nhất các hằng *b* và *d* để

$$a_n = b\alpha_1^n + d\alpha_2^n$$

Ví dụ

Giải hệ thức truy hồi

$a_n=5a_{n-1}-6a_{n-2}$ với $a_0 = 0$, $a_1 = 1$.

Giải

Phương trình đặc trưng: $x^2 = 5x - 6$ có 2 nghiệm phân biệt $x_1=3$, $x_2=2$.

$a_n = b3^n + d2^n$

$a_0 = 0$, $a_1 = 1$ suy ra b=1 và d=-1. Vậy

$a_n = 3^n - 2^n$

Ví dụ

Giả sử mỗi cặp thỏ sau 2 tháng tuổi sinh liên tục mỗi tháng một cặp. Thả lên đảo hoang một cặp thỏ mới sinh và thỏ không chết. Tính số cặp thỏ sau *n* tháng.

Giải

Gọi F_n là số cặp thỏ sau *n* tháng. F_n bằng số cặp thỏ tháng trước (F_{n-1}) cộng với số cặp thỏ mới sinh. Mà số cặp thỏ mới sinh bằng số cặp thỏ đã có cách đó 2 tháng (F_{n-2}). Có hệ thức truy hồi:

$F_n=F_{n-1}+F_{n-2}$ với $F_0=F_1=1$.

Phương trình đặc trưng $x^2=x+1$ có hai nghiệm phân biệt $\alpha_1=\dfrac{1+\sqrt{5}}{2}$ và $\alpha_2=\dfrac{1-\sqrt{5}}{2}$.

$$F_n= b\left[\frac{1+\sqrt{5}}{2}\right]^n +d\left[\frac{1-\sqrt{5}}{2}\right]^n$$

$F_0=F_1=1 \Leftrightarrow b+d=1$ và $b\left[\frac{1+\sqrt{5}}{2}\right]+d\left[\frac{1-\sqrt{5}}{2}\right]=1$

$\Leftrightarrow b=\frac{1}{\sqrt{5}}\left[\frac{1+\sqrt{5}}{2}\right]$ và $d=-\frac{1}{\sqrt{5}}\left[\frac{1-\sqrt{5}}{2}\right]$

Vậy $F_n=\frac{1}{\sqrt{5}}\left[\frac{1+\sqrt{5}}{2}\right]^{n+1}-\frac{1}{\sqrt{5}}\left[\frac{1-\sqrt{5}}{2}\right]^{n+1}$

Ví dụ.

Cho tập chữ số X={0, 1} và tập chữ cái Y={A, B, C }. Đếm số xâu ký tự *n* gồm các chữ cái trong $X \cup Y$ ={0, 1, A, B, C } sao cho không chứa hai chữ số liền nhau.

Giải. Gọi $S=s_1s_2\ldots s_{n-1}s_n$ là xâu chữ trên và a_n là số xâu S.

Xét 2 trường hợp:

a. $s_n \in Y$: có 3 khả năng xảy ra cho s_n và số xâu S cho mỗi khả năng bằng số xâu $s_1s_2\ldots s_{n-1}$ không chứa hai chữ số liền nhau và bằng a_{n-1}. Trường hợp này, theo nguyên lý nhân, số xâu S bằng $3a_{n-1}$.

b. $s_n \in X$: có 2 khả năng xảy ra cho s_n và s_{n-1} $s_n \in Y$, có 3 khả năng xảy ra cho s_{n-1}. Theo nguyên lý nhân, có 2x3=6 khả năng xảy ra cho $s_{n-1}s_n$. Số xâu S cho mỗi khả năng bằng số xâu $s_1s_2\ldots s_{n-2}$ không chứa hai chữ số liền nhau và bằng a_{n-2}. Trường hợp này, theo nguyên lý nhân, số xâu S bằng $6a_{n-2}$.

Theo nguyên lý cộng, có hệ thức truy hồi cho a_n là:

$a_n = 3a_{n-1} + 6a_{n-2}$.

Đây là hệ thức truy hồi bậc 2, có 2 giá trị đầu là $a_1 = 5$, $a_2 = 21$.

Phương trình đặc trưng $x^2=3x+6$ có hai nghiệm phân biệt $\alpha_1=\frac{3+\sqrt{33}}{2}$ và $\alpha_2=\frac{3-\sqrt{33}}{2}$.

$a_n= b\left[\frac{3+\sqrt{33}}{2}\right]^n+d\left[\frac{3-\sqrt{33}}{2}\right]^n$

Có thể mở rộng đến a_0.

$a_2 = 3a_1 + 6a_0 \Rightarrow a0 =1$.

Dùng 2 giá trị đầu là $a_0 = 1, a_1 = 5$.

$a_0 = 1$, $a_1= 5 \Rightarrow b+d=1$ và $b\left[\frac{3+\sqrt{33}}{2}\right]+d\left[\frac{3-\sqrt{33}}{2}\right]=5$

$\Rightarrow b=\frac{1}{\sqrt{33}}\left[\frac{7+\sqrt{33}}{2}\right]$ và $d=-\frac{1}{\sqrt{33}}\left[\frac{7-\sqrt{33}}{2}\right]$

$$\text{Vậy } a_n = \frac{1}{\sqrt{33}}\left[\frac{7+\sqrt{33}}{2}\right]\left[\frac{3+\sqrt{33}}{2}\right]^n - \frac{1}{\sqrt{33}}\left[\frac{7-\sqrt{33}}{2}\right]\left[\frac{3-\sqrt{33}}{2}\right]^n$$

Ví dụ. Cho hàm

```
int A(int n)
{
    if (n<2) return n+1;
    else return 5*A(n-1)-6*A(n-2);
}
```

Đếm số lệnh *return* khi gọi *A(n)*.

Giải.

Gọi a_n là số *return* khi gọi lệnh *A(n)*.

Có hệ thức truy hồi

$a_n=a_{n-1}+a_{n-2}$ với a0=a1=1.

Trùng với dãy Fibonacci.

Vậy $a_n= \frac{1}{\sqrt{5}}\left[\frac{1+\sqrt{5}}{2}\right]^{n+1} - \frac{1}{\sqrt{5}}\left[\frac{1-\sqrt{5}}{2}\right]^{n+1}$

Định lý 2.

Nếu α nghiệm kép của (2) thì tồn tại duy nhất các hằng *b* và *d* để

$$a_n = b\alpha^n + dn\alpha^n$$

Ví dụ.

Giải hệ thức truy hồi

$a_n=4a_{n-1}-4a_{n-2}$ với $a_0 = 1$, $a_1 = 3$.

Giải.

Phương trình đặc trưng: $x^2 = 4x - 4$ có nghiệm kép x=2.

$$a_n = b2^n + dn2^n$$

$a_0 = 1$ và $a_1 = 3$ suy ra b=1 và d=$\frac{1}{2}$. Vậy

$$a_n = 2^n + n2^{n-1}$$

****

Bài tập

Lập hệ thức truy hồi

1. Tìm hệ thức truy hồi và điều kiện đầu để tính số chuỗi nhị phân độ dài n không có 3 bit 0 liên tiếp.

2. Tìm hệ thức truy hồi và điều kiện đầu để tính số cách đi lên n bậc thang nếu có thể đi 1, 2, hoặc 3 bước một lần

3. Tìm hệ thức truy hồi mà R_n thoả mãn, trong đó R_n là số miền của mặt phẳng bị phân chia bởi n đường thẳng nếu không có hai đường nào song song và không có 3 đường nào cùng đi qua một điểm.

4. Tìm hệ thức truy hồi và điều kiện đầu để tính số chuỗi ký tự gồm A, B, C có độ dài n chứa hai ký tự liên tiếp giống nhau

5. Cho dãy S = $\{S_n\}$, S_n là số chuỗi *n* bit không chứa hai bit 0 liên tiếp.
a) Tìm hệ thức truy hồi và các điều kiện đầu cho dãy $\{S_n\}$
b) Chứng tỏ rằng $S_n = f_{n+1}$, n=1,2,... với *f* là dãy Fibonacci.

6. Tìm hệ thức truy hồi và điều kiện đầu cho số cách đóng cặp ngoặc đơn trong biểu thức $a_1*a_2*...*a_n$ với $n \geq 2$.

Giải hệ thức truy hồi

7. Gọi a_n là số các chữ số 0 có trong các số tự nhiên từ 0 đến 10^n - 1.
a) Chứng tỏ rằng a_n thoả mãn hệ thức truy hồi $a_n=a_{n-1}+9(n-1)10^{n-2}$.
b) Biết giá trị đầu $a_1=1$, giải hệ thức truy hồi trên.

8. Gọi a_n là số dãy bit độ dài n không có 2 bit 0 kề nhau.
a) Tìm hệ thức truy hồi cho a_n.
b) Biết giá trị đầu $a_1=2$ và $a_2=3$, giải hệ thức truy hồi trên.

9. Giả sử dân số thế giới năm 2011 là 8 tỉ người và tốc độ tăng dân số là 0,2% mỗi năm.
a) Lập hệ thức truy hồi cho dân số thế giới n năm sau năm 2011.
b) Giải hệ thức truy hồi cho dân số thế giới n năm sau năm 2011.
c) Dân số thế giới năm 2020 là bao nhiêu ?

10. Giả sử lãi suất tiền gửi ngân hàng là 2% một năm. Tính tổng số tiền có trong tài khoản sau 10 năm, nếu tiền gửi là 10 triệu.

11. Cho dãy số $\{a_n\}$ thoả mãn hệ thức truy hồi:

$a_n= 5a_{n-1}- 6a_{n-2}$; $a_0=0$ và $a_1=1$.

a) Giải hệ thức truy hồi trên.
b) Viết hàm A(n) để tính a_n.

12. Cho dãy số $\{a_n\}$ thoả mãn hệ thức truy hồi:

$a_n= 6a_{n-1}- 9a_{n-2}$; $a_0=1$ và $a_1=3$.

a) Giải hệ thức truy hồi trên.
b) Viết hàm A(n) để tính a_n.

****

Chương 3. Bài toán liệt kê

3.1. Phương pháp quay lui (đệ quy)

3.1.1. Nguyên lý chung

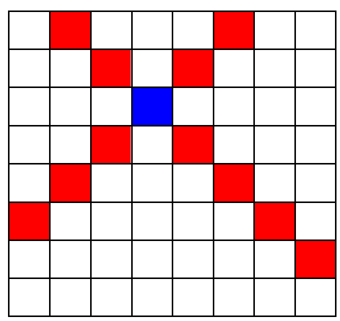
Để liệt kê tất cả các cấu hình $S=s_1s_2..s_k$ thuật toán giả sử đã có cấu hình con $s_1s_2\ldots s_{i-1}$. Ở bước thử tìm giá trị cho s_i , *Try(i)*, duyệt qua mọi giá trị *j* đề cử được cho s_i và thực hiện 4 bước sau**:**

a) s_i=j;
b) <*Thay đổi trạng thái*>;
c) Nếu đủ cấu hình (i=k) thì Print(S) ngược lại, gọi đệ quy để thử cho s_{i+1}, *Try(i+1)*;
d) <*Trả lại trạng thái cũ*>;

Chương trình chính gọi *Try(1).*

Lưu ý *Một số cấu hình không có hai bước* b) *và* d)

Bài toán 8 hậu hòa bình (Niklaus Wirth)



Các khai báo:

- Bàn cờ: `int S[8];`

ý nghĩa: S[i] = j ⇔ hàng i đặt hậu tại ô (i, j).

- Các mảng cần đánh dấu:

```
int a[8], b[15], c[15];
```

a[j] = TRUE ⇔ cột j còn trống.
c[i-j+7] = TRUE ⇔ đường chéo song song chéo chính qua ô (i, j) còn trống.
c[i+j] = TRUE ⇔ đường chéo song song chéo phụ qua ô (i, j) còn trống.

Trước khi gọi Try(0) cần lấp đầy các mảng a,b,c giá trị 1 (TRUE).
Chương trình được cài đặt như sau.

//8 hau hoa binh

```
#include <stdio.h>
#include <conio.h>

int S[8], a[8], b[15], c[15], sol=0;

void Try(int), print();

void main(){
  int i;
  clrscr();
  for (i=0; i<8; i++) a[i]=1;
  for (i=0; i<15; i++) b[i]=1;
  for (i=0; i<15; i++) c[i]=1;
  Try(0);
  getchar();
}
void Try(int i)
{
  int j;
  for (j=0; j<8; j++) if (a[j]&&b[i-j+7]&&c[i+j]){
    S[i]=j;
    a[j]=0; b[i-j+7]=0; c[i+j]=0;
    if (i==7) print(); else Try(i+1);
    a[j]=1; b[i-j+7]=1; c[i+j]=1;
  }
}

void print()
{
  int i;
  printf("\n%3d:",++sol);
  for (i=0; i<8; i++) printf("%d ", S[i]);
}
```

Trong việc mô tả các thuật toán liệt kê các cấu hình tổ hợp đơn giản, luôn giả sử tập nguồn X ={1..n}. Trong trường hợp cụ thể thì lấy các phần tử của X làm chỉ số của mảng. Để gần với lý thuyết, mặc dù C luôn đánh chỉ số mảng từ 0, chương trình chỉ in từ S[1] đến S[k].

3.1.2. Liệt kê chỉnh hợp lặp

Không có hai bước b) *và* d)

```
void Try(int i)
{
  int j;
  for (j=1; j<=n; j++){
    S[i]=j;
    if (i==k) print(); else Try(i+1);
  }
}
```

3.1.3. Liệt kê chỉnh hợp không lặp

Cần mảng ***a*** đánh dấu các giá trị *j* đã dùng, với ý nghĩa a[j] = TRUE ↔ j chưa dùng.

```
void Try(int i)
{
  int j;
  for (j=1; j<=n; j++) if (a[j]){
    S[i]=j;
    a[j]=0;
    if (i==k) print(); else Try(i+1);
    a[j]=1;
  }
}
```

3.1.4. Liệt kê tổ hợp

Tổ hợp là bộ không thứ tự. Liệt kê theo thứ tự tăng ($s_i<s_{i+1}$).
Giá trị j đề cử được cho s_i : $1+s_i \le j \le n-k+i$.
Chương trình chính gọi Try(1) nên mảng S có S[0]=0.

```
void Try(int i)
{
  int j;
  for (j=1+S[i-1]; j<=n-k+i; j++){
    S[i]=j;
    if (i==k) print(); else Try(i+1);
  }
}
```

3.2. Phương pháp lặp (sinh kế tiếp)

3.2.1. Nguyên lý chung

Phương pháp lặp liệt kê các cấu hình tổ hợp theo thứ tự từ điển.
Thứ tự từ điển được định nghĩa như sau:
Cho hai dãy $S = s_1s_2...s_k$ và $T = t_1t_2...t_n$. Gọi S nhỏ hơn T theo thứ tự từ điển, ký hiệu S<T, nếu thỏa mãn một trong hai điều kiện sau:

- $\forall i=1..k$: $s_i = t_i$ và $k<n$.
- $\exists s_i<t_i$ và $\forall j=1..i-1$: $s_j=t_j$.

Tuy nhiên, với các cấu hình tổ hợp cùng loại thì có độ dài bằng nhau (n=k) nên chỉ dùng điều kiện thứ hai.
Để liệt kê tất cả các cấu hình $S = s_1s_2...s_k$ theo thứ tự từ điển cần thực hiện các bước sau:
Bước 1 Tìm S nhỏ nhất; Print(S);
Bước 2 Lặp cho đến khi đủ số cấu hình:

- Tìm s_i bên phải nhất có thể tăng được.
- Tăng s_i ít nhất.
- Hạ cấu hình con $s_{i+1}s_{i+2}...s_k$ xuống nhỏ nhất.
- Print(S);

Cũng có thể thay bước 2 bởi bước sau: vừa kiểm tra lớn nhất vừa tìm s_i bên phải nhất có thể tăng được.

