TOÁN RỜI RẠC 1

CHƯƠNG 1

Giảng viên: Vũ Văn Thỏa

CHƯƠNG 1. KIẾN THỨC CƠ SỞ

- Logic mệnh đề
- Vị ngữ và lượng từ
- Ứng dụng trên máy tính
- Cơ bản về tập hợp
- Thuật toán

1.1 Logic mệnh đề

- Khái niệm mệnh đề
- Các toán tử logic
- Một số loại mệnh đề
- Sự tương đương của các mệnh đề

1.1.1 Khái niệm mệnh đề

- Một **mệnh đề** là một khẳng định (câu) chỉ nhận 1 trong hai giá trị đúng hoặc sai.

- Các chữ cái hay được dùng để ký hiệu các mệnh đề: p, q, r, s, ...

- ***Giá trị chân lý*** của mệnh đề là đúng ký hiệu là T (hoặc 1) hoặc sai ký hiệu là F (hoặc 0).

Ví dụ 1

Tất cả các câu sau đây đều là mệnh đề:

1. "Hà nội là thủ đô của Việt nam"

   (Mệnh đề đúng)

2. "1 + 1 = 2"

   (Mệnh đề đúng)

3. "2 + 2 = 5"

   (Mệnh đề sai)

Ví dụ 2

Tất cả các câu sau đây không phải là mệnh đề:

1. “Anh đi đâu đấy?”
2. “Hãy cẩn thận trong khi làm bài”
3. “ $x + 2 = 5$”
4. “$x+y = z$”

1.1.2 Các toán tử logic

a) Toán tử phủ định (not)

- Cho mệnh đề p.

Mệnh đề phủ định của p được ký hiệu là $\neg p$ hoặc $\overline{p}$:

$\neg p$ = "không phải p";

$\neg p$ nhận giá trị đúng khi p sai và nhận giá trị sai khi p đúng.

Bảng giá trị chân lý toán tử phủ định:

p	¬p
0	1
1	0

b) *Hội (giao) của 2 mệnh đề* ($p \wedge q$)

- *Cho p và q là hai mệnh đề. Câu "p và q" cũng là một mệnh đề, ký hiệu $p \wedge q$ và đọc là p hội q.*
- *$p \wedge q$ đúng khi cả p và q đều đúng, sai trong các trường hợp còn lại.*
- ***Ví dụ:***

Cho mệnh đề p = "Hôm nay thứ sáu", mệnh đề q = "Hôm nay trời mưa“

$\Rightarrow$ *$p \wedge q$ = “Hôm nay thứ sáu và trời mưa”*

Bảng giá trị chân lý toán tử hội:

p	q	$p \wedge q$
0	0	0
0	1	0
1	0	0
1	1	1

c) Tuyển (hợp) của 2 mệnh đề ($p \vee q$)

- Cho p và q là hai mệnh đề. Câu "p hoặc q" là một mệnh đề, ký hiệu $p \vee q$, đọc là p hoặc q hay tuyển của p và q.
- $p \vee q$ đúng khi p hoặc q đúng, sai khi cả p và q đều sai.
- **Ví dụ:** Cho mệnh đề p = "Hôm nay thứ sáu", mệnh đề q = "Hôm nay trời mưa"

$\Rightarrow$ $p \vee q$ = "Hôm nay hoặc là thứ sáu, hoặc là trời mưa"

Bảng giá trị chân lý toán tử tuyển:

p	q	$p \vee q$
0	0	0
0	1	1
1	0	1
1	1	1

d) *Tuyển loại trừ của 2 mệnh đề* ($p \oplus q$)

- Cho p và q là hai mệnh đề. Câu "p hoặc q nhưng không cả hai" là một mệnh đề, ký hiệu $p \oplus q$, đọc là p hoặc q nhưng không cả hai hay tuyển loại trừ của p và q.
- $p \oplus q$ đúng khi p đúng, q sai hoặc q đúng, p sai và sai trong các trường hợp còn lại.
- **Ví dụ:** Cho mệnh đề p = "Hôm nay thứ sáu", mệnh đề q = "Hôm nay trời mưa"

$\Rightarrow$ $p \oplus q$ = “Hôm nay hoặc là thứ sáu, hoặc là trời mưa, nhưng không cả hai”

Bảng giá trị chân lý toán tử tuyển loại trừ:

p	q	$p \oplus q$
0	0	0
0	1	1
1	0	1
1	1	0

e) *Phép kéo theo của 2 mệnh đề* ($p \to q$)

- Cho p và q là hai mệnh đề. Câu "p suy ra q" cũng là một mệnh đề gọi là phép kéo theo, ký hiệu $p \to q$;
- Mệnh đề $p \to q$ chỉ sai khi p đúng mà q sai.
- Trong phép kéo theo thì p được gọi là giả thiết và q được gọi là kết luận.

Bảng giá trị chân lý toán tử kéo theo:

p	q	$p \to q$
0	0	1
0	1	1
1	0	0
1	1	1

Nhận xét:

- Vì phép kéo theo xuất hiện nhiều trong các suy luận toán học, nên có rất nhiều thuật ngữ được dùng để biểu đạt $p \to q$:

- Nếu p thì q
- p kéo theo q
- p là điều kiện đủ của q
- q là điều kiện cần của p

Ví dụ phép kéo theo:

- "Nếu tôi (A) trúng số độc đắc, tôi sẽ tặng anh (B) một chiếc dream"
- **Nhận xét:**

- Phép kéo theo trên là một phép kéo theo được dùng trong ngôn ngữ thông thường, vì ở đây có mối quan hệ giữa giả thiết và kết luận.

- Phép kéo theo này là đúng trừ phi người A trúng số độc đắc nhưng lại không mua cho người B xe dream.

