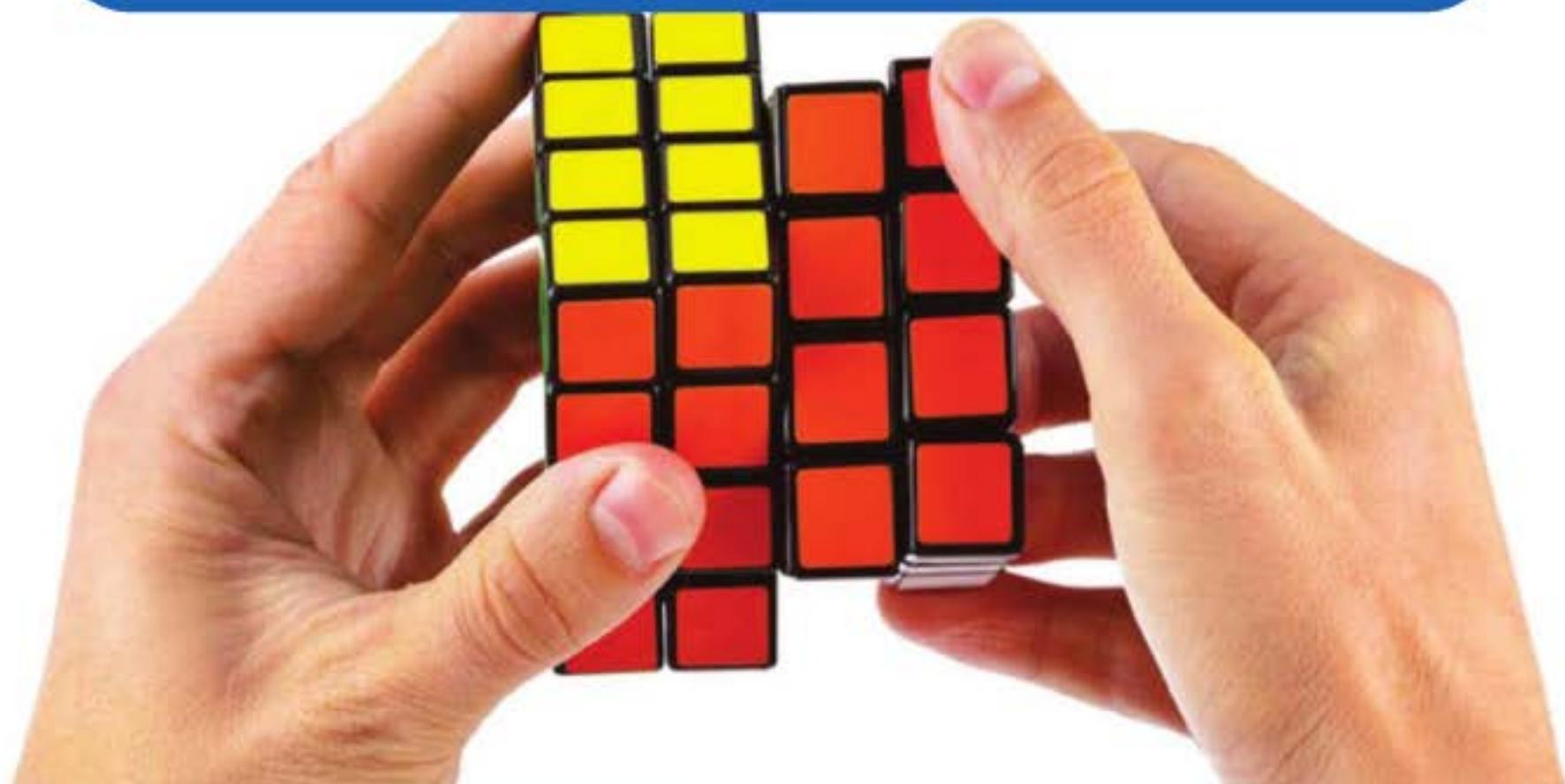
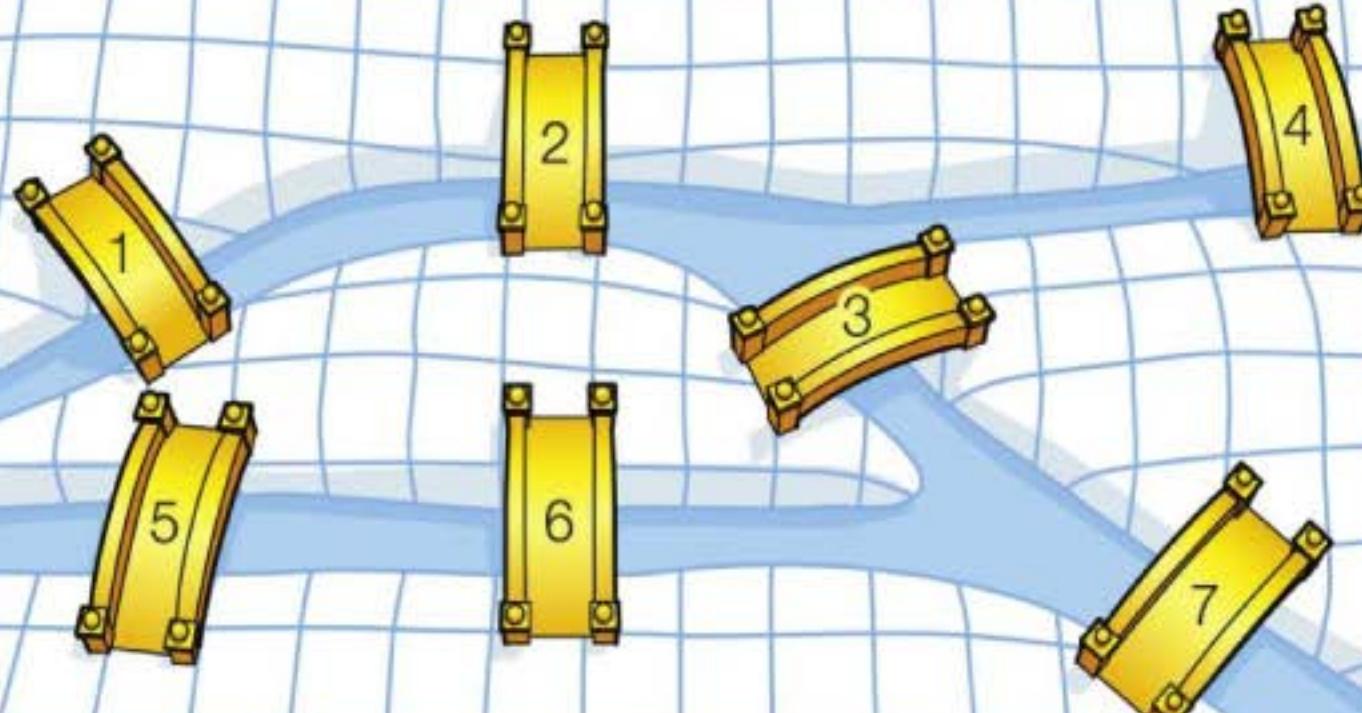


NGUYỄN THÀNH SƠN - ĐẶNG TRƯỜNG SƠN - LÊ VĂN VINH
TRẦN CÔNG TÚ - NGUYỄN QUANG NGỌC - NGUYỄN PHƯƠNG

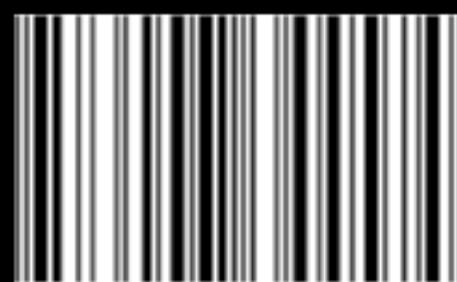


GIÁO TRÌNH
**TOÁN RỜI RẠC
VÀ LÝ THUYẾT ĐỒ THỊ**



NHÀ XUẤT BẢN
ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH

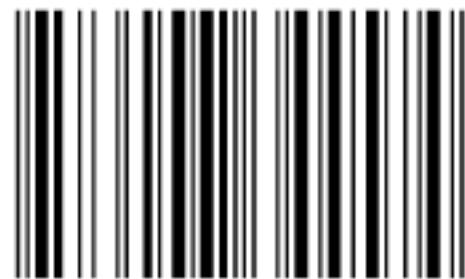
A I



**Nguyễn Thành Sơn, Đặng Trường Sơn, Lê Văn Vinh, Trần Công Tú,
Nguyễn Quang Ngọc, Nguyễn Phương**

**Giáo trình
TOÁN RỜI RẠC
vÀ LÝ THUYẾT ĐỒ THỊ**

**NHÀ XUẤT BẢN ĐẠI HỌC QUỐC GIA
THÀNH PHỐ HỒ CHÍ MINH - 2016**



ISBN: 978-604-73-4457-4

LỜI NÓI ĐẦU

Toán rời rạc và Lý thuyết đồ thị là môn học tích hợp từ hai môn Toán rời rạc và môn Lý thuyết đồ thị. Đây là một trong những môn học căn bản và quan trọng trong lĩnh vực ứng dụng toán trong tin học. Nội dung “Toán rời rạc” trang bị cho người học những kiến thức cơ bản về logic mệnh đề, logic vị từ, suy diễn logic, quan hệ tương đương, quan hệ thứ tự, dàn, đại số Bool và cung cấp cho người học kiến thức và kỹ năng trong việc phân tích, nhìn nhận vấn đề, cũng như trong việc xác định công thức đa thức tối thiểu bằng phương pháp biểu đồ Karnaugh. Còn kiến thức về “Lý thuyết đồ thị” có ứng dụng đa dạng trong cuộc sống. Nó cung cấp các kiến thức về công cụ, phương pháp, thuật toán và hỗ trợ chúng ta xây dựng các mô hình nhằm giải quyết nhiều bài toán thực tiễn. Toán rời rạc và Lý thuyết đồ thị hiện là môn học bắt buộc trong chương trình đào tạo các ngành Công nghệ thông tin, Toán tin, Khoa học máy tính, ...

Sau khi học xong môn Toán rời rạc và Lý thuyết đồ thị, sinh viên sẽ có khả năng phân tích, giải thích, tư duy và lập luận giải quyết các vấn đề về toán rời rạc và lý thuyết đồ thị. Sinh viên sẽ được cung cấp kiến thức để hiểu và vận dụng được những quy trình, giải thuật trên đồ thị, có kỹ năng trong việc lập trình để giải quyết các bài toán trên đồ thị. Một năng lực quan trọng khác là khả năng phân tích, nhìn nhận vấn đề một cách khoa học, không phiến diện hay tư duy theo lối mòn. Ngoài ra, sinh viên sẽ biết cách sử dụng đồ thị như một công cụ mô hình hóa trong việc mô phỏng các vấn đề thực tế để chuyển thành các bài toán có thể giải được trên đồ thị.

Nội dung giáo trình gồm có 11 chương, bao quát hầu hết các vấn đề cốt lõi của môn học. Giáo trình không đi sâu vào các vấn đề lý thuyết mà tập trung vào các vấn đề cơ bản của toán rời rạc và các giải thuật cũng như tính ứng dụng của môn học. Cuối mỗi chương đều có phần bài tập để sinh viên có thể tự kiểm tra kiến thức của mình. Các thuật toán trong giáo trình hầu hết được trình bày dưới dạng mã giả. Phần phụ lục có mã nguồn của một số thuật toán.

Với kinh nghiệm nhiều năm giảng dạy môn Toán rời rạc và Lý thuyết đồ thị tại trường đại học, chúng tôi đã cố gắng đem những kiến thức và kinh nghiệm của mình để trình bày các vấn đề trong giáo trình một cách rõ ràng và đơn giản nhất. Tuy nhiên, những thiếu sót là điều không thể tránh khỏi. Trong quá trình biên soạn giáo trình này chúng tôi đã nhận được nhiều lời động viên và góp ý của các đồng nghiệp, xin chân

thành cảm ơn và mong muốn tiếp tục nhận được ý kiến đóng góp của các giảng viên và các bạn sinh viên để giáo trình ngày càng hoàn thiện hơn.

Các tác giả

MỤC LỤC

LỜI NÓI ĐẦU.....	3
MỤC LỤC	5
Chương 1. CƠ SỞ LOGIC.....	9
1.1 Phép tính mệnh đề	9
1.2 Suy diễn logic.....	19
1.3 Vị từ và lượng từ	27
Bài tập chương 1	34
Chương 2. QUAN HỆ HAI NGÔI.....	37
2.1 Khái niệm chung	37
2.2 Quan hệ tương đương.....	39
2.3 Quan hệ thứ tự.....	41
2.4 Dàn (Lattice)	47
Bài tập chương 2	53
Chương 3. ĐẠI SỐ BOOL – HÀM BOOL.....	55
3.1 Đại số Bool.....	55
3.2 Hàm Bool	59
3.3 Mạng các cỗng	62
3.4 Phương pháp biểu đồ Karnaugh	63
Bài tập chương 3	72
Chương 4. MỞ ĐẦU VỀ LÝ THUYẾT ĐỒ THỊ	75
4.1. Mở đầu	75
4.2. Định nghĩa và phân loại đồ thị	76
4.3. Các thuật ngữ cơ bản.....	80
4.4. Đường đi, chu trình, đồ thị liên thông.....	82
4.5. Một số dạng đồ thị đặc biệt	86
Bài tập chương 4	94

Chương 5. BIỂU DIỄN ĐỒ THỊ TRÊN MÁY TÍNH	97
5.1. Ma trận kè, ma trận trọng số	97
5.2. Danh sách cạnh (cung)	102
5.3. Danh sách kè	103
5.4. Ma trận liên thuộc	106
5.5. Sự đẳng cấu của đồ thị	107
Bài tập chương 5	109
Chương 6. DUYỆT ĐỒ THỊ.....	113
6.1. Duyệt đồ thị theo chiều sâu	113
6.2. Duyệt đồ thị theo chiều rộng	115
6.3. Tìm đường đi và kiểm tra tính liên thông	119
Bài tập chương 6	124
Chương 7. ĐỒ THỊ EULER VÀ ĐỒ THỊ HAMILTON	127
7.1. Đồ thị Euler	127
7.2. Đồ thị Hamilton.....	132
Bài tập chương 7	136
Chương 8. CÂY.....	139
8.1. Định nghĩa và các tính chất cơ bản của cây	139
8.2. Cây khung của đồ thị.....	141
8.3. Bài toán cây khung nhỏ nhất	145
8.4. Cây có gốc.....	151
Bài tập chương 8	159
Chương 9. TÔ MÀU ĐỒ THỊ.....	161
9.1. Mở đầu	161
9.2. Định lý bốn màu	162
9.3. Đồ thị hai màu	162
9.4. Thuật toán sequentialcolor	163
9.5. Những ứng dụng của bài toán tô màu đồ thị	165
Bài tập chương 9	168

Chương 10. ĐƯỜNG ĐI NGẮN NHẤT	171
10.1. Định nghĩa.....	171
10.2. Thuật toán Dijkstra.....	171
10.3. Thuật toán Ford-Bellman	174
10.4. Thuật toán Floyd	177
Bài tập chương 10	181
Chương 11. LUỒNG TRONG MẠNG	185
11.1. Một số khái niệm cơ bản	185
11.2. Định lý Ford-Fulkerson.....	188
11.3. Thuật toán tìm luồng cực đại trong mạng	192
Bài tập chương 11	198
PHỤ LỤC A. HƯỚNG DẪN VÀ ĐÁP ÁN MỘT SỐ BÀI TẬP.....	199
PHỤ LỤC B. CHƯƠNG TRÌNH MẪU.....	212
TÀI LIỆU THAM KHẢO.....	231

Chương 1

CƠ SỞ LOGIC

1.1. PHÉP TÍNH MỆNH ĐỀ

1.1.1. Định nghĩa 1

i. Mệnh đề toán học là những phát biểu mà chúng ta có thể xác định đúng hay sai.

Những câu cảm thán, câu mệnh lệnh không phải là mệnh đề.

Giá trị đúng sai của một mệnh đề được gọi là chân trị của nó. Chúng ta ký hiệu **(1)** hay **(T)** để chỉ mệnh đề đúng và **(0)** hay **(F)** để chỉ mệnh đề sai.

Nếu P là mệnh đề thì bảng sau gọi là bảng chân trị của P.

P
0
1

Ví dụ 1:

“ $2 + 2 = 5$ ” là mệnh đề sai (0)

“ $2 \times 7 = 14$ ” là mệnh đề đúng (1)

“Trời nóng quá”

“x là số nguyên tố”

}

Không phải là mệnh đề

ii. Một mệnh đề được gọi là nguyên thủy nếu nó không thể phân tích thành những mệnh đề nhỏ hơn.

Mệnh đề ghép hay mệnh đề phức hợp thì ngược lại.

Ví dụ 2:

$2 + 2 = 5$: Mệnh đề nguyên thủy

$2 + 2 = 5$ và Nguyễn Du là tác giả của truyện Kiều: Mệnh đề phức hợp

1.1.2. Định nghĩa 2

Cho P, Q là các mệnh đề nguyên thủy

i. Phép phủ định

Mệnh đề phủ định của P ký hiệu là $\neg P$, là mệnh đề có giá trị sai khi P đúng và đúng khi P sai.

P	$\neg P$
0	1
1	0

ii. Phép nối liền

Mệnh đề nối liền của P và Q , ký hiệu $P \wedge Q$ (đọc là “ P và Q ”) là mệnh đề chỉ có giá trị đúng khi cả P và Q đều đúng.

P	Q	$P \wedge Q$
0	0	0
0	1	0
1	0	0
1	1	1

iii. Phép nối rời

Mệnh đề nối rời của P và Q , ký hiệu $P \vee Q$ (đọc là “ P hay Q ”) là mệnh đề chỉ sai khi cả P lẫn Q đều sai.

P	Q	$P \vee Q$
0	0	0
0	1	1
1	0	1
1	1	1

Ngoài ra ta còn định nghĩa “ $P \vee Q$ ” (đọc là “ P hoặc Q ”) là mệnh đề chỉ đúng khi chỉ một trong hai mệnh đề P, Q đúng.

P	Q	$P \vee Q$
0	0	0
0	1	1
1	0	1
1	1	0

iv. Phép kéo theo

Mệnh đề “nếu P thì Q ” hay “ P là điều kiện đủ của Q ”, ký hiệu $P \rightarrow Q$ (đọc là “nếu P thì Q ” hay “ P kéo theo Q ”) là mệnh đề chỉ sai khi P đúng, Q sai.

P	Q	$P \rightarrow Q$
0	0	1
0	1	1
1	0	0
1	1	1

v. Phép kéo theo hai chiều

Mệnh đề $P \leftrightarrow Q$ có nghĩa là $P \rightarrow Q$ và $Q \rightarrow P$

P	Q	$P \leftrightarrow Q$
0	0	1
0	1	0
1	0	0
1	1	1

Nhận xét: $P \leftrightarrow Q$ có chân trị 1 khi P và Q có cùng chân trị và có chân trị 0 khi P và Q có chân trị khác nhau.

Ví dụ 3:

P = “Thúy đi chơi”

Q = “Trăng tàn”

R = “Trời mưa”

a/ Mô tả bằng ngôn ngữ mệnh đề: $(Q \wedge \neg R) \rightarrow P$

Nếu trăng tàn và trời không mưa thì Thúy đi chơi

b/ Mô tả bằng mệnh đề: Trời không mưa nhưng Thúy vẫn đi chơi

$$\neg R \wedge P$$

Lưu ý: Trong logic toán “nhưng” với “và” là như nhau.

Ví dụ 4:

P = “Anh là vua”

Q = “Em là hoàng hậu”

$P \rightarrow Q$ = “Nếu anh là vua thì em là hoàng hậu”

1.1.3. Định nghĩa 3

Dạng mệnh đề được xác định từ các hằng mệnh đề, các biến mệnh đề cùng với các phép nối logic theo một trật tự nhất định. Ký hiệu $E(p, q, r\dots)$

Ví dụ 5:

Cho $S_0 = “2 \times 2 = 8”$

$E(p, q, r) = (\neg p \wedge q) \vee [(\neg q \rightarrow r) \wedge (r \vee S_0)]$ là một dạng mệnh đề với các biến mệnh đề p, q, r.