Bước 2' While (!stop){
```
i=k; while (i>0 && s_i ...s_k lớn nhất) i--;
if (i= =0) stop=1;
else{
  - Tăng s_i ít nhất.
  - Hạ cấu hình con s_{i+1}s_{i+2}...s_k xuống nhỏ nhất.
  - Print(S);
}
```

Bước Tìm S tiếp theo S hiện tại tùy theo từng cấu hình cụ thể mà thay đổi S như sau:

3.2.2. Liệt kê chỉnh hợp lặp

```
for (i=1; i<=k; i++) S[i]=1; print();
while (c<nk){
  i=k; while (S[i]==n) i--;
  S[i]++;
  for (j=i+1; j<=k; j++) S[j]=1;
  print();
}
```

3.2.3. Liệt kê tổ hợp

Tổ hợp là bộ không thứ tự. Liệt kê theo thứ tự tăng (s_i<s_{i+1}).
Giá trị lớn nhất cho s_i là n-k+i.

```
for (i=1; i<=k; i++) S[i]=i; print();
while (c<C(n,k)){
  i=k; while (S[i]==n-k+i) i--;
  S[i]++;
  for (j=i+1; j<=k; j++) S[j]=S[j-1]+1;
  print();
}
```

3.2.4. Liệt kê hoán vị

Trong các hoán vị thì nhỏ nhất theo thứ tự tăng và lớn nhất theo thứ tự giảm
Ví du: n=5. Nhỏ nhất 1 2 3 4 5 và lớn nhất 5 4 3 2 1
Giả sử hiện thời S = 32541 thì phần tử tăng được là s_i = 2, vì 541 lớn nhất. Tăng s_i bằng cách đổi giá trị cho s_j nhỏ nhất lớn hơn ở bên phải là s_j=4. Sau khi hoán đổi có S=34521 thì 521 vẫn lớn nhất, để trở thành nhỏ nhất thì đảo thứ tự của chúng. Vậy hoán vị kế tiếp S=32541 là S=34125.

```
for (i=1; i<=n; i++) S[i]=i; print();
while (c<n!){
      i=n-1; while (S[i]>S[i+1]) i--;
      j=n; while (S[j]<S[i]) j--;
      tam=S[i]; S[i]=S[j]; S[j]=tam;
```

```
        j=i+1; k=n;
    while (j<k){ tam=S[j]; S[j]=S[k]; S[k]=tam; j++; k--;}
        print();
  }
```

3.2.5. Liệt kê chỉnh hợp không lặp

Đưa thuật toán phát sinh hoán vị vào mỗi bước sinh tổ hợp có được thuật toán sinh chỉnh hợp không lặp.
Chương trình được cài đặt như sau.

```
#include <stdio.h>
#include <conio.h>

#define MAX 20
int S[MAX],A[MAX], c=0, cc=0, dem=0,n, k; //bo qua S[0].

void print();
int Com(int, int), fact(int k);

void main(){
  int i,j,ii,jj,kk,tam;

  clrscr();
  printf("nhap n, k:"); scanf("%d%d%*c", &n,&k);

  for (i=1; i<=k; i++) S[i]=i; c++;
    cc=0;
    for (ii=1; ii<=k; ii++) A[ii]=ii;
    cc++; print();
    while (cc<fact(k)){
      ii=k-1; while (A[ii]>A[ii+1]) ii--;
      jj=k; while (A[jj]<A[ii]) jj--;
      tam=A[ii]; A[ii]=A[jj]; A[jj]=tam;
      jj=ii+1; kk=k;
      while (jj<kk){ tam=A[jj]; A[jj]=A[kk]; A[kk]=tam; jj++; kk--;}
      cc++; print();
    }
  while (c<Com(n,k)){
    i=k; while (S[i]==n-k+i) i--;
    S[i]++;
    for (j=i+1; j<=k; j++) S[j]=S[j-1]+1; c++;
    // Phát sinh k! hoán vị của tổ hợp hiện tại
    cc=0;
    for (ii=1; ii<=k; ii++) A[ii]=ii;
    cc++; print();
    while (cc<fact(k)){
      ii=k-1; while (A[ii]>A[ii+1]) ii--;
      jj=k; while (A[jj]<A[ii]) jj--;
      tam=A[ii]; A[ii]=A[jj]; A[jj]=tam;
      jj=ii+1; kk=k;
```

```
      while (jj<kk){ tam=A[jj]; A[jj]=A[kk]; A[kk]=tam; jj++; kk--;}
      cc++; print();
    }
  }
  getchar();
}

void print()
{
  int i;
  printf("\n%3d:",++dem);
  for (i=1; i<=k; i++) printf("%d ", S[A[i]]);
}

int Com(int n, int k)
{
  if (k==0||k==n) return 1; else return Com(n-1,k-1)+Com(n-1,k);
}

int fact(int k)
{
  if (k==0) return 1; else return k*fact(k-1);
}
```

Cũng có thể , không dùng bài toán đếm, kết hợp vừa kiểm tra dừng vừa tìm s_i bên phải nhất có thể tăng được. Chẳng hạn,

với chỉnh hợp không lặp:
```
i=k; while (i>0 && S[i]==n-k+i)i--;
if (i==0) dừng; else S[i]tăng được…
```
với hoán vị:
```
i=n-1; while (i>0 && S[i]>S[i+1])i--;
if (i==0) dừng; else S[i] tăng được…
```

****

Bài tập

Phương pháp quay lui

13. Viết chương trình sinh tất cả các hoán vị của tập X={1..n}.

14. Viết chương trình sinh tất cả các chỉnh hợp lặp chập k của tập X={1..n}.

15. Viết chương trình sinh tất cả các chỉnh hợp chập k của tập X={1..n}.

16. Viết chương trình phát sinh tất cả các tổ hợp chập k của tập X={1..n}.

17. Viết chương trình phát sinh tất cả các tổ hợp chập k của tập X gồm *n* phần tử bằng cách nhập phần tử s $\in$X, chia bài toán thành 2 bài toán con:
a) Liệt kê các tổ hợp chứa s.
b) Liệt kê các tổ hợp không chứa s.

18. Dùng thuật toán phát sinh cấu hình tổ hợp để viết chương trình liệt kê tất cả các số từ 1 đến 1000000 có tổng các chữ số bằng 20.

19. Viết chương trình liệt kê tất cả các nghiệm nguyên không âm của phương trình $x_1+x_2+\ldots+x_k = n$. Với n, k nhập từ bàn phím.

Phương pháp lặp

Viết lại các chương trình trên bằng phương pháp lặp.

****

Chương 4. Đại số Boole

4.1. Đại số Boole

Đại số Boole là một hệ thống <B; +, . , $\overline{}$, 0, 1> , trong đó B là một tập hợp, thỏa mãn các tiên đề sau:

1. Luật kết hợp (*Associative*)
   $\forall$ **x, y, z $\in$ B : (x + y) + z = x + (y + z)** và **(x.y).z = x.(y.z)**
2. Luật giao hoán (*Commutative*)
   $\forall$ **x, y $\in$ B : x + y = y + x** và **x.y = y.x**
3. Luật đồng nhất (*Identity*)
   $\forall$ **x $\in$ B : x + 0 = x** và **x.1 = x**
4. Luật phân phối (*distributive*)
   x.(y + z) = (x.y) + (x.z) và **x+(y.z) = (x+y).(x+z)**
5. Luật bù (*Inverse*/*Complement*)
   $\forall$ **x $\in$ B,** $\exists$ $\overline{x}$ $\in$ **B : x +** $\overline{x}$ **= 1** và **x .** $\overline{x}$ **= 0**
   Phần tử $\overline{x}$ gọi là phần tử bù của x.

Suy ra các luật sau

6. *Luật lũy đẳng* (*Idempotent*)
   $\forall$ **x $\in$ B : x + x = x** và **x.x = x**
7. *Luật nuốt* (giới nội-*Boundedness*)
   $\forall$ **x $\in$ B : 0.x = 0** và **1 + x = 1.**
8. *Luật hấp thụ* (Absorption)
   $\forall$ **x, y $\in$ B :** **x + xy = x** và **x(x + y) = x.**
   $x + \overline{x}y = x + y$ và $x(\overline{x} + y) = xy$.
9. *Luật bù kép* (*Involution*)
   $\forall$ **x $\in$ B:** $\overline{\overline{x}} = x$.
10. *Luật De Morgan*
   $\forall$ **x, y $\in$ B :** $\overline{x+y} = \overline{x}\,\overline{y}$ **và** $\overline{xy} = \overline{x} + \overline{y}$.
11. *Luật 0-1*
   $\overline{0} = 1$ **và** $\overline{1} = 0$

4.2. Các khái niệm

Cho B={0,1}.

Biến Boole là biến chỉ nhận các giá trị trong B.

Tục biến (*Literal*) là một biến hoặc bù của một biến, như x, $\overline{x}$, y, $\overline{y}$ ...

***Biểu thức Boole*:** Cho *n* biến Boole $x_1, x_2, \ldots, x_n$. Biểu thức Boole được định nghĩa đệ quy như sau:

a) $x_1, x_2, \ldots, x_n$, 0, 1 là các biểu thức Boole.
b) Nếu E_1, E_2 là biểu thức Boole thì E_1+E_2, $E_1.E_2$, $\overline{E_1}$ cũng là Biểu thức Boole.

Thường ký hiệu $E=E(x_1, x_2, \ldots, x_n)$.

***Hàm Boole bậc* n** là hàm f: $B^n \rightarrow B$. Mỗi hàm Boole có thể biểu diễn bằng một biểu thức.

Hội sơ cấp (*Fundamental Product*) là một tục biến hoặc tích của hai hoặc nhiều tục biến, trong đó không có tục biến của cùng một biến.

Tuyển chuẩn tắc là một biểu thức Boole gồm một hội sơ cấp hoặc tổng của các hội sơ cấp không chứa trong nhau.

Tuyển chuẩn tắc đầy đủ (*Complete/Full disjuntive normal form*)

Biểu thức $E = E(x_1, x_2, \ldots, x_n)$ là tuyển chuẩn tắc đầy đủ nếu mỗi hội sơ cấp đều chứa tất cả *n* biến. Một hội sơ cấp chứa tất cả *n* biến được gọi là một ***tiểu hạng***. Có 2^n tiểu hạng *n* biến và có 2^{2^n} hàm Boole bậc *n*.

Định lý 1 *Mỗi hàm Boole bậc n có một dạng tuyển chuẩn tắc đầy đủ duy nhất.*

Định lý 2 *Nếu* $x + y = 1$ *và* $x.y = 0$ *thì* $y = \bar{x}$.

4.3. Thuật toán tìm dạng tuyển chuẩn tắc

Bước 1. Dùng luật De Morgan và luật bù kép để đưa tất cả các phép bù vào trong các cặp ngoặc đơn cho đến khi phép bù chỉ dùng cho các biến. Biểu thức chỉ gồm tổng và tích của các hội sơ cấp.

Bước 2. Dùng luật phân phối để biến đổi tiếp thành tổng các tích.

Bước 3. Dùng luật giao hoán, luật lũy đẳng, luật bù để biến đổi mỗi tích thành 0 hoặc hội sơ cấp.

Bước 4. Cuối cùng dùng luật hấp thụ và luật đồng nhất để biến đổi thành tuyển chuẩn tắc.

Ví dụ E $= \overline{\overline{(xy)}\,z}\ \ \overline{(\bar{x}+z)(\bar{y}+\bar{z})}$

$=(\overline{\overline{xy}} + \bar{z})((\overline{\bar{x}+z}) + (\overline{\bar{y}+\bar{z}}))$

$=(xy + \bar{z})(\bar{\bar{x}}\,\bar{z} + \bar{\bar{y}}\,\bar{\bar{z}})$

$=(xy + \bar{z})(x\bar{z} + yz)$

$= xyx\bar{z} + xyyz + \bar{z}x\bar{z} + \bar{z}yz$

$= xy\bar{z} + xyz + x\bar{z} + 0$

$= xyz + x\bar{z}$

4.4. Thuật toán tìm dạng tuyển chuẩn tắc đầy đủ

Bước 1. Tìm hội sơ cấp P trong E không chứa biến x_i, nhân P cho $x_i + \bar{x}_i$, xóa các hội sơ cấp lặp (vì $x_i + \bar{x}_i = 1$ và P+P=P).

Bước 2. Lặp bước 1 cho đến khi mọi hội sơ cấp P trong E đều là tiểu hạng, nghĩa là chứa tất cả *n* biến

Ví dụ. E $= yz + x\bar{z}$

$= (x + \bar{x})yz + x\bar{z}(y + \bar{y})$

$= xyz + \bar{x}yz + xy\bar{z} + x\bar{y}\bar{z}$

4.5. Bảng chân trị

Bảng chân trị của hàm Boole bậc *n* là bảng kê giá trị của hàm ứng với tất cả các giá trị khác nhau của các biến.

Ví dụ. Cho hàm Boole bậc 3 dạng tuyển chuẩn tắc đầy đủ

$$f(x,y,z) = xyz + \bar{x}yz + xy\bar{z} + x\bar{y}\bar{z}$$

x	y	z	f(x,y,z)
1	1	1	1
1	1	0	1
1	0	1	0
1	0	0	1
0	1	1	1
0	1	0	0
0	0	1	0
0	0	0	0

Nhận xét. Với hàm Boole dạng tuyển chuẩn tắc đầy đủ thì các giá trị 1 của hàm ứng với các tiểu hạng có trong biểu thức biểu diễn hàm. Ngược lại, nếu cho hàm dưới dạng bảng chân trị thì dạng tuyển chuẩn tắc đầy đủ của hàm gồm các tiểu hạng ứng với các giá trị 1.

4.6. Tìm biểu thức tối thiểu bằng bản đồ Karnaugh

4.6.1. Các khái biệm

Hai hội sơ cấp P_1 và P_2 gọi là ***kề nhau*** nếu P_1 và P_2 khác nhau đúng một tục biến (một biến bù trong một hội sơ cấp và không bù trong hội sơ cấp khác).

Đặc biệt, tổng của hai hội sơ cấp kề nhau như thế sẽ là một hội sơ cấp thiếu một tục biến. $xE+\bar{x}E = E$.

Ví dụ Tìm tổng của hai hội sơ cấp kề nhau P_1, P_2 với:

a) $P_1 = xy\bar{z}$ và $P_2 = x\bar{y}\,\bar{z}$

   $P_1+P_2 = xy\bar{z} + x\bar{y}\,\bar{z} = x\bar{z}(y+\bar{y}) = x\bar{z}(1) = x\bar{z}$

b) $P_1 = \bar{x}yzt$ và $P_2 = \bar{x}y\bar{z}t$

c) $P_1+P_2 = \bar{x}y\bar{z}t + \bar{x}y\bar{z}t = \bar{x}yt(z+\bar{z}) = \bar{x}yt(1) = \bar{x}yt$

d) $P_1 = \bar{x}yzt$ và $P_2 = xy\bar{z}t$

   Hai tiểu hạng này không kề nhau vì chúng khác nhau đến hai tục biến. Đặc biệt

e) $P_1 = xy\bar{z}$ và $P_2 = xy\bar{z}t$

   P_1 và P_2 không kề nhau vì chúng có các biến khác nhau. Do đó, chúng không xuất hiện trong cùng một bản đồ Karnaugh.

Bản đồ Karnaugh của biểu thức* n *biến (hàm Boole bậc *n*) là bảng chữ nhật gồm 2^n ô, mỗi ô là một tiểu hạng sao cho hai ô kề nhau là hai tiểu hạng kề nhau (khác nhau đúng một tục biến).

4.6.2. Mã Gray

Mã Gray là 2^n xâu bit độ dài *n* được sắp xếp sao cho hai xâu kề nhau chỉ khác nhau đúng một bit, kể cả hai xâu đầu và cuối . Phương pháp liệt kê theo thứ tự từ điển không có được tính chất này.

Mã Gray 2^n xâu bit độ dài *n* được tạo ra từ 2^{n-1} xâu bit độ dài *n-1* bằng cách phản chiếu gấp đôi để có được 2^n xâu bit độ dài *n-1*. Rồi nửa đầu thêm trước là bit *1*, nửa sau thêm trước là bit *0*.