Ví dụ phép kéo theo:

- "Nếu hôm nay là thứ sáu, thì 2 + 3 = 7"
- **Nhận xét**:

- Phép kéo theo trên là không có ý nghĩa, vì không có mối quan hệ giữa giả thiết và kết luận.
- Trong suy luận toán học thì cách nói trên có thể chấp nhận và đúng cho mọi ngày kể cả thứ sáu.

$\Rightarrow$ khái niệm toán học về sự kéo theo độc lập với mối quan hệ nhân quả giữa giả thiết và kết luận trong thực tiễn.

f) Phép tương đương ($p \leftrightarrow q$)

- Giả sử p và q là hai mệnh đề. Mệnh đề tương đương (ký hiệu $p \leftrightarrow q$) là mệnh đề chỉ đúng khi p và q cùng đúng hoặc cùng sai.
- Mệnh đề $p \leftrightarrow q$ đọc là ***"p nếu và chỉ nếu q"***, "p là cần và đủ đối với q" hoặc "nếu p thì q và ngược lại“.
- K***hông đọc*** là "p tương đương với q", thuật ngữ này mang một ý nghĩa khác sẽ đề cập đến trong khái niệm "tương đương logic".

Bảng giá trị chân lý toán tử tương đương:

p	q	$p \leftrightarrow q$
0	0	1
0	1	0
1	0	0
1	1	1

Bảng tổng hợp các toán tử logic

p	q	$\bar{p}$	$p \wedge q$	$p \vee q$	$p \oplus q$	$p \rightarrow q$	$p \leftrightarrow q$
0	0	1	0	0	0	1	1
0	1	1	0	1	1	1	0
1	0	0	0	1	1	0	0
1	1	0	1	1	0	1	1

1.1.3 Một số loại mệnh đề

- ***Mệnh đề đảo và mệnh đề phản đảo***

- Giả sử p và q là hai mệnh đề và có phép kéo theo $p \rightarrow q$.

- Mệnh đề $q \rightarrow p$ được gọi là ***mệnh đề đảo*** của $p \rightarrow q$

- Mệnh đề $\neg q \rightarrow \neg p$ được gọi là ***mệnh đề phản đảo*** của $p \rightarrow q$

Đối ngẫu của một mệnh đề

- Đối ngẫu của một mệnh đề phức hợp s *chỉ chứa các toán tử* $\vee$, $\wedge$ *và* $\neg$ là một mệnh đề nhận được từ mệnh đề cũ bằng cách thay $\vee$ bằng $\wedge$, $\wedge$ bằng $\vee$, 1 bằng 0 và 0 bằng 1. Đối ngẫu của s được ký hiệu là s*.

- **Tính chất**:

$$(s^*)^* = s$$

1.1.4 Tương đương của các mệnh đề

■ ***Hằng đúng (tautology), mâu thuẫn và tiếp liên***

- Một mệnh đề phức hợp luôn đúng bất kể giá trị của các mệnh đề thành phần gọi là ***hằng đúng***.

- Một mệnh đề luôn sai gọi là ***mâu thuẫn***.

- Một mệnh đề không phải hằng đúng, cũng không phải mâu thuẫn gọi là ***tiếp liên*** (contigency) hoặc ***thỏa được***.

- Hằng đúng và mâu thuẫn có vai trò quan trọng trong suy luận toán học.

Ví dụ

- Mệnh đề $p \vee \neg p$ là hằng đúng.
- Mệnh đề $p \wedge \neg p$ là mâu thuẫn.

Tương đương logic

Định nghĩa:

- Các mệnh đề (phức hợp) p và q luôn có cùng giá trị được gọi là ***tương đương logic***.

- Hai mệnh đề p và q được gọi là tương đương logic nếu $p \leftrightarrow q$ là hằng đúng và ký hiệu là $p \Leftrightarrow q$ hoặc $p = q$.

Chú ý:

- Dấu $\Leftrightarrow$, hoặc =, hoặc $\equiv$ là một ký hiệu để chỉ ra rằng mệnh đề p tương đương với q, chứ không phải là một phép toán logic.

- Cho hai mệnh đề p , q $\Rightarrow$ có thể nói về giá trị của quan hệ tương đương $p \leftrightarrow q$. Quan hệ này có thể đúng hoặc sai. Nhưng chỉ nói $p \Leftrightarrow q$ hoặc p = q nếu $p \leftrightarrow q$ là hằng đúng.

- **<u>Định lý</u>**

Giá trị của một mệnh đề (phức hợp) không thay đổi nếu thay một mệnh đề thành phần của nó bằng mệnh đề tương đương.

Bảng các tương đương logic

STT	Tương đương	Tên gọi
1	$p \wedge 1 = p, \quad p \vee 0 = p$	Luật đồng nhất
2	$p \vee 1 = 1 \quad p \wedge 0 = 0$	Luật nuốt
3	$p \vee p = p \quad p \wedge p = p$	Luật lũy đẳng
4	$\neg(\neg p) = p$	Luật phủ định kép
5	$p \vee q = q \vee p, p \wedge q = q \wedge p$	Luật giao hoán
6a	$(p \vee q) \vee r = p \vee (q \vee r)$	Luật kết hợp
6b	$(p \wedge q) \wedge r = p \wedge (q \wedge r)$	Luật kết hợp

Bảng các tương đương logic (tiếp)

STT	Tương đương	Tên gọi
7a	$p \lor (q \land r) = (p \lor q) \land (p \lor r)$	Luật phân phối
7b	$p \land (q \lor r) = (p \land q) \lor (p \land r)$	Luật phân phối
8a	$\neg(p \land q) = \neg p \lor \neg q$	Luật De Morgan
8b	$\neg(p \lor q) = \neg p \land \neg q$	Luật De Morgan
9a	$p \lor \neg p = 1$	
9b	$p \land \neg p = 0$	
9c	$(p \to q) = (\neg p \lor q)$	

Ví dụ về chứng minh tương đương logic

- **<u>Định lý</u>**

$$p \leftrightarrow q = (p \rightarrow q) \land (q \rightarrow p)$$

Có thể chứng minh định lý bằng cách:

(1) Lập bảng giá trị chân lý của các mệnh đề hai vế và chứng tỏ chúng nhận giá trị như nhau tương ứng với giá trị của các mệnh đề thành phần.