Ví dụ 6:

Cho p, q, r là các mệnh đề nguyên thủy. Đặt:

$$E = E(p, q, r) = p \vee (q \wedge r)$$

$$F = F(p, q, r) = (p \vee q) \wedge r$$

Lập bảng chân trị của E và F

p	q	r	$q \wedge r$	$E = p \vee (q \wedge r)$	$p \vee q$	$F = (p \vee q) \wedge r$
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	1	0
0	1	1	1	1	1	1
1	0	0	0	1	1	0
1	0	1	0	1	1	1
1	1	0	0	1	1	0
1	1	1	1	1	1	1

Nhận xét: E \neq F

Ví dụ 7:

$$E = p \rightarrow q$$

$$F = \neg p \vee q$$

p	q	$E = p \rightarrow q$	$\neg p$	$F = \neg p \vee q$
0	0	1	1	1
0	1	1	1	1
1	0	0	0	0
1	1	1	0	1

Nhận xét: E và F có cùng chân trị.

1.1.4. Định nghĩa 4

Hai mệnh đề E và F được gọi là tương đương nếu chúng có cùng chân trị. Ký hiệu $E \Leftrightarrow F$.

Theo ví dụ 7 thì

Định lý 1: $(P \rightarrow Q) \Leftrightarrow (\neg P \vee Q)$

Ví dụ 8:

Lập bảng chân trị của $E = p \wedge q \rightarrow p$

p	q	$p \wedge q$	$E = p \wedge q \rightarrow p$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	1

Nhận xét: E luôn luôn đúng.

1.1.5. Định nghĩa 5

E là một hằng sai hay một mâu thuẫn luôn có chân trị 0. Ký hiệu F₀ hay 0.

E là một hằng đúng nếu nó luôn luôn có chân trị 1. Ký hiệu T₀ hay 1.

Nhận xét: $E \Leftrightarrow F$ có nghĩa là $E \leftrightarrow F$ là một hằng đúng.

Ví dụ 9:

Lập bảng chân trị của $E = q \rightarrow (p \vee q)$

p	q	$p \vee q$	$E = q \rightarrow (p \vee q)$
0	0	0	1
0	1	1	1
1	0	1	1
1	1	1	1

1.1.6. Định nghĩa 6

Cho hai mệnh đề E và F. Ta nói F là hệ quả logic của E nếu $E \rightarrow F$ là một hằng đúng. Ký hiệu $E \Rightarrow F$.

Theo ví dụ 9 ta có $q \Rightarrow (p \vee q)$

Ví dụ 10: Lập bảng chân trị của $(p \rightarrow q) \leftrightarrow (\neg q \rightarrow \neg p)$

p	q	$E = p \rightarrow q$	$\neg p$	$\neg q$	$F = (\neg q \rightarrow \neg p)$	$E \leftrightarrow F$
0	0	1	1	1	1	1
0	1	1	1	0	1	1
1	0	0	0	1	0	1
1	1	1	0	0	1	1

Định lý 2: $(p \rightarrow q) \Leftrightarrow (\neg q \rightarrow \neg p)$

Định lý 3: (Quy tắc thay thế)

Quy tắc 1: Cho dạng mệnh đề E và F là một “biểu thức con” của E. Nếu ta thay F bởi F' ta có dạng mệnh đề E' và nếu $F \Leftrightarrow F'$ thì $E \Leftrightarrow E'$.

Quy tắc 2: Cho dạng mệnh đề E(p, q, r,...) là một hằng đúng. Nếu ta thay p bởi dạng mệnh đề E'(p', q') thì ta có dạng mệnh đề E(p', q', r,...) cũng là một hằng đúng.

Ví dụ 11: $E = [p \wedge (p \rightarrow q)] \rightarrow q$

Bằng cách lập bảng chân trị, ta thấy E là một hằng đúng (T_0). Trong E, thay p bởi $r \rightarrow s$

$$E' = [(r \rightarrow s) \wedge ((r \rightarrow s) \rightarrow q)] \rightarrow q$$

Theo quy tắc 2 thì E' là T₀.

Ví dụ 12: $E = (p \rightarrow q) \rightarrow r$

$p \rightarrow q$ là một biểu thức con của E

Theo định lý 1: $(p \rightarrow q) \Leftrightarrow (\neg p \vee q)$

Trong E thay $p \rightarrow q$ bởi $\neg p \vee q$, ta có: $E': (\neg p \vee q) \rightarrow r$

Theo quy tắc 1: $E \Leftrightarrow E'$

Nhận xét:

- $E \Leftrightarrow E$
- $E \Leftrightarrow F$ thì $F \Leftrightarrow E$
- $E \Leftrightarrow F$ và $F \Leftrightarrow G$ thì $E \Leftrightarrow G$

Ngoài hai quy tắc thay thế trên ta còn sử dụng 10 quy luật logic được phát biểu dưới dạng các tương đương logic để rút gọn một dạng mệnh đề cho trước.

Định lý 4: (Luật logic)

i. **Phủ định của phủ định:**

$$\neg\neg p \Leftrightarrow p$$

ii. **Luật DeMorgan:**

$$\neg(p \wedge q) \Leftrightarrow (\neg p \vee \neg q)$$

$$\neg(p \vee q) \Leftrightarrow (\neg p \wedge \neg q)$$

iii. **Luật giao hoán**

$$p \wedge q \Leftrightarrow q \wedge p$$

$$p \vee q \Leftrightarrow q \vee p$$

iv. **Luật kết hợp**

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

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

v. **Luật phân bố**

$$p \wedge (q \vee r) \Leftrightarrow (p \wedge q) \vee (p \wedge r)$$

$$p \vee (q \wedge r) \Leftrightarrow (p \vee q) \wedge (p \vee r)$$

vì. Luật lũy đồng

$$p \wedge p \Leftrightarrow p$$

$$p \vee p \Leftrightarrow p$$

vìi. Luật về phần tử trung hòa

$$p \wedge T_0 \Leftrightarrow p$$

$$p \vee F_0 \Leftrightarrow p$$

viii. Luật về phần tử đảo (phần tử bù)

$$p \wedge (\neg p) \Leftrightarrow F_0$$

$$p \vee (\neg p) \Leftrightarrow T_0$$

ix. Luật thống trị

$$p \wedge F_0 \Leftrightarrow F_0$$

$$p \vee T_0 \Leftrightarrow T_0$$

x. Luật hấp thụ

$$p \wedge (p \vee q) \Leftrightarrow p$$

$$p \vee (p \wedge q) \Leftrightarrow p$$

Ví dụ 13:

p = “Điệp lấy vợ”

q = “Lan đi tu”

$\neg(p \rightarrow q) = ?$

Bước	Lý do
1. $\neg(p \rightarrow q) \Leftrightarrow \neg(\neg p \vee q)$	Tương đương logic
2. $\Leftrightarrow (\neg \neg p) \wedge (\neg q)$	DeMorgan
3. $\Leftrightarrow p \wedge (\neg q)$	Phủ định của phủ định

Tóm lại: $\neg(p \rightarrow q) \Leftrightarrow p \wedge (\neg q)$

Vậy: $\neg(p \rightarrow q)$: Điệp lấy vợ nhưng Lan không đi tu

Ví dụ 14: Rút gọn $E = (p \vee q) \wedge \neg(\neg p \wedge q)$

Bước	Lý do
1. $E \Leftrightarrow (p \vee q) \wedge (\neg \neg p \vee \neg q)$	DeMorgan
2. $\Leftrightarrow (p \vee q) \wedge (p \vee \neg q)$	Phủ định của phủ định
3. $\Leftrightarrow p \vee (q \wedge \neg q)$	Phân bố
4. $\Leftrightarrow p \vee F_0$	Phản tử bù
5. $\Leftrightarrow p$	Phản tử trung hòa

Vậy: $E \Leftrightarrow p$

Ví dụ 15: Rút gọn $E = \neg [\neg ((p \vee q) \wedge r) \vee \neg q]$

Bước	Lý do
1. $E \Leftrightarrow \neg [(p \vee q) \wedge r] \wedge \neg \neg q$	DeMorgan
2. $\Leftrightarrow [(p \vee q) \wedge r] \wedge q$	Phủ định của phủ định
3. $\Leftrightarrow (p \vee q) \wedge (r \wedge q)$	Kết hợp
4. $\Leftrightarrow (p \vee q) \wedge (q \wedge r)$	Giao hoán
5. $\Leftrightarrow [(p \vee q) \wedge q] \wedge r$	Kết hợp
6. $\Leftrightarrow q \wedge r$	Hấp thụ

Vậy: $E \Leftrightarrow q \wedge r$

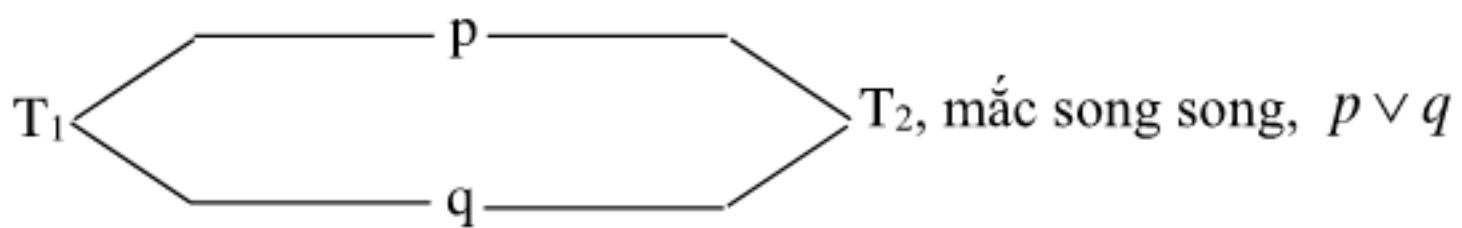
❖ Một mạng chuyên gồm dây dẫn và công tắc ở hai đầu T₁, T₂

T₁ ————— p ————— T₂

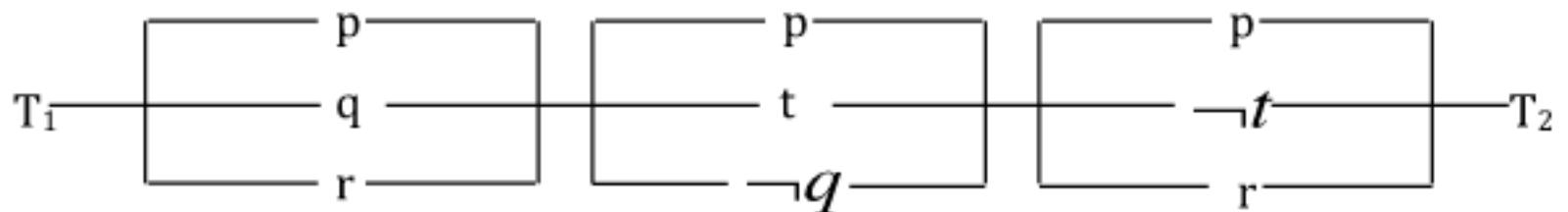
Nếu công tắc mở (hở) thì không có dòng điện đi qua, ghi là 0.

Nếu công tắc đóng (kín) thì có dòng điện đi qua, ghi là 1.

T₁ ————— p ————— q ————— T₂, mắc nối tiếp, $p \wedge q$



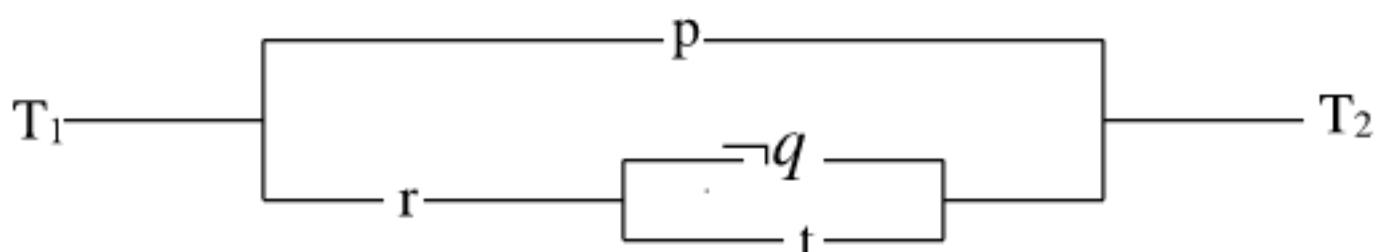
Ví dụ 16: Rút gọn mạng sau



Mạng trên được viết dưới dạng

$$E = (p \vee q \vee r) \wedge (p \vee t \vee \neg q) \wedge (p \vee \neg t \vee r)$$

Bước	Lý do
1. $E \Leftrightarrow p \vee [(q \vee r) \wedge (t \vee \neg q) \wedge (\neg t \vee r)]$	Phân bố
2. $\Leftrightarrow p \vee [(q \vee r) \wedge (\neg t \vee r) \wedge (t \vee \neg q)]$	Giao hoán
3. $\Leftrightarrow p \vee [(r \vee (q \wedge \neg t)) \wedge (t \vee \neg q)]$	Phân bố
4. $\Leftrightarrow p \vee [(r \vee (q \wedge \neg t)) \wedge \neg \neg (t \vee \neg q)]$	Phủ định của phủ định
5. $\Leftrightarrow p \vee [(r \vee (q \wedge \neg t)) \wedge \neg (\neg t \wedge \neg \neg q)]$	DeMorgan
6. $\Leftrightarrow p \vee [(r \vee (q \wedge \neg t)) \wedge \neg (\neg t \wedge q)]$	Phủ định của phủ định
7. $\Leftrightarrow p \vee [(r \vee (q \wedge \neg t)) \wedge \neg (q \wedge \neg t)]$	Giao hoán
8. $\Leftrightarrow p \vee [(r \wedge \neg (q \wedge \neg t)) \vee ((q \wedge \neg t) \wedge \neg (q \wedge \neg t))]$	Phân bố
9. $\Leftrightarrow p \vee [(r \wedge \neg (q \wedge \neg t)) \vee F_0]$	Phân tử bù
10. $\Leftrightarrow p \vee [r \wedge \neg (q \wedge \neg t)]$	Phân tử trung hòa
11. $\Leftrightarrow p \vee [r \wedge (\neg q \vee t)]$	DeMorgan



1.2. SUY DIỄN LOGIC

Trong chứng minh toán học người ta thường dựa vào các tiền đề có sẵn p_1, p_2, \dots, p_n để đưa ra một kết luận q .

1.2.1. Định nghĩa 7: Cho các mệnh đề p_1, p_2, \dots, p_n, q . Nếu $p_1 \wedge p_2 \wedge \dots \wedge p_n \rightarrow q$ là một hằng đúng thì ta nói đó là suy luận đúng.

p_1, p_2, \dots, p_n gọi là các tiền đề, q gọi là kết luận. Ký hiệu

$$\frac{\begin{array}{c} p_1 \\ \vdots \\ p_n \end{array}}{\therefore q}$$

Ngược lại ta có suy luận sai.

Ví dụ 17: Cho p, r, s là các mệnh đề nguyên thủy

Cho $p_1=p$, $p_2 = p \wedge r \rightarrow s$, $q = r \rightarrow s$. Suy luận $p_1 \wedge p_2 \rightarrow q$ đúng hay sai?

p ₁	r	s	$p \wedge r$			$p_1 \wedge p_2$	$p_1 \wedge p_2 \rightarrow q$
				p ₂	q		
p				$p \wedge r \rightarrow s$	$r \rightarrow s$		
0	0	0	0	1	1	0	1
0	0	1	0	1	1	0	1
0	1	0	0	1	0	0	1
0	1	1	0	1	1	0	1
1	0	0	0	1	1	1	1
1	0	1	0	1	1	1	1
1	1	0	1	0	0	0	1
1	1	1	1	1	1	1	1

Vậy suy luận $p_1 \wedge p_2 \rightarrow q$ là đúng.

Ví dụ 18: Suy luận sau đúng hay sai?

$$(p \rightarrow q) \wedge p \rightarrow q$$

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

Đây là một suy luận đúng, gọi là Modus Ponens hay phương pháp khẳng định.

☞ Các định lý sau đây cho phép chúng ta xét từng bước xem một suy luận có đúng hay không mà không cần lập bảng chân trị.

Định lý 5: Luật suy diễn

Luật

Tên luật

$$p \rightarrow q$$

$$1. \frac{p}{\therefore q}$$

Phương pháp khẳng định

$$\therefore p \rightarrow r$$

$$2. \frac{q \rightarrow r}{\therefore p \rightarrow r}$$

Tam đoạn luận

$$\therefore p \rightarrow r$$

$$3. \frac{\neg q}{\therefore \neg p}$$

Phương pháp phủ định

4.
$$\frac{\neg p}{\therefore q}$$
 Tam đoạn luận rời
5.
$$\frac{q}{\therefore p \wedge q}$$
 Luật nối liền
6.
$$\frac{p \wedge q}{\therefore p}$$
 Luật đơn giản nối liền
7.
$$\frac{p}{\therefore p \vee q}$$
 Luật khuyếch đại rời
8.
$$\frac{\neg p \rightarrow F_0}{\therefore p}$$
 Luật mâu thuẫn
9.
$$\frac{\begin{array}{c} p \rightarrow r \\ q \rightarrow r \end{array}}{\therefore p \vee q \rightarrow r}$$
 Quy tắc C/M theo trường hợp

Chứng minh định lý 5: Lập bảng chân trị cho từng luật như trong ví dụ 18.

Ví dụ 19: Suy luận sau đúng hay sai?

$$\frac{\begin{array}{c} p \rightarrow r \\ \neg p \rightarrow q \\ q \rightarrow s \end{array}}{\therefore \neg r \rightarrow s}$$

Bước	Lý do
1. $\neg p \rightarrow q$	Tiền đề
2. $q \rightarrow s$	Tiền đề
3. $\neg p \rightarrow s$	B ₁ , B ₂ và tam đoạn luận
4. $p \rightarrow r$	Tiền đề
5. $\neg r \rightarrow \neg p$	B ₄ và tương đương logic
6. $\therefore \neg r \rightarrow s$	B ₅ , B ₃ và tam đoạn luận

Vậy suy luận trên là suy luận đúng.

Ví dụ 20: Xét suy luận sau

$$\begin{array}{c} p \rightarrow q \\ q \rightarrow (r \wedge s) \\ \neg r \vee (\neg t \vee u) \\ \hline \therefore u \end{array}$$

Bước	Lý do
1. $p \rightarrow q$	Tiền đề
2. $q \rightarrow (r \wedge s)$	Tiền đề
3. $p \rightarrow (r \wedge s)$	B ₁ , B ₂ và tam đoạn luận
4. $p \wedge t$	Tiền đề
5. p	B ₄ và đơn giản nối liền
6. $r \wedge s$	B ₅ , B ₃ và phương pháp khẳng định
7. r	B ₆ và đơn giản nối liền
8. $\neg r \vee (\neg t \vee u)$	Tiền đề
9. $\neg t \vee u$	B ₇ , B ₈ và tam đoạn luận rời
10. t	B ₄ và đơn giản nối liền
11. $\therefore u$	B ₉ , B ₁₀ và tam đoạn luận rời

Vậy suy luận trên là suy luận đúng.

Ví dụ 21: Xét suy luận sau:

Nếu cô Địệu đi chơi thì cô Địệu không học bài.

Nếu cô Địệu không học bài thì cô Địệu thi rót.

Mà cô Địệu đang đi chơi. Vậy cô Địệu thi rót.

Suy luận trên có dạng

$$p \rightarrow \neg q$$

$$\neg q \rightarrow r$$

$$\frac{p}{\therefore r}$$

Bước

Lý do

1. $p \rightarrow \neg q$

Tiền đề

2. $\neg q \rightarrow r$

Tiền đề

3. $p \rightarrow r$

B₁, B₂ và tam đoạn luận

4. p

Tiền đề

5. $\therefore r$

B₃, B₄ và phương pháp khẳng định

Vậy suy luận trên là suy luận đúng

Ví dụ 22: Xét suy luận sau:

Nếu ban nhạc không biết chơi nhạc rock hay nước giải khát không được đưa đến đúng giờ thì bữa tiệc sẽ bị hủy bỏ và cô Địệu sẽ rất giận.

Nếu bữa tiệc bị hủy bỏ thì tiền sẽ được trả lại.

Mà tiền không được trả lại. Vậy ban nhạc biết chơi nhạc rock.

Suy luận trên có dạng

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

$$r \rightarrow t$$

$$\frac{\neg t}{\therefore p}$$

Bước

Lý do

1. $r \rightarrow t$

Tiền đề

2. $\neg t$

Tiền đề

- | | |
|--|---|
| 3. $\neg r$ | B ₁ , B ₂ và phương pháp phủ định |
| 4. $\neg r \vee \neg s$ | B ₃ và khuyếch đại rời |
| 5. $\neg(r \wedge s)$ | B ₄ và DeMorgan |
| 6. $(\neg p \vee \neg q) \rightarrow (r \wedge s)$ | Tiền đề |
| 7. $\neg(\neg p \vee \neg q)$ | B ₅ , B ₆ và phương pháp phủ định |
| 8. $p \wedge q$ | B ₇ và DeMorgan |
| 9. $\therefore p$ | B ₈ và đơn giản nối liền |

Lưu ý 1:

$$\begin{array}{c}
 (1) \quad \begin{array}{c} p_1 \\ \vdots \\ p_n \end{array} \\
 \hline
 \therefore p \rightarrow q
 \end{array}
 \qquad
 \begin{array}{c}
 (2) \quad \begin{array}{c} p_1 \\ \vdots \\ p_n \end{array} \\
 \hline
 \begin{array}{c} p \\ \hline \therefore q \end{array}
 \end{array}$$

Hai suy luận trên là như nhau nên ta có thể xét (2) thay vì xét (1) và ngược lại.