Mỗi tục biến được biểu diễn bằng một bit: 1 ứng với biến và 0 ứng với bù của biến. Mục đích là hai tiểu hạng kề nhau (ngang/dọc) khác nhau đúng một tục biến có thể rút gọn được biến đó: $xE+\bar{x}E = E$.

n=1
1
0

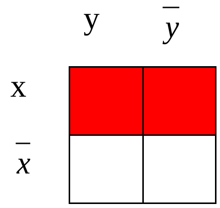
n=2	
1	1
1	0
0	0
0	1

n=3		
1	1	1
1	1	0
1	0	0
1	0	1
0	0	1
0	0	0
0	1	0
0	1	1

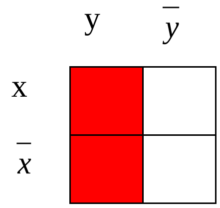
Trường hợp n=2

	y	$\bar{y}$
x	xy	$x\bar{y}$
$\bar{x}$	$\bar{x}y$	$\bar{x}\bar{y}$

(a)



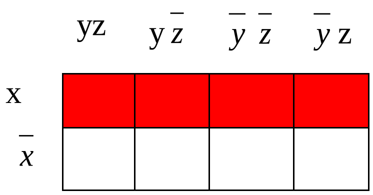
(b) x



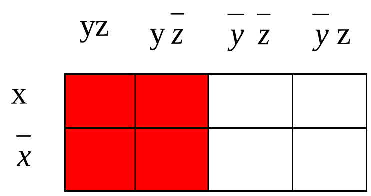
(c) y

Trường hợp n=3

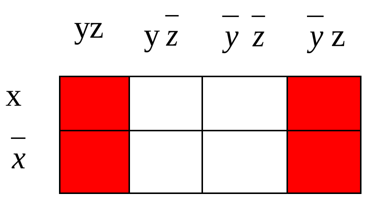
	yz	$y\bar{z}$	$\bar{y}\bar{z}$	$\bar{y}z$
x	xyz	$xy\bar{z}$	$x\bar{y}\bar{z}$	$x\bar{y}z$
$\bar{x}$	$\bar{x}yz$	$\bar{x}y\bar{z}$	$\bar{x}\bar{y}\bar{z}$	$\bar{x}\bar{y}z$



(a) x



(b) y

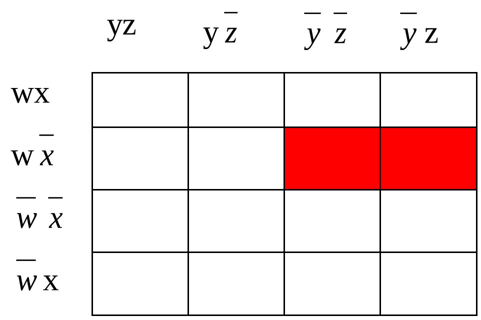


(c) z

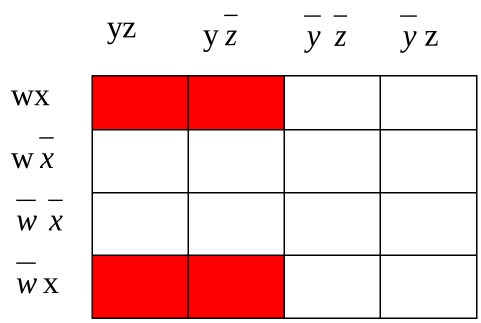
Trường hợp n=4

	yz	$y\bar{z}$	$\bar{y}\,\bar{z}$	$\bar{y}z$
wx	wx yz	$wx\,y\bar{z}$	$wx\,\bar{y}\,\bar{z}$	$wx\,\bar{y}z$
$w\bar{x}$	$w\bar{x}\,yz$	$w\bar{x}\,y\bar{z}$	$w\bar{x}\,\bar{y}\,\bar{z}$	$w\bar{x}\,\bar{y}z$
$\bar{w}\,\bar{x}$	$\bar{w}\,\bar{x}\,yz$	$\bar{w}\,\bar{x}\,y\bar{z}$	$\bar{w}\,\bar{x}\,\bar{y}\,\bar{z}$	$\bar{w}\,\bar{x}\,\bar{y}z$
$\bar{w}x$	$\bar{w}x\,yz$	$\bar{w}x\,y\bar{z}$	$\bar{w}x\,\bar{y}\,\bar{z}$	$\bar{w}x\,\bar{y}z$

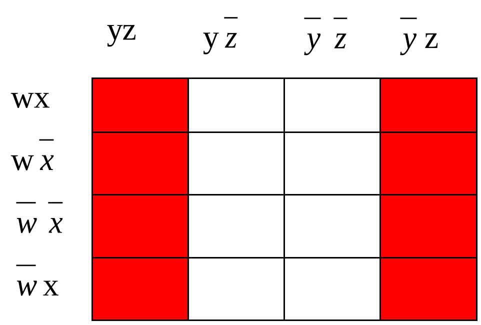
$w\bar{x}\,\bar{y}$



xy



z



4.6.3. Các bước tìm biểu thức tối thiểu

Để tìm biểu thức tối thiểu của một hàm Boole bậc *n* dạng tuyển chuẩn tắc đầy đủ, thực hiện các bước sau:

Bước 1. Tạo bản đồ Karnaugh gồm 2^n ô. Mỗi tiểu hạng có trong biểu thức được đánh số *1* vào ô tương ứng.

Bước 2. Khoanh tròn các số *1* kề nhau cùng nằm trong một hình chữ nhật lớn nhất dạng 1x1, 1x2, 2x1, 1x4, 4x1, 2x2, 2x4, 4x2,…

Bước 3. Tìm số ít nhất các hình chữ nhật được khoanh tròn phủ được tất cả các ô số *1*. Mỗi hình chữ nhật là một hội sơ cấp của biểu thức tối thiểu.

Ví dụ. Tìm biểu thức tối thiểu của mỗi biểu thức Boole dạng tuyển chuẩn tắc đầy đủ sau

a) $E_1 = xy + x\overline{y}$

b) $E_2 = xy + \overline{x}y + \overline{x}\,\overline{y}$

c) $E_3 = xy + \overline{x}\,\overline{y}$

Giải

	y	$\overline{y}$
x	1	1
$\overline{x}$		

E_1

	y	$\overline{y}$
x	1	
$\overline{x}$	1	1

E_2

	y	$\overline{y}$
x	1	
$\overline{x}$		1

E_3

(a) $E_1 = x$ là tuyển chuẩn tắc tối thiểu.

(b) $E_2 = \overline{x} + y$ là tuyển chuẩn tắc tối thiểu.

(c) E_3 gồm hai hình vuông cô lập biểu diễn cho xy và $\overline{x}\,\overline{y}$.
$E_3 = xy + \overline{x}\,\overline{y}$ là tuyển chuẩn tắc tối thiểu.

Ví dụ. Tìm biểu thức tối thiểu của mỗi biểu thức Boole dạng tuyển chuẩn tắc đầy đủ sau

a) $E_1 = xyz + xy\overline{z} + \overline{x}y\overline{z} + \overline{x}\,\overline{y}z$

b) $E_2 = xyz + xy\overline{z} + \overline{x}y\overline{z} + \overline{x}\,\overline{y}z + \overline{x}\,\overline{y}\,\overline{z}$

c) $E_3 = xyz + xy\overline{z} + x\overline{y}z + \overline{x}yz + \overline{x}\,\overline{y}\,\overline{z}$

Giải.

a)

	yz	$y\bar{z}$	$\bar{y}\bar{z}$	$\bar{y}z$
x	1	1		
$\bar{x}$		1		1

c)

	yz	$y\bar{z}$	$\bar{y}\bar{z}$	$\bar{y}z$
x	1	1		1
$\bar{x}$	1			1

b)

	yz	$y\bar{z}$	$\bar{y}\bar{z}$	$\bar{y}z$
x	1	1		
$\bar{x}$		1	1	1

a) $E_1 = xy + y\bar{z} + \bar{x}\,\bar{y}z$

b) E_2 có bốn khoanh tròn là xy, $y\bar{z}$, $\bar{x}\,\bar{z}$, và $\bar{x}\,\bar{y}$. Tuy nhiên, chỉ cần một trong hai khoanh tròn chấm chấm, nghĩa là $y\bar{z}$ hoặc $\bar{x}\,\bar{z}$ là cần trong phủ nhỏ nhất của E_2. Do đó E_2 có hai tuyển chuẩn tắc tối thiểu là

$E_2 = xy + y\bar{z} + \bar{x}\,\bar{y}$ hoặc $E_2 = xy + \bar{x}\,\bar{z} + \bar{x}\,\bar{y}$

c) $E_3 = xy + z$

Ví dụ. $E = w\bar{x} + wxy + \bar{w}\,\bar{x}\,\bar{y} + \bar{w}xy\bar{z}$

	yz	$y\bar{z}$	$\bar{y}\bar{z}$	$\bar{y}z$
wx	1	1		
$w\bar{x}$	1	1	1	1
$\bar{w}\,\bar{x}$			1	1
$\bar{w}x$		1		

Tuyển chuẩn tắc tối thiểu của E là $E = wy + \bar{x}\,\bar{y} + xy\bar{z}$

4.7.Tìm biểu thức tối thiểu bằng phương pháp Quine-McCluskey

Bản đồ Karnaugh khó dùng trong trường hợp số biến nhiều hơn bốn biến. Hơn nữa phương pháp này được thực hiện bằng cách nhận dạng trực quan. Phương pháp này được đưa ra bởi W.V. Quine và E.J. McCluskey vào những năm 1950 và có thể cài đặt được trên máy tính.

Phương pháp này gồm hai phần. Phần đầu tìm biểu thức rút gọn, là tìm các hội sơ cấp có thể đưa vào trong biểu thức tối thiểu. Phần thứ hai tìm biểu thức tối thiểu từ các hội sơ cấp có được từ phần đầu bằng cách xác định số ít nhất các các hội sơ cấp này trong biểu thức tối thiểu.

Để tìm biểu thức tối thiểu của một biểu thức Boole dạng tuyển chuẩn tắc đầy đủ *n* biến gồm *k* tiểu hạng, thuật toán Quine-McCluskey thực hiện các bước sau:

Bước 1. Tạo bảng ban đầu gồm *k* xâu bit ứng với *k* tiểu hạng. Sắp xếp theo thứ tự giảm dần của số bit 1 trong mỗi xâu. Để tạo thành từng nhóm có số bit 1 bằng nhau

Bước 2. Lặp cho đến khi không ghép được nữa:
Tạo bảng mới từ bảng hiện thời bằng cách sau: với mỗi xâu gồm *m* bit 1, duyệt qua từng xâu gồm *m-1* bit 1 nếu cùng vị trí dấu "-" và chỉ khác nhau đúng 1 bit và(ghép được) thì đưa xâu này vào bảng mới bằng cách thay bit đó bởi dấu "-" (rút gọn biến này-luật đồng nhất); nếu không có xâu nào ghép được thì đưa nguyên xâu này vào. Nếu xâu đưa vào đã có thì không đưa vào nữa (luật lũy đẳng).

Bước 3 Dựa vào bảng cuối ở bước 3 có được *dạng tuyển chuẩn tắc rút gọn* của biểu thức gồm *q* hội sơ cấp. Để tìm dạng tuyển chuẩn tắc tối thiểu, tạo bảng gồm *p* dòng hội sơ cấp và *k* cột tiểu hạng. Hội sơ cấp *i* chứa trong tiểu hạng *j* thì đánh dấu "*" vào ô (i,j) trong bảng.

Tìm số ít nhất các hàng phủ được tất cả các cột (dấu "*") trong bảng bằng quy tắc sau:

a) Với cột chỉ có duy nhất một dấu "*" thì dòng đó gọi là *dòng cần thiết* và hội sơ cấp đó là cần thiết phải có trong biểu thức tối thiểu.

b) Tìm các dòng còn lại bằng thuật toán tham lam như sau: Đưa dòng có nhiều dấu "*" nhất vào biểu thức tối thiểu và đánh dấu xóa các cột có chứa dấu "*" mà nó phủ được. Lặp cho đến khi tất cả các dấu "*"đều phủ được, hay tất cả các cột được đánh dấu xóa.

Thuật toán này chấp nhận được, tuy nhiên có trường hợp biểu thức nhận được không phải là biểu thức tối thiểu.

Ví dụ. Cho biểu thức Boole bằng bảng chân trị sau

w	x	y	z	f(w,x,y,z)
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	0

Dạng tuyển chuẩn tắc đầy đủ là:
$f(x,y,z)=\overline{w}\,\overline{x}\,\overline{y}\,z+\overline{w}\,\overline{x}\,yz+\overline{w}\,x\,\overline{y}\,z+\overline{w}\,xyz+w\,\overline{x}\,y\,\overline{z}+w\,\overline{x}\,yz+wxy\,\overline{z}$

Bảng đầu			Bảng 1			Bảng 2		
STT	Xâu	∑1	Ghép	Xâu	∑1	Ghép	Xâu	∑1
1	1110	3	(1,4)	1-10	2	(1)	1-10	2
2	1011	3	(2,4)	101-	2	(2)	101-	2
3	0111	3	(2,6)	-011	2	(3)	-011	2
4	1010	2	(3,5)	01-1	2	(4,7)	0--1	1
5	0101	2	(3,6)	0-11	2	(5,6)	đã có	1
6	0011	2	(5,7)	0-01	1			
7	0001	1	(6,7)	00-1	1			

Vậy biểu thức rút gọn gồm 4 hội sơ cấp là: $wy\bar{z}+w\bar{x}y+\bar{x}yz+\bar{w}z$

	$\bar{w}\bar{x}\bar{y}z$	$\bar{w}\bar{x}yz$	$\bar{w}x\bar{y}z$	$\bar{w}xyz$	$w\bar{x}y\bar{z}$	$w\bar{x}yz$	$wxy\bar{z}$
$wy\bar{z}$					*		*
$w\bar{x}y$					*	*	
$\bar{x}yz$		*				*	
$\bar{w}z$	*	*	*	*			

Cột 7 cho thấy dòng 1 là cần thiết. Đưa dòng 1 vào biểu thức tối thiểu phủ được 2 cột 5 và 7.

Các cột 1, 3, 4 cho thấy dòng 4 là cần thiết. Đưa dòng 4 vào biểu thức tối thiểu phủ được 4 cột 1,2,3,4.

Cuối cùng, để phủ cột 6, chỉ cần đưa vào biểu thức tối thiểu một trong hai dòng 2 hoặc 3.

Vậy, có hai biểu thức tối thiểu là

$wy\bar{z}+w\bar{x}y+\bar{w}z$

$wy\bar{z}+\bar{x}yz+\bar{w}z$

Ví dụ.

$f(w,x,y,z) = wxyz+wx\overline{y}z+wx\overline{y}\,\overline{z}+w\overline{x}y\overline{z}+w\overline{x}\,\overline{y}\,\overline{z}+\overline{w}xy\overline{z}+\overline{w}x\overline{y}z+\overline{w}x\overline{y}\,\overline{z}+\overline{w}\,\overline{x}y\overline{z}$

Bảng đầu			Bảng 1			Bảng 2		
STT	Xâu	∑1	Ghép	Xâu	∑1	Ghép	Xâu	∑1
1	1111	4	(1,2)	11-1	3	(1)	11-1	3
2	1101	3	(2,3)	110-	2	(2,3,5)	-10-	1
3	1100	2	(2,6)	-101	2	(4)	1-00	1
4	1010	2	(3,7)	1-00	1	(6)	10-0	1
5	0110	2	(3,8)	-100	1	(7)	-010	1
6	0101	2	(4,7)	10-0	1	(8)	01-0	1
7	1000	1	(4,9)	-010	1	(9)	0-10	1
8	0100	1	(5,8)	01-0	1			
9	0010	1	(5,9)	0-10	1			

Vậy biểu thức rút gọn gồm 7 hội sơ cấp là: $wxz+x\overline{y}+w\overline{y}\,\overline{z}+w\overline{x}\,\overline{z}+\overline{x}y\overline{z}+\overline{w}x\overline{z}+\overline{w}y\overline{z}$

	$wxyz$	$wx\overline{y}z$	$wx\overline{y}\,\overline{z}$	$w\overline{x}y\overline{z}$	$w\overline{x}\,\overline{y}\,\overline{z}$	$\overline{w}xy\overline{z}$	$\overline{w}x\overline{y}z$	$\overline{w}x\overline{y}\,\overline{z}$	$\overline{w}\,\overline{x}y\overline{z}$
wxz	*	*							
$x\overline{y}$		*	*				*	*	
$w\overline{y}\,\overline{z}$			*		*				
$w\overline{x}\,\overline{z}$				*	*				
$\overline{x}y\overline{z}$				*					*
$\overline{w}x\overline{z}$						*		*	
$\overline{w}y\overline{z}$						*			*

Cột 1 cho thấy dòng 1 là cần thiết. Đưa dòng 1 vào biểu thức tối thiểu phủ được 2 cột 1 và 2.

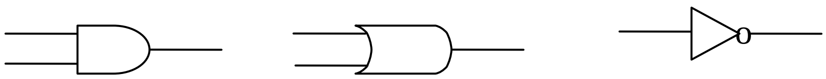
Cột 7 cho thấy dòng 2 là cần thiết. Đưa dòng 2 vào biểu thức tối thiểu phủ thêm được 3 cột 3, 7 và 8.

Dòng 4 phủ thêm được 2 cột 4 và 5.

Cuối cùng, dòng 7 phủ được 2 cột 6 và 9.

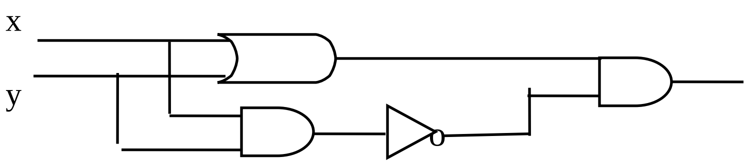
Vậy, biểu thức tối thiểu là: $f(w,x,y,z)=wxz+x\overline{y}+w\overline{x}\,\overline{z}+\overline{w}y\overline{z}$

4.8. Mạch logic



Cổng AND (.) Cổng OR (+) Cổng NOT (¯)

Ví dụ. f(x,y) = (x+y) $\overline{xy}$



****

Bài tập

1. Xét hàm Boole f(x,y,z) được cho bởi bảng chân trị sau đây:

x	y	z	F(x, y, z)
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

a) Tìm dạng tuyển chuẩn tắc đầy đủ của f(x,y,z).
b) Tìm dạng tuyển chuẩn tắc tối thiểu của f(x,y,z).