(2) Chứng minh bằng phản chứng.

(3) Chứng minh bằng biến đổi tương đương logic.

Bảng giá trị chân lý:

p	q	$p\rightarrow q$	$q\rightarrow p$	$(p\rightarrow q)\wedge(q\rightarrow p)$	$p\leftrightarrow q$
0	0	1	1	1	1
0	1	1	0	0	0
1	0	0	1	0	0
1	1	1	1	1	1

1.2 Vị ngữ và lượng từ

- Vị ngữ
- Lượng từ

1.2.1 Khái niệm vị ngữ

- Các câu như "x>3", "y= x+3",... thường gặp trong toán học và các chương trình máy tính.
- Các câu trên không phải là mệnh đề nếu không biết các giá trị của x,y.
- Câu "x lớn hơn 3" có 2 bộ phận: x là chủ ngữ, còn "lớn hơn 3" là *vị ngữ*. Ký hiệu P(x) = "x lớn hơn 3", với P ký hiệu vị ngữ và x là biến.
- Với mỗi giá trị x, P(x) là một mệnh đề, do đó P(x) gọi là hàm mệnh đề $\Rightarrow$ P(4) là đúng , P(2) sai.

1.2.1 Khái niệm vị ngữ (tiếp)

- Vị ngữ P(x, y, …, z) là một hàm mệnh đề với các biến x, y, …, z.
- Với mỗi bộ giá trị cụ thể của các biến x, y, …, z vị ngữ P(x, y, …, z) là một mệnh đề nhận giá trị đúng hoặc sai.

1.2.2 Lượng từ

- Có một cách khác để biến hàm mệnh đề thành các mệnh đề là ***sự lượng hóa***.
- Xét hai loại lượng từ:

- lượng từ phổ dụng hay còn gọi là lượng từ "với mọi" (∀);

- lượng từ "tồn tại" (∃).

- Lượng từ "với mọi" thường được hiểu là với mọi giá trị biến trong không gian dữ liệu nào đó gọi tắt là ***không gian***.

Bảng các lượng từ:

Mệnh đề	Trường hợp đúng	Trường hợp sai
$\forall x\, P(x)$	P(x) đúng với mọi x	Có một giá trị x để P(x) sai
$\exists x\, P(x)$	Có một giá trị x để P(x) đúng	P(x) sai với mọi x

Bảng các phủ định lượng từ:

Phủ định	Mệnh đề tương đương	Trường hợp đúng	Trường hợp sai
$\neg(\exists x\, P(x))$	$\forall x\, \neg P(x)$	P(x) sai với mọi x	Có một giá trị x để P(x) đúng
$\neg(\forall x\, P(x))$	$\exists x\, \neg P(x)$	Có một giá trị x để P(x) sai	P(x) đúng với mọi x

1.3 Ứng dụng trên máy tính

- **Các phép toán bit**
- **Dịch những câu thông thường thành biểu thức logic**

1.3.1 Các phép toán bit

- Các máy tính dùng bit để biểu diễn thông tin. Một bit có giá trị là 0 hoặc 1. Bit cũng được dùng để biểu diễn giá trị logic: 1 là đúng và 0 là sai. Một biến chỉ nhận các giá trị đúng sai được gọi là biến logic hay biến Boole (Boolean variable).
- Các phép toán bit trong máy tính tương ứng với các phép toán logic $\neg$, $\wedge$, $\vee$, $\oplus$ và ký kiệu là NOT, AND, OR, XOR

Bảng giá trị các phép toán bit:

¬ NOT		∧ AND	0	1	∨ OR	0	1	⊕ XOR	0	1
0	1	0	0	0	0	0	1	0	0	1
1	0	1	0	1	1	1	1	1	1	0

Xâu bit (hoặc xâu nhị phân)

- Xâu bit là dãy rỗng hoặc chứa một số bit.
- Chiều dài xâu là số bit trong xâu đó.
- **Ví dụ**

101010011 là một xâu bit có chiều dài là 9.

Định nghĩa

- Phép toán NOT bit của xâu bit a là xâu c có cùng độ dài như a và xâu c có các bit là NOT của các bit tương ứng trong a.
- Các phép toán AND bit, OR bit và XOR bit đối với hai xâu bit a, b có cùng độ dài là xâu c có cùng độ dài như a, b. Xâu c có các bit là AND, OR và XOR của các bit tương ứng trong hai xâu a, b.

Ví dụ:

- Cho hai xâu bit:

a = 10101001

b = 01001010

- c = NOT a = 01010110
- c = a AND b = 00001000
- c = a OR b = 11101011
- c = a XOR b = 11100011

1.3.2 Dịch câu thông thường thành biểu thức logic

- Trong tin học cần dịch những câu thông thường thành những biểu thức logic để máy tính có thể hiểu được.
- **<u>Ví dụ 1:</u>**

Cần dịch câu "Bạn không được lái xe nếu bạn cao dưới 1,5 m trừ phi bạn trên 18 tuổi".