Ví dụ 23: Xét suy luận sau

$$\begin{array}{c}
 u \rightarrow r \\
 (r \wedge s) \rightarrow (p \vee t) \\
 q \rightarrow (u \wedge s) \\
 \hline
 \neg t \\
 \hline
 \therefore q \rightarrow p
 \end{array}$$

Ta xét

$$\begin{array}{c}
 u \rightarrow r \\
 (r \wedge s) \rightarrow (p \vee t) \\
 q \rightarrow (u \wedge s) \\
 \neg t \\
 \hline
 q \\
 \hline
 \therefore p
 \end{array}$$

Bước	Lý do
1. $q \rightarrow (u \wedge s)$	Tiền đề
2. q	Tiền đề
3. $u \wedge s$	B ₁ , B ₂ và phương pháp khẳng định
4. u	B ₄ và đơn giản nối liền
5. $u \rightarrow r$	Tiền đề
6. r	B ₄ , B ₅ và phương pháp khẳng định
7. s	B ₃ và đơn giản nối liền
8. $r \wedge s$	B ₆ , B ₇ và nối liền
9. $(r \wedge s) \rightarrow (p \vee t)$	Tiền đề
10. $p \vee t$	B ₈ , B ₉ và phương pháp khẳng định
11. $\neg t$	Tiền đề
12. $\therefore p$	B ₁₀ , B ₁₁ và tam đoạn luận rời

Vậy suy luận trên đúng

Lưu ý 2: Chứng minh bằng phản chứng

$$(1) \quad \begin{array}{c} p_1 \\ \vdots \\ p_n \\ \hline \therefore q \end{array}$$

$$(2) \quad \begin{array}{c} p_1 \\ \vdots \\ p_n \\ \hline \neg q \\ \hline \therefore F_0 \end{array}$$

Hai suy luận trên là như nhau nên ta có thể xét (2) thay vì xét (1) và ngược lại.

Ví dụ 24: Xét suy luận sau

$$\begin{array}{c} \neg p \leftrightarrow q \\ q \rightarrow r \\ \hline \therefore p \end{array}$$

Chứng minh bằng phản chứng. Ta xét

$$\neg p \leftrightarrow q$$

$$q \rightarrow r$$

$$\neg r$$

$$\frac{\neg p}{\therefore F_0}$$

Bước**Lý do**

- | | |
|-------------------------------|---|
| 1. $q \rightarrow r$ | Tiền đề |
| 2. $\neg r$ | Tiền đề |
| 3. $\neg q$ | B_1, B_2 và phương pháp phủ định |
| 4. $\neg p \leftrightarrow q$ | Tiền đề |
| 5. $\neg p \rightarrow q$ | B_4 và tương đương logic |
| 6. p | B_3, B_5 , phương pháp phủ định và DeMorgan |
| 7. $\neg p$ | Tiền đề |
| 8. $\therefore F_0$ | B_6, B_7 , luật nối liền và phần tử bù |

Lưu ý 3: Cho

$$\frac{\begin{matrix} p_1 \\ \vdots \\ p_n \end{matrix}}{\therefore q}$$

Để chỉ ra rằng suy luận bên trên sai, ta chỉ cần chỉ ra một trường hợp trong đó tất cả p_i đều đúng mà q sai.

Ví dụ 25: Chứng tỏ rằng suy luận sau đây sai

$$\frac{\begin{matrix} p \\ p \vee q \\ q \rightarrow (r \rightarrow s) \\ t \rightarrow r \end{matrix}}{\therefore \neg s \rightarrow \neg t}$$

$\neg s \rightarrow \neg t$ sai $\Rightarrow \neg s$ đúng, $\neg t$ sai $\Rightarrow s=0, t=1$.

$t \rightarrow r$ đúng và $t=1 \Rightarrow r=1$

$r=1, s=0$ nên $r \rightarrow s$ sai

$$\text{mà } \begin{cases} q \rightarrow (r \rightarrow s) \text{ đúng} \\ r \rightarrow s \text{ sai} \end{cases} \Rightarrow q=0$$

Vậy với $p=1, q=0, t=1, r=1, s=0$ và $t=1$ thì các tiền đề đều đúng nhưng kết luận sai nên suy luận trên sai.

1.2.2. Những suy luận sai cần lưu ý

1. Nếu là tổng thống Mỹ thì lớn hơn hay bằng 35 tuổi. Mà tôi lớn hơn hay bằng 35 tuổi, vậy tôi là tổng thống Mỹ.

Suy luận trên có dạng

$$\frac{p \rightarrow q}{\therefore p}$$

2. Nếu $2+3=6$ thì $2+4=6$, mà $2+3 \neq 6$ nên $2+4 \neq 6$

$$\frac{p \rightarrow q}{\frac{\neg p}{\therefore \neg q}}$$

1.3. VỊ TỪ VÀ LUẬNG TÙ

1.3.1. Định nghĩa 8

Một phát biểu $p(x, y, \dots)$ với $x \in A, y \in B$ gọi là vị từ nếu

- $p(x, y, \dots)$ không là mệnh đề;
- Nếu thay x, y, \dots bởi a, b, \dots cụ thể thì $p(a, b, \dots)$ là mệnh đề;
 A, B, \dots gọi là vũ trụ. x, y, \dots gọi là các biến tự do.

Ví dụ 26:

a) Với $x \in R$, ta gọi $P(x) = "x^2 - 4 > 0"$

Đây là vị từ theo biến tự do x với vũ trụ R

P(1) là mệnh đề sai

P(-3) là mệnh đề đúng

b) Với $x, y \in N$, xét

$P(x,y) = "3x+y \text{ là số nguyên tố}"$

P(1,2) là mệnh đề đúng

P(3,5) là mệnh đề sai

$P(x,y,\dots)$ là vị từ theo hai biến tự do $x \in N, y \in N$

c) Với $n \in N, x \in R$, ta gọi $P(n, x) = "nx \geq 10"$ là vị từ theo 2 biến $n \in N, x \in R$

P(3,5) là mệnh đề đúng

P(1,3) là mệnh đề sai

1.3.2. Định nghĩa 9

Cho $p(x), q(x)$ là các vị từ theo $x \in A$

1. $\neg p(x)$ là vị từ mà khi thay x bằng a ta có mệnh đề $\neg p(a)$;

2. $p(x) \wedge q(x)$ là vị từ mà khi thay x bằng a ta có mệnh đề $p(a) \wedge q(a)$.

Tương tự cho $p(x) \vee q(x), p(x) \rightarrow q(x)$.

Ví dụ 27:

$$p(x) = "x^2 - 3x + 2 = 0" \quad x \in R$$

$$q(x) = "x - 1 > 0" \quad x \in R$$

$$\neg p(x) = "x^2 - 3x + 2 \neq 0" \quad x \in R$$

$$\neg q(x) = "x - 1 \leq 0" \quad x \in R$$

$$p(x) \wedge q(x) = "x^2 - 3x + 2 = 0 \text{ và } x - 1 > 0"$$

Cho vị từ $P(x), x \in A$

1. $P(a)$ đúng với $a \in A$

2. Một số $P(a)$ đúng, một số $P(a)$ sai

Trường hợp 1: ta viết $\forall x \in A, P(x)$

Trường hợp 2: ta viết $\exists x \in A, P(x)$

\forall là lượng từ phổ dụng, \exists là lượng từ tồn tại

1.3.3. Định nghĩa 10

Cho vị từ $P(x)$, $x \in A$. Các mệnh đề $\forall x \in A, p(x)$, $\exists x \in A, p(x)$ gọi là các mệnh đề lượng tử hóa của vị từ $p(x)$. Biến x gọi là biến buộc.

Ví dụ 28: Với $x \in R$, đặt

$$p(x) = "x \geq 0"$$

$$q(x) = "x^2 \geq 0"$$

$$r(x) = "x^2 - 3x - 4 = 0"$$

$$s(x) = "x^2 - 3 > 0"$$

Hãy cho biết chân trị của các mệnh đề sau đây

- a) $\exists x \in R, (p(x) \wedge r(x))$ đúng ($x = 4$)
- b) $\forall x \in R, (p(x) \rightarrow q(x))$ đúng
- c) $\exists x \in R, (p(x) \rightarrow q(x))$ đúng
- d) $\forall x \in R, (p(x) \rightarrow s(x))$ sai ($x = 1$)
- e) $\forall x \in R, (r(x) \vee s(x))$ sai ($x = 0$)
- f) $\forall x \in R, (r(x) \rightarrow p(x))$ sai ($x = -1$)

Cho vị từ $p(x,y)$, $x \in A$, $y \in B$

Lấy $x = a \in A$, coi $q(y) = p(a,y)$ là vị từ theo biến $y \in B$

Ta có thể lượng tử hóa $q(y)$

$$\exists y \in B, q(y) \text{ tức là } \exists y \in B, p(a,y)$$

$$\forall y \in B, q(y) \text{ tức là } \forall y \in B, p(a,y)$$

Coi $r(x) = \forall y \in B, p(x,y)$ $x \in A$

là vị từ theo x . Ta có thể lượng tử hóa

$$\forall x \in A, r(x) \text{ từ } \forall x \in A, \forall y \in B, p(x,y)$$

$$\exists x \in A, r(x) \text{ từ } \exists x \in A, \forall y \in B, p(x,y)$$

tương tự ta xét $s(x) = \exists y \in B, p(x,y)$

ta có $\forall x \in A, \exists y \in B, p(x,y)$

$$\exists x \in A, \exists y \in B, p(x,y)$$

Vậy với vị từ $p(x,y)$, $x \in A$, $y \in B$

Ta có 4 mệnh đề lượng tử hóa sau

$\forall x \in A, \forall y \in B, p(x, y)$

$\exists x \in A, \forall y \in B, p(x, y)$

$\forall x \in A, \exists y \in B, p(x, y)$

$\exists x \in A, \exists y \in B, p(x, y)$

Ví dụ 29:

Với $x, y \in R$, xét $p(x, y) = "x+y=5"$

Với $x = a \in R$, (tùy ý) lấy $y = 5-a$ thì $x+y = 5$

Vậy $\forall x \in R, \exists y \in R, p(x, y)$ là mệnh đề đúng (*)

Với $x \in R$ lấy $y = -x$ thì $x+y = 0 \neq 5$

Vậy $\exists x \in R, \forall y, p(x, y)$ sai (**)

Vậy $(*) \rightarrow (**)$ là mệnh đề sai

Định lý 6

Trong một mệnh đề lượng tử hoá từ vị từ $p(x, y, \dots)$ nếu ta hoán vị hai lượng tử đúng cạnh nhau thì mệnh đề mới tương đương logic với mệnh đề cũ nếu hai lượng tử này cùng loại. Mệnh đề mới là một hệ quả logic của mệnh đề cũ nếu hai lượng tử trước khi hoán vị có dạng \exists, \forall . Suy ra

$$[\forall x \in A, \forall y \in B, P(x, y)] \Leftrightarrow [\forall y \in B, \forall x \in A, p(x, y)]$$

$$[\exists x \in A, \exists y \in B, p(x, y)] \Leftrightarrow [\exists y \in B, \exists x \in A, p(x, y)]$$

$$[\exists x \in A, \forall y \in B, p(x, y)] \Rightarrow [\forall y \in B, \exists x \in A, p(x, y)]$$

Ví dụ 30:

$$\exists x \in R, \forall y \in R, x^*y=0 \Rightarrow \forall y \in R, \exists x \in R, x^*y=0$$

$$\forall x \in R, \exists y \in R, x^*y=x^2 \quad \times \quad \exists y \in R, \forall x \in R, x^*y=x^2$$

Định lý 7

Phủ định của một mệnh đề lượng tử hoá từ vị từ $p(x, y, \dots)$ có được bằng cách thay các lượng tử \forall bởi lượng tử \exists , các lượng tử \exists thì thay bởi các lượng tử phủ dụng và $p(x, y, \dots)$ thì thay bởi $\neg p(x, y, \dots)$

Ví dụ 30': (SV tự cho)

Định lý 8 (quy tắc đặc biệt hoá phủ dụng)

Cho vị từ $p(x), x \in A$. Nếu $\forall x \in A, p(x)$ là một mệnh đề đúng thì $p(a)$ đúng với $a \in A$.

Ví dụ 31:

Mọi giảng viên toán đều học toán rời rạc. Ông X là giảng viên toán.
Vậy ông X học toán rời rạc.

$$A = \{x / x \text{ là giảng viên}\}$$

$$p(x) = x \text{ là giảng viên toán}$$

$$q(x) = x \text{ học toán rời rạc}$$

Suy luận trên có dạng

$$\forall x, (p(x) \rightarrow q(x))$$

$$\frac{p(X)}{\therefore q(X)}$$

Giải

Bước

Lý do

- | | |
|---|-----------------------------------|
| 1. $\forall x, (p(x) \rightarrow q(x))$ | Tiền đề |
| 2. $p(X) \rightarrow q(X)$ | B1 và QT ĐB hoá phổ dụng |
| 3. $p(X)$ | Tiền đề |
| 4. $\therefore q(X)$ | B2, B3, và phương pháp khẳng định |

Ví dụ 32: Xét suy luận sau

Không có sinh viên toán hay CNTT học môn sinh, mà sinh viên M học môn sinh. Vậy M không phải là sinh viên toán

Đặt $p(x) = x \text{ là sinh viên toán}$

$q(x) = x \text{ là sinh viên CNTT}$

$r(x) = x \text{ học môn sinh}$

Suy luận trên có dạng

$$\forall x, p(x) \vee q(x) \rightarrow \neg(r(x))$$

$$\frac{r(M)}{\therefore \neg p(M)}$$

Giải

Bước

Lý do

- | | |
|--|---------|
| 1. $\forall x, p(x) \vee q(x) \rightarrow \neg r(x)$ | Tiền đề |
|--|---------|

2. $p(M) \vee q(M) \rightarrow \neg r(M)$	B1 và đặc biệt hoá phô dụng
3. $r(M)$	Tiền đề
4. $\neg(p(M) \vee q(M))$	B2, 3 và pp phủ định
5. $\neg p(M) \wedge \neg q(M)$	B4 và Demorgan
6. $\therefore \neg p(M)$	B5 và đơn giản nối liền

Quy tắc tổng quát hoá phô dụng:

Nếu với $a \in A$, a tùy ý. $P(a)$ là mệnh đề đúng thì $\forall x \in A, p(x)$ đúng

Ví dụ 33: Chứng minh

$$\begin{array}{c} \forall x \in A, p(x) \rightarrow q(x) \\ \hline \forall x \in A, q(x) \rightarrow r(x) \\ \hline \therefore \forall x \in A, p(x) \rightarrow r(x) \end{array}$$

Giải

Bước	Lý do
1. $\forall x \in A, p(x) \rightarrow q(x)$	Tiền đề
2. $\forall x \in A, q(x) \rightarrow r(x)$	Tiền đề
3. $p(a) \rightarrow q(a)$	B1 và đặc biệt hoá phô dụng
4. $q(a) \rightarrow r(a)$	B2 và đặc biệt hoá phô dụng
5. $p(a) \rightarrow r(a)$	B2, 4 và tam đoạn luận
$\therefore \forall x \in A, p(x) \rightarrow r(x)$	B5 và tổng quát hoá phô dụng

Ví dụ 34:

$\begin{array}{c} (1) \\ \hline \forall x, p(x) \vee q(x) \\ \hline \forall x, (\neg p(x) \wedge q(x)) \rightarrow r(x) \\ \hline \therefore \forall x, \neg r(x) \rightarrow p(x) \end{array}$	$\begin{array}{c} (2) \\ \hline \forall x, (p(x) \vee q(x)) \\ \forall x, (\neg p(x) \wedge q(x)) \rightarrow r(x) \\ \hline \forall x, \neg r(x) \\ \hline \therefore \forall x, p(x) \end{array}$
---	---

Hai suy luận trên là như nhau nên ta xét suy luận (2) thay vì xét suy luận (1)

Bước	Lý do
1. $\forall x, (p(x) \vee q(x))$	Tiền đề
2. $p(a) \vee q(a)$	B1 và ĐBHPD
3. $\forall x, (\neg p(x) \wedge q(x)) \rightarrow r(x)$	Tiền đề
4. $(\neg p(a) \wedge q(a)) \rightarrow r(a)$	B3 và ĐBHPD
5. $\neg r(a) \rightarrow \neg(\neg p(a) \wedge q(a))$	B4 và tđ logic (pp phủ định)
6. $\neg r(a) \rightarrow p(a) \vee \neg q(a)$	B5 và DeMorgan
7. $\forall x, \neg r(x)$	Tiền đề
8. $\neg r(a)$	B7 và ĐBHPD
9. $p(a) \vee \neg q(a)$	B6, 8 và pp khẳng định
10. $(p(a) \vee q(a)) \wedge (p(a) \vee \neg q(a))$	B2, 9 và pp nối liền
11. $p(a) \vee (q(a) \wedge \neg q(a))$	B10 và phân phối
12. $p(a) \vee F_0$	B11 và pp phân tử bù
13. $p(a)$	B12 và pp phân tử trung hòa

BÀI TẬP CHƯƠNG 1

1. Cho hai dạng mệnh đề $p \rightarrow q$ và $p \wedge \neg q$.
- Lập bảng chọn trị cho $\neg(p \rightarrow q)$ và $p \wedge \neg q$
 - Dựng quy tắc thay thế để kiểm tra dạng mệnh đề sau là hằng đúng
 $\neg(p \rightarrow q) \leftrightarrow (p \wedge \neg q)$.

Cho biết các quy luật logic nào được áp dụng trong mỗi bước tương đương?

2. Hãy chỉ ra các hằng đúng trong các dạng mệnh đề sau
- | | |
|---|--|
| a) $(p \vee q) \rightarrow (p \wedge q)$ | b) $(p \wedge q) \rightarrow (p \vee q)$ |
| c) $p \rightarrow (\neg q \rightarrow p)$ | d) $p \rightarrow (p \rightarrow q)$ |
3. Trong các khẳng định sau, hãy chỉ ra các khẳng định đúng:
- $q \Rightarrow p \rightarrow q$
 - $\neg(p \rightarrow q) \Rightarrow p$
 - $(p \wedge q) \vee r \Rightarrow p \wedge (q \vee r)$
4. Có thể nói gì về một dạng mệnh đề:
- Có hệ quả logic là một mâu thuẫn?
 - Có hệ quả logic là một hằng đúng?
 - Là hệ quả logic của một mâu thuẫn?
 - Là hệ quả logic của một hằng đúng?
5. Cho biết các quy luật logic nào được áp dụng trong mỗi bước tương đương sau

Biểu thức	Quy luật logic
$(q \wedge p) \vee \neg(q \rightarrow p)$	
$\Leftrightarrow (q \wedge p) \vee \neg(\neg q \vee p)$	
$\Leftrightarrow (q \wedge p) \vee (\neg(\neg q) \wedge \neg p)$	
$\Leftrightarrow (q \wedge p) \vee (q \wedge \neg p)$	
$\Leftrightarrow q \wedge (p \vee \neg p)$	
$\Leftrightarrow q \wedge 1$	
$\Leftrightarrow q$	

6. Xét vị từ

$$p(x): x^2 - 4x + 3 = 0,$$

$$q(x): x > 0.$$

Cho \mathbf{R} là tập các số thực. Xác định chân trị của các mệnh đề

- a) $\forall x \in \mathbf{R}, p(x) \rightarrow q(x)$
- b) $\exists x \in \mathbf{R}, p(x) \rightarrow \neg q(x)$
- c) $\forall x \in \mathbf{R}, p(x) \rightarrow \neg q(x)$

7. Kiểm tra các suy luận sau

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

$$p \vee s$$

$$t \rightarrow q$$

$$\frac{\neg s}{\therefore \neg r \rightarrow \neg t}$$

8. Rút gọn

$$E = \neg [\neg ((p \vee q) \wedge r) \vee \neg q]$$

9. Suy luận sau đúng hay sai?

$$p \rightarrow r$$

$$\neg p \rightarrow q$$

$$\frac{q \rightarrow s}{\therefore \neg r \rightarrow s}$$

10.

$$\forall x \in A, p(x) \rightarrow q(x)$$

Chứng minh $\frac{\forall x \in A, q(x) \rightarrow r(x)}{\therefore \forall x \in A, p(x) \rightarrow r(x)}$

Chương 2

QUAN HỆ HAI NGÔI

2.1. KHÁI NIỆM CHUNG

Định nghĩa 1

Một quan hệ giữa tập A và tập B là một tập con \mathcal{R} của $A \times B$. Nếu $(a,b) \in \mathcal{R}$ ta viết $a \mathcal{R} b$. Nếu $B = A$, ta có một quan hệ trên A (quan hệ 2 ngôi)

Ví dụ 1:

$$A = \{1, 2, 3\}$$

$$B = \{x, y, z, t\}$$

$$\mathcal{R} = \{(1,x), (1,z), (2,y), (3,y), (3,t)\} \subset A \times B$$

$$\mathcal{R}_1 = \{(1,2), (1,3), (2,3)\} \subset A \times A \text{ (quan hệ 2 ngôi trên A)}$$

Nhận xét 1:

Nếu $|A| = m$, $|B| = n$ thì có $2^{m \times n}$ quan hệ giữa A và B.