2. Xét hàm Boole f(w,x,y,z) được cho bởi bảng chân trị sau đây:

w	x	y	z	f(w,x,y,z)
1	1	1	1	1
1	1	1	0	0
1	1	0	1	1
1	1	0	0	1
1	0	1	1	0
1	0	1	0	1
1	0	0	1	0
1	0	0	0	1
0	1	1	1	0
0	1	1	0	1
0	1	0	1	1
0	1	0	0	1
0	0	1	1	0
0	0	1	0	1
0	0	0	1	0
0	0	0	0	0

a) Tìm dạng tuyển chuẩn tắc đầy đủ của f(w,x,y,z).
b) Tìm dạng tuyển chuẩn tắc tối thiểu của f(w,x,y,z).

3. Tìm dạng tuyển chuẩn tắc của biểu thức sau:
f(x, y, z) = $\overline{\overline{(xy)}\,z}\;\;\overline{(\overline{x}+z)(\overline{y}+\overline{z})}$

4. Tìm dạng tuyển chuẩn tắc đầy đủ của biểu thức sau:
a) f(x, y, z) = $yz+x\overline{z}$
b) f(x, y, z) = $x+y+x\overline{z}$

5. Tìm biểu thức tối thiểu của các biểu thức Boole bậc hai sau đây:
a) $E_1= xy + x\overline{y}$
b) $E_2= xy + \overline{x}y + \overline{x}\,\overline{y}$
c) $E_3= xy + \overline{x}\,\overline{y}$

6. Tìm biểu thức tối thiểu của các biểu thức Boole bậc ba sau:
a) $E_1 = x + yz + z\overline{x}y + \overline{y}xz$
b) $E_2= xyz + xy\overline{z} + \overline{x}y\overline{z} + \overline{x}\,\overline{y}z$
c) $E_3= xyz + xy\overline{z} + x\overline{y}z + \overline{x}yz + \overline{x}\,\overline{y}z$
d) $E_4= xyz + xy\overline{z} + \overline{x}y\overline{z} + \overline{x}\,\overline{y}z$

7. Tìm biểu thức tối thiểu của các biểu thức Boole bậc bốn sau đây:
a) $E_1 = w\overline{x} + wxy + \overline{w}\,\overline{x}\,\overline{y} + \overline{w}xy\overline{z}$
b) $E_2= wxyz + wxy\overline{z} + \overline{w}\,\overline{x}y\overline{z} + \overline{w}\,\overline{x}\,\overline{y}z$

****

Chương 5. Đại số mệnh đề

5.1. Mệnh đề và chân trị

- Khái niệm về mệnh đề:

Mệnh đề toán học là khái niệm cơ bản của toán học không được định nghĩa mà chỉ được mô tả.

Mệnh đề toán học (gọi tắt là mệnh đề) là một khẳng định có giá trị chân lý xác định (*đúng* hoặc *sai*). Cũng không thể *vừa đúng vừa sai* hay không thể *không đúng cũng không sai*.

- Ví dụ:
  - "Số 123 chia hết cho 3" là 1 mệnh đề đúng
  - "Thành phố Hồ Chí Minh là thủ đô của nước Việt Nam" là một mệnh đề sai.
  - "Bạn có khỏe không ? " không phải là một mệnh đề toán học vì đây là một câu hỏi không thể phản ánh một điều đúng hay một điều sai

- Kiểm tra xem các khẳng định sau có là mệnh đề không? Nếu có, đó là mệnh đề đúng hay sai?
  - Môn Toán rời rạc là môn bắt buộc chung cho ngành tin học.
  - 97 là số nguyên tố.
  - N là số nguyên tố
- Ký hiệu mệnh đề :

Người ta thường dùng các ký hiệu : P, Q, R, …

- *Chú ý*: Mệnh đề phức hợp là mệnh đề được xây dựng từ các mệnh đề khác nhờ liên kết chúng lại bằng các liên từ(và, hay, nếu…thì…) hoặc trạng từ "không". Ví dụ : Nếu trời tốt thì tôi đi dạo.
- Chân trị của mệnh đề:

Một mệnh đề chỉ có thể đúng hoặc sai, không thể đồng thời vừa đúng vừa sai. Khi mệnh đề P đúng ta nói P có **chân trị** đúng, ngược lại ta nói P có **chân trị** sai.

Chân trị đúng và chân trị sai sẽ được ký hiệu lần lượt là 1 (hay Đ/T) và 0 (hay S/F).

5.2. Phép tính mệnh đề

- Mục đích của phép tính mệnh đề:

Nghiên cứu chân trị của một mệnh đề phức hợp từ chân trị của các mệnh đề đơn giản hơn và các phép nối những mệnh đề này biểu hiện qua liên từ hoặc trạng từ "không".

Các phép tính

Tên	**Phép toán**	**Ký hiệu**
Phủ định	NOT	⌉
Hội	AND	$\wedge$
Tuyển	OR	$\vee$
Tuyển chọn	XOR	$\oplus$
Kéo theo	IMPLIES	$\rightarrow$
Tương đương	IFF	$\leftrightarrow$

5.2.1. Phép phủ định của mệnh đề

Phủ định của mệnh đề P ký hiệu ⌉P hay $\overline{p}$ đúng khi và chỉ khi P sai
Toán tử đơn hạng *phủ định* "⌉" (*NOT*) biến một mệnh đề thành mệnh đề phủ định của nó.
Nếu *p* = "Tôi có tóc nâu."
thì ⌉*p* = "Tôi không có tóc nâu."

Bảng chân trị phép phủ định

p	$\neg p$
T	F
F	T

5.2.2. Phép hội

Mệnh đề hội của hai mệnh đề P, Q được kí hiệu bởi P $\wedge$ Q (đọc là "P và Q"), là mệnh đề được định bởi : P $\wedge$ Q đúng khi và chỉ khi P và Q đồng thời đúng

Ví dụ. Mệnh đề "Hôm nay, An giúp mẹ lau nhà và rửa chén" chỉ đúng khi hôm nay An giúp mẹ cả hai công việc lau nhà và rửa chén. Ngược lại, nếu hôm nay An chỉ giúp mẹ một trong hai công việc trên, hoặc không giúp mẹ cả hai thì mệnh đề trên sai.

Bảng chân trị phép hội

p	q	$p \wedge q$
F	F	F
F	T	F
T	F	F
T	T	T

Lưu ý : Một hội $p_1 \wedge p_2 \wedge \ldots \wedge p_n$ của *n* mệnh đề sẽ có 2^n dòng trên bảng chân trị.

5.2.3. Phép tuyển

Mệnh đề tuyển của hai mệnh đề P, Q được kí hiệu bởi P ∨ Q (đọc là "P hay Q"), là mệnh đề được định bởi : P ∨ Q sai khi và chỉ khi P và Q đồng thời sai

Ví dụ. Mệnh đề "Tôi đang chơi bóng đá hay bóng rổ".
Mệnh đề này chỉ sai khi tôi vừa không đang chơi bóng đá cũng như vừa không đang chơi bóng rổ. Ngược lại, tôi chơi bóng đá hay đang chơi bóng rổ hay đang chơi cả hai thì mệnh đề trên đúng.

Bảng chân trị phép tuyển

p	*q*	*p*∨*q*
F	F	F
F	T	**T**
T	F	**T**
T	T	T

Lưu ý. *p*∨*q* nghĩa là *p* đúng, hoặc q đúng, hoặc **cả hai** cùng đúng!

5.2.4. Phép tuyển chọn (XOR)

P ⊕ Q sai ⇔ P và Q đồng thời cùng đúng hoặc cùng sai.

p = "Tôi sẽ đạt loại A môn học này."
q = "Tôi sẽ hỏng môn học này."
p ⊕ *q* = "Hoặc là tôi sẽ đạt loại A môn học này, hoặc là tôi sẽ hỏng môn học này. (nhưng không cả hai!)"

Bảng chân trị phép tuyển chọn

p	*q*	*p*⊕*q*
F	F	F
F	T	T
T	F	T
T	T	**F**

- Lưu ý rằng *p*⊕*q* nghĩa là *p* đúng, hoặc *q* đúng, nhưng không phải **cả hai**!
- Phép toán này gọi là tuyển chọn vì loại trừ trường hợp cả hai *p* và *q* cùng đúng.

5.2.5. Phép kéo theo

Mệnh đề P kéo theo Q của hai mệnh đề P và Q, ký hiệu bởi P → Q (đọc là "P kéo theo Q" hay "Nếu P thì Q" hay "P là điều kiện đủ của Q" hay "Q là điều kiện cần của P") là mệnh đề được định bởi: P → Q sai khi và chỉ khi P đúng mà Q sai .

Ví dụ. Mệnh đề "Chiều nay, nếu rảnh tôi sẽ ghé thăm bạn" chỉ sai khi chiều nay tôi rảnh nhưng tôi không ghé thăm bạn.

Ngược lại, nếu chiều nay tôi bận thì dù tôi có ghé thăm bạn hay không, mệnh đề trên vẫn đúng. Ngoài ra, tất nhiên nếu chiều nay tôi có ghé thăm bạn thì mệnh đề trên đúng (dù tôi có rảnh hay không!).

Cho p = "Bạn học chăm."

q = "Bạn đạt kết quả tốt."

$p \rightarrow q$ = "Nếu bạn học chăm thì bạn đạt kết quả tốt." (ngược lại, có thể đạt kết quả tốt hoặc không tốt)

Bảng chân trị phép kéo theo

p	q	$p \rightarrow q$
F	F	T
F	T	T
T	F	**F**
T	T	T

- $p \rightarrow q$ *sai chỉ khi* p đúng nhưng q không đúng.
- $p \rightarrow q$ *không nói lên được* rằng p suy ra q!
- $p \rightarrow q$ *không đòi hỏi* p hay q phải đúng!

Ví dụ. "(1=0)" $\rightarrow$ "lợn có thể bay" là đúng!

- "Nếu bài giảng này kết thúc thì ngày mai trời mọc." ***T*** hay *F* ?
- "Nếu thứ Ba là ngày trong tuần thì tôi là một con chim." *T* hay ***F*** ?
- "Nếu 1+1=6, thì Obama là tổng thống." ***T*** hay *F* ?
- "Nếu mặt trăng làm bằng pho mát thì tôi giàu hơn Bill Gates." ***T*** hay *F* ?

5.2.6. Phép tương đương

Mệnh đề P kéo theo Q và ngược lại của hai mệnh đề P và Q, ký hiệu bởi P $\leftrightarrow$ Q đọc là "P nếu và chỉ nếu Q" hay P khi và chỉ khi Q" hay "P là điều kiện cần và đủ của Q", là mệnh đề xác định bởi: P $\leftrightarrow$ Q đúng khi **v**à chỉ khi P và Q có cùng chân trị.

Mệnh đề tương đương $p \leftrightarrow q$ *nói lên rằng* p đúng nếu và chỉ nếu q đúng.

Ví dụ.

p = "Bush thắng cử năm 2004."

q = "Bush là tổng thống năm 2005."

$p \leftrightarrow q$ = "Nếu và chỉ nếu Bush thắng cử năm 2004 thì Bush là tổng thống năm 2005."

Bảng chân trị phép tương đương

p	q	$p \leftrightarrow q$
F	F	T
F	T	F
T	F	F
T	T	T

- $p \leftrightarrow q$ nghĩa là p và q có cùng chân trị.
- Lưu ý rằng bảng chân trị ngược với phép $\oplus$

$p \leftrightarrow q$ nghĩa là $\neg(p \oplus q)$

- $p \leftrightarrow q$ không phải p và q đều đúng.

Có 1 phép toán đơn hạng và 5 phép toán nhị hạng. Bảng chân trị của chúng như sau.

p	q	$\neg p$	$p\wedge q$	$p\vee q$	$p\oplus q$	$p\rightarrow q$	$p\leftrightarrow q$
F	F	T	F	F	F	T	T
F	T	T	F	T	T	T	F
T	F	F	F	T	T	F	F
T	T	F	T	T	F	T	T

5.3. Dạng mệnh đề

Dạng mệnh đề hay *Mệnh đề phức hợp* là một biểu thức được cấu tạo từ:

- Các hạng mệnh đề, tức là các mệnh đề đã xét ở trên.
- Các biến mệnh đề, tức là các biến lấy giá trị là các mệnh đề, thông qua các phép toán mệnh đề đã xét ở trên
- Các hạng mệnh đề và các biến mệnh đề theo một trình tự nhất định nào đó, thường được chỉ rõ bởi các dấu ngoặc đơn.

- Với E là một dạng mệnh đề các biến mệnh đề p, q, r ứng với mỗi giá trị cụ thể P, Q, R (là các mệnh đề) của p, q, r thì ta có duy nhất một mệnh đề E(P, Q, R). Ta viết E = E(p, q, r).
- Bảng chân trị là bảng ghi tất cả các trường hợp chân trị có thể xảy ra đối với dạng mệnh đề E theo chân trị của các biến mệnh đề p, q, r. Nếu có *n* biến, bảng này sẽ có 2^n dòng, chưa kể dòng tiêu đề.

Ví dụ. Xét dạng mệnh đề

$E = (p\vee q) \rightarrow r$

Có bảng chân trị là hình bên

p	q	r	p∨q	E
1	1	1	1	1
1	1	0	1	0
1	0	1	1	1
1	0	0	1	0
0	1	1	1	1
0	1	0	1	0
0	0	1	0	1
0	0	0	0	1

5.4. Hằng đúng và mâu thuẩn

Một hằng đúng là mệnh đề phức hợp luôn đúng với mọi giá trị của các mệnh đề thành phần.

Ví dụ. $p \vee \rceil p$ [Bảng chân trị?]
Một mâu thuẩn (hằng sai) là mệnh đề phức hợp luôn sai với mọi giá trị của các mệnh đề thành phần.
Ví dụ. $p \wedge \rceil p$ [Bảng chân trị?]
Ngoài ra thì được gọi là các *tiếp liên*.
Mệnh đề phức hợp *p* gọi là *tương đương logic* với mệnh đề phức hợp *q*, viết là $p \Leftrightarrow q$, nếu và chỉ nếu mệnh đề phức hợp $p \leftrightarrow q$ *là hằng đúng*.
Các mệnh đề phức hợp *p* và *q* tương đương logic với nhau nếu và chỉ nếu chúng có cùng bảng chân trị.

Cho E và F là hai dạng mệnh đề.

- F là *hệ quả* của E, ký hiệu E⇒F, nếu E→F là hằng đúng.
- E *tương đương logic* (hay *tương đương*)với F, ký hiệu E⇔F, nếu E↔F là hằng đúng.

Chứng minh tương đương qua bảng chân trị

Ví dụ. Chứng minh rằng $p \vee q \Leftrightarrow \neg(\neg p \wedge \neg q)$.

P	Q	$p \vee q$	$\neg p$	$\neg q$	$\neg p \wedge \neg q$	$\neg(\neg p \wedge \neg q)$
T	T	T	F	F	F	T
T	F	T	F	T	F	T
F	T	T	T	F	F	T
F	F	F	T	T	T	F

Quy tắc thay thế thứ 1

Trong dạng mệnh đề E, nếu ta thay thế biểu thức con F bởi một dạng mệnh đề tương đương logic thì dạng mệnh đề thu được vẫn còn tương đương logic với E.

Quy tắc thay thế thứ 2

Giả sử dạng mệnh đề E(p,q,r…) là một hằng đúng. Nếu ta thay thế những nơi p xuất hiện trong E bởi một F(p',q',r') thì dạng mệnh đề nhận được theo các biến q, r… ,p',q',r',…vẫn còn là 1 hằng đúng.

5.5. Tương đương Logic

Các luật logic: Với p, q, r là các biến mệnh đề, **1** là hằng đúng và **0** là hằng sai, có các tương đương logic sau đây:

- Luật *kết hợp:* $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$ và $(p \wedge q) \wedge r \Leftrightarrow p \wedge (q \wedge r)$
- Luật *giao hoán:* $p \vee q \Leftrightarrow q \vee p$ và $p \wedge q \Leftrightarrow q \wedge p$

- Luật *phân phối*: $p\vee(q\wedge r) \Leftrightarrow (p\vee q)\wedge(p\vee r)$ và
  $p\wedge(q\vee r) \Leftrightarrow (p\wedge q)\vee(p\wedge r)$
- Luật *đồng nhất*: $p\wedge\mathbf{1} \Leftrightarrow p$ và $p\vee\mathbf{0} \Leftrightarrow p$
- Luật *Đúng/Sai tầm thường*: $p \vee \neg p \Leftrightarrow \mathbf{1}$
  $p \wedge \neg p \Leftrightarrow \mathbf{0}$
- Luật *luỹ đẳng*: $p \wedge p \Leftrightarrow p$ và $p \vee p \Leftrightarrow p$
- Luật *trội(nuốt)*: $p\vee\mathbf{1} \Leftrightarrow \mathbf{1}$ và $p\wedge\mathbf{0} \Leftrightarrow \mathbf{0}$
- Luật *bù kép:* $\neg\neg p \Leftrightarrow p$
- Luật *De Morgan*: $\neg(p\wedge q) \Leftrightarrow \neg p \vee \neg q$
  $\neg(p\vee q) \Leftrightarrow \neg p \wedge \neg q$

Dùng Tương đương Logic để định nghĩa các phép toán qua các phép toán khác

- Tuyển chọn: $p\oplus q \Leftrightarrow (p\vee q) \wedge\neg (p\wedge q)$
  $p\oplus q \Leftrightarrow (p\wedge\neg q)\vee(q\wedge\neg p)$
- Kéo theo: $p\rightarrow q \Leftrightarrow \neg p \vee q$
- Tương đương: $p\leftrightarrow q \Leftrightarrow (p\rightarrow q) \wedge (q\rightarrow p)$ hay $p\leftrightarrow q \Leftrightarrow \neg(p\oplus q)$

Ví dụ.