Đặt: p = "Bạn được lái xe“; r = "Bạn cao dưới 1,5m“; s = "Bạn trên 18 tuổi“;

$\Rightarrow$ câu trên có thể dịch thành: $(r \wedge \neg s) \rightarrow \neg p$

■ **Ví dụ 2:**

Viết biểu thức logic mô tả điều kiện để phương trình a x^2 + b x + c = 0 có hai nghiệm thực phân biệt.

Ta có: p = $(a \neq 0) \wedge (b*b - 4*a*c > 0)$;

r = “Phương trình a x^2 + b x + c = 0 có hai nghiệm thực phân biệt”.

Suy ra: $p \rightarrow r$

■ Ví dụ 3:

Biểu diễn câu "Mọi người đều có đúng một người bạn tốt nhất“;

Đặt B(x,y) = "y là bạn tốt nhất của x“.

Có

$$\forall x\ \exists y\ \forall z\ [B(x,y) \wedge ((z \neq y) \rightarrow \neg B(x,z))]$$

1.4 Cơ bản về tập hợp

- Mở đầu
- Các phép toán trên tập hợp
- Đa tập

1.4.1 Mở đầu

- **Định nghĩa**

*Tập hợp bao gồm các đối tượng gọi là các phần tử của tập hợp đó. Tập hợp chỉ **chứa** các phần tử của nó.*

Tập hợp xác định khi chỉ rõ các phần tử của nó.

- **Mô tả một tập hợp**

- Liệt kê các phần tử: {a,1,2,Fred}, {1,2,3,...,99}

- Chỉ rõ các thuộc tính đặc trưng: A = {x | x là các số nguyên dương là bội của 5}

1.4.1 Mở đầu (tiếp)

- **Một số tập hợp quen thuộc:**

N = Tập hợp các số tự nhiên {0,1,2,...}

Z = Tập hợp các số nguyên ()

R = Tập hợp các số thực

- Tập hợp không có phần tử nào gọi là ***tập rỗng,*** ký hiệu là $\varnothing$, hoặc {}.

- **Hai tập hợp bằng nhau:**

- Hai tập hợp bằng nhau ⇔ chúng có cùng các phần tử.

- Trong tập hợp, thứ tự hay sự lặp lại các phần tử là không quan trọng.

⇒ Các tập hợp {1,2,5}, {5,1,2}, {2,1,5} là như nhau

1.4.1 Mở đầu (tiếp)

- ***Tập con của một tập hợp***

\- Tập A gọi là một tập con của tập B, ký hiệu là $A \subseteq B \Leftrightarrow$ lượng từ $\forall x\ (x \in A \rightarrow x \in B)$ là đúng.

\- Tập rỗng là tập con của mọi tập hợp, do với tập S bất kỳ, lượng từ $\forall x\ (x \in \emptyset \rightarrow x \in S)$ luôn đúng, vì mệnh đề $x \in \emptyset$ luôn luôn sai.

- Nếu $A \subseteq B$ và $A \neq B$ thì A là tập con thực sự của B và ký hiệu $A \subset B$.

- Chứng minh $A = B$ bằng cách chứng minh

$$A \subseteq B \text{ và } B \subseteq A.$$

1.4.1 Mở đầu (tiếp)

- ***Bản số hay lực lượng của một tập hợp***

Cho tập hợp S.

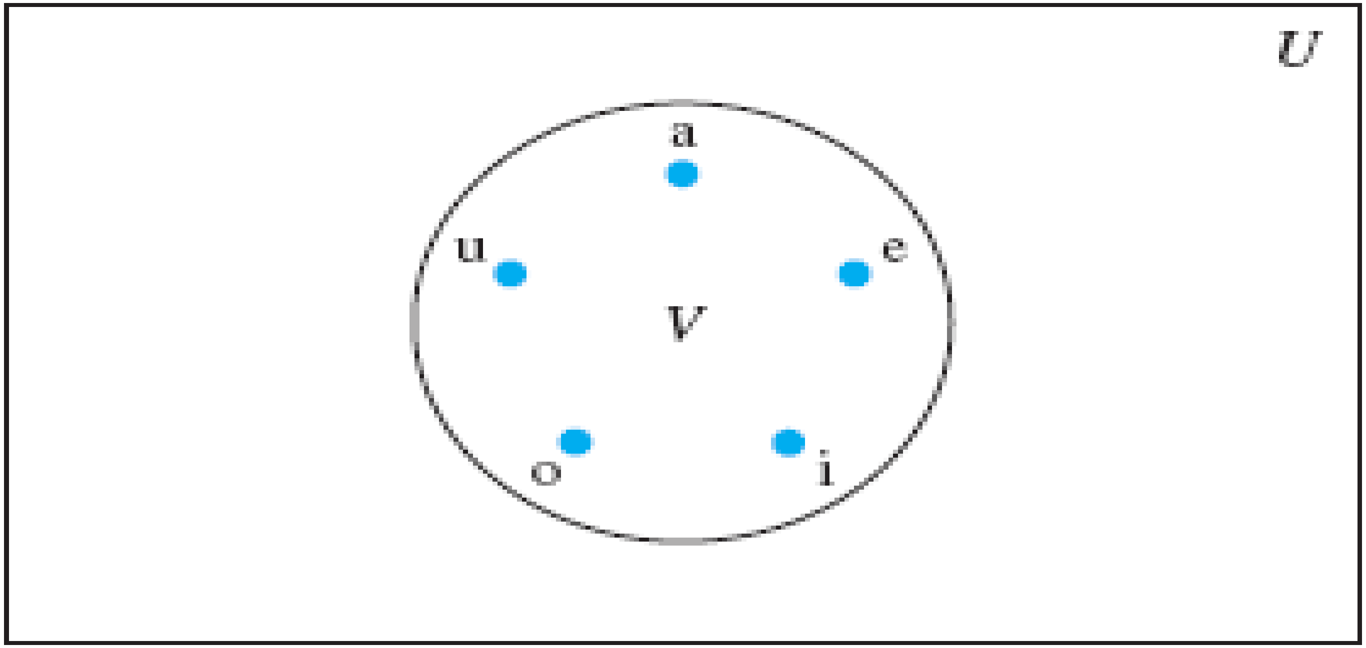
- Nếu có n phần tử phân biệt trong S $\Rightarrow$ S là tập hợp hữu hạn và n là bản số của S.