Định nghĩa 2

Cho \mathcal{R} là một quan hệ trên A. Ta nói

i. \mathcal{R} phản xạ nếu $(x,x) \in \mathcal{R} \quad \forall x \in A$

ii. \mathcal{R} đối xứng nếu $(x,y) \in \mathcal{R} \Rightarrow (y,x) \in \mathcal{R}$

iii. \mathcal{R} phản xứng nếu $x, y \in A, (x,y) \in \mathcal{R} \Rightarrow (y,x) \notin \mathcal{R}$

Hay $\begin{cases} (x,y) \in R \\ (y,x) \in R \end{cases} \Rightarrow x = y$

iv. \mathcal{R} bắc cầu nếu $\begin{cases} (x,y) \in R \\ (y,z) \in R \end{cases} \Rightarrow (x,z) \in R$

Ví dụ 2:

$$A = \{1, 2, 3\}$$

$$\mathcal{R} = \{(1,1), (3,3), (1,2), (2,3), (3,2)\}$$

\mathcal{R} không phản xạ vì $(2,2) \notin \mathcal{R}$

\mathcal{R} không đối xứng vì $(1,2) \in \mathcal{R}$ và $(2,1) \notin \mathcal{R}$

\mathcal{R} không phản xứng vì $(2,3) \in \mathcal{R}$ và $(3,2) \in \mathcal{R}$

\mathcal{R} không bắc cầu vì $\begin{cases} (1,2) \in R \\ (2,3) \in R \end{cases}$ mà $(1,3) \notin \mathcal{R}$

Ví dụ 3:

Cho $A = \{x/x \text{ là sinh viên khoa CNTT}\}$

Trên A ta định nghĩa quan hệ \mathcal{R} như sau:

$x, y \in A, x \mathcal{R} y \Leftrightarrow x \text{ cùng tuổi với } y$

\mathcal{R} phản xạ vì $x \in A, x \text{ cùng tuổi với } x$

$\Rightarrow x \mathcal{R} x, \forall x \in A \Rightarrow \mathcal{R}$ phản xạ

\mathcal{R} đối xứng vì $x \mathcal{R} y \Rightarrow x \text{ cùng tuổi với } y \Rightarrow y \text{ cùng tuổi với } x \Rightarrow y \mathcal{R} x \Rightarrow \mathcal{R}$ đối xứng

\mathcal{R} bắc cầu vì $x \mathcal{R} y$ và $y \mathcal{R} z \Rightarrow x \text{ cùng tuổi với } y \text{ và } y \text{ cùng tuổi với } z$

$\Rightarrow x \text{ cùng tuổi với } z \Rightarrow x \mathcal{R} z \Rightarrow \mathcal{R}$ bắc cầu

Ví dụ 4:

Cho $A = \{\text{Đường thẳng trong không gian}\}$

Trên A ta định nghĩa quan hệ \mathcal{R} như sau:

$d, d' \in A, d \mathcal{R} d' \Leftrightarrow d \text{ song song } d'$ hoặc d trùng d'

\mathcal{R} có tính chất phản xạ, đối xứng, bắc cầu

Ví dụ 5:

Cho $n \in \mathbb{N}$, trên \mathbb{Z} ta định nghĩa quan hệ \mathcal{R} như sau:

$x, y \in \mathbb{Z}, x \mathcal{R} y \Leftrightarrow n|x-y \Leftrightarrow x-y=kn, k \in \mathbb{Z}$

\mathcal{R} phản xạ: $\forall x \in \mathbb{Z}, x-x=0n \Rightarrow x \mathcal{R} x \quad \forall x \in \mathbb{Z}$

\mathcal{R} đối xứng:

$x \mathcal{R} y \Rightarrow x-y=kn, k \in \mathbb{Z} \Rightarrow y-x=(-k)n=k'n, k' \in \mathbb{Z} \Rightarrow y \mathcal{R} x$

\mathcal{R} bắc cầu:

$\begin{cases} x \mathcal{R} y \Rightarrow \begin{cases} x-y=kn \\ y-z=k'n \end{cases} \Rightarrow x-z=(k+k')n=k''n & k'' \in \mathbb{Z} \Rightarrow x \mathcal{R} z \end{cases}$

Ví dụ 6:

Cho $S \neq \emptyset$ và $A = \mathcal{P}(S)$. Trên A ta định nghĩa quan hệ \mathcal{R} như sau:

$$X, Y \in A, X R Y \Leftrightarrow X \subset Y$$

\mathcal{R} có tính chất phản xạ, phản xứng và bắc cầu

Ví dụ 7:

Cho $n \in \mathbb{N}$, đặt $\mathcal{U}_n = \{x / x \text{ là ước của } n\}$

$$\begin{aligned} \mathcal{U}_6 &= \{1, 2, 3, 6\}, & \mathcal{U}_{12} &= \{1, 2, 3, 4, 6, 12\}, & \mathcal{U}_{18} &= \{1, 2, 3, 6, 9, 18\}, \\ \mathcal{U}_{24} &= \{1, 2, 3, 4, 6, 8, 12, 24\} \end{aligned}$$

Trên \mathcal{U}_n ta định nghĩa quan hệ \mathcal{R} như sau:

$$x, y \in \mathcal{U}_n: x R y \Leftrightarrow x | y$$

\mathcal{R} có tính chất phản xạ, phản xứng và bắc cầu

2.2. QUAN HỆ TƯƠNG ĐƯƠNG

Định nghĩa 3

Cho \mathcal{R} là quan hệ trên A , \mathcal{R} gọi là quan hệ tương đương nếu \mathcal{R} phản xạ, đối xứng và bắc cầu

Ví dụ 8: Cho $A \neq \emptyset$, trên A ta định nghĩa quan hệ \mathcal{R} như sau:

$$x, y \in A, x R y \Leftrightarrow x = y$$

Dễ dàng ta kiểm chứng được \mathcal{R} là một quan hệ tương đương

Ký hiệu: Nếu \mathcal{R} là một quan hệ tương đương và $x R y$ ta thường viết $x \sim y$

Ví dụ 9: A với quan hệ \mathcal{R} trong các ví dụ 3, 4, 5 là quan hệ tương đương

Định nghĩa 4

Cho \mathcal{R} là 1 quan hệ tương đương trên A

i. Với $x \in A$, ta định nghĩa lớp tương đương chứa x là

$$C|(x) = \bar{x} = \{y \in A, y R x\}$$

ii. Tập hợp thương A/\mathcal{R} là tập hợp các lớp tương đương

$$A/\mathcal{R} = \{\bar{x} / x \in A\}$$

Ví dụ 10:

$$A = \{-3, -2, -1, 0, 1, 2, 3\}$$

Trên A ta định nghĩa quan hệ \mathcal{R} như sau:

$$x, y \in A, x \mathcal{R} y \Leftrightarrow x^2 = y^2$$

Ta có \mathcal{R} là một quan hệ tương đương

$$\overline{x} = \{y \in A / y \mathcal{R} x\} = \{y \in A / y^2 = x^2\} = \{y \in A / y = x \text{ hay } y = -x\} = \{x, -x\}$$

$$\overline{0} = \{0\}, \quad \overline{1} = \{-1, 1\}, \quad \overline{2} = \{-2, 2\}, \quad \overline{3} = \{-3, 3\}$$

$$\text{Tập hợp thương: } A/R = \{\overline{0}, \overline{1}, \overline{2}, \overline{3}\}$$

Ví dụ 11:

A = tập hợp các học viên lớp X gồm

a, d, m	: 17 tuổi
b, c, e, g	: 18 tuổi
f, p	: 25 tuổi
h, k, n, q, r	: 60 tuổi
s	: 80 tuổi
u	: 10 tuổi

Trên A ta định nghĩa quan hệ \mathcal{R} như sau:

$$x, y \in A, x \mathcal{R} y \Leftrightarrow x \text{ cùng tuổi với } y$$

Ta có \mathcal{R} là một quan hệ tương đương

Lớp tương đương

$$\overline{a} = \{x / x \mathcal{R} a\} = \{x / x \text{ cùng tuổi với } a\} = \{a, d, m\}$$

$$\overline{b} = \{b, c, e, g\}, \quad \overline{f} = \{f, p\}, \quad \overline{h} = \{h, k, n, q, r\},$$

$$\overline{s} = \{s\}, \quad \overline{u} = \{u\}$$

$$\text{Tập hợp thương: } A/R = \{\overline{a}, \overline{b}, \overline{f}, \overline{h}, \overline{s}, \overline{u}\}$$

2.3. QUAN HỆ THỨ TỰ

Định nghĩa 5: Một quan hệ \mathcal{R} trên A gọi là một quan hệ thứ tự nếu nó có các tính chất phản xạ, phản xứng và bắc cầu.

Ví dụ 12:

Cho $A \in \mathbb{R}$, trên A ta định nghĩa $x \mathcal{R} y \Leftrightarrow x \leq y$

Dễ dàng kiểm chứng được \mathcal{R} là một quan hệ thứ tự.

Ví dụ 13:

Trên $\mathcal{P}(S)$, ta định nghĩa $X \mathcal{R} Y \Leftrightarrow X \subset Y$.

Theo ví dụ 6, \mathcal{R} là một quan hệ thứ tự.

Ví dụ 14:

Trên \mathcal{U}_n ta định nghĩa $x \mathcal{R} y \Leftrightarrow x | y$

Theo ví dụ 7, \mathcal{R} là một quan hệ thứ tự.

Ký hiệu

Nếu \mathcal{R} là quan hệ thứ tự trên A và $x \mathcal{R} y$ ta viết $x \prec y$.

(A, \prec) gọi là một tập hợp có thứ tự

Định nghĩa 6

Cho (A, \prec) và $x, y \in A$.

- i. Viết $x \prec y$, ta nói y là trội của x hay x được trội bởi y
- ii. y gọi là trội trực tiếp của x nếu

$x \prec y$ và không có z sao cho $x \prec z \prec y$

Ví dụ 15: $\mathcal{U}_{12} = \{1, 2, 3, 4, 6, 12\}$

$$x \prec y \Leftrightarrow x | y$$

Ta có $\begin{cases} 3 \prec 6 \\ 3 \prec 12 \end{cases}$ 6, 12 là trội của 3

$3 \prec 6 \prec 12$ 6 là trội trực tiếp của 3

Các trội của 2 là 4, 6, 12

Trội trực tiếp của 2 là 4, 6.

Ví dụ 16:

$$S = \{1, 2, 3, 4\}$$

Xét $(\mathcal{P}(S), \subset)$

Trội trực tiếp của $X = \{1, 3\}$ là $\{1, 2, 3\}, \{1, 3, 4\}$

Trội trực tiếp của $X = \{2\}$ là $\{1, 2\}, \{2, 3\}, \{2, 4\}$

Định nghĩa 7

Cho (A, \prec) với A hữu hạn.

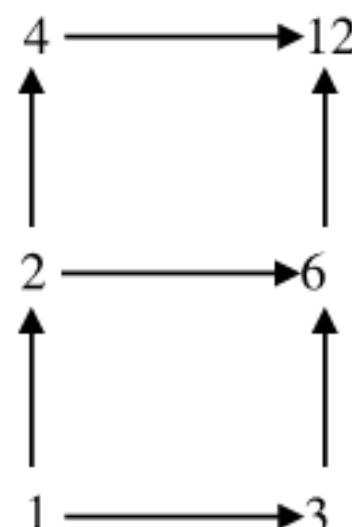
Biểu đồ Hasse của (A, \prec) gồm:

Tập điểm: Mỗi điểm biểu diễn 1 phần tử của A;

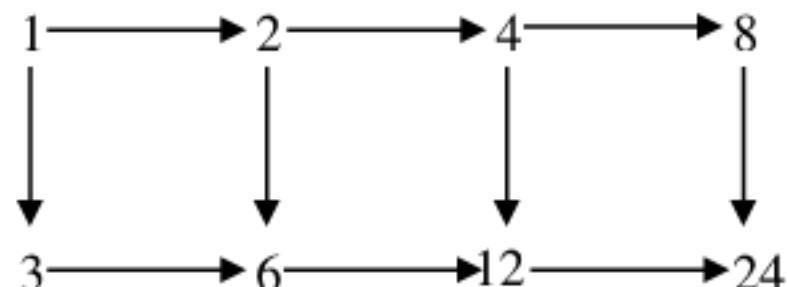
Tập cung: Vẽ một cung từ x đến y nếu y là trội trực tiếp của x.

Ví dụ 17:

$$\mathcal{U}_{12} = \{1, 2, 3, 4, 6, 12\}$$



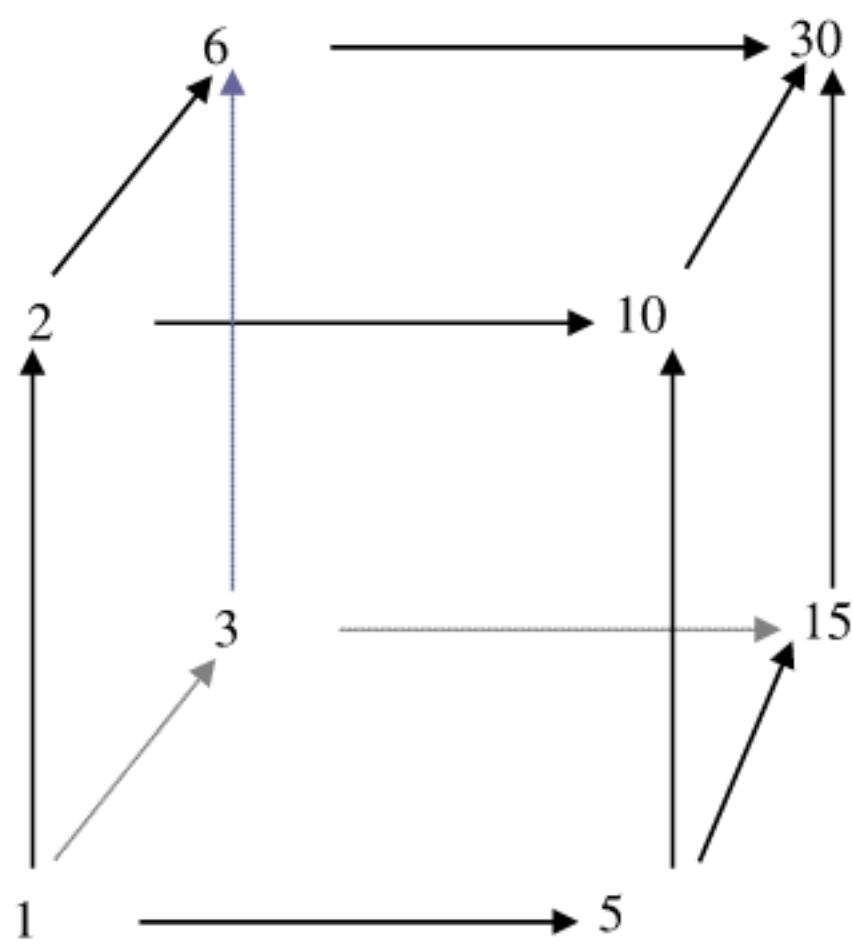
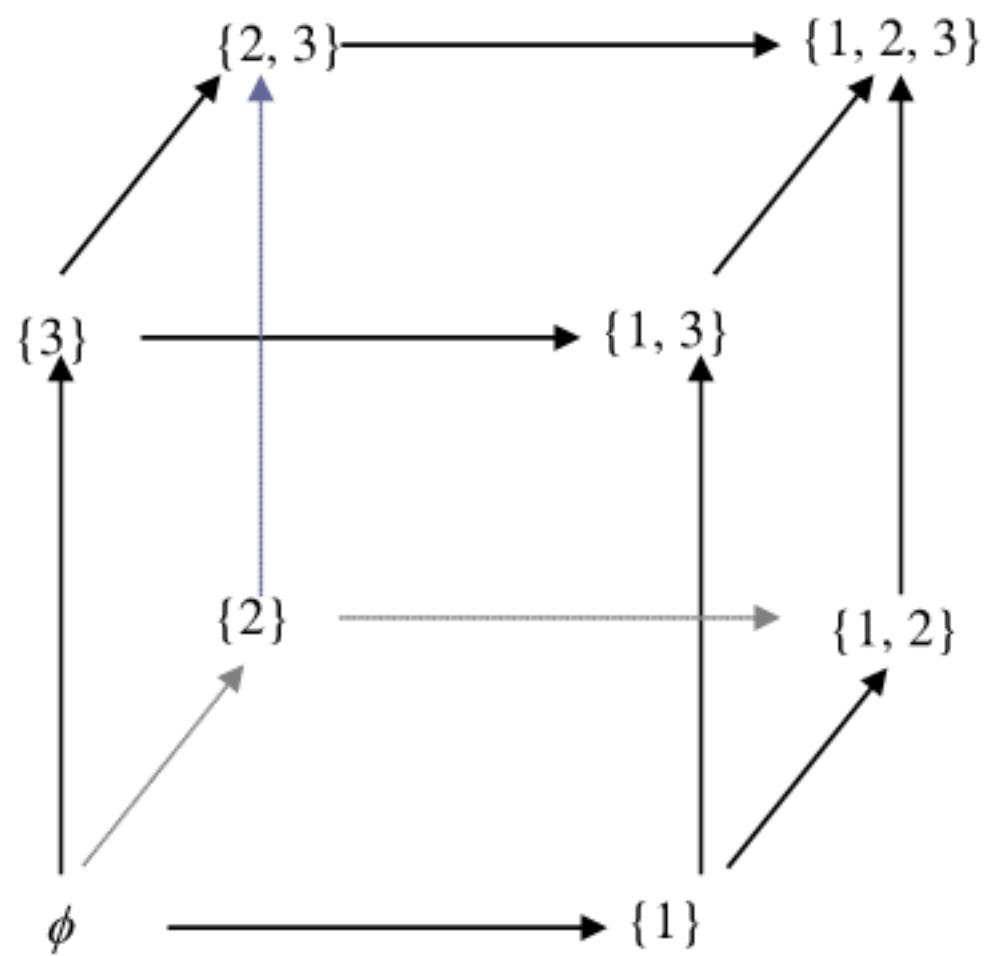
$$\mathcal{U}_{24} = \{1, 2, 3, 4, 6, 8, 12, 24\}$$



Ví dụ 18:

$$S = \{1, 2, 3\} \quad (\mathcal{P}(S), \subset)$$

$$\mathcal{U}_{30} = \{1, 2, 3, 5, 6, 10, 15, 30\}$$



Định nghĩa 8

Cho (A, \prec)

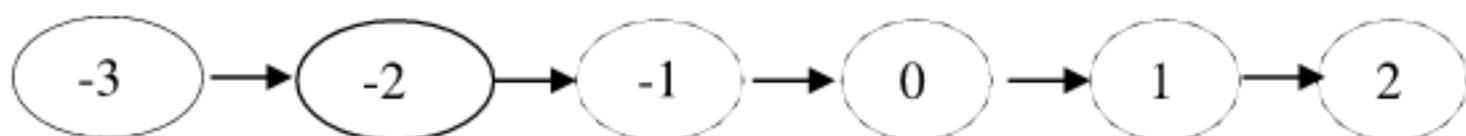
- i) Với $x, y \in A$ ta bảo x và y so sánh được nếu $x \prec y$ hay $y \prec x$;
- ii) Nếu $\forall x, y \in A$, x và y so sánh được thì ta nói \prec là quan hệ thứ tự toàn phần. Ngược lại \prec gọi là quan hệ thứ tự bán phần và A gọi là một Poset (Partially ordered set).