$(p \wedge \neg q) \rightarrow (p \oplus r) \Leftrightarrow \neg p \vee q \vee \neg r.$

Giải.

$(p\wedge\neg q)\rightarrow(p\oplus r)$
$\Leftrightarrow \neg(p \wedge\neg q) \vee (p \oplus r)$ [Đ/n →]
$\Leftrightarrow \neg(p \wedge\neg q)\vee((p \vee r) \wedge \neg(p \wedge r))$ [Đ/n ⊕]
$\Leftrightarrow (\neg p \vee q) \vee ((p \vee r) \wedge \neg(p \wedge r))$ [luật DeMorgan]
$\Leftrightarrow (q \vee \neg p) \vee ((p \vee r) \wedge \neg(p \wedge r))$ [luật giao hoán]
$\Leftrightarrow q \vee (\neg p \vee ((p \vee r) \wedge \neg(p \wedge r)))$ [luật kết hợp]
$\Leftrightarrow q \vee ((\neg p \vee (p \vee r) \wedge (\neg p \vee \neg(p \wedge r))$ [luật phân phối]
$\Leftrightarrow q \vee ((\neg p \vee p) \vee r) \wedge (\neg p \vee \neg(p \wedge r))$ [luật kết hợp]
$\Leftrightarrow q \vee ((\mathbf{T} \vee r) \wedge (\neg p \vee \neg(p \wedge r)))$ [luật bù]
$\Leftrightarrow q \vee (\mathbf{T} \wedge (\neg p \vee \neg(p \wedge r)))$ [luật nuốt]
$\Leftrightarrow q \vee (\neg p \vee \neg(p \wedge r))$ [luật đồng nhất]
$\Leftrightarrow q \vee (\neg p \vee (\neg p \vee \neg r))$ [luật De Morgan]
$\Leftrightarrow q \vee ((\neg p \vee \neg p) \vee \neg r)$ [luật kết hợp]
$\Leftrightarrow q \vee (\neg p \vee \neg r)$ [luật lũy đẳng]
$\Leftrightarrow (q \vee \neg p) \vee \neg r$ [luật kết hợp]
$\Leftrightarrow \neg p \vee q \vee \neg r$ [luật giao hoán]
Xong

5.6. Chứng minh hằng đúng/sai

Để chứng minh một dạng mệnh đề là hằng đúng, hằng sai, các dạng mệnh đề là tương đương lôgic, dạng mệnh đề này là hệ quả logic của dạng mệnh đề kia và ngược lại, có các cách sau:

- Lập bảng chân trị.
- Sử dụng phép thay thế.

Ví dụ. Cho p, q, r là các biến mệnh đề. Chứng minh rằng:
$(\neg p \to r) \wedge (q \to r) \Leftrightarrow (p \to q) \to r$ (1)
Có thể chứng minh (1) bằng hai cách sau.

Cách 1 Lập bảng chân trị (như ví dụ trên)

Cách 2 Biến đổi

$(\neg p \to r) \wedge (q \to r)$	$\Leftrightarrow (p \vee r) \wedge (\neg q \vee r)$	[Đ/n →]
	$\Leftrightarrow (p \wedge \neg q) \vee r$	[luật phân phối]
	$\Leftrightarrow \neg(\neg p \vee \neg\neg q) \vee r$	[luật De Morgan]
	$\Leftrightarrow \neg(\neg p \vee q) \vee r$	[luật bù kép]
	$\Leftrightarrow \neg(p \to q) \vee r$	[Đ/n →]
	$\Leftrightarrow (p \to q) \to r$	[Đ/n →]

5.7. *Qui tắc suy diễn*

- Trong các chứng minh toán học, xuất phát từ một số khẳng định đúng *p, q, r*...(tiền đề), ta áp dụng các qui tắc suy diễn để suy ra chân lí của một mệnh đề *h* mà ta gọi là kết luận.
- Nói cách khác, dùng các qui tắc suy diễn để chứng minh:

($p \wedge q \wedge r \wedge \ldots$) có hệ quả logic là h

Thường mô hình hóa phép suy luận đó dưới dạng:

$$\begin{array}{l} p \\ q \\ r \\ \underline{\ldots} \\ \therefore h \end{array}$$

Ký hiệu ∴ đọc là: *do đó, vậy thì*, …

5.7.1. Qui tắc Modus Ponens (Phương pháp khẳng định)

Qui tắc này được thể hiện bằng hằng đúng:

$$\left[(p \to q) \wedge p\right] \to q$$

Hoặc dưới dạng sơ đồ

$$\begin{array}{l} p \to q \\ \underline{p \qquad} \\ \therefore q \end{array}$$

- Nếu An học chăm thì An học tốt.
- Mà An học chăm

Suy ra An học tốt

- Hình vuông là hình bình hành
- Mà hình bình hành có hai đường chéo cắt nhau tại trung điểm mỗi đường.

Suy ra hình vuông có hai đường chéo cắt nhau tại trung điểm mỗi đường

5.7.2. Qui tắc Tan đoạn luận (Syllogism)

Qui tắc này được thể hiện bằng hằng đúng:

$$\left[(p \to q) \wedge (q \to r)\right] \to (p \to r)$$

Hoặc dưới dạng sơ đồ

$$\begin{array}{l} p \to q \\ \underline{q \to r} \\ \therefore\, p \to r \end{array}$$

- Hai tam giác có cạnh bằng nhau kèm giữa hai góc bằng nhau thì chúng bằng nhau

Suy ra hai tam giác vuông có cạnh huyền và 1 cặp góc nhọn bằng nhau thì bằng nhau.

- Hai tam giác vuông có cạnh huyền và 1 cặp góc nhọn bằng nhau thì chúng ta có một cạnh bằng nhau kèm giữa hai góc bằng nhau.

- Một con ngựa rẻ là một con ngựa hiếm
- Cái gì hiếm thì đắt

Suy ra một con ngựa rẻ thì đắt (!)

5.7.3. Qui tắc Modus Tollens (Phương pháp phủ định)

Qui tắc này được thể hiện bằng hằng đúng:

$$\left[(p \to q) \wedge \neg q\right] \to \neg p$$

- Hoặc dưới dạng sơ đồ

$$\begin{array}{l} p \to q \\ \underline{\neg q} \\ \therefore\, \neg p \end{array}$$

- Xét chứng minh   Suy luận

$$\begin{array}{l} p \to r \\ r \to s \\ t \vee \neg s \\ \neg t \vee u \\ \underline{\neg u} \\ \therefore\, \neg p \end{array} \qquad\qquad \begin{array}{l} p \to r \\ r \to s \\ s \to t \\ t \to u \\ \underline{\neg u} \\ \therefore\, \neg p \end{array}$$

5.7.4. Qui tắc Tam đoạn luận rời

Qui tắc này được thể hiện bằng hằng đúng:

$$\left[(p \vee q) \wedge \neg q\right] \to p \qquad \left[(p \vee q) \wedge \neg p\right] \to q$$

Ý nghĩa của qui tắc: nếu trong hai trường hợp có thể xảy ra, chúng ta biết có một trường hợp sai thì chắc chắn trường hợp còn lại sẽ đúng

5.7.5. Qui tắc Mâu thuẩn (Chứng minh phản chứng)

Ta có tương đương logic

$$\left[(p_1 \wedge p_2 \wedge \ldots \wedge p_n) \to q\right] \Leftrightarrow \left[(p_1 \wedge p_2 \wedge \ldots \wedge p_n \wedge \neg q) \to 0\right]$$

- Để chứng minh vế trái g là một hằng đúng ta chứng minh nếu thêm phủ định của q vào các tiền đề thì được một mâu thuẫn.

Ví dụ

Hãy chứng minh

$$\begin{array}{l} p \to r \\ \neg p \to q \\ q \to s \\ \hline \therefore \neg r \to s \end{array}$$

C/m bằng phản chứng

$$\begin{array}{l} p \to r \\ \neg p \to q \\ q \to s \\ \neg r \\ \neg s \\ \hline \therefore 0 \end{array}$$

5.7.6. Chứng minh theo trường hợp

Dựa trên hằng đúng:

$$\left[(p \to r) \wedge (q \to r)\right] \to \left[(p \vee q) \to r\right]$$

Ý nghĩa: *nếu p suy ra r và q suy ra r thì p hay q cũng có thể suy ra r.*

Ví dụ. Chứng minh rằng: $(n^3 - 4n) \vdots 3$

Giải.

Xét các trường hợp sau:

- n $\vdots$ 3: suy ra $(n^3 - 4n) \vdots 3$
- n =3k+1: n^3-4n =$(3k+1)^3$-4(3k+1) = $27k^3+27k^2$+9k+1-12k-4 = $27k^3+27k^2$-3k-3 $\vdots$ 3
- n =3k+2: n^3-4n =$(3k+2)^3$-4(3k+2) = $27k^3+54k^2$+12k+8-12k-8 = $27k^3+54k^2$ $\vdots$ 3

Một số luật thêm

$$\frac{p}{\therefore p \vee q}$$ Phép thêm

$$\frac{p \wedge q}{\therefore p}$$ Phép đơn giản nối liền

$$\frac{\begin{matrix} p \\ q \end{matrix}}{\therefore p \wedge q}$$ Phép nối liền

1. Nếu nghệ sĩ Trương Ba không trình diễn hay số vé bán ra ít hơn 100 thì đêm diễn sẽ bi hủy bỏ và ông bầu sẽ rất buồn.
2. Nếu đêm diễn bị hủy bỏ thì tiềnvé phải trả lại cho người xem.
3. Nhưng tiền vé đã không trả lại cho người xem.
   Vậy nghệ sỹ Trương Ba đã trình diễn

- p: Nghệ sĩ Trương Ba đã trình diễn.
- q: số vé bán ra ít hơn 100.
- r: đêm diễn bị hủy bỏ.
- s: ông bầu buồn.
- t: trả lại tiền vé cho người xem

5.7.7. Phản ví dụ

Để chứng minh một phép suy luận là sai hay

$$p_1 \wedge p_2 \wedge \ldots \wedge p_n \rightarrow q$$

không là một hằng đúng. Ta chỉ cần chỉ ra một phản ví dụ.

- Ông Minh nói rằng nếu không được tăng lương thì ông ta sẽ nghỉ việc. Mặt khác, nếu ông ấy nghỉ việc và vợ ông ấy bị mất việc thì phải bán xe. Biết rằng nếu vợ ông Minh hay đi làm trễ thì trước sau gì cũng sẽ bị mất việc và cuối cùng ông Minh đã được tăng lương.
- Suy ra nếu ông Minh không bán xe thì vợ ông ta đã không đi làm trễ
- p: ông Minh được tăng lương.
- q: ông Minh nghỉ việc.
- r: vợ ông Minh mất việc.
- s: gia đình phải bán xe.
- t: vợ ông hay đi làm trễ

$$\begin{array}{l} \neg p \rightarrow q \\ q \wedge r \rightarrow s \\ t \rightarrow r \\ \underline{p} \\ \therefore \neg s \rightarrow \neg t \end{array}$$

s=0
t=1
p=1
q=0
r=1

Ví dụ chứng minh

- Giả sử có các giả thiết sau:
  "Trời không nắng và trời lạnh."
  "Chúng ta chỉ bơi nếu trời nắng."
  "Nếu chúng ta không bới thì chúng ta sẽ đi xuồng."
  "Nếu chúng ta đi xuồng thì chúng ta sẽ về nhà sớm."
- Cho các giả thiết trên, chứng minh rằng **"chúng ta sẽ về nhà sớm"** bằng cách dùng các qui tắc suy diễn.
- Hãy ký hiệu gọn các mệnh đề như sau:

nắng = **"trời nắng"**; *lạnh* = **"trời lạnh"**;
bơi = **"chúng ta sẽ bơi"**; *xuồng* = **"chúng ta sẽ đi xuồng"**; *sớm* = **"chúng ta sẽ về sớm"**.

- Các giả thiết có thể viết lại như:
  (1) ¬ *nắng* ∧ *lạnh* (2) *bơi* → *nắng*
  (3) ¬ *bơi* → *xuồng* (4) *xuồng* → *sớm*

Các bước	Chứng minh
1. ¬ *nắng* ∧ *lạnh*	giả thiết #1.
2. ¬ *nắng*	rút gọn 1.
3. *bơi* → *nắng*	giả thiết #2.
4. ¬ *bơi*	phản đảo 2,3.
5. ¬ *bơi* → *xuồng*	giả thiết #3.
6. *xuồng*	phản đảo 4,5.
7. *xuồng* → *sớm*	giả thiết #4.
8. *sớm*	phản đảo 6,7.

Ví dụ . Kiểm tra suy luận sau

$$\begin{array}{l} p\rightarrow(q\rightarrow r) \\ p\vee s \\ t \rightarrow q \\ \underline{\neg s} \\ \therefore \neg r\rightarrow \neg t \end{array}$$

Giải.

1) $\neg s$	[tiền đề]
2) $p \vee s$	[tiền đề]
3) p	[tam đoạn luận rời]
4) $p \rightarrow (q \rightarrow r)$	[tiền đề]
5) $q \rightarrow r$	[quy tắc khẳng định]
6) $t \rightarrow q$	[tiền đề]
7) $t \rightarrow r$	[tam đoạn luận]
$\therefore \neg r \rightarrow \neg t$	[luật phản đảo]

Vậy suy luận trên là đúng.

Ví dụ. Kiểm tra suy luận sau

$p \rightarrow q$
$\neg r \vee s$
$p \vee r$
____

$\therefore \neg q \rightarrow s$

Giải.

1) Giả sử $\neg(\neg q \rightarrow s)$	[giả thiết phản chưng]
2) $\neg(\neg\neg q \vee s)$	[Đn$\rightarrow$]
3) $\neg(q \vee s)$	[luật bù kép]
4) $\neg q \wedge \neg s$	[luật De Morgan]
5) $p \rightarrow q$	[tiền đề]
6) $\neg p$	[quy tắc phủ định]
7) $\neg r \vee s$	[tiền đề]
8) $\neg r$	[4,6 tam đoạn luận rời]
9) $\neg p \wedge \neg r$	[phép nối liền]
10) $\neg(p \vee r)$	[luật De Morgan]
11) $(p \vee r)$	[tiền đề]
12) 0	[luật phần tử bù]

Vậy suy luận trên là đúng (quy tắc phản chứng).

Ví dụ. Kiểm tra suy luận sau

p
$p \rightarrow r$
$p \rightarrow (q \vee \neg r)$

____

$\neg q \vee \neg s$

$\therefore$ s

Kiểm tra trường hợp thỏa mãn giả thiết: p=1, r=1, q=1, s=0.

Vậy suy luận trên là sai.

****

Bài tập

1. Kiểm tra lại dạng mệnh đề sau là hằng đúng

$[p\to(q \vee r)] \to[(p \to q) \vee(p \to r)]$

2. Kiểm tra lại tính đúng đắn của suy luận sau

p

$q \to r$

$p \to \neg r$

$\therefore \neg q$

3. Cho p, q, r là các biến mệnh đề. Chứng minh các dạng mệnh đề sau là các hằng đúng:

a) $((p \to q) \wedge p) \to q$

b) $((p \to q) \neg\wedge q) \to\neg p$

c) $((p \vee q) \wedge \neg q) \to p$

d) $(p \to q) \leftrightarrow ((p \wedge\neg q) \to 0)$

e) $((p \to q) \wedge (q \to r)) \to (p \to r)$

f) $((p \vee q) \to r) \leftrightarrow ((p \to r) \wedge (q \to r))$

g) $(p \to q) \to ((q \to r) \to (p \to r))$

4. Cho p, q, r là các biến mệnh đề. Chứng minh:

a) $((p \to r) \vee (q \to r))\to(p \to r)\Leftrightarrow p \to (q \vee r)$

b) $((\neg p\wedge q \neg\wedge r) \neg\to q) \to(p \vee r) \Leftrightarrow p\vee q \vee r$

c) $((p \to r)\wedge(q \to r))\to(p\to q) \Leftrightarrow\neg p\vee q \vee \neg r$

d) $(p \to q) \wedge (p \to r) \Leftrightarrow p \to (q \wedge r)$

5. Hãy kiểm tra các suy luận sau:

a)

$p \to q$

$\bar{q}$

$\bar{r}$

$\therefore \overline{p \vee r}$

b)

$p \wedge q$

$p \to (r \wedge q)$

$r \to (s \vee t)$

$\bar{s}$

$\therefore t$

c)

$p \to (q \to r)$

$\bar{q} \to \bar{p}$

p

$\therefore r$

d) $\mathbf{p} \leftrightarrow \mathbf{q}$
$\mathbf{q} \rightarrow \mathbf{r}$
$\mathbf{r} \vee \overline{\mathbf{s}}$
$\overline{\mathbf{s}} \rightarrow \overline{\mathbf{q}}$

$\therefore \mathbf{s}$

e) p
$\overline{p} \rightarrow q$
$(q \wedge r) \rightarrow s$
$t \rightarrow r$

$\therefore \overline{s} \rightarrow \overline{t}$

****

Chương 6. Đồ thị

6.1. Các khái niêm

- *Đồ thị vô hướng G = (V,E)* bao gồm V là tập các *đỉnh*, và E là tập các *cạnh*, mỗi cạnh nối hai đỉnh lại với nhau - không phân biệt thứ tự hai đỉnh đầu mút. Trường hợp mỗi cạnh đều có phân biệt thứ tự hai đỉnh đầu mút thì gọi là *đồ thị có hướng* và cạnh thường gọi là *cung*. Trường hợp có một số cạnh có hướng và một số cạnh vô hướng thì gọi là *đồ thị hỗn hợp*, khi đó có thể xem là đồ thị có hướng bằng cách thay mỗi cạnh bởi hai cung ngược chiều nhau.