- Bản số của S được ký hiệu là |S|.

- Nếu tập hợp S không phải là hữu hạn $\Rightarrow$ S là tập hợp vô hạn.

1.4.1 Mở đầu (tiếp)

- Biểu diễn tập hợp bằng sơ đồ Venn



1.4.1 Mở đầu (tiếp)

- Biểu diễn tập hợp bằng tính thuộc của các phần tử:

  $x \in S$ được biểu diễn là 1

  $x \notin S$ được biểu diễn là 0

1.4.1 Mở đầu (tiếp)

- **Tập hợp lũy thừa hay tập hợp tất cả các tập con của một tập hợp**

Cho tập S.

Tập hợp lũy thừa của S, ký hiệu P(S) là tập hợp của tất cả các tập con của S:

$$P(S) = \{ A \mid A \subseteq S\}$$

Nhận xét: Nếu tập S có n phần tử thì tập P(S) có 2^n phần tử.

Ví dụ:

1. Cho $S = \{0,1,2\}$

$\Rightarrow$ $P(S) = \{\varnothing,\{0\},\{1\},\{2\},\{0,1\},\{0,2\},\{1,2\},\{0,1,2\}\}$

2. Tập rỗng chỉ có một tập con là chính nó

$\Rightarrow P(\varnothing) = \{\varnothing\}$

3. Tập $\{\varnothing\}$ là tập hợp có phần tử nó là tập rỗng

$\Rightarrow$ $\{\varnothing\}$ không phải là tập rỗng, nó có duy nhất một tập con thực sự là tập $\varnothing$.

$\Rightarrow P(\{\varnothing\}) = P(P(\varnothing)) = \{ \varnothing,\{\varnothing\}\}$

1.4.2 Các phép toán trên tập hợp

- ***Hợp của 2 tập hợp***

$$A \cup B = \{x \mid x \in A \vee x \in B\}$$

- **Hợp của n tập hợp**

$$\bigcup_{i=1}^{n} A_i = A_1 \cup A_2 \cup \ldots \cup A_n = \{x \mid \exists i,\ x \in A_i\}$$

Phép giao:

- ***Giao của 2 tập hợp***

$$A \cap B = \{x \mid x \in A \wedge x \in B\}$$

- ***Hai tập hợp rời nhau***

A và B rời nhau nếu $A \cap B = \varnothing$

- **Tính chất:**

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

- **Giao của n tập hợp:**

$$\bigcap_{i=1}^{n} A_i = A_1 \cap A_2 \cap \ldots \cap A_n = \{x \mid x \in A_i, \forall i\}$$

1.4.2 Các phép toán trên tập hợp (tiếp)

- ***Hiệu của 2 tập hợp***

$$A-B = \{x \mid x \in A \wedge x \notin B\}$$

- ***Hiệu đối xứng của 2 tập hợp***

$$A \oplus B = (A \cup B) - (A \cap B) =$$

$$\{x \mid (x \in A \vee x \in B) \wedge \neg(x \in A \wedge x \in B) \}$$

- ***Phần bù của một tập hợp***

Cho U là tập vũ trụ và tập hợp A.

Phần bù của A là $\overline{A} = U-A = \{x \mid x \notin A\}$

Bảng giá trị thuộc các phép toán tập hợp

A	B	$\bar{A}$	$A \cap B$	$A \cup B$	$A - B$	$A \oplus B$
0	0	1	0	0	0	0
0	1	1	0	1	0	1
1	0	0	0	1	1	1
1	1	0	1	1	0	0

Tích Đề các (Decartes)

- Tích Đề các của các tập A_1 , A_2 ..., A_n, ký hiệu $A_1 x A_2 x \ldots x A_n$, là tập hợp các dãy sắp thứ tự $(a_1, a_2, \ldots, a_n)$, $a_i \in A_i$, $i = 1, 2, \ldots, n$:

$$A_1 x A_2 x \ldots x A_n = \{(a_1, a_2, \ldots, a_n) | a_i \in A_i, i = 1, \ldots, n\}$$

- Dãy $(a_1, a_2, \ldots, a_n)$ có thứ tự $\Rightarrow$ tích Đề các không có tính giao hoán

Các hằng đẳng thức tập hợp

STT	Hằng đẳng thức	Tên gọi
1	$A \cap U = A$	Luật đồng nhất
2	$A \cup \varnothing = A$	Luật đồng nhất
3	$A \cup U = U$	Luật nuốt
4	$A \cap \varnothing = \varnothing$	Luật nuốt
5	$A \cup A = A$	Luật lũy đẳng
6	$A \cap A = A$	Luật lũy đẳng
7	$\overline{\overline{A}} = A$	Luật bù
8	$A \cup B = B \cup A$	Luật giao hoán

Các hằng đẳng thức tập hợp

STT	Hằng đẳng thức	Tên gọi
9	$A \cap B = B \cap A$	Giao hoán
10	$(A \cup B) \cup C = A \cup (B \cup C)$	Luật kết hợp
11	$(A \cap B) \cap C = A \cap (B \cap C)$	Luật kết hợp
12	$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$	Phân phối
13	$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$	Phân phối
14	$\overline{A \bigcup B} = \overline{A} \bigcap \overline{B}$	De Morgan
15	$\overline{A \bigcap B} = \overline{A} \bigcup \overline{B}$	De Morgan