Ví dụ 19:

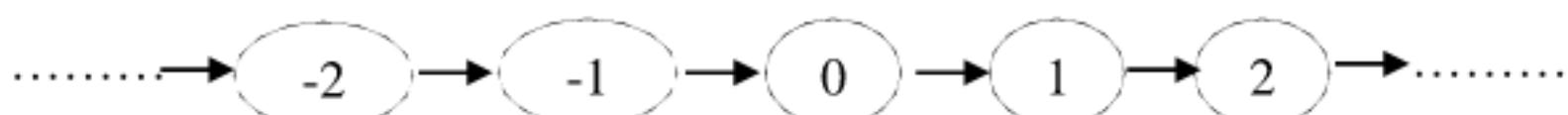
a/ Cho $A = \{-3, -2, -1, 0, 1, 2\}$

Với $x \prec y \Leftrightarrow x \leq y$

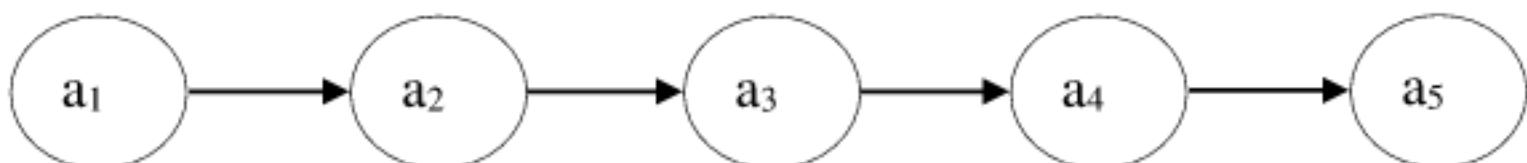
(A, \leq) là tập hợp có thứ tự toàn phần



b/ $A = \mathbb{Z}$ với thứ tự thông thường là một quan hệ thứ tự toàn phần



c/ Đảo lại cho (A, \prec) với sơ đồ Hasse



(A, \prec) là tập có thứ tự toàn phần.

Ví dụ 20:

Cho $U_{24} = \{1, 2, 3, 4, 6, 8, 12, 24\}$

Với thứ tự $x \prec y \Leftrightarrow x \mid y$

3 và 6 so sánh được vì $3 \prec 6$.

3 và 8 không so sánh được.

Vậy đây là quan hệ thứ tự bán phần.

Ví dụ 21:

Xét $(\mathcal{P}(S), \subset)$

$A, B \in \mathcal{P}(S): A \prec B \Leftrightarrow A \subset B$

$|S|=2$ thì đây là quan hệ thứ tự bán phần.

Chứng minh:

Gọi 2 phần tử của S là a, b . Vậy $S = \{a, b\}$.

Suy ra $\mathcal{P}(S) = \{\emptyset, \{a\}, \{b\}, \{a, b\}\}$.

Do $\{a\}$ và $\{b\}$ không so sánh được

nên $(\mathcal{P}(S), \subset)$ là quan hệ thứ tự bán phần.

Từ ví dụ 19 ta có Nhận xét I:

Sơ đồ Hasse của (A, \prec) là 1 dây chuyền $\Leftrightarrow (A, \prec)$ là một tập thứ tự toàn phần.

Định nghĩa 9

Cho (A, \prec)

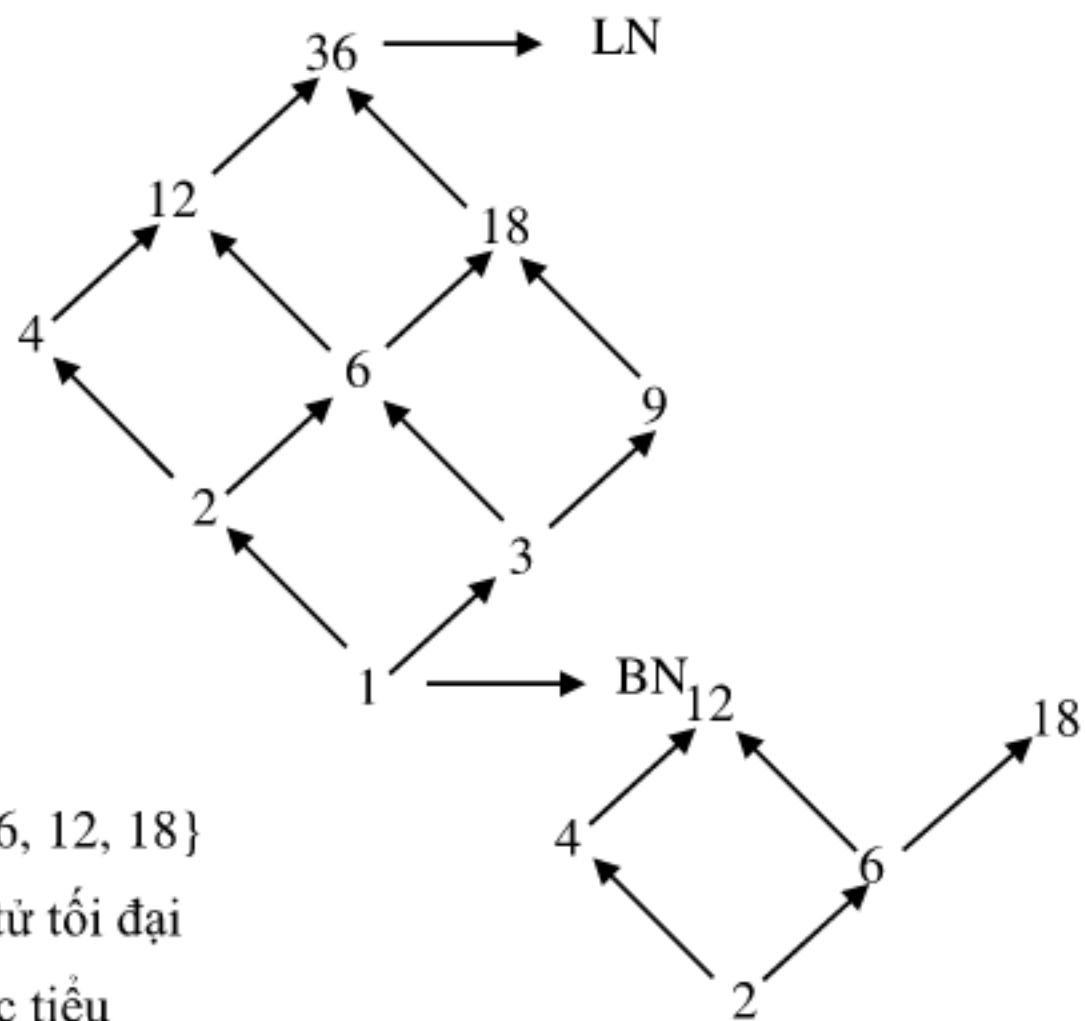
i/ Phần tử $m \in A$ gọi là

- Tối đại nếu $m \prec x \Rightarrow x = m$
- Cực đại hay lớn nhất nếu $\forall x \in A, x \prec m$

ii/ Phần tử $m' \in A$ gọi là

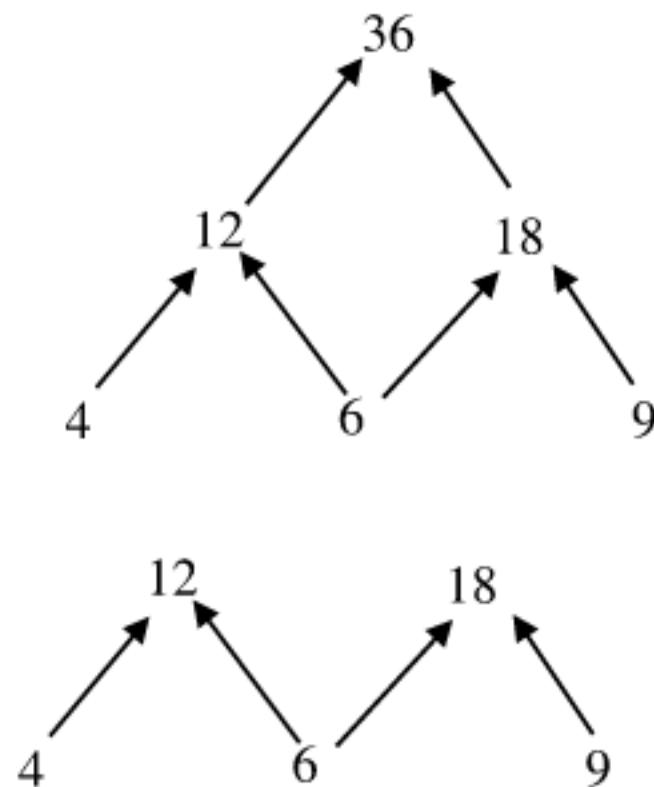
- Tối thiểu nếu $x \prec m' \Rightarrow x = m'$
- Cực thiểu hay nhỏ nhất nếu $\forall x \in A, m' \prec x$

Ví dụ 22: Cho $U_{36} = \{1, 2, 3, 4, 6, 9, 12, 18, 36\}$



Xét $B = \{4, 6, 9, 12, 18, 36\}$
 36 là lớn nhất
 4, 6, 9 là các phần tử tối thiểu

Xét $C = \{4, 6, 9, 12, 18\}$
 12, 18 là phần tử tối đại
 4, 6, 9 là phần tử tối thiểu



Nhận xét 2

Phần tử lớn nhất nếu có là duy nhất.

Nếu n là phần tử tối đại duy nhất thì đó cũng là phần tử cực đại. Tương tự cho tối thiểu, cực thiểu.

2.4. DÀN (LATTICE)

Định nghĩa 10

Cho (A, \preceq) và $B \subset A$

i. $c \in A$ gọi là chặn trên của B nếu

$$x \preceq c \quad \forall x \in B$$

Đặt $X = \{c \in A; c \text{ là chặn trên của } B\} \subset A$. Phần tử nhỏ nhất của X nếu có gọi là chặn trên nhỏ nhất của B . Ký hiệu $\text{Sup}B$

ii. $c \in A$ gọi là chặn dưới của B nếu

$$c \preceq x \quad \forall x \in B$$

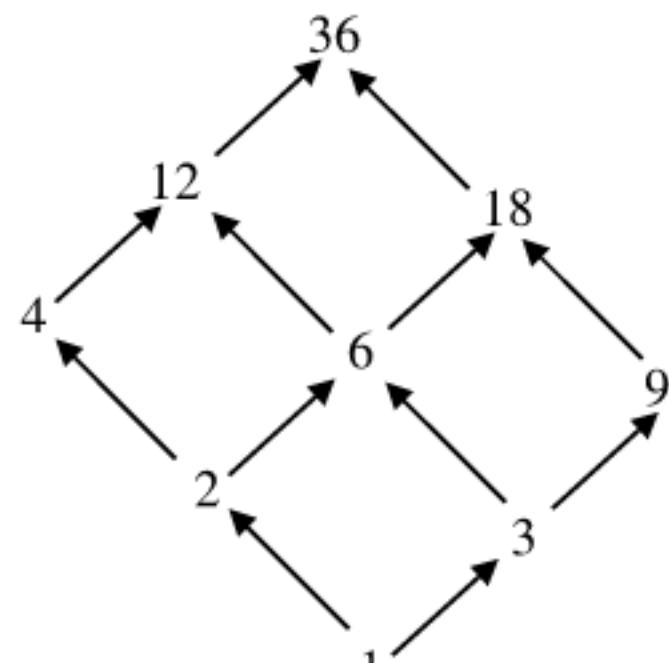
Đặt $Y = \{c \in A; c \text{ là chặn dưới của } B\} \subset A$. Phần tử lớn nhất của Y nếu có gọi là chặn dưới lớn nhất của B . Ký hiệu $\text{Inf}B$

Ví dụ 23:

$$B = \{2, 4, 6\} \Rightarrow \begin{cases} \text{Sup}B = 12 \\ \text{Inf}B = 2 \end{cases}$$

$$B = \{6, 9, 18\} \Rightarrow \begin{cases} \text{Sup}B = 18 \\ \text{Inf}B = 3 \end{cases}$$

$$B = \{3, 4\} \Rightarrow \begin{cases} \text{Sup}B = 12 \\ \text{Inf}B = 1 \end{cases}$$



Nhận xét 3:

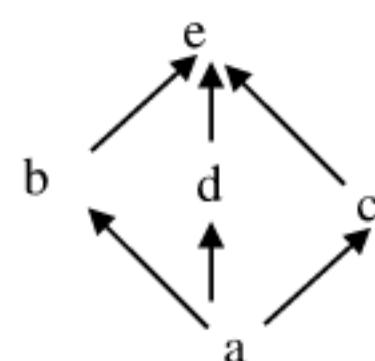
Nếu B có phần tử lớn nhất là b thì $\text{Sup}B = b$.

Nếu B có phần tử bé nhất là c thì $\text{Inf}B = c$.

Ví dụ 24:

$$B = \{b, c\} \Rightarrow \begin{cases} \text{Sup}B = e \\ \text{Inf}B = a \end{cases}$$

$$B = \{a, d, e\} \Rightarrow \begin{cases} \text{Sup}B = e \\ \text{Inf}B = a \end{cases}$$

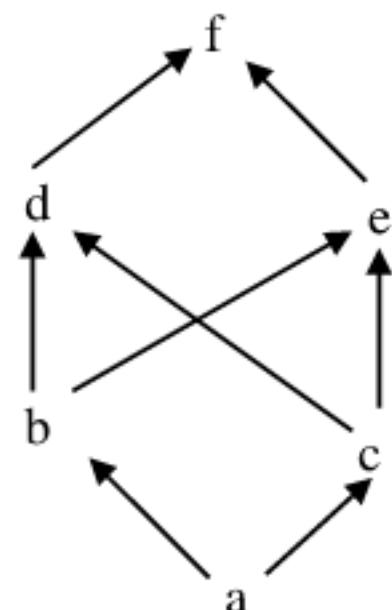


Ví dụ 25:

$$B = \{b, c, d\} \Rightarrow \begin{cases} SupB = d \\ InfB = a \end{cases}$$

$$B = \{b, c\} \Rightarrow \begin{cases} SupB & \text{Không có} \\ InfB = a & \end{cases}$$

$$X = \{c \in A / x \prec c \quad \forall x \in B\} = \{d, e, f\}$$



Phần tử bé nhất của X không tồn tại.

$$B = \{d, e\} \Rightarrow supB = f$$

Tập các chặn dưới của B là Y = {b, c, a}, tập này không có phần tử lớn nhất.

Ví dụ 26:

Cho $B = \{B_1, \dots, B_k\} \subset \mathcal{P}(S)$ với quan hệ bao hàm.

$$\text{Ta có } SupB = \bigcup_{i=1}^k B_i$$

$$InfB = \bigcap_{i=1}^k B_i$$

Chứng minh:

$$\text{Đặt } B_0 = \bigcup_{i=1}^k B_i, \text{ ta có } B_i \subset B_0 \quad \forall i = 1, \dots, k$$

Cho A là một chặn trên của B, tức $B_i \subset A \quad \forall i = 1, \dots, k$

$$\Rightarrow \bigcup_{i=1}^k B_i \subset A \Rightarrow B_0 \subset A$$

$$\text{Tóm lại } \left\{ \begin{array}{l} B_i \prec B_0 \quad \forall i = 1, \dots, k \\ B_0 = \min \{X / X \text{ là chặn trên của } B\} \end{array} \right.$$

Vậy $B_0 = SubB$

Tương tự cho InfB

Định nghĩa 11

Cho (A, \prec) , A gọi là một dàn nếu $\forall x, y \in A$ thì $\sup\{x, y\}$ và $\inf\{x, y\}$ tồn tại.

Khi đó ta viết $\sup\{x, y\} = x \vee y$, $\inf\{x, y\} = x \wedge y$

Ví dụ 27:

Xét $(\mathcal{P}(S), \subset)$

$\forall A, B \in \mathcal{P}(S)$, ta có (theo VD 26)

$$A \vee B = \text{Sup}\{A, B\} = A \cup B$$

$$A \wedge B = \text{Inf}\{A, B\} = A \cap B$$

Vậy $(\mathcal{P}(S), \subset)$ là một dàn

Ví dụ 28:

$$\mathcal{U}_8 = \{1, 2, 4, 8\}$$



$x, y \in \mathcal{U}_8$, ta có

$$\begin{cases} x \prec y \\ y \prec x \end{cases}$$

Nếu $x \prec y$ thì $x \vee y = y$, $x \wedge y = x$

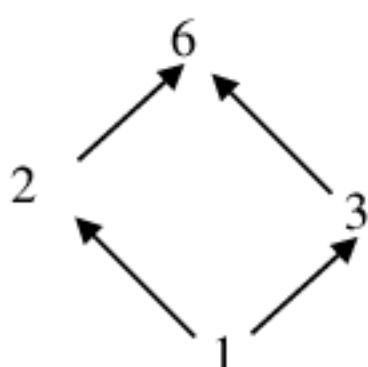
Nếu $y \prec x$ thì $x \vee y = x$, $x \wedge y = y$

Nhận xét 4:

(A, \prec) với thứ tự toàn phần là một dàn.

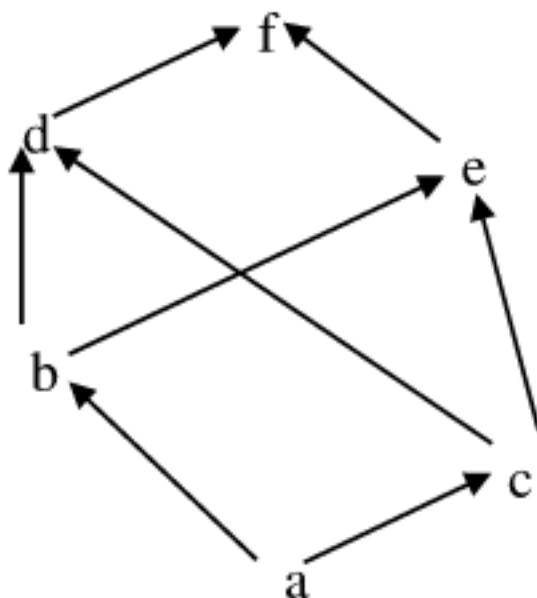
Ví dụ 29:

$$\mathcal{U}_6 = \{1, 2, 3, 6\}$$



$\{x, y\}$	$\text{Sup}\{x, y\}$	$\text{Inf}\{x, y\}$
{1,2}	2	1
{1,3}	3	1
{1,6}	6	1
{2,3}	6	1
{2,6}	6	2
{3,6}	6	3

Ví dụ 30:



Theo VD25, $\{b, c\}$ không có sup, suy ra đây không phải là một dàn.

Định lý 2

Cho (A, \prec) là một dàn và $x, y, z \in A$, ta có

i. $x \vee y = y \vee x$

$$x \wedge y = y \wedge x$$

ii. $x \vee (y \vee z) = (x \vee y) \vee z$

$$x \wedge (y \wedge z) = (x \wedge y) \wedge z$$

Nhận xét 5:

$B = \{x_1, \dots, x_n\}$ thì

$$\text{Sup } B = x_1 \vee x_2 \vee \dots \vee x_n$$

$$\text{Inf } B = x_1 \wedge x_2 \wedge \dots \wedge x_n$$

Đặc biệt $A = \{a_1, \dots, a_n\}$ thì

$$a_1 \vee a_2 \vee \dots \vee a_n = \text{phần tử lớn nhất của } A$$

$$a_1 \wedge a_2 \wedge \dots \wedge a_n = \text{phần tử nhỏ nhất của } A$$

Định nghĩa 13

Một dàn (A, \prec) gọi là dàn phân bố nếu với mọi $x, y, z \in A$, ta có

$$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$$

$$x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$$

Ví dụ 31:

Xét dàn $(\mathcal{P}(S), \subset)$

$$A, B \in \mathcal{P}(S), \text{ theo VD 27 ta có} \quad A \vee B = A \cup B$$

$$A \wedge B = A \cap B$$

Vậy với $A, B, C \in \mathcal{P}(S)$ thì

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

$$\text{Tương tự } A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C)$$

Vậy $(\mathcal{P}(S), \subset)$ là một dàn phân bố.

Ví dụ 32:

Xét \mathcal{U}_8 , cho $x, y, z \in \mathcal{U}_8$, ta có thể giả sử $x \prec y \prec z$

$$x \vee (y \wedge z) = x \vee y = y$$

$$(x \vee y) \wedge (x \vee z) = y \wedge z = y$$

$$\text{Suy ra } x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$$

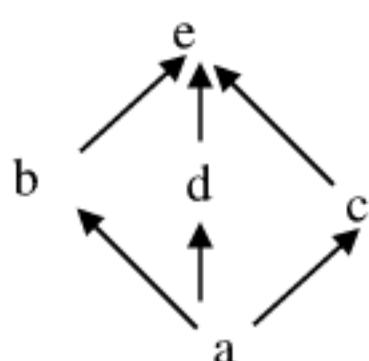
$$\text{Tương tự } x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$$

Vậy \mathcal{U}_8 là một dàn phân bố.