Các thuật ngữ sau dùng trên đồ thị vô hướng.

- *Hai đỉnh u và v của đồ thị vô hướng G được gọi là kề nhau* nếu (u,v) là cạnh của đồ thị G. Nếu e = (u, v) là cạnh của đồ thị thì e *liên thuộc* với hai đỉnh u và v,vài ngược lại đỉnh u và đỉnh v liên thuộc e.

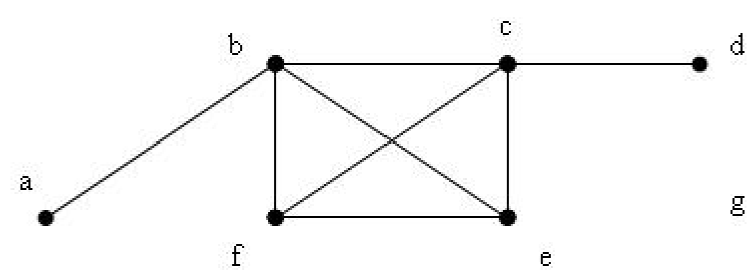
- *Hai cạnh e_1 và e_2 gọi là cạnh song song (cạnh bội*) nếu chúng cùng liên thuộc với một cặp đỉnh. Ký hiệu e_1 // e_2.

- *Khuyên* là cạnh liên thuộc với 1 đỉnh duy nhất. Cạnh e là khuyên có dạng e = (u, u).

- *Đơn đồ thị* là đồ thị không có khuyên và không có cạnh song song. Ngược lại gọi là *đa đồ thị*.

- *Bậc của đỉnh v* là số cạnh liên thuộc v, ký hiệu là deg(v) hay ∂(v). Với đồ thị có hướng, *bậc ra (bậc vào) của đỉnh v* là số cung liên thuộc đi ra khỏi nó (đi vào nó) và ký hiệu là $deg^+(v)$ ($deg^-(v)$). Trên đồ thị vô hướng thì khuyên đếm bậc cho đỉnh 2 đơn vị.

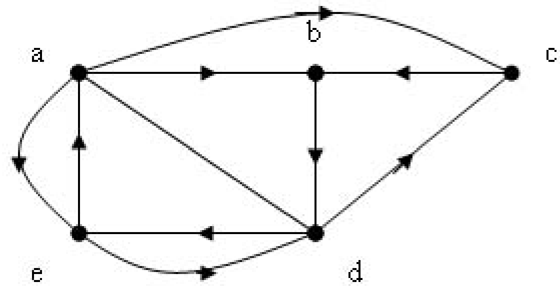
Ví dụ.



Đồ thị vô hướng

deg(a) = 1, deg(b) = 4, deg(c) = 4, deg(f) = 3,

deg(d) = 1, deg(e) = 3, deg(g) = 0



Đồ thị có hướng. Cạnh e có 2 cung ngược nhau.

$deg^{-}(a)=2, deg^{-}(b)=2, deg^{-}(c)=2, deg^{-}(d)=3, deg^{-}(e) = 2.$

$deg^{+}(a)=4, deg^{+}(b)=1, deg^{+}(c)=1, deg^{+}(d)=3, deg^{+}(e)=2.$

- *Đỉnh cô lập* là đỉnh không liên thuộc với cạnh nào cả. Đỉnh cô lập có bậc 0.

- *Đỉnh treo* là đỉnh bậc 1.

- *Đường đi độ dài n từ v_0 đến* v_n là dãy xen kẻ gồm *n+1* đỉnh và *n* cạnh liên thuộc nhau, bắt đầu từ v_0 và kết thúc là v_n. *Hai đỉnh liên thông* nếu có đường đi nối chúng. Với đơn đồ thị thì có thể mô tả đường đi bằng dãy đỉnh.

- *Đường đi đơn* là đường đi không lặp đỉnh.

- *Đồ thị liên thông* là đồ thị mà mọi cặp đỉnh đều liên thông. Đồ thị có hướng liên thông thì gọi là *liên thông mạnh*. Đồ thị có hướng không liên thông, nhưng xóa hướng tất cả các cung để có được đồ thị vô hướng liên thông thì gọi là *liên thông yếu*.

- *Chu trình* là đường đi không lặp cạnh có đỉnh đầu và đỉnh cuối trùng nhau.

- *Chu trình đơn* là chu trình không lặp đỉnh.

- *Đường đi Euler* là đường đi qua tất cả các đỉnh và tất cả các cạnh của đồ thị, mỗi cạnh một lần. *Đồ thị bán Euler* là đồ thị có đường đi Euler.

- *Chu trình Euler* là đường đi qua tất cả các đỉnh và tất cả các cạnh của đồ thị, mỗi cạnh một lần. *Đồ thị Euler* là đồ thị có chu trình Euler.

- *Đường đi Hamilton* là đường đi qua tất cả các đỉnh của đồ thị, mỗi đỉnh một lần. *Đồ thị bán Hamilton* là đồ thị có đường đi Hamilton.

- *Chu trình Hamilton* là chu trình qua tất cả các đỉnh của đồ thị, mỗi đỉnh một lần, trừ đỉnh đầu và cuối. *Đồ thị Hamilton* là đồ thị có chu trình Hamilton.

- *Đồ thị có trọng số* là đồ thị mà mỗi cạnh được gán cho một nhãn số. Ký hiệu G=(V,E,W), W là tập trọng số của các cạnh. Cạnh e có nhãn k thì k gọi là *trọng số* hay *độ dài* của cạnh e. *Trọng số* *của đồ thị* là tổng trọng số của các cạnh trên đồ thị. *Độ dài của đường đi* là tổng trọng số của các cạnh trên đường đi.

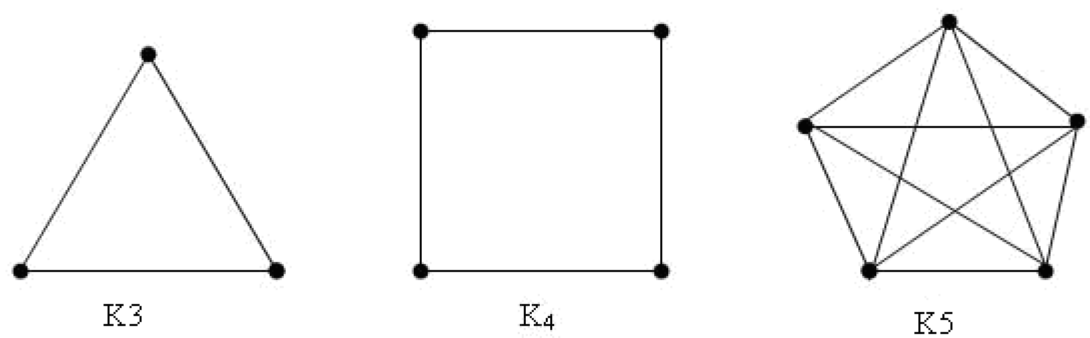
- *Đồ thị con* G' = (V', E') của đồ thị G = (V, E) là đồ thị thỏa mãn 2 điều kiện sau:

1) $V' \subseteq V$ và $E' \subseteq E$.
2) Nếu $e=(u,v) \in E'$ thì $u, v \in V'$.

Trong trường hợp đồ thị không liên thông, nó sẽ rã ra thành một số đồ thị con liên thông đôi một không có đỉnh chung. Những đồ thị con liên thông như vậy ta sẽ gọi là các *thành phần liên thông* của đồ thị. Đồ thị liên thông chỉ có 1 thành phần liên thông.

- *Cầu* là cạnh mà nếu loại bỏ nó khỏi đồ thị làm tăng số thành phần liên thông của đồ thị.

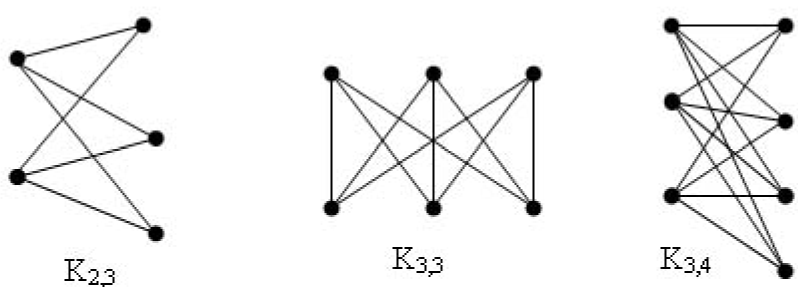
- *Đồ thị đầy đủ* là đơn đồ thị vô hướng mà giữa hai đỉnh bất kỳ có 1 cạnh nối chúng. Đồ thị đầy đủ *n* đỉnh, ký hiệu là K_n.



Các đồ thị đầy đủ

Đồ thị đầy đủ K_n có tất cả n(n-1)/2 cạnh, nó là đơn đồ thị có nhiều cạnh nhất.

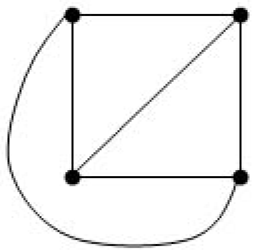
- Đồ thị phân đôi (hai phía)G=(V,E) là đồ thị mà tập đỉnh V của nó được phân hoạch thành hai tập X và Y sao cho mỗi cạnh e=(u,v) thì $u \in X$ và $v \in Y$.

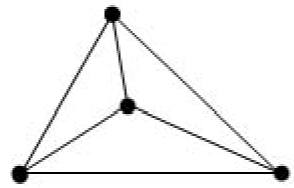


Đồ thị hai phía đầy đủ

- *Đồ thị phẳng* là đồ thị có thể vẽ nó trên mặt phẳng sao cho các cạnh của nó không cắt nhau ngoài đỉnh. Cách vẽ như vậy sẽ được gọi là biểu diễn phẳng của đồ thị.

Đồ thị K_4 là phẳng, vì có thể vẽ nó trên mặt phẳng sao cho các cạnh của nó không cắt nhau ngoài đỉnh.





- Cho đồ thị G = (*V*, *E*). B $\subseteq$ *V* gọi là *tập ổn định trong* của đồ thị G nếu: $\forall$ *v*,*w* $\in$ B không có cạnh (v,w).

Ví dụ. Hãy đặt 8 quân hậu lên một bàn cờ sao cho chúng không ăn lẫn nhau.

-*Tập ổn định trong lớn nhất* là tập ổn định trong có nhiều phần tử nhất. Lực lượng của tập ổn định trong lớn nhất được gọi là *số ổn định trong* của đồ thị đó.

- *Tập ổn định ngoài* C của đồ thị G nếu: $\forall$ w $\notin$ C , $\exists$ *v* $\in$ C : w kề với *v*.

Ví dụ. Hãy đặt 5 quân hậu lên một bàn cờ sao cho chúng kiểm soát được toàn bộ bàn cờ.

- *Tập ổn định ngoài bé nhất* là tập ổn định ngoài có ít phần tử nhất. Lực lượng của tập ổn định ngoài bé nhất được gọi là *số ổn định ngoài* của đồ thị**.**

- *Nhân* của đồ thị là tập vừa ổn định trong vừa ổn định ngoài của đồ thị.

- *Sắc số* của một đồ thị là số màu ít nhất dùng để tô các đỉnh của đồ thị đó.

6.2. Các định lý

Định lý 1(Euler). *Cho G = (V, E) là đồ thị vô hướng liên thông.*
G là đồ thị Euler khi và chỉ khi G không có đỉnh bậc lẻ.

Hệ quả. *Cho G = (V, E) là đồ thị vô hướng liên thông.*
G là đồ thị bán Euler khi và chỉ khi G không có đỉnh bậc lẻ hoặc có 2 đỉnh bậc lẻ.

Định lý 2. *Cho G = (V, E) là đồ thị vô hướng.*
Tổng bậc của tất cả các đỉnh bằng hai lần số cạnh.

*Ví dụ***.** Đồ thị với *n* đỉnh có bậc là 6 có bao nhiêu cạnh?

*Giải***.** Theo định lý 1 ta có $2m = 6n$. Từ đó suy ra số cạnh của đồ thị là $3n$.

*Ví dụ***.** Trong kỳ nghỉ hè, một nhóm gồm 9 sinh viên quy ước: mỗi người viết thư cho đúng 3 người trong nhóm. Chứng tỏ rằng phải có ít nhất một người không hồi âm lại người viết thư cho mình.

*Giải***.** Biểu diễn bằng đồ thị như sau: Mỗi người là một đỉnh; mỗi lá thư là một cung. Nếu ai nhận được thư đều có hồi âm thì trở thành đồ thị vô hướng. Tổng bậc các đỉnh là 9x3=27. Theo định lý trên, đây phải là số chẵn (mâu thuẩn).

Hệ quả. *Trong đồ thị vô hướng, số đỉnh bậc lẻ là một số chẵn.*

Định lý 3. *Đơn đồ thị là đồ thị phân đôi khi và chỉ khi nó không chứa chu trình độ dài lẻ.*

Định lý 4 (Kuratovski). *Đồ thị là phẳng khi và chỉ khi nó không chứa đồ thị con đồng cấu với* $K_{3,3}$ *hoặc* K_5.

Định lý 5 (Công thức Euler). *Giả sử G là đồ thị phẳng liên thông với v đỉnh, e cạnh. Gọi f là số miền của mặt phẳng bị chia bởi biểu diễn phẳng của G. Khi đó*

$$f = e-v + 2$$

Ví dụ. Cho G là đồ thị phẳng liên thông với 20 đỉnh, mỗi đỉnh đều có bậc là 3. Hỏi mặt phẳng bị chia làm bao nhiêu phần bởi biểu diễn phẳng của đồ thị G?

Giải. Do mỗi đỉnh của đồ thị đều có bậc là 3, nên tổng bậc của các đỉnh là 3x20=60. Từ đó suy ra số cạnh của đồ thị e = 60/20 = 30. Vì vậy, theo công thức Euler, số miền cần tìm là f = 30-20+2 = 12.

6.3. Ma trận kề-Ma trận trọng số

Cho G=(V,E). Ma trận kề của đồ thị G là ma trận A= (a_{ij})nxn được xác định như sau:

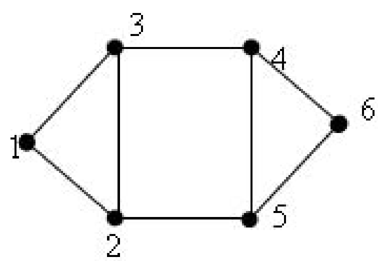
a_{ij} = 1 nếu có cạnh/cung (i,j) và a_{ij} = 0 nếu không có cạnh (i,j).