Ví dụ về chứng minh đẳng thức tập hợp

- Cần chứng minh

$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$$

- Lập bảng tính thuộc để so sánh và suy ra kết quả cần chứng minh

Vế trái:

A	B	C	$B \cap C$	$A \cup (B \cap C)$
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	1	1
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Vế phải:

A	B	C	A ∪ B	A ∪ C	VP
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	1	0	0
0	1	1	1	1	1
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	1	1	1
1	1	1	1	1	1

1.4.3 Đa tập

a) Khái niệm

- Đa tập là một tập hợp không quan tâm đến thứ tự các phần tử, nhưng lại chú ý đến sự lặp lại của chúng.
- Ký hiệu đa tập $P = \{m_1.a_1, m_2.a_2,..., m_k.a_k\}$ để chỉ một đa tập có phần tử a_1 xuất hiện m_1 lần, phần tử a_2 xuất hiện m_2 lần, ..., phần tử a_k xuất hiện m_k lần
- Số m_i gọi là bội của phần tử a_i, $i =1, 2, ..., k$.

b) Các phép toán trên đa tập

Cho P và Q là hai đa tập.

- ***Hợp*** của hai đa tập P và Q là tập $P \cup Q$, trong đó bội của một phần tử trong $P \cup Q$ là số lớn nhất trong hai bội của phần tử đó trong P và Q (Luật Max).

- ***Giao*** của hai đa tập P và Q là tập $P \cap Q$, trong đó bội của một phần tử trong $P \cap Q$ là số bé nhất trong hai bội của phần tử đó trong P và Q (Luật Min).

- ***Hiệu*** của hai đa tập P và Q là tập P - Q, trong đó bội của một phần tử trong P - Q là hiệu của bội của nó trong P và bội của nó trong Q nếu hiệu không âm, ngược lại thì bội của phần tử trong đa tập P - Q bằng 0.

- ***Tổng*** của hai đa tập P và Q là tập $P \oplus Q$, trong đó bội của một phần tử trong $P \oplus Q$ là tổng của bội của nó trong P và trong Q.

1.5 Biểu diễn tập hợp trên máy tính

- Khái niệm
- Ví dụ

1.5.1 Khái niệm

- Có nhiều cách để biểu diễn một tập hợp trên máy tính. ***Thường lưu trữ tập vũ trụ bằng cách sắp xếp các phần tử*** theo thứ tự nào đó.
- Xét tập vũ trụ U hữu hạn gồm n phần tử:

- Sắp xếp các phần tử của U dưới dạng danh sách: $a_1, a_2, \ldots, a_n$;

- Biểu diễn tập con A của U bằng một xâu bit có chiều dài n, bit thứ i là 1 nếu $a_i \in A$ và 0 nếu ai $\notin$ A.

- Các phép toán trên các tập hợp tương đương các phép toán trên các xâu bit:

1.5.2 Ví dụ

- Cho U = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10} và sự sắp các phần tử trong U là theo thứ tự tăng dần, tức là $a_i = i$.
- Xác định xâu bit biểu diễn tập con A gồm các số nguyên lẻ trong U, tập con B gồm các số nguyên chẵn trong U và tập con C gồm các số nguyên không vượt quá 5 trong U?
- Xác định $A \cap C$, $B \cup C$, $A \oplus B$, $A - C$?

Giải

Các tập con của U được biểu diễn bởi các xâu bít độ dài 10. Xâu bit biễu diễn các tập con tương ứng:

- Tập hợp các số lẻ trong U là

  A = 1010101010

- Tập hợp các số chẵn trong U là

  B = 0101010101

- Tập các số nguyên ≤ 5 trong U là

  C = 1111100000

- $A \cap C = 1010100000$
- $B \cup C = 1111110101$
- $A \oplus B = 1111111111$
- $A - C = 0000001010$

1.6 Khái niệm về thuật toán

- Định nghĩa và ví dụ
- Các đặc trưng của thuật toán
- Biểu diễn thuật toán

1.6.1 Định nghĩa và ví dụ

a) Định nghĩa thuật toán

- Thuật toán (Algorithm) để giải một bài toán là một dãy *hữu hạn các thao tác được sắp xếp theo một trình tự xác đinh sao cho sau khi thực hiện dãy thao tác ấy, từ Input của bài toán, ta nhận được Output cần tìm.*
- *Theo Oxford Dictionary, Algorithm: set of well-defined rules for solving a problem in a finite number of steps.*

b) Ví dụ

Thuật toán Euclid tìm g = UCLN(m, n) của hai số nguyên dương m và n.

$m = 15, n = 80 \Rightarrow g = 5$

$m = 2013, n = 13 \Rightarrow g = 1$

Thực hiện liên tiếp các phép chia:

$$15 = 0*85 + 15 \Rightarrow (15, 85) = (85, 15)$$

$$85 = 5*15 + 10 \Rightarrow (85, 15) = (15, 10)$$

$$15 = 1*10 + 5 \Rightarrow (15, 10) = (10, 5)$$

$$10 = 2*5 + 0 \Rightarrow (10, 5) = (5, 0) = 5$$

Phương pháp tìm g = UCLN(m, n) gọi là thuật toán Euclid

Thuật toán Euclid:

Input: m, n nguyên dương;

Output: g = UCLN(m, n);

E1. Nếu $n \neq 0$ chuyển E2; Ngược lại chuyển E5;

E2. r = m mod n;

E3. m = n; n = r ;

E4. Quay lại E1;

E5. Xuất g = m và kết thúc;

TEST thuật toán Euclid:

Lượt thực hiện	m	n	r
0	15	85	15
1	85	15	10
2	15	10	5
3	10	5	0
4	5	0	

1.6.2 Các đặc trưng của thuật toán

(1) *Đại lượng vào* (Input): Các thuật toán thường có các giá trị nhập (input values).