Nhận xét 6

Như đã chứng minh trong ví dụ 32, (A, \prec) với thứ tự toàn phần là một dàn phân bố.

Ví dụ 33:



$$d \wedge (b \vee c) = d \wedge e = d$$

$$(d \wedge b) \vee (d \wedge c) = a \vee a = a$$

$$\Rightarrow d \wedge (b \vee c) \neq (d \wedge b) \vee (d \wedge c)$$

Suy ra đây không là dàn phân bố.

Định nghĩa 13

Cho (A, \prec) là một dàn. Giả sử A có phần tử lớn nhất là 1, phần tử nhỏ nhất là 0.

i. \bar{x} gọi là phần tử bù của $x \in A$ nếu

$$x \vee \bar{x} = 1$$

$$x \wedge \bar{x} = 0$$

ii. A gọi là dàn bù nếu với mọi $x \in A$, x có phần tử bù.

Ví dụ 34:

Xét U_6 , U_6 có phần tử lớn nhất là 6 và phần tử nhỏ nhất là 1. Vậy U_6 là một dàn bù.

x	\bar{x}
1	6
6	1
2	3
3	2

BÀI TẬP CHƯƠNG 2

1. Trong các quan hệ sau (được định nghĩa trên tập các số nguyên dương) hãy cho biết quan hệ nào có tính phản xạ, đối xứng, phản xứng, bắc cầu, một thứ tự? Chứng minh.

a) $(x, y) \in R \Leftrightarrow x = y^2$

b) $(x, y) \in R \Leftrightarrow x > y$

c) $(x, y) \in R \Leftrightarrow x \geq y$

2. Cho $A = \{ a, b, c, d, e, x, y, z, t, v \}$ với thứ tự xác định bởi biểu đồ Hasse ở bên. Hãy tính

a) Sup {e, b}

b) Inf {e, t}

c) Sup {d, v}

3. Cho $A = \{ 1, 2, 3, 4, 5 \} \times \{ 1, 2, 3, 4, 5 \}$ và R là một quan hệ trên A sao cho

(a, b) R (c, d) $\Leftrightarrow a+b=c+d$

- a) Kiểm tra lại R là một quan hệ tương đương.

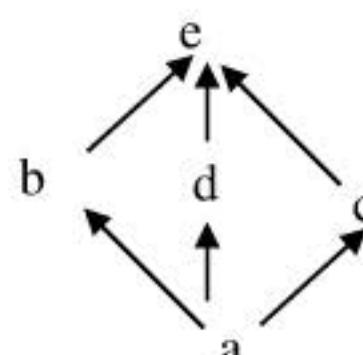
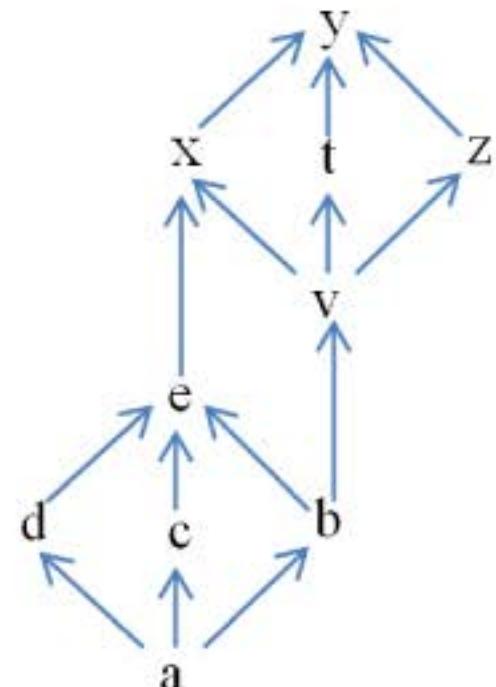
- b) Xác định các lớp tương đương $[(1,3)]$, $[(2,4)]$ và $[(1,1)]$.

c) Chỉ ra phân hoạch của A thành các lớp tương đương.

4. Cho $A = \{ a, b, c, d, e \}$ với thứ tự xác định bởi biểu đồ Hasse dưới đây. Hãy tính

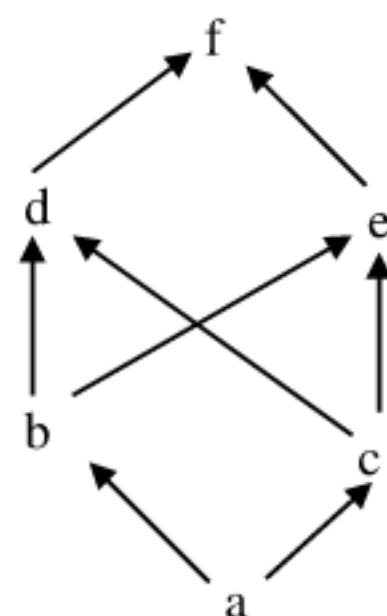
a) sup {b, c}, inf {b, c}

b) sup {a, d, e}, inf {a, d, e}



5. Cho $A = \{a, b, c, d, e, f\}$ với thứ tự xác định bởi biểu đồ Hasse bên. Hãy tính

- a) $\sup\{b, c, d\}, \inf\{b, c, d\}$
- b) $\sup\{b, c\}, \inf\{b, c\}$
- c) $\sup\{d, e\}, \inf\{d, e\}$



6. Cho $X = \{1, 2, 3, 4\}$. Hãy cho biết quan hệ R nào sau đây là phản xạ:

- a) $R = \{(1,1), (2,2), (3,3), (4,4), (1,4), (4,1)\}$
- b) $R = \{(1,1), (2,2), (4,4), (1,4), (4,1)\}$

7. Cho $X = \{1, a, 9, u\}$. Hãy cho biết quan hệ R nào sau đây là phản xạ:

- a) $R = \{(1,1), (a,a), (9,9), (1,u), (a,u)\}$
- b) $R = \{(1,1), (a,a), (9,9), (1,u), (a,u), (u,u)\}$

8. Cho $X = \{1, 2, 3, 4, 5, 6, 7, 8\}$. Hãy cho biết quan hệ R nào sau đây là phản xạ:

- a) $x R y \Leftrightarrow x + y$ chẵn.
- b) $(x, y) \in R \Leftrightarrow x + y$ lẻ.
- c) $(x, y) \in R \Leftrightarrow x < y$.
- d) $(x, y) \in R \Leftrightarrow x \leq y$.

9. Cho $X = \mathbb{Z}$. Hãy cho biết quan hệ R nào sau đây là phản xạ:

- a) $x R y \Leftrightarrow x + y$ chẵn.
- b) $(x, y) \in R \Leftrightarrow x + y$ lẻ.
- c) $(x, y) \in R \Leftrightarrow y$ chỉ hết cho x .

10. Cho $X = \{1, 2, 3, 4, 5, 6, 7, 8\} \times \{1, 2, 3, 4, 5, 6, 7, 8\}$. Hãy cho biết quan hệ R nào sau đây là phản xạ:

- a) $(a, b) R (c, d) \Leftrightarrow a + c$ chẵn.
- b) $((a, b), (c, d)) \in R \Leftrightarrow a + c$ lẻ.
- c) $(a, b) R (c, d) \Leftrightarrow a = c$.

Chương 3

ĐẠI SỐ BOOL – HÀM BOOL

3.1. ĐẠI SỐ BOOL

Định nghĩa 1: Cho $A \neq \emptyset$ và hai phép toán có 2 ngôi.

$$A \times A \rightarrow A$$

$$(a, b) \mapsto a \vee b$$

$$(a, b) \mapsto a \wedge b$$

Khi đó (A, \vee, \wedge) gọi là một đại số Bool nếu thỏa mãn các tính chất sau:

Nếu $\forall x, y, z \in A$ có

1. Tính giao hoán

$$x \vee y = y \vee x \quad x \wedge y = y \wedge x$$

2. Tính kết hợp

$$(x \vee y) \vee z = x \vee (y \vee z) \quad (x \wedge y) \wedge z = x \wedge (y \wedge z)$$

3. Tính phân bố

$$x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z) \quad x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$$

4. Phản tử trung hòa

$$\text{Có } 0, 1 \in A \quad x \vee 0 = x \quad x \wedge 1 = x$$

5. Phản tử bù

$$\text{Có } \bar{x} : x \vee \bar{x} = 1 \quad x \wedge \bar{x} = 0$$

Ví dụ 1:

Một dàn bù phân bố là một đại số Bool.

Ví dụ 2:

Cho $B = \{0, 1\}$

Trên B ta định nghĩa 2 luật \vee và \wedge

\vee		1
0	0	1
1	1	1

$$x \vee y = x + y - xy$$

\wedge	0	1
0	0	0
1	0	1

$$x \wedge y = xy$$

k	0	1
\bar{k}	1	0

(B, \vee, \wedge) là một đại số Bool.

Chú ý 1:

Cho A là một đại số Bool, với $x, y \in A$ ta có:

$$\overline{x \vee y} = \overline{x} \wedge \overline{y}$$

Luật DeMorgan

$$\overline{x \wedge y} = \overline{x} \vee \overline{y}$$

Kiểm chứng?

Gợi ý: Để chứng minh

$$\overline{x \vee y} = \overline{x} \wedge \overline{y}$$

Ta cần chứng minh $\begin{cases} (x \vee y) \vee (\overline{x} \wedge \overline{y}) = 1 \\ (x \vee y) \wedge (\overline{x} \wedge \overline{y}) = 0 \end{cases}$

Ta có

$$\begin{aligned} (x \vee y) \vee (\overline{x} \wedge \overline{y}) &= x \vee [y \vee (\overline{x} \wedge \overline{y})] \\ &= x \vee [(y \vee \overline{x}) \wedge (y \vee \overline{y})] = x \vee [(y \vee \overline{x}) \wedge 1] \\ &= x \vee (y \vee \overline{x}) = (x \vee \overline{x}) \vee y = 1 \vee y = 1 \end{aligned}$$

Tương tự $(x \vee y) \wedge (\overline{x} \wedge \overline{y}) = 0$

Định lý 1

Cho A là một đại số Bool, trên A ta định nghĩa quan hệ

$$x \prec y \Leftrightarrow x \wedge y = x$$

- i. Ta có \prec là một quan hệ thứ tự
- ii. (A, \prec) là một dàn bù phân bố

Hơn nữa, với $x, y \in A$ ta có:

$$sup(x, y) = x \vee y$$

$$inf(x, y) = x \wedge y$$

Chứng minh:

i. \prec là một quan hệ thứ tự

$$x \wedge x = x \Rightarrow x \prec x \Rightarrow \prec \text{ phản xạ}$$

$$\begin{cases} x \prec y \Rightarrow \\ y \prec x \end{cases} \Rightarrow \begin{cases} x \wedge y = x \\ y \wedge x = y \end{cases} \Rightarrow x = x \wedge y = y \wedge x = y \Rightarrow \prec \text{ phản xứng}$$

$$\begin{cases} x \prec y \Rightarrow \\ y \prec z \end{cases} \Rightarrow \begin{cases} x \wedge y = x \\ y \wedge z = y \end{cases} \Rightarrow x = x \wedge y = x \wedge (y \wedge z) = (x \wedge y) \wedge z \Rightarrow x \wedge z$$

$\Rightarrow \prec$ bắc cầu

iii. Với $x, y \in A$, ta chứng minh:

$$\sup(x, y) = x \vee y; \quad \inf(x, y) = x \wedge y$$

$$\sup(x, y) = x \vee y = c$$

Ta chứng minh $\begin{cases} x \prec c, y \prec c \\ x \prec d, y \prec d \end{cases} \Rightarrow c \prec d$

Ta có

$$\begin{aligned} x \wedge c &= x \wedge (x \vee y) \\ &= (x \vee 0) \wedge (x \vee y) \\ &= x \vee (0 \wedge y) \\ &= x \vee 0 \\ &= x \end{aligned}$$

$$\Rightarrow x \prec c$$

$$\begin{aligned}
x \wedge c &= x \wedge (x \vee y) \\
&= (x \vee 0) \wedge (x \vee y) = [x \wedge (x \vee y)] \vee [0 \wedge (x \vee y)] \\
&= (x \wedge x) \vee (x \wedge y) \vee 0 \\
&= x \vee [(x \wedge y) \vee 0] \\
&= x \vee [(x \vee 0) \wedge (y \vee 0)] \\
&= [x \vee (x \vee 0)] \wedge [x \vee (y \vee 0)] \\
&= x \wedge (x \vee y) \\
&= (x \wedge x) \vee (x \wedge y) \\
&= x \vee (x \wedge y) \\
&= x \vee (0 \wedge y) \\
&= x \vee 0 \\
&= x
\end{aligned}$$

$\Rightarrow x \prec c$

Tương tự $y \prec c$

Cho $\begin{cases} x \prec d \\ y \prec d \end{cases} \Rightarrow \begin{cases} x \wedge d = x \\ y \wedge d = y \end{cases}$

$$c \wedge d = (x \vee y) \wedge d = (x \wedge d) \vee (y \wedge d) = x \vee y = c \Rightarrow c \prec d$$

Suy ra $c = \text{sup}(x, y)$

Tương tự $\text{inf}(x, y) = x \wedge y$.

Định nghĩa 2

Cho A là đại số Bool, các trội trực tiếp của phần tử bé nhất gọi là 1 nguyên tử của A.

Ví dụ 3:

- i. Các nguyên tử trong $\mathcal{P}(E)$ với $E = \{a, b, c\}$ là $\{a\}, \{b\}, \{c\}$
- ii. Cho $E = \{a_1, a_2, \dots, a_n\}$ thì các nguyên tử của $\mathcal{P}(E)$ là $\{a_1\}, \{a_2\}, \dots, \{a_n\}$

Ví dụ 4:

U_{30} có các nguyên tử là 2, 3 và 5.

U_{210} có các nguyên tử là 2, 3, 5 và 7.

3.2. HÀM BOOL

Nhắc lại $B = \{0,1\}$ với các phép toán

\vee	0	1
0	0	1
	1	1

\wedge	0	1
0	0	0
	1	0

x	0	1
\bar{x}	1	0

$$B^n = \underbrace{B \times B \times \dots \times B}_{n \text{ lần}} = \{(x_1, \dots, x_n) / x_i \in B\}$$

$$|B^n| = 2^n$$

Định nghĩa 4

- i. Một hàm Bool hay hàm logic, hàm nhị phân theo n biến là một ánh xạ

$$f: B^n \rightarrow B$$

Đặt $\mathcal{F}_n = B^{B^n}$, ta có $|\mathcal{F}_n| = 2^{(2^n)}$

- ii. Các hàm 0, 1 được định nghĩa bởi

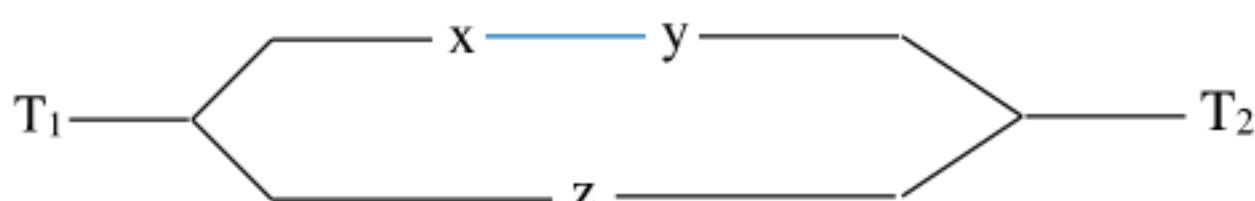
$$0: B^n \rightarrow B$$

$$\forall x, 0(x) = 0$$

$$1: B^n \rightarrow B$$

$$\forall x, 1(x) = 1$$

Ví dụ 5:



$$f(x, y, z) = \begin{cases} 1 & \text{Nếu có dòng điện từ } T_1 \text{ đến } T_2 \\ 0 & \text{Nếu ngược lại} \end{cases}$$

Vậy $f \in \mathcal{F}_3$

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

Định nghĩa 5

Với $f, g \in \mathcal{F}_n$, trên \mathcal{F}_n ta định nghĩa quan hệ \prec như sau:

$$f \prec g \Leftrightarrow f(a) \leq g(a) \quad \forall a = (a_1, a_2, \dots, a_n) \in B^n$$

(\mathcal{F}_n, \prec) là một đại số Bool và ta có sup và inf của 2 hàm f và g được cho bởi:

$$(f \vee g)(a) = f(a) \vee g(a) = f(a) + g(a) - f(a)g(a), \quad \forall a \in B^n$$

$$(f \wedge g)(a) = f(a) \wedge g(a) = f(a).g(a), \quad \forall a \in B^n$$

$$\overline{f}(a) = \overline{f(a)} = 1 - f(a), \quad \forall a \in B^n$$

Định lý 2

i. Các nguyên tử trong \mathcal{F}_n là các hàm Bool chỉ bằng 1 tại một điểm duy nhất gọi là các từ tối thiểu của \mathcal{F}_n .

ii. $\forall f \in \mathcal{F}_n$, f có thể viết

$$f = m_1 \vee m_2 \vee \dots \vee m_i \vee \dots \vee m_r$$

Với m_i là các từ tối thiểu trội bởi f .

Ví dụ 6: Với $f(x, y, z)$ biểu diễn mạch điện ở ví dụ trên ta có

x	y	z	$f(x, y, z)$	m_1	m_2	m_3	m_4	m_5
0	0	0	0	0	0	0	0	0
0	0	1	1	1	0	0	0	0
0	1	0	0	0	0	0	0	0
0	1	1	1	0	1	0	0	0
1	0	0	0	0	0	0	0	0
1	0	1	1	0	0	1	0	0
1	1	0	1	0	0	0	1	0
1	1	1	1	0	0	0	0	1

Do đó $f = m_1 \vee m_2 \vee m_3 \vee m_4 \vee m_5$

Định nghĩa 6: x_i, \bar{x}_i gọi là các từ đơn

Định lý 3: Mọi từ tối thiểu trong \mathcal{F}_n đều có thể viết dưới dạng $m = b_1b_2\dots b_n$, trong đó b_i là các từ đơn.

Định nghĩa 7

Cho $f \in \mathcal{F}_n$, nếu $f = m_1 \vee m_2 \vee \dots \vee m_r$ với m_i là các từ tối thiểu thì dạng này gọi là dạng nối rời chính tắc (d.n.f) của f

Ví dụ 8:

$$f \in \mathcal{F}_4, \quad f = x_1x_2\bar{x}_3 \vee x_1\bar{x}_2$$

Viết dạng nối rời chính tắc của f

Giải:

$$\begin{aligned} f &= x_1x_2\bar{x}_3(x_4 \vee \bar{x}_4) \vee x_1\bar{x}_2(x_3 \vee \bar{x}_3)(x_4 \vee \bar{x}_4) \\ &= x_1x_2\bar{x}_3x_4 \vee x_1x_2\bar{x}_3\bar{x}_4 \vee (x_1\bar{x}_2x_3 \vee x_1\bar{x}_2\bar{x}_3)(x_4 \vee \bar{x}_4) \\ &= x_1x_2\bar{x}_3x_4 \vee x_1x_2\bar{x}_3\bar{x}_4 \vee x_1\bar{x}_2x_3x_4 \vee x_1\bar{x}_2x_3\bar{x}_4 \vee x_1\bar{x}_2\bar{x}_3x_4 \vee x_1\bar{x}_2\bar{x}_3\bar{x}_4 \end{aligned}$$

Ví dụ 9:

$$f \in \mathcal{F}_3, \quad f = xyz \vee x\bar{z} \vee \bar{y}z$$

Viết dạng nối rời chính tắc của f .

Giải:

$$\begin{aligned}
 f &= xyz \vee x\bar{z} \vee \bar{y}z \\
 &= xyz \vee x(y \vee \bar{y})\bar{z} \vee (x \vee \bar{x})\bar{y}z \\
 &= xyz \vee xy\bar{z} \vee x\bar{y}\bar{z} \vee x\bar{y}z \vee \bar{x}\bar{y}z
 \end{aligned}$$

Định nghĩa 8

Một đơn thức là một tích khác 0 của các tử đơn. Một công thức đa thức của f là một công thức biểu diễn f dưới dạng $f = f_1 \vee f_2 \vee \dots \vee f_r$ với f_i là đơn thức.