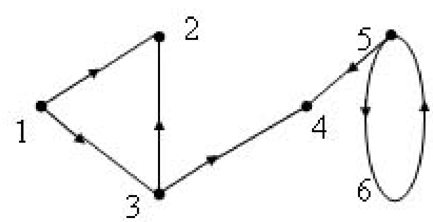
Đồ thị vô hướng có ma trận kề đối xứng.

	v1	v2	v3	v4	v5	v6
v1	0	1	1	0	0	0
v2	1	0	1	0	1	0
v3	1	1	0	1	0	0
v4	0	0	1	0	1	1
v5	0	1	0	1	0	1
v6	0	0	0	1	1	0

	v1	v2	v3	v4	v5	v6
v1	0	1	1	0	0	0
v2	0	0	0	0	0	0
v3	0	1	0	1	0	0
v4	0	0	0	0	0	0
v5	0	0	0	1	0	1
v6	0	0	0	0	1	0

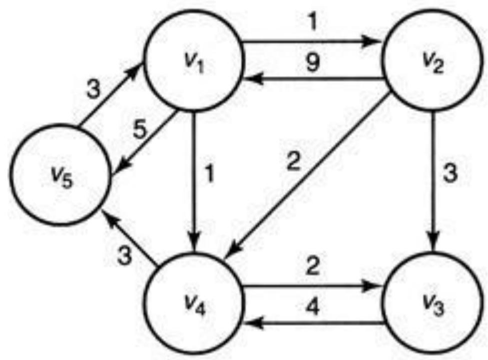
là ma trận kề của các đồ thị:





Đồ thị vô hướng G và Đồ thị có hướng G_1

Cho G=(V,E,W). Ma trận kề của đồ thị G là ma trận trọng số A= $(a_{ij})_{nxn}$ được xác định như sau: a_{ij} = là trọng số cạnh (i,j) , $a_{ij} = \infty$ nếu không có cạnh (i,j) và a_{ii} =0 nếu không có khuyên.



có ma trận trọng số là

	v1	v2	v3	v4	v5
v1	0	1	∞	1	5
v2	9	0	3	2	∞
v3	∞	∞	0	4	∞
v4	∞	∞	2	0	3
v5	3	∞	∞	∞	0

6.4. Thuật toán tìm chu trình Euler

Cho đồ thị Euler G=(V,E). Các bước của tìm chu trình Euler C của G được mô tả như sau:

Bước 1 Xuất phát từ v bất kỳ, tìm chu trình C.

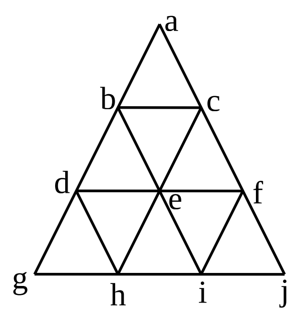
Bước 2 Xóa các cạnh trong C ra khỏi G và các đỉnh cô lập.

Bước 3 Nếu G=Ø thì C là chu trình Euler cần tìm.

Bước 4 - Tìm đỉnh chung v của C và G.
- Sắp xếp C bắt đầu từ v.
- Xuất phát từ v, tìm chu trình C'.

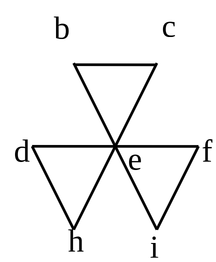
- Nối C’ vào C.
- Quay lại *bước 2*.

Ví dụ. Cho đồ thị G=(V, E) như sau.
a) Chứng tỏ G là đồ thị Euler.
b) Tìm chu trình Euler C của G

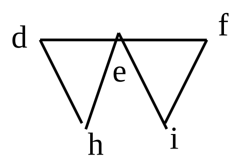


G=(V,E)

Giải.
a) G liên thông và tất cả các đỉnh bậc chẵn.
b) Xuất phát từ a, C = (a, b, d, g, h, i, j, f, c, a). Đồ thị còn lại G_1.



C và G_1 có đỉnh chung b. Sắp xếp C bắt đầu từ b, C=(b, d, g, h, i, j, f, c, a, b).
Trên G_1 có chu trình bắt đầu từ b, C_1=(b, c, e, b).
Nối C_1 vào C, C==(b, d, g, h, i, j, f, c, a, b, c, e, b).
Đồ thị còn lại G_2.



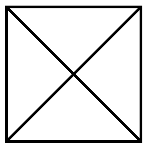
C và G_2 có đỉnh chung d. Sắp xếp C bắt đầu từ d, C= (d, g, h, i, j, f, c, a, b, c, e, b, d).
Trên G_2 có chu trình bắt đầu từ d, C_2 = (d, e, f, i, e, h, d).
Nối C_2 vào C, C=(d, g, h, i, j, f, c, a, b, c, e, b, d, e, f, i, e, h, d).
G_3=Ø thì C là chu trình Euler cần tìm.
Hay C= (a, b, c, e, b, d, e, f, i, e, h, d, g, h, i, j, f, c, a).
Tương tự cho việc tìm đường đi Euler trên đồ thị bán Euler.

****

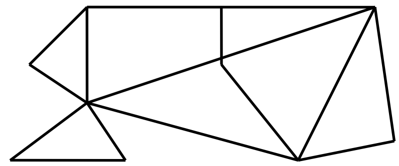
Bài tập

1. Các đồ thị sau có phải là đồ thị Euler không ? Nếu đúng thì chỉ ra chu trình Euler

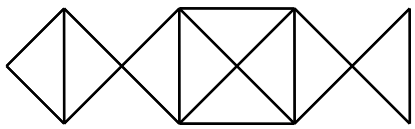
a)



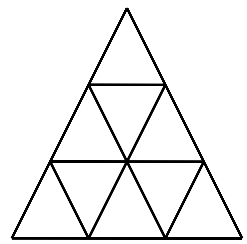
b)



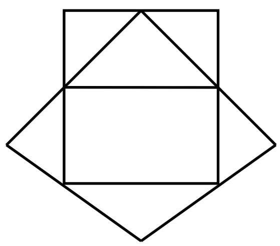
c)



d)



2. Chứng tỏ đồ thị sau là đồ thị bán Euler. Hãy chỉ ra một đường đi Euler.



3. Lập ma trận kề của các đồ thị trong các bài tập trên.

****

Chương 7. Các bài toán tối ưu trên đồ thị

7.1. Bài toán tìm đường đi ngắn nhất

7.1.1. Thuật toán Dijkstra

a) *Nội dung*

Thuật toán Dijkstra để tìm đường đi ngắn nhất từ đỉnh *a* đến đỉnh *z* trên đồ thị G=(V,E,W) bao gồm việc đánh nhãn cho các đỉnh. Ở mỗi bước, một số đỉnh có nhãn cố định và một số đỉnh có nhãn tam thời. Khi đỉnh *z* có nhãn cố định *Dz* thì *Dz* là độ dài của đường đi ngắn nhất từ *a* đến *z*.

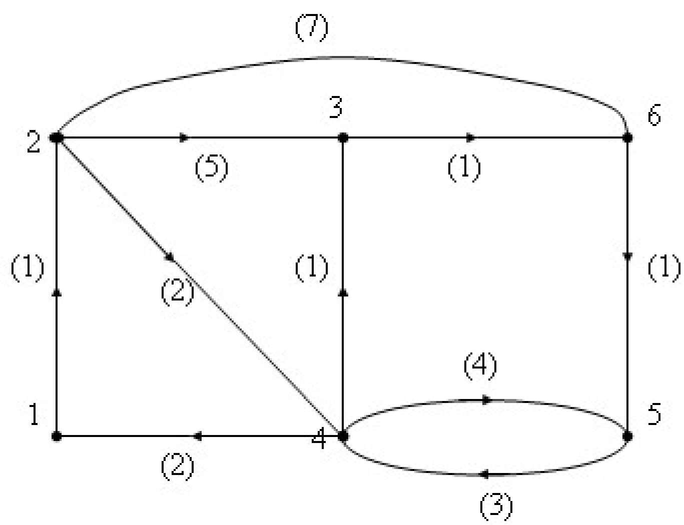
b) *Các bước của thuật toán*

Bước 1 $T=V$; $D_a = 0$; $D_i = \infty$, $v_i \neq a$.

Bước 2 Lặp cho đến khi $z \notin T$:

- Lấy ra khỏi T đỉnh v_i có D_i nhỏ nhất
- Đánh nhãn lại cho mọi v_j kề v_i và $v_j \in T$ theo công thức:
  $D_j = \min\{D_j , D_i + W_{ij}\}$

Ví dụ. Cho đồ thị G=(V,E,W), trọng số cạnh là số trong ngoặc đơn. Tìm đường đi ngắn nhất từ v_1 đến v_6.



G=(V,E,W)

Kết quả tính toán được trình bày theo bảng dưới đây. Qui ước v_i là đỉnh có nhãn nhỏ nhất được lấy ra khỏi T. Khi một đỉnh được lấy ra khỏi T thì có nhãn cố định, đánh dấu "*", các bước sau biến đổi, vì thế ta đánh dấu -.

T	v_i	D_1	D_2	D_3	D_4	D_5	D_6
{1..6}	–	0	∞	∞	∞	∞	∞
{2..6}	1	**0***	1	∞	∞	∞	∞
{3..6}	2	–	**1***	6	3	∞	7
{3,5,6}	4	–	–	4	3*	7	7
{5,6}	3	–	–	**4***	–	7	5
{5}	6	–	–	–	–	6	**5***

$v_6 \notin T$, thuật toán dừng. Độ dài đường đi ngắn nhất từ v_1 đến v_6 là $D_6 = 5$.

Nhận xét.

a) Khi đỉnh v_j có nhãn cố định D_j nhờ việc lấy ra khỏi T đỉnh v_i thì có cung (i,j) trên đường đi ngắn nhất từ *a* đến v_j. Để lấy cả đường đi ngắn nhất từ *a* đến *z* thì lần ngược từ *z* đến *a*.
b) Cũng có thể trình bày các bước của thuật toán như sau: Đánh cho mỗi đỉnh v_j một cặp nhãn [Dj, v_i], trong đó Dj như trên và v_i là đỉnh nối đỉnh *a* với v_j – nghĩa là có cạnh (i,j) trên đường đi từ *a* đến v_j. Cuối bảng, dựa vào nhãn thứ hai để lấy đường đi ngắn nhất từ *a* đến *z*.
c) Để lấy đường đi ngắn nhất từ *a* đến mọi đỉnh, thay vòng lặp "Lặp cho đến khi $z \notin T$" bởi "Lặp cho đến khi $T=\varnothing$"

Với nhận xét b), có thể dùng bảng sau

T	v_i	D_1	D_2	D_3	D_4	D_5	D_6
{1..6}	–	[0,1]	[∞,1]	[∞,1]	[∞,1]	[∞,1]	[∞,1]
{2..6}	1	*	[1,1]	[∞,1]	[∞,1]	[∞,1]	[∞,1]
{3..6}	2	–	*	[6,2]	[3,2]	[∞,1]	[7,2]
{3,5,6}	4	–	–	[4,4]	*	[7,4]	[7,2]
{5,6}	3	–	–	*	–	[7,4]	[5,3]
{5}	6	–	–	–	–	[6,6]	*

Đường đi ngắn nhất từ v1 đến v6 là: v6<-v3<-v4<-v2<-v1

7.1.2. Thuật toán Floyd

a) *Nội dung*

Giả sử đồ thị G=(V,E,W), |V|=*n*, được biểu diễn bằng ma trận trọng số A_{nxn}. Thuật toán Floyd để tìm đường đi ngắn nhất giữa mọi cặp đỉnh trên đồ thị G và lưu độ dài trong chính ma trận A. Thuật toán thực hiện *n* bước lặp, ở bước *k* thì a_{ij} là độ dài đường đi ngắn nhất từ v_i đến v_j không qua mọi đỉnh lớn hơn *k*, và ở bước này đặt

$$a_{ij} = \min \{a_{ij}, a_{ik} + a_{kj}\}$$

b) *Mô tả thuật toán*

Giả sử ma trận A_{nxn} được khai báo toàn cục, thuật toán được mô tả như sau:

```
void Floyd( )
{
	for (k=1; k<=n; k++)
		for (i=1; i<=n; i++)
			for (j=1; j<=n; j++)
				if (a_ik + a_kj < a_ij) a_ij = a_ik + a_kj;
}
```

7.2. Bài toán tìm cây phủ nhỏ nhất

7.2.1. Các khái niệm

- *Cây* là đồ thị vô hướng liên thông không chu trình.

- Cho đồ thị vô hướng G = (V,E), *cây phủ T* của đồ thị G là đồ thị con chứa tất cả các đỉnh của G và T là một cây.

- Cho đồ thị vô hướng G = (V,E,W), *cây phủ nhỏ nhất* của đồ thị G là cây phủ có trọng số nhỏ nhất trong tất cả các các phủ của G.

Định lý 1. Giả sử T=(V,E) là đồ thị vô hướng gồm *n* đỉnh. Khi đó các mệnh đề sau đây là tương đương:

(1) T là cây
(2) T không chứa chu trình và có *n-1* cạnh
(3) T liên thông và có *n-1* cạnh

Định lý 2. *G có cây phủ khi và chỉ khi G liên thông*.

7.2.2. Thuật toán Prim

a) *Nội dung*
Cho đồ thị G=(V,E,W), thuật toán Prim để tìm cây phủ nhỏ nhất T của G gồm một số bước lặp. Ở mỗi bước, tâp đỉnh V được chia thành 3 loại:
- Đỉnh trên T.
- Đỉnh rìa: đỉnh chưa thuộc T nhưng kề với T.
- Đỉnh khác.

Và ở mỗi bước chọn trong số đỉnh rìa một đỉnh có cạnh nối cây T với trọng số nhỏ nhất để đưa cạnh đó (và đỉnh đó) vào T.

b) *Mô tả thuật toán*

Các bước chính của thuật toán Prim tìm cây phủ nhỏ nhất T của đồ thị liên thông có trọng số G được mô tả như sau:

Bước 1 T:={v}; v bất kỳ

Bước 2 Lặp n-1 lần:

- Tìm đỉnh rìa v có cạnh e nối T với w(e) nhỏ nhất.
- Đưa e và v vào T.

Ví dụ. Tìm cây phủ nhỏ nhất T của đồ thị được cho bởi ma trận trọng số sau.

	v1	v2	v3	v4	v5	v6
v1	0	33	17	∞	∞	∞
v2	33	0	18	20	∞	∞
v3	17	18	0	16	4	∞
v4	∞	20	16	0	9	8
v5	∞	∞	4	9	0	14
v6	∞	∞	∞	8	14	0

Bảng dưới đây đánh nhãn của các đỉnh trong các bước lặp của thuật toán. Mỗi đỉnh v_j được đánh một cặp nhãn [w_{ij}, v_i], trong đó $v_i \in T$ nối v_j có cạnh nối (i,j) với w_{ij} nhỏ nhất. Ở mỗi bước tìm nhãn thứ nhất w_{ij} nhỏ nhất để đánh dấu * cho v_j là đỉnh được chọn để bổ sung vào cây phủ nhỏ nhất T, và đánh lại nhãn cho cho các đỉnh v_k kề v_j nếu cần (w_{jk} nhỏ hơn nhãn cũ của v_k). Khi đưa v_j vào cây phủ nhỏ nhất T thì nhãn của nó không còn bị biến đổi trong các bước lặp tiếp theo, vì vậy được đánh dấu "–":

v1	v2	v3	v4	v5	v6	v	e
*	[33,1]	[17,1]	[∞ ,1]	[∞ ,1]	[∞ ,1]	1	–
–	[18,3]	*	[16,3]	[4,3]	[∞ ,1]	3	(1,3)
–	[18,3]	–	[9,5]	*	[14,5]	5	(3,5)
–	[18,3]	–	*	–	[8,4]	4	(5,4)
–	[18,3]	–	–	–	*	6	(4,6)
–	*	–	–	–	–	2	(3,2)

7.2.3. Thuật toán Kruskal

Các bước của thuật toán Kruskal tìm cây phủ nhỏ nhất T của đồ thị liên thông có trọng số G được mô tả như sau:

Bước 1 T:=V; T không có cạnh

Bước 2 Lặp n-1 lần:

- Tìm cạnh e có trọng số nhỏ nhất và đưa vào T không tạo chu trình.
- Đưa e vào T.

Thuật toán thuận tiện bằng cách sắp xếp E theo trọng số tăng dần, rồi lần lượt duyệt từng cạnh để kiểm tra.
Cụ thể, thuật toán có thể mô tả như sau:

```
void  Kruskal( )
{
        T:=V;
        // sắp xếp E tăng dần theo trọng số.
        i=1; c=0;
        While (c < n-1) {
                if (T ∪ { eᵢ} không chứa chu trình) { T= T∪ { eᵢ} ; c++;}
                i++;
        }
}
```

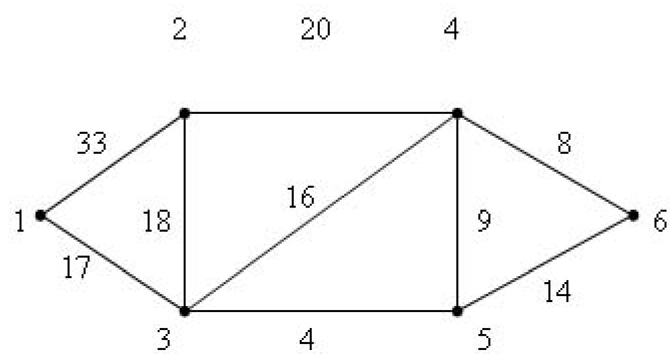
*Ví dụ.*Tìm cây khung nhỏ nhất của đồ thị cho trong hình dưới.

Bước khởi tạo. Đặt T:=∅ . Sắp xếp các cạnh của đồ thị theo thứ tự trọng số tăng dần có dãy cạnh:

(3,5) , (4,6) , (4,5) , (5,6) , (3,4) , (1,3) , (2,3) , (2,4) , (1,2)

với độ dài tương ứng của chúng

4, 8, 9, 14, 16, 17, 18, 20, 23.