(2) *Đại lượng ra* (Output): Thuật toán thường tạo ra các giá trị xuất (output values) thể hiện lời giải cho bài toán hay vấn đề.

(3) *Tính xác định* (Definiteness). Các bước trong thuật toán phải chính xác rõ ràng

1.6.2 Các đặc trưng của thuật toán (tiếp)

(4) *Tính kết thúc* (Finiteness) hay *tính dừng*: Thuật toán phải cho ra lời giải (hay kết quả) sau một số hữu hạn bước thực hiện. Tính dừng của thuật toán thường được chứng minh bằng toán học hay các bộ Test mẫu.

1.6.2 Các đặc trưng của thuật toán (tiếp)

(5) Tính hiệu quả:

- *Sự đúng đắn:* Với dữ liệu vào cho trước, thuật toán hoạt động sau một số hữu hạn bước sẽ dừng và cho kết quả mong muốn.

Bằng các phép thử chỉ có thể xác định được tính sai của thuật toán.

Muốn xác định tính đúng phải chứng minh chặt chẽ bằng toán học.

(5) Tính hiệu quả:

- *Tính hữu hiệu hay tính khả thi:* Tính hữu hiệu thường được đánh giá qua thời gian thực hiện, dung lượng lưu trữ trên thiết bị nhớ khi thực hiện thuật toán trên máy tính.

1.6.2 Các đặc trưng của thuật toán (tiếp)

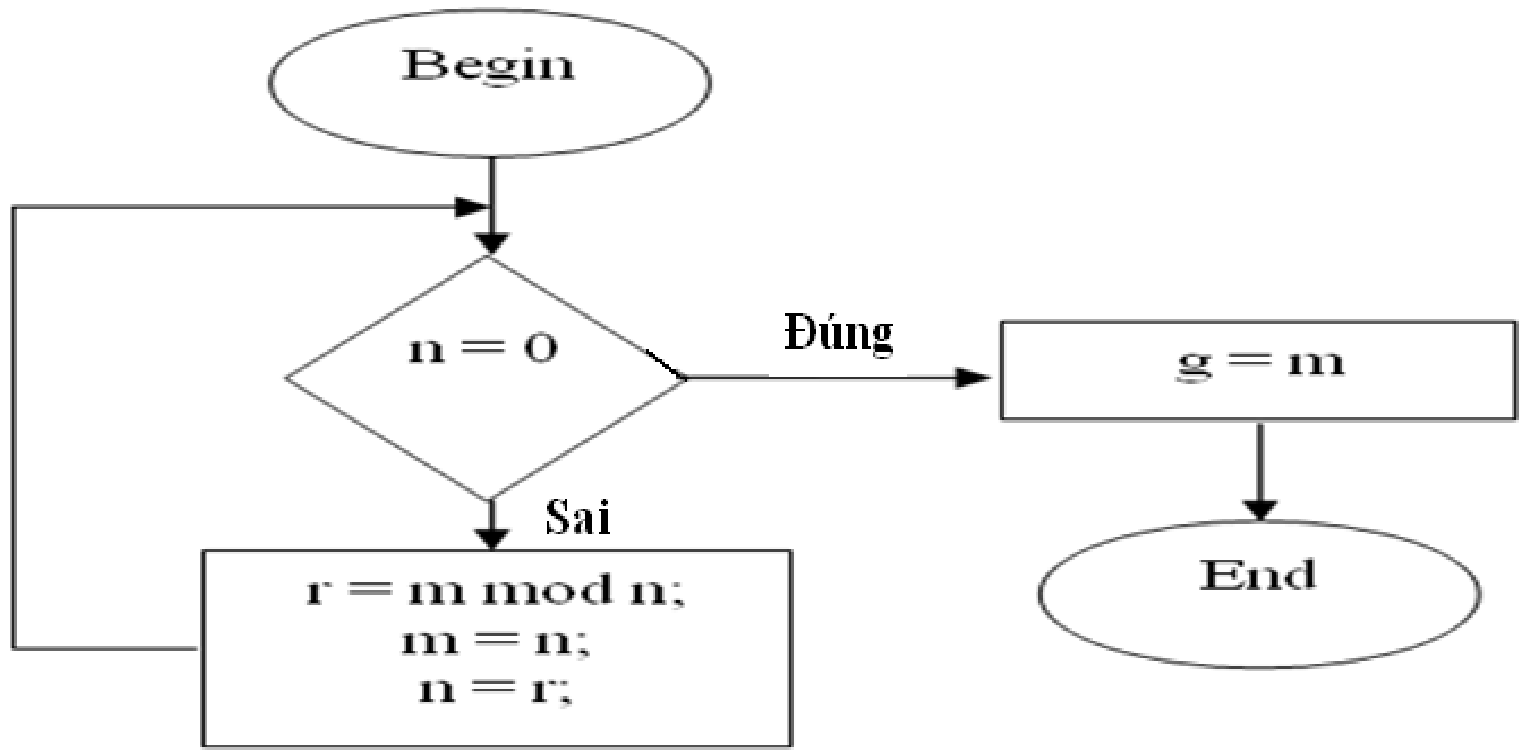
(6) *Tính tổng quát (phổ dụng):* Thuật toán phải được áp dụng được cho tất cả các bài toán có dạng như mong muốn, chứ không phải chỉ áp dụng cho một số trường hợp riêng lẻ.