Ví dụ 10:

Trong \mathcal{F}_4 , $f = \underbrace{xyt}_{f_1} \vee \underbrace{\bar{x}y\bar{z}t}_{f_2} \vee \underbrace{yt}_{f_3}$

f_i : đơn thức

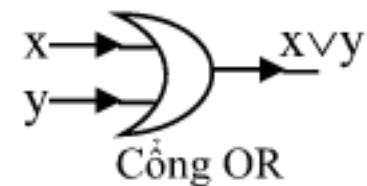
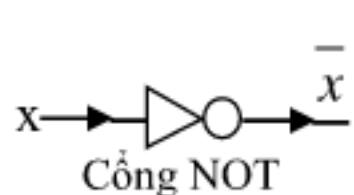
Đây là một công thức đa thức của f .

3.3. MẠNG CÁC CÔNG

Định nghĩa 9

a. Cỗng logic

Là một công cụ trong máy tính thực hiện các nhiệm vụ đặc thù trong xử lý số liệu tương ứng với các phép toán trong đại số Bool. Ta có các phép toán sau:



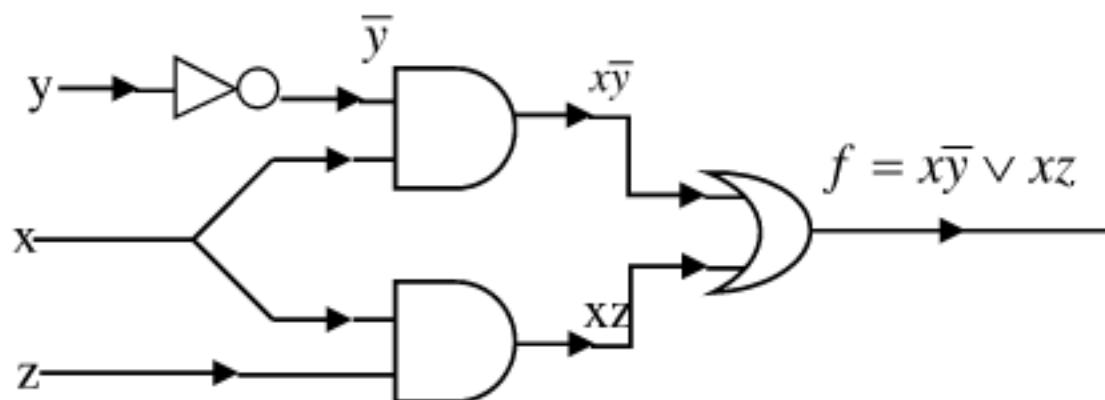
b. Mạng logic

Gồm nhiều cỗng logic liên kết với nhau theo những quy tắc sau đây:

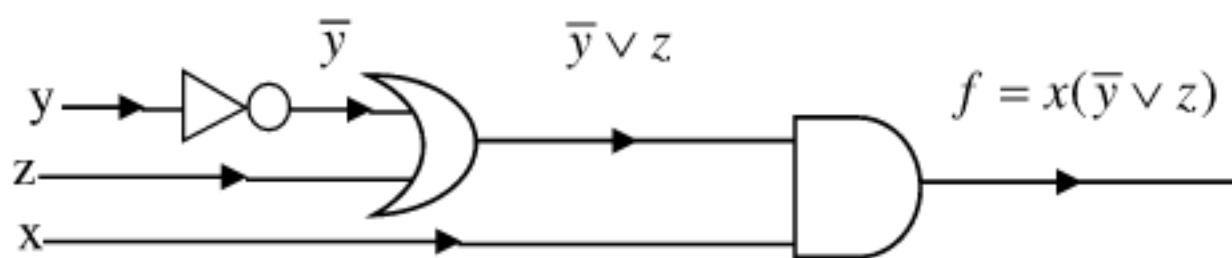
- Một đường vào có thể tách ra cho nhiều cỗng.
- Không có đường ngược, nghĩa là đường ra của một cỗng không thể là đường vào của chính nó.

Ví dụ 13:

a. Vẽ mạng của cỗng $f = x\bar{y} \vee xz$



b. Vẽ mạng của cỗng $f = x(\bar{y} \vee z)$



Nhận xét: Mạng trong VD13a sử dụng 4 cỗng, trong VD13b sử dụng 3 cỗng. Vậy số cỗng tối thiểu để biểu diễn f là bao nhiêu? Ta tìm công thức tối thiểu của f .

3.4. PHƯƠNG PHÁP BIỂU ĐỒ KARNAUGH

Định nghĩa 10

Cho $f \in \mathcal{F}_n$, xét 2 công thức đa thức của f

$$f = m_1 \vee m_2 \vee \dots \vee m_p \quad (1)$$

Với $\alpha_1 \leq \alpha_2 \leq \dots \leq \alpha_p$ là độ dài (số từ đơn) của đơn thức m_i

$$f = M_1 \vee M_2 \vee \dots \vee M_q \quad (2)$$

Với $\beta_1 \leq \beta_2 \leq \dots \leq \beta_q$ là độ dài (số từ đơn) của đơn thức M_i

i. Ta nói công thức (1) đơn giản hơn công thức (2) nếu

$$\begin{cases} p \leq q \\ \alpha_i \leq \beta_i, \quad i = 1, 2, \dots, p \end{cases}$$

ii. Ta nói công thức (1) và công thức (2) đơn giản như nhau nếu

$$\begin{cases} p = q \\ \alpha_i = \beta_i, \quad i = 1, 2, \dots, p \end{cases}$$

iii. Công thức (1) gọi là công thức tối thiểu nếu không có công thức nào đơn giản hơn công thức (1). Có thể có một số công thức tối thiểu nếu chúng đơn giản như nhau.

Ví dụ 14: Cho $f \in \mathcal{F}_4$, $f = \underbrace{xy}_{m_1} \vee \underbrace{x\bar{z}}_{m_2} \vee \underbrace{\bar{x}zt}_{m_3} \vee \underbrace{\bar{x}\bar{y}t}_{m_4}$ (1)

$$\alpha_1 = 2 \leq \alpha_2 = 2 \leq \alpha_3 = 3 \leq \alpha_4 = 3$$

$$f = \underbrace{xy}_{M_1} \vee \underbrace{x\bar{z}}_{M_2} \vee \underbrace{yzt}_{M_3} \vee \underbrace{\bar{y}\bar{z}t}_{M_4} \vee \underbrace{\bar{x}zt}_{M_5} \quad (2)$$

$$\beta_1 = \beta_2 = 2 \leq \beta_3 = \beta_4 = \beta_5 = 3$$

Công thức (1) đơn giản hơn công thức (2) vì $\begin{cases} p = 4 < q = 5 \\ \alpha_i = \beta_i, \quad i = 1, 2, 3, 4 \end{cases}$

$$f = xy \vee x\bar{z} \vee \bar{x}\bar{y}t \vee yzt \quad (3)$$

$$f = xy \vee x\bar{z} \vee \bar{x}zt \vee \bar{x}\bar{y}t \quad (1)$$

Vậy công thức (1) và công thức (3) đơn giản như nhau.

Nhận xét

Giả sử f có công thức đa thức tối thiểu là $f = m_1 \vee m_2 \vee \dots \vee m_p$ (1)

Giả sử có đơn thức m sao cho $m_1 \preceq m \preceq f$. Khi đó ta có

$$f = m \vee f = (m \vee m_1) \vee m_2 \vee \dots \vee m_p = m \vee m_2 \vee \dots \vee m_p \quad (\text{vì } m_1 \prec m) \quad (2)$$

Cũng do $m_1 \prec m$, ta có thể viết $m_1 m = m_1$ nên mọi thừa số là từ đơn của m cũng là một thừa số của m_1 . Hơn nữa do $m_1 \neq m$ nên có ít nhất một thừa số là từ đơn của m_1 không xuất hiện trong m . Khi đó công thức (2) rõ ràng “đơn giản hơn” (1) nhưng không “đơn giản như nhau”. Điều này trái giả thiết ban đầu (1) là công thức tối thiểu. Vậy m_i là đơn thức tối đại trội bởi f .

Định lý 6

Trong một công thức đa thức tối thiểu $f = m_1 \vee m_2 \vee \dots \vee m_p$ thì mỗi m_i là một đơn thức tối đại trội bởi f , gọi là các tiền đề nguyên tố của f .

Phương pháp biểu đồ Karnaugh

Ta xét $f \in \mathcal{F}_n$ với $n=3, 4$. Trước tiên ta xét $\mathcal{F}_4 = \{f / f: B^4 \rightarrow B\}$.

B^4 có 16 phần tử có dạng (x,y,z,t) được (thể hiện) sắp xếp trong bảng gọi là hình vuông lớn sau đây.

	x		\bar{x}	
z	1010	1110	0110	0010
	1011	1111	0111	0011
\bar{z}	1001	1101	0101	0001
	1000	1100	0100	0000
	y	y	y	\bar{y}
				t
			t	
				\bar{t}

Trong đó:

x để chỉ cột mà ở đó thành phần thứ nhất lấy giá trị 1.

\bar{x} để chỉ cột mà ở đó thành phần thứ nhất lấy giá trị 0.

Tương tự cho y .

z để chỉ dòng mà ở đó thành phần thứ ba lấy giá trị 1.

\bar{z} để chỉ dòng mà ở đó thành phần thứ ba lấy giá trị 0.

Tương tự cho t .

Định nghĩa 11

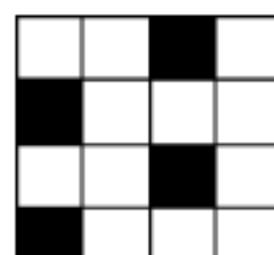
Hai ô trên hình vuông lớn gọi là kề nhau nếu chúng chỉ khác nhau 1 thành phần. Theo định nghĩa này thì ô (1,1) và ô (1,4) là kề nhau, ô (1,2) kề với ô (4,2)...Nói cách khác, hai ô gọi là kề nhau nếu chúng kề nhau trên hình trụ khi ta cuốn hình vuông lớn theo chiều dọc hay chiều ngang.

Định nghĩa 12

Cho $f \in \mathcal{F}_4$, ta gạch chéo những ô của hình vuông lớn mà ở đó $f = 1$. Hình vẽ có được gọi là biểu đồ Karnaugh của f . Ký hiệu K_f .

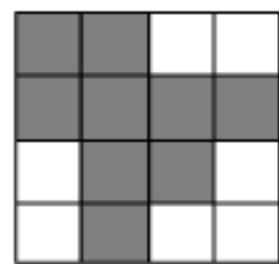
Ví dụ 16: (Bài 19a)

a. $f^1(1) = \{0101, 0110, 1000, 1011\}$



K_f

b. $f^{-1}(0) = \{0000, 0001, 0010, 0100, 1000, 1001, 0110\}$



K_f

Định lý 7

Cho $f, g \in \mathcal{F}_4$, ta có

- i. $f \leq g \Leftrightarrow K_f \subset K_g$
- ii. $K_{f \vee g} = K_f \cup K_g; \quad K_{f \cdot g} = K_f \cap K_g$
- iii. $K_{\overline{f}} = \overline{K}_f$

Định lý 8: (Biểu đồ Karnaugh của một đơn thức)

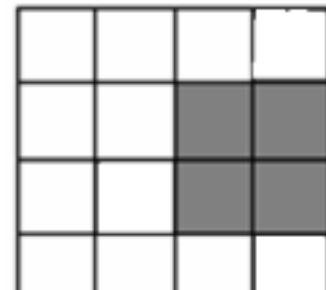
Cho $m = y_1 \dots y_p$ ($1 \leq p \leq 4$) là đơn thức trong \mathcal{F}_4 . Biểu đồ Karnaugh của m gồm 2^{4-p} ô kề nhau. Tức là một hình chữ nhật theo nghĩa rộng mà ta gọi là tế bào.

Ví dụ 17:

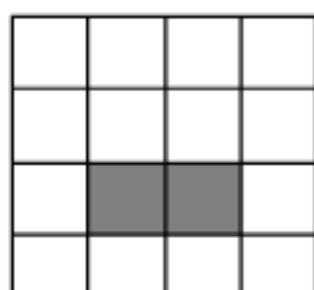
$m = z$ (8 ô)



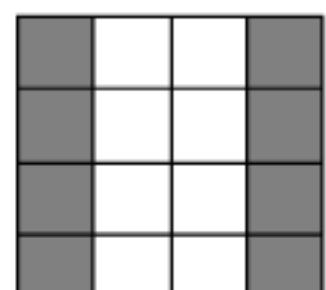
$m = \bar{x}t$ (4 ô)



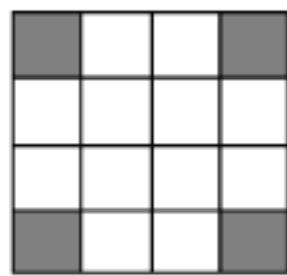
$m = y\bar{z}t$ (2 ô)



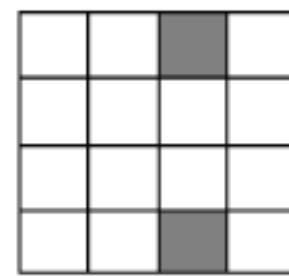
$m = \bar{y}$ (8 ô)



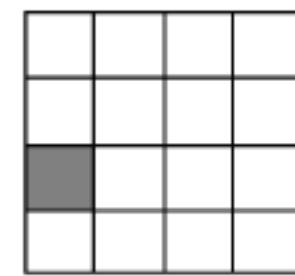
$$m = \bar{y}\bar{t} \quad (4 \text{ ô})$$



$$m = \bar{x}y\bar{t} \quad (2 \text{ ô})$$



$$m = x\bar{y}\bar{z}t \quad (1 \text{ ô})$$



Nhận xét

Cho $f \in \mathcal{F}_4$, f có công thức đa thức tối thiểu là

$$f = m_1 \vee m_2 \vee \dots \vee m_p$$

Theo định lý 6, m_i là đơn thức tối đại trội bởi f , gọi là tiền đề nguyên tố. Ta có

$$K_f = K_{m_1} \cup K_{m_2} \cup \dots \cup K_{m_p} \quad (\text{Định lý 7})$$

Vậy K_{m_i} tối đại $\subset K_f$

$K_{m_1}, K_{m_2}, \dots, K_{m_p}$ gọi là một phủ của K_f

Nếu $\bigcup_{j \in J} K_{m_j}, J \neq \{1, 2, \dots, p\}$ thì ta nói

$K_{m_1} \cup K_{m_2} \cup \dots \cup K_{m_p}$ là phủ tối thiểu.

Định nghĩa 13

Cho công thức đa thức tối thiểu của $f \in \mathcal{F}_4$

$$f = m_1 \vee m_2 \vee \dots \vee m_p$$

i. Ta có K_{m_i} tối đại trong K_f , gọi là các tế bào lớn của K_f

ii. Nếu $K_f = K_{m_1} \cup K_{m_2} \cup \dots \cup K_{m_p}$ và $K_f \neq \bigcup_{j \in J} K_{m_j}, J \neq \{1, 2, \dots, p\}$ thì $K_{m_1} \cup K_{m_2} \cup \dots \cup K_{m_p}$ gọi là phủ tối thiểu của K_f

Thuật toán tìm công thức đa thức tối thiểu của hàm Bool theo 4 biến ($f \in \mathcal{F}_4$)

Bước 1: Tìm tất cả các tế bào lớn của f .

Bước 2: Nếu có một ô chỉ ở trong một tế bào lớn duy nhất, ta chọn tế bào lớn này để phủ. Lặp lại quá trình trên cho tới khi không còn ô nào nằm trong một tế bào lớn duy nhất.

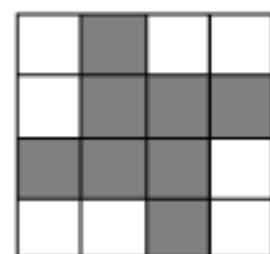
Bước 3: Nếu các tế bào lớn trong bước 2 đã phủ kín K_f thì đến bước 4, ngược lại chọn 1 ô còn lại và 1 tế bào lớn chứa ô này để phủ. Tiếp tục cho đến khi phủ kín K_f .

Bước 4: Loại bỏ các phép phủ không tối thiểu và suy ra công thức đa thức tối thiểu của f .

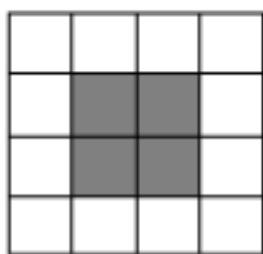
Ví dụ 18:

Tìm công thức đa thức tối thiểu của f .

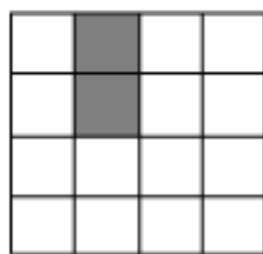
Bước 1:



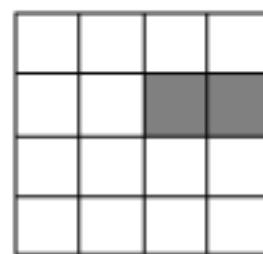
K_f



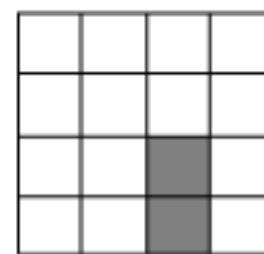
yz



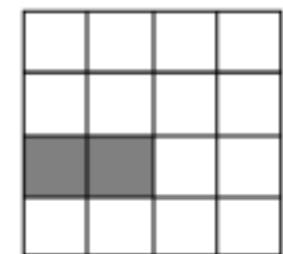
xyz



$\bar{x}zt$



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



$x\bar{z}t$

Bước 2:

Ô (1,2) ở trong tế bào lớn duy nhất xyz

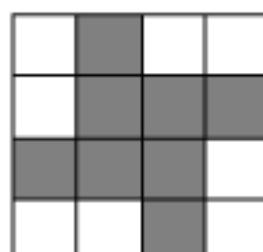
Ô (2,4) ở trong tế bào lớn duy nhất $\bar{x}zt$

Ô (3,1) ở trong tế bào lớn duy nhất $x\bar{z}t$

Ô (4,3) ở trong tế bào lớn duy nhất $\bar{x}y\bar{z}$

Bước 3:

Dùng các tế bào lớn này để phủ K_f



K_f đã được phủ kín

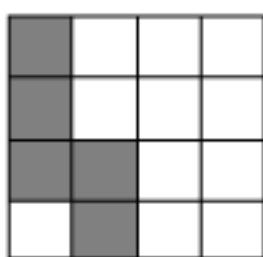
Bước 4:

Công thức đa thức tối thiểu của f là

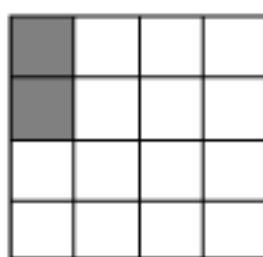
$$f = xyz \vee \bar{x}zt \vee x\bar{z}t \vee \bar{x}y\bar{z}$$

Ví dụ 19: Tìm công thức đa thức tối thiểu của f

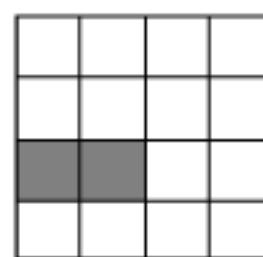
Bước 1:



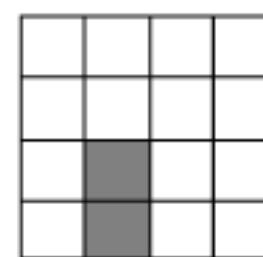
K_f



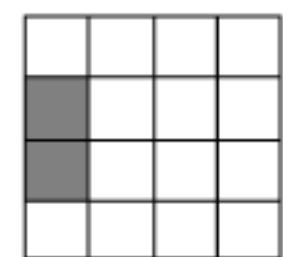
$x\bar{y}z$



$x\bar{z}t$



$xy\bar{z}$



$x\bar{y}t$

Bước 2:

Ô (1,1) ở trong tế bào lớn duy nhất $x\bar{y}z$

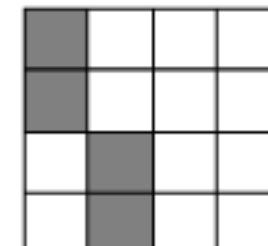
Ô (4,2) ở trong tế bào lớn duy nhất $xy\bar{z}$

Bước 3:

Dùng 2 tế bào lớn này để phủ K_f .

K_f chưa được phủ kín.