Đồ thị và cây khung nhỏ nhất

Ở ba lần lặp đầu tiên ta lần lượt bổ sung vào tập T các cạnh (3,5) , (4,6) , (4,5). Rõ ràng nếu thêm cạnh (5,6) vào T thì sẽ cùng 2 cạnh (4,5), (4,6) đã có trong T tạo thành chu trình. Tình huống tương tự cũng xảy ra đối với cạnh (3,4) là cạnh tiếp theo của dãy. Tiếp theo ta bổ sung cạnh (1,3), (2,3) vào T và thu được tập T gồm 5 cạnh:

T = { (3,5) , (4,6) , (4,5) , (1,3) , (2,3) }

Chính là tập cạnh của cây khung nhỏ nhất cần tìm.

Mặc dù thuật toán Kruskal thuận tiện trong tính toán thủ công, nhưng thuật toán Prim được ưa chuộng hơn.

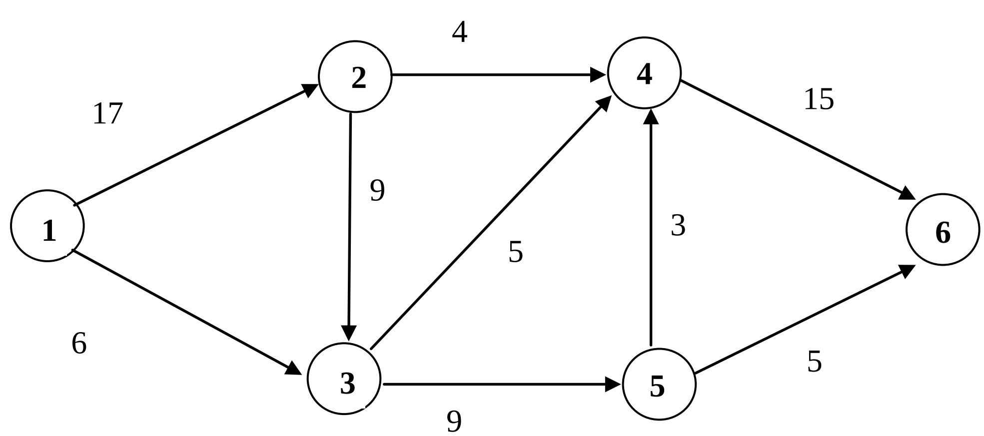
7.3. Bài toán tìm luồng cực đại

7.3.1. Các khái niệm

- *Mạng* là đồ thị có hướng, có trọng số G=(V,E,C) thỏa mãn:

- G liên thông yếu (bỏ hướng các cạnh thì liên thông).
- Có duy nhất một đỉnh *s* không có cung vào gọi là *đỉnh phát* và duy nhất một đỉnh *t* không có cung ra gọi là *điểm thu*.
- Mỗi cung $(i,j)\in E$ được gán một số $c_{ij} \geq 0$ gọi là khả năng thông qua của cung (i,j).

Ví dụ.



- Cho mạng G=(V,E,C). *Luồng* $F=(f_{ij})$ trên mạng G là việc gán cho mỗi cung $(i,j)\in E$ một số f_{ij} thoả mãn:

- $\forall (i,j)\in E: 0\leq f_{ij} \leq c_{ij}$
- $\forall v_i \notin \{s, t\}$: $\sum_{(j,i)\in E} f_{ji} = \sum_{(i,j)\in E} f_{ij}$

Rõ ràng có: $\sum_{(s,j)\in E} f_{sj} = \sum_{(j,t)\in E} f_{jt}$

Giá trị này gọi là *giá trị của luồng F*, ký hiệu V_f.

- Cho mạng G=(V,E,C). *Luồng cực đại* trên mạng G là luồng có giá trị lớn nhất trong tất cả các luồng trên G.

- Cho mạng G=(V,E,C). *Lát cắt* $S=(V_1,V_2)$ là một phân hoạch tập đỉnh V của mạng G thành hai tập V_1 và $V_2 = V \setminus V_1$, trong đó $s \in V_1$, $t \in V_2$. Khả năng thông qua hay giá trị của lát cắt $S=(V_1, V_2)$ là số $c(V_1, V_2) = \sum c_{ij}$ với $v_i \in V_1$ và $v_j \in V_2$.

Định lý Ford-Fulkerson. *Luồng cực đại bằng lát cắt cực tiểu.*

7.3.2. Thuật toán Ford-Fulkerson

Thuật toán Ford-Fulkerson để tìm luồng cực đại F trên mạng G = (V,E,C) gồm các bước sau:

Bước 1 F=0; // khởi đầu luồng 0, $\forall (i,j) \in E: f_{ij}=0$

Bước 2 Lặp cho đến khi hết đường tăng luồng:

- Tìm đường tăng luồng P: s→t. với lượng tăng luồng δ.
- Tăng luồng dọc theo P một lượng δ.

Đường tăng luồng P tìm được có dạng

P: s→…→**i**→**j**→…→t thì (i,j) là cung thuận

P: s→…→**i**←**j**→…→t thì (j,i) là cung nghịch

Bước tìm đường tăng luồng P như sau:

- Đặt nhãn s là ∞.

- Lặp cho đến khi *t* có nhãn δ_t: khi đỉnh v_i vừa có nhãn thì đánh nhãn cho mọi v_j kề vô hướng v_i nếu thỏa mãn một trong hai trường hợp sau:

a) Nếu có cung (i,j) và $c_{ij}- f_{ij}>0$ (chưa bão hòa) thì đặt $\delta_j=\min\{\delta_i,c_{ij}- f_{ij} \}$, nạp cung thuận (i,j) vào P.

b) Nếu có cung (j,i) và $f_{ji}>0$ thì đặt $\delta_j= \min\{\delta_i, f_{ji} \}$, nạp cung nghịch (j,i) vào P.

Khi đỉnh *t* có nhãn δ_t thì có lượng tăng luồng là $\delta = \delta_t$; ngược lại, nếu không đánh nhãn được đỉnh *t* thì hết đường tăng luồng (thuật toán dừng).

Tăng luồng dọc theo P một lượng δ theo công thức:

$f'_{ij} =f_{ij} +\delta$, *nếu (i,j) là cung thuận.*

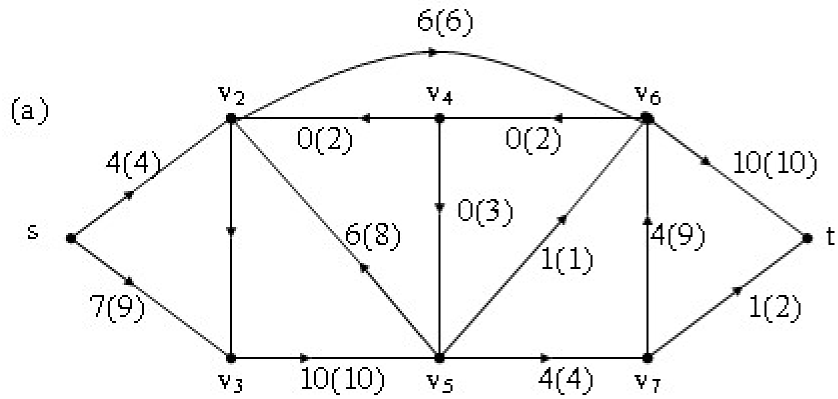
$f'_{ji} =f_{ji} - \delta$, *nếu (j,i) là cung nghịch.*

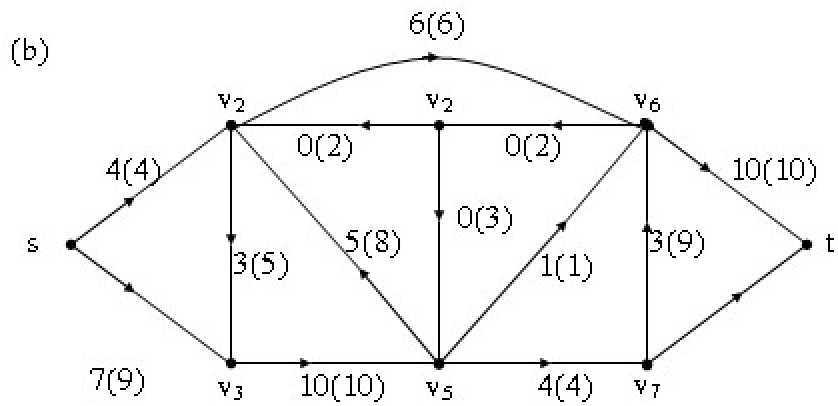
Thuật toán Ford-Fulkerson có thể mô tả như sau:

```
void Max_Flow( ){
        F=0; // Khởi đầu từ luồng với giá trị 0
        stop:=0;
        while (!stop){
                if (Tìm được đường tăng luồng P) <Tăng luồng dọc theo P>;
        else stop=1;
```

}

Ví dụ.





Hai số viết bên cạnh mỗi cung là khả năng thông qua của cung (số trong ngoặc) và luồng trên cung.

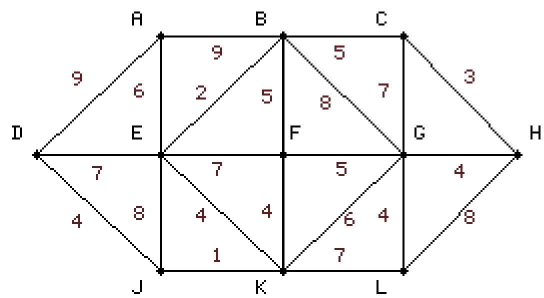
Luồng trên là cực đại. Lát cắt cực tiểu là X = { s, v_2, v_3, $v_{5\}}$, X^*= { v_4, v_6, v_7, t} .

Giá trị luồng cực đại là F_{max}=11.

****

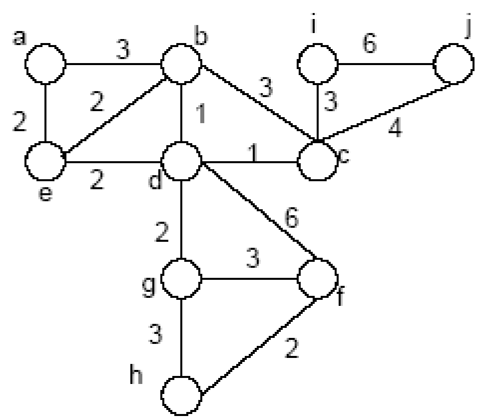
Bài tập

1. Cho đồ thị G=(V,E,W)



a) Viết ma trận trọng số của G.
b) Tìm đường đi ngắn nhất từ A đến H.
c) Tìm cây phủ nhỏ nhất của G.

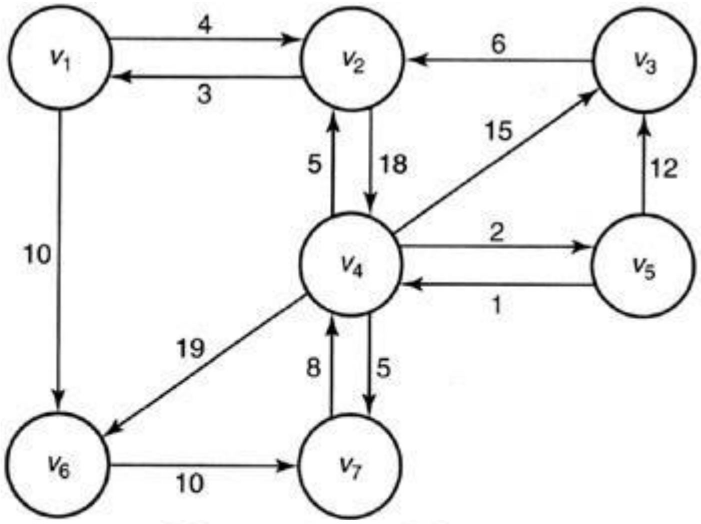
2. Cho đồ thị G=(V,E,W)

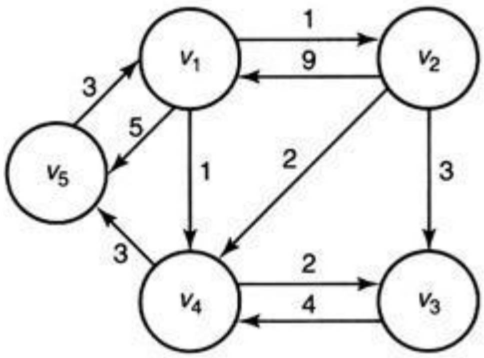


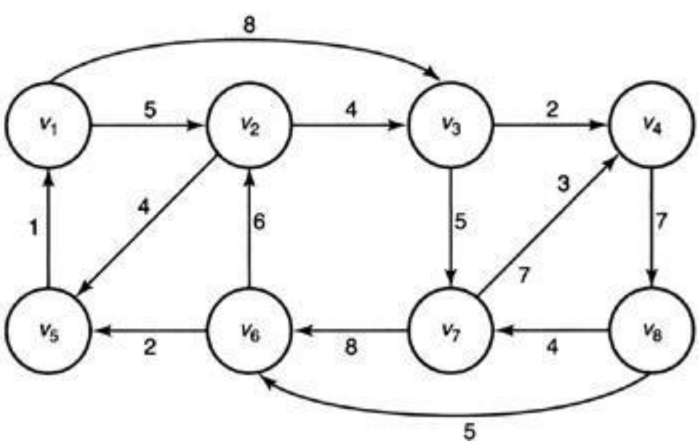
a) Tìm đường đi ngắn nhất từ *a* đến *j*.

b) Tìm cây phủ nhỏ nhất của G.

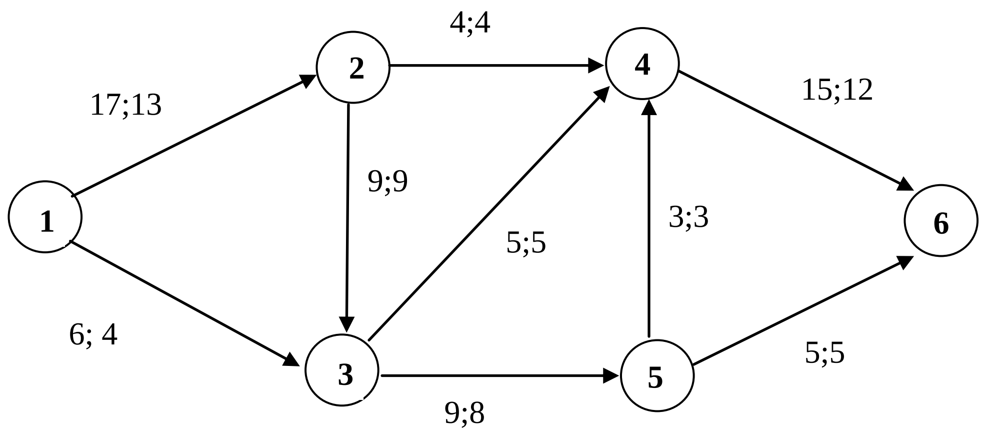
3. Dùng thuật toán Dijkstra để tìm đường đi ngắn nhất từ đỉnh v_1 đến mọi đỉnh khác trong các đồ thị sau.





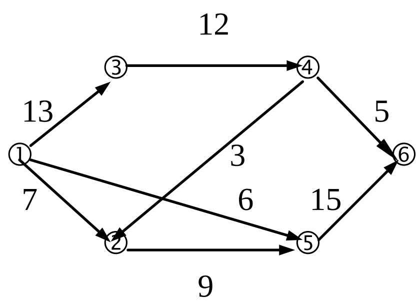


4. Chứng tỏ luồng sau là luồng cực đại

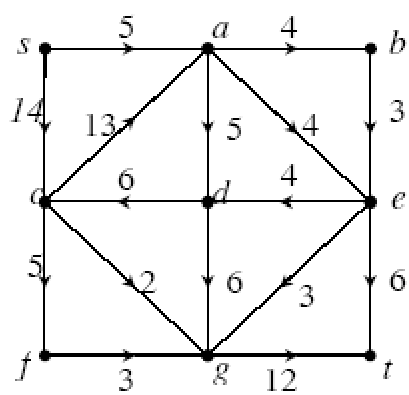


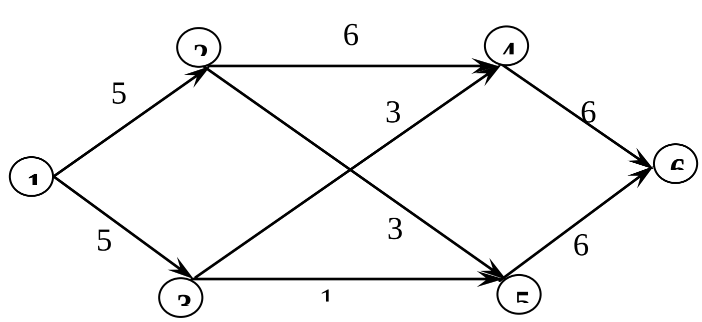
5. Tìm luồng cực đại trên các mạng sau

a)



b)





****