1.6.3 Biểu diễn thuật toán

- Các phương pháp biểu diễn thuật toán:

- Ngôn ngữ tự nhiên (Liệt kê các bước)
- Lưu đồ (sơ đồ khối).
- Tựa ngôn ngữ lập trình (pseudo code = mã giả).
- Ngôn ngữ lập trình

Biểu diễn thuật toán Euclid bằng lưu đồ



Biểu diễn thuật toán Euclid bằng chương trình C/C++

```
long UCLN(long m, long n) {
    while (n != 0) {
     long r = m % n;
     m = n;  n = r;
     }
     return (m);
    }
```

Biểu diễn thuật toán đổi chỗ bằng mã giả

```
//Khai báo kiểu Item (nếu cần)
void swap(Item &x, Item &y)
 { Item tmp=x;
     x=y;
     y=tmp;
 }
```

1.6.3 Biểu diễn thuật toán (tiếp)

- Biểu diễn thuật toán bằng ngôn ngữ tự nhiên: liệt kê các bước của thuật toán bằng lời.

Trong một số trường hợp cách làm này không dễ dàng và khó hiểu.

- Biểu diễn thuật toán bằng lưu đồ cho cách nhìn tổng quan về toàn cảnh của quá trình xử lý theo thuật toán.

Tuy nhiên lưu đồ thường rất cồng kềnh nên tốn rất nhiều công để xây dựng lưu đồ, nhất là khi dùng các phần mềm soạn thảo văn bản để biên soạn chúng.

1.6.3 Biểu diễn thuật toán (tiếp)

- *Trong các tài liệu về cấu trúc dữ liệu và giải thuật thường dùng tựa ngôn ngữ Pascal để diễn tả thuật toán $\Rightarrow$ khó nhớ các thuật toán nếu không cài đặt thực sự trên máy tính.*
- *Mã giả giúp diễn đạt thuật toán gọn hơn nhưng lại không chặt chẽ bằng ngôn ngữ lập trình và đôi khi khó hiểu được một cách chính xác.*
- **Kết luận**:

Cách tốt nhất để có thể hiểu rõ thuật toán là cài đặt và chạy thử trên máy tính.

Phương pháp xây dựng thuật toán:

(a) Xác định bài toán:

- Xác định Input;

- Xác định Output;

(b) Ý tưởng:

- Nêu hướng tiếp cận giải quyết bài toán;

- Các trường hợp ngoại lệ của bài toán;

(c) Biểu diễn thuật toán:

- Chọn cách biểu diễn thuật toán phù hợp;

Phương pháp xây dựng thuật toán (tiếp):

(d) Test thuật toán:

- Mô phỏng các bước thực hiện của thuật toán trên dữ liệu cụ thể;

- Chứng tỏ kết quả thực hiện thuật toán phù hợp;

(e) Đánh giá thuật toán:

- Đánh giá số lượng các phép tính sơ cấp (đơn) cần thực hiện trên Input kích thước n để nhận được Output;

- Đánh giá hiệu quả của thuật toán.

1.6.4 Độ phức tạp của thuật toán

- Khái niêm độ phức tạp của thuật toán
- Đánh giá độ phức tạp của thuật toán

Khái niệm độ phức tạp của thuật toán

- **Thước đo hiệu quả của thuật toán:**

\- *Thời gian* máy tính sử dụng giải bài toán với INPUT có kích thước xác định.

\- Dung lượng hay *không gian* bộ nhớ đòi hỏi để thực hiện thuật toán với INPUT có kích thước xác định.

Khái niêm độ phức tạp của thuật toán (tiếp)

- **Độ *phức tạp tính toán* của một thuật toán:**

\- *Độ phức tạp thời gian* là thời gian cần thiết để thực hiện thuật toán.

\- *Độ phức tạp không gian* là dung lượng đĩa và bộ nhớ cần thiết để thực hiện thuật toán.

Khái niêm độ phức tạp của thuật toán (tiếp)

- Thường dùng tiêu chuẩn số phép tính sơ cấp cần thiết T(n) để thực hiện thuật toán với n là kích thước dữ liệu input
- T(n) không phải là hàm đơn trị đối với một thuật toán.
- Trong thực tế chỉ có thể đưa ra đánh giá ước lượng T(n) trong trường hợp *xấu nhất*, trường hợp *tốt nhất* và giá trị *trung bình*

Đánh giá độ phức tạp của thuật toán

- **Hai phương pháp đánh giá**:

- Phương pháp thử nghiệm qua các bộ TEST
- Phương pháp lý thuyết nhờ các công cụ toán học

Đánh giá độ phức tạp của thuật toán (tiếp)

- **Định nghĩa khái niệm O lớn**

Cho f và g *là các hàm thực không âm* có miền xác định trong tập số tự nhiên.

Hàm f(n) = O(g(n)) (đọc là: f(n) là O lớn của g(n)) nếu tồn tại hằng số C > 0 và số tự nhiên k sao cho f(n) ≤ C*g(n), với mọi n > k.

Các quy tắc để đánh giá thời gian thực hiện thuật toán

Nếu $T_1(n) = O(f(n))$, $T_2(n) = O(g(n))$ thì:

- Qui tắc cộng:

$$T_1(n) + T_2(n) = O(\max(f(n),g(n)))$$

- Quy tắc nhân:

$$T_1(n)\,T_2(n) = O(f(n)\,g(n))$$

Dạng đánh giá T(n):

- O(1): hằng số
- O(log n)
- O(n)
- O(nlog n)
- $O(n^k)$, k > 1
- $O(2^n)$
- O(n!)

Thảo luận


THANK YOU