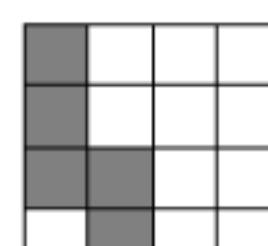
Còn ô (3,1) chưa được phủ.



a. Chọn $x\bar{y}t$ để phủ ô (3,1)

K_f được phủ kín

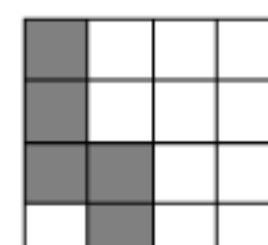
$$f = x\bar{y}z \vee xy\bar{z} \vee x\bar{y}t \quad (1)$$



b. Chọn $x\bar{z}t$ để phủ ô (3,1)

K_f được phủ kín

$$f = x\bar{y}z \vee xy\bar{z} \vee x\bar{z}t \quad (2)$$



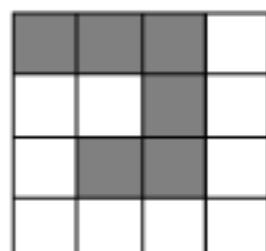
Bước 4:

Công thức (1) và công thức (2) đơn giản như nhau nên cả hai là công thức đa thức tối thiểu của f.

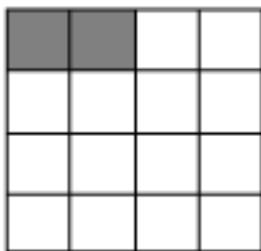
$$\begin{aligned} f &= x\bar{y}z \vee xy\bar{z} \vee x\bar{y}t \\ &= x\bar{y}z \vee xy\bar{z} \vee x\bar{z}t \end{aligned}$$

Ví dụ 20: Tìm công thức đa thức tối thiểu của f.

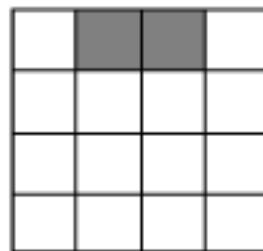
Bước 1:



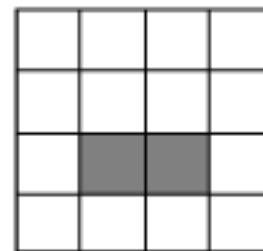
K_f



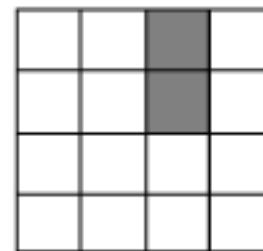
$xz\bar{t}$



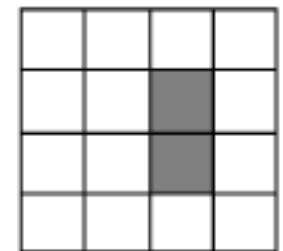
$yz\bar{t}$



$y\bar{z}\bar{t}$



$\bar{x}yz$



$\bar{x}yt$

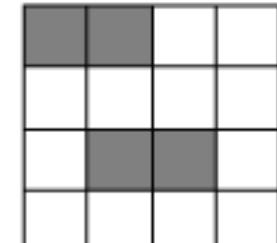
Bước 2:

Ô (1,1) ở trong tế bào lớn duy nhất $xz\bar{t}$

Ô (3,2) ở trong tế bào lớn duy nhất $y\bar{z}\bar{t}$

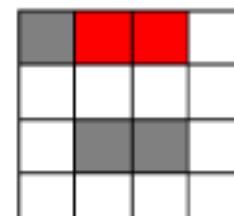
Dùng 2 tế bào lớn này để phủ K_f .

Còn 2 ô là (1,3) và (2,3) chưa được phủ

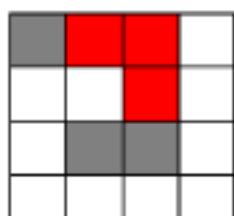


Bước 3:

a. Chọn $yz\bar{t}$ để phủ ô (1,3)

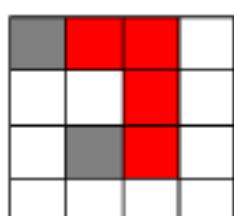


a₁. Chọn $\bar{x}yz$ để phủ ô (2,3)



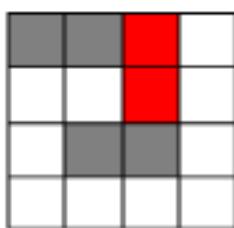
$$f = xz\bar{t} \vee y\bar{z}\bar{t} \vee yz\bar{t} \vee \bar{x}yz \quad (1)$$

a₂. Chọn $\bar{x}yt$ để phủ ô (2,3)



$$f = xz\bar{t} \vee y\bar{z}\bar{t} \vee yz\bar{t} \vee \bar{x}yt \quad (2)$$

b. Chọn $\bar{x}yz$ để phủ ô (1,3)



K_f được phủ kín

$$f = xz\bar{t} \vee y\bar{z}\bar{t} \vee \bar{x}yz \quad (3)$$

Bước 4:

Công thức (3) đơn giản hơn công thức (1) và (2) nên (3) là công thức đa thức tối thiểu của f

$$f = xz\bar{t} \vee y\bar{z}\bar{t} \vee \bar{x}yz$$

Trường hợp 3 biến

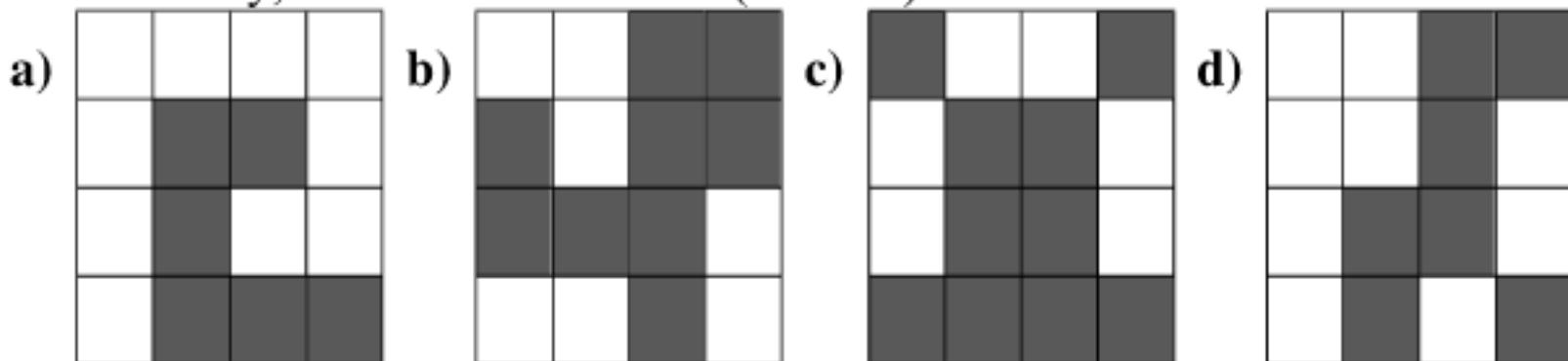
B^3 có 8 phần tử có dạng (x,y,z) được (thể hiện) sắp xếp trong bảng sau:

		x				\bar{x}	
		101	111	011	001		
z	100	110	010	000			
	\bar{z}	\bar{y}	y	\bar{y}			

Tất cả các bước còn lại tương tự như trường hợp 4 biến.

BÀI TẬP CHƯƠNG 3

1. Tìm công thức đa thức tối thiểu của hàm Bool có biểu đồ Karnaugh dưới đây, chỉ ra các tế bào lớn (vẽ hình):



2. Tìm công thức đa thức tối thiểu của hàm Bool 4 biến x, y, z, t :

a) $\bar{z}(\bar{x}\bar{y} \vee yt) \vee y(x\bar{z} \vee \bar{x}z)$

b) $xyzt \vee \bar{x}\bar{y} \vee x\bar{z}t \vee y\bar{z}t$

3. Tìm công thức đa thức tối thiểu của các hàm Bool f dưới đây.

a) f là hàm Bool 3 biến và lấy giá trị 1 khi và chỉ khi có đúng 2 biến lấy giá trị 1

b) f là hàm Bool 3 biến và lấy giá trị 1 khi và chỉ khi có ít nhất 2 biến lấy giá trị 1

c) f là hàm Bool 4 biến và lấy giá trị 1 khi và chỉ khi số biến lấy giá trị 1 là số lẻ.

4. Tìm công thức đa thức tối thiểu của $f \in F_3$ có:

$$f = (x \vee \bar{y} \vee z)(\bar{x} \vee \bar{y} \vee z)(x \vee y \vee \bar{z})$$

5. Tìm công thức đa thức tối thiểu của hàm Bool 4 biến có

$$f^{-1}(0) = \{(1,1); (1,2); (2,2); (3,4); (4,1); (4,2); (4,4)\}$$

Yêu cầu chỉ ra và vẽ hình các tế bào lớn.

6. Tìm công thức đa thức tối thiểu của hàm $f \in F_4$ có:

$$f^{-1}(0) = \{0000, 0001, 0010, 0011, 0100, 1000, 1001, 1011, 1100, 1111\}$$

Yêu cầu chỉ ra và vẽ hình các tế bào lớn.

7. Dùng phương pháp biểu đồ Karnaugh để tìm công thức đa thức tối thiểu của hàm sau (yêu cầu chỉ ra và vẽ hình các tế bào lớn)

$$f = (\bar{x} \vee y \vee \bar{t})(\bar{y} \vee \bar{z} \vee t)(x \vee y \vee \bar{t})$$

8. Tìm công thức đa thức tối thiểu của hàm Bool 4 biến có

$$f = (\bar{z} \vee zt \vee \bar{y}z\bar{t})(\bar{x} \vee xy \vee x\bar{y}\bar{t})(z \vee \bar{z}t \vee \bar{y}\bar{z}\bar{t})$$

Yêu cầu chỉ ra và vẽ hình các tế bào lớn.

9. Dùng phương pháp biểu đồ Karnaugh để tìm công thức đa thức tối thiểu của hàm f có:

$$f^{-1}(1) = \{(1,1), (1,4), (2,1), (2,2), (2,3), (3,3), (3,4)\}$$

Yêu cầu chỉ ra và vẽ hình các tế bào lớn.

Chương 4

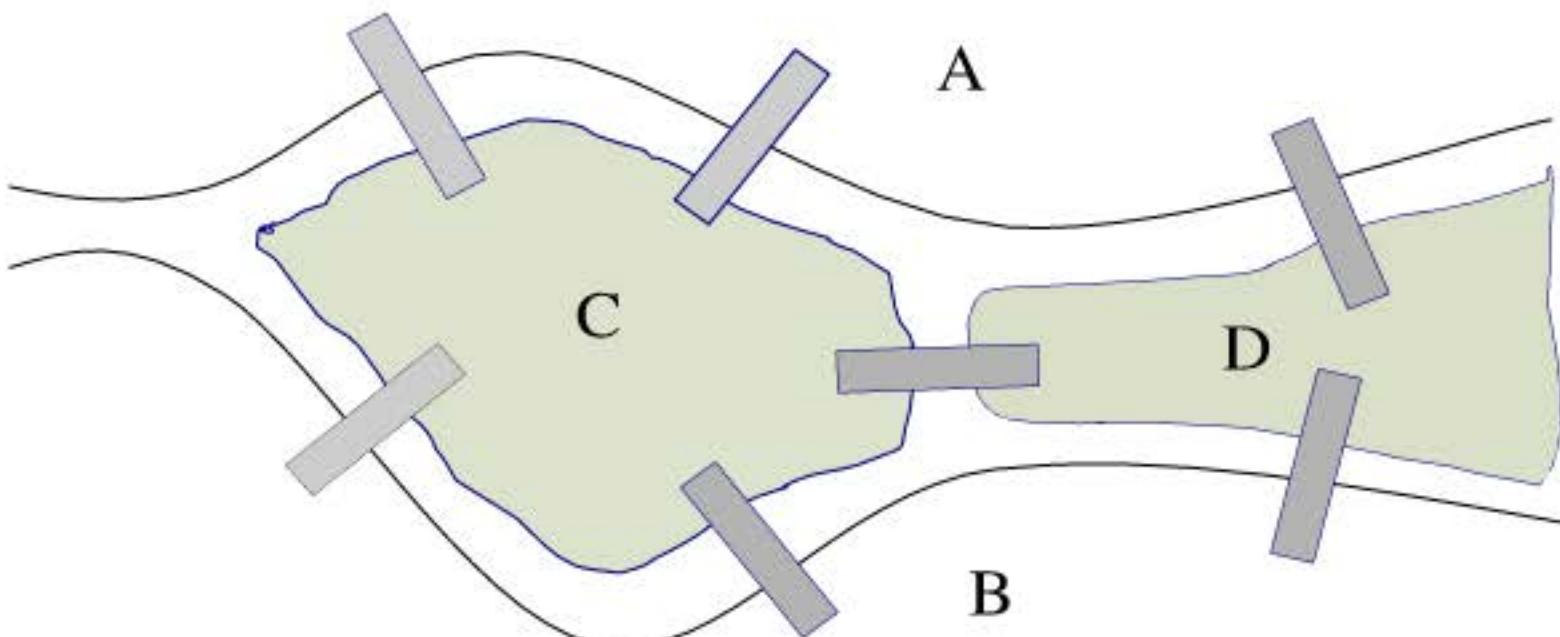
MỞ ĐẦU VỀ LÝ THUYẾT ĐỒ THỊ

Chương này cung cấp cho người đọc những kiến thức nền tảng về lý thuyết đồ thị, bao gồm các khái niệm, tính chất cơ bản, phân loại đồ thị và các dạng đồ thị đặc biệt.

4.1. MỞ ĐẦU

Lý thuyết đồ thị thuộc một nhánh nghiên cứu của toán học. Nhà toán học được xem là người đặt nền tảng đầu tiên cho lĩnh vực này là Leonhard Euler (người Thụy Sĩ). Ông chính là người đưa ra lời giải cho bài toán bảy cây cầu ở thành phố Königsberg (hay còn gọi là bài toán bảy cây cầu Euler) vào năm 1736.

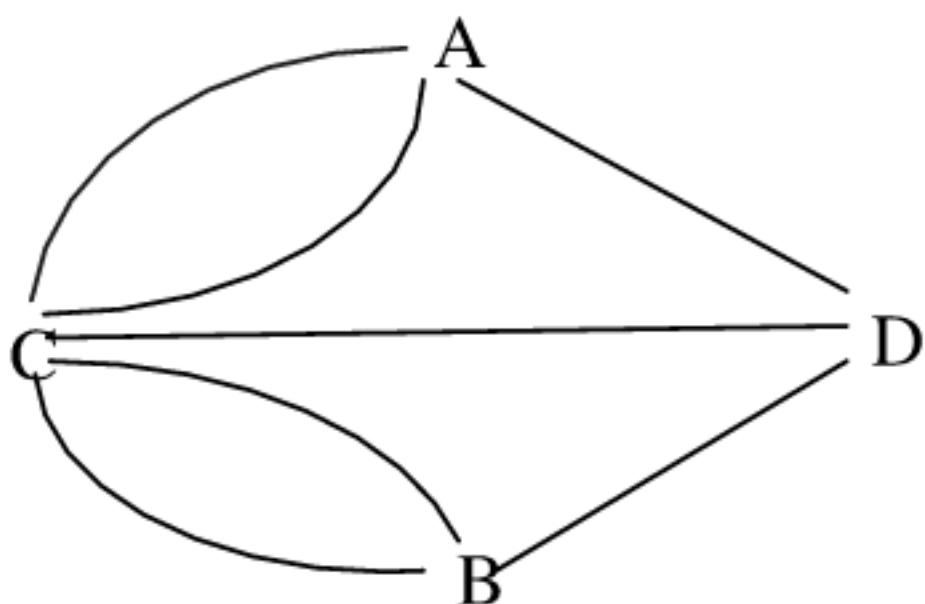
Bài toán được đặt ra dựa trên hoàn cảnh địa lý ở thành phố Königsberg (nay là Kaliningrad, Nga) vốn có con sông Pregel chảy qua. Con sông tách thành hai nhánh, hình thành hai hòn đảo lớn nối với nhau và nối với đất liền bởi bảy cây cầu. Người ta đặt ra câu hỏi là có thể đi theo một tuyến đường, đi qua 7 cây cầu, mỗi cây cầu chỉ được đi qua đúng một lần, rồi trở về điểm xuất phát được hay không.



Hình 4.1: Bảy chiếc cầu ở thành phố Königsberg

Để giải quyết bài toán này, Euler đã chuyển nó về dạng đồ thị. Xem mỗi vùng là một đỉnh, mỗi chiếc cầu là một cạnh nối giữa hai đỉnh. Từ đó bài toán có thể được phát biểu lại như sau:

Có thể xuất phát từ một điểm, đi qua tất cả các cạnh của đồ thị sao cho mỗi cạnh chỉ đi qua đúng một lần, rồi trở về đỉnh xuất phát được không?



Hình 4.2: Đồ thị của bài toán 7 chiếc cầu

Như vậy, từ một bài toán trong thực tế, Euler đã chuyển nó thành bài toán trên đồ thị và chứng minh rằng không tồn tại một tuyến đường như vậy dựa trên việc vận dụng các tính chất của đồ thị (người đọc sẽ tìm ra lời giải cụ thể cho bài toán này sau khi đọc xong chương 4). Đó cũng là ý tưởng chung để ứng dụng lý thuyết đồ thị giải quyết các bài toán trong các lĩnh vực khác nhau.

4.2. ĐỊNH NGHĨA VÀ PHÂN LOẠI ĐỒ THỊ

4.2.1. Định nghĩa đồ thị

Đồ thị G là một tập hợp bao gồm tập các đỉnh và tập các cạnh. Ta thường ký hiệu: $G = (V, E)$, trong đó:

V : tập các đỉnh;

E : tập các cạnh.

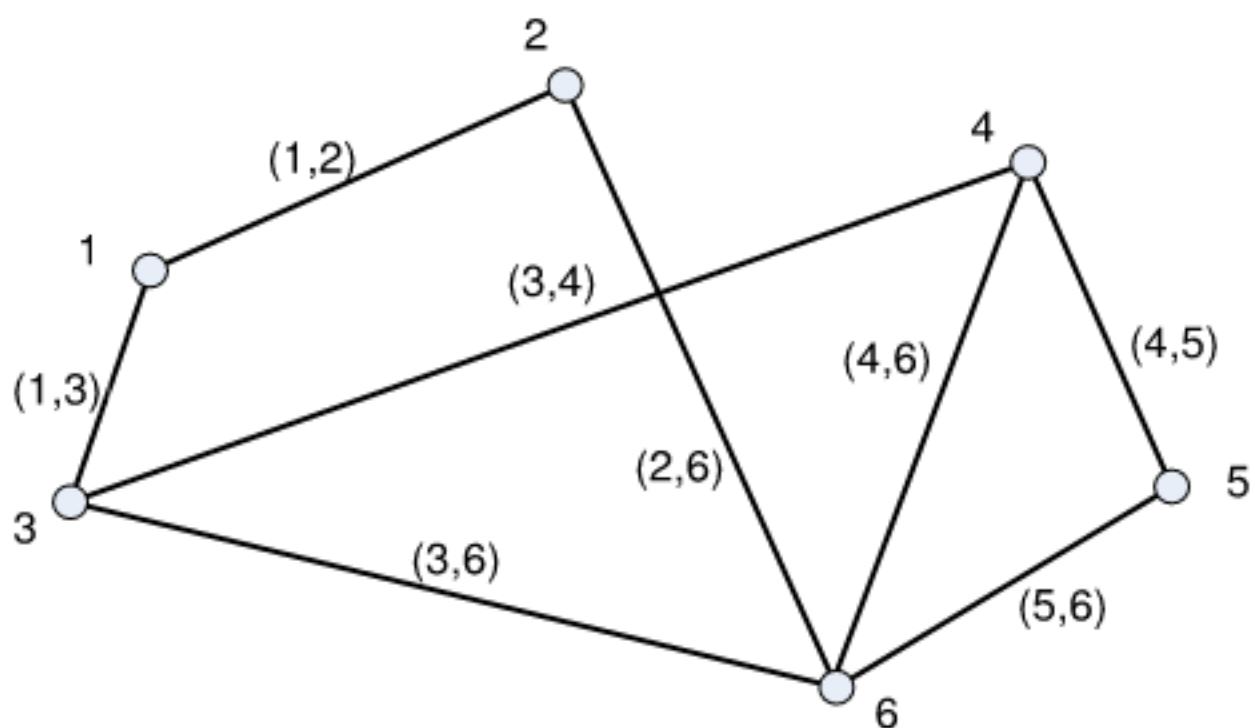
Các đồ thị thường được phân loại thành đồ thị vô hướng và có hướng; đơn đồ thị và đa đồ thị.

4.2.2. Phân loại đồ thị

a. Đồ thị vô hướng

- **Đơn đồ thị vô hướng**

Đồ thị $G = (V, E)$ được gọi là đơn đồ thị vô hướng nếu tập các cạnh E là **tập** các cặp **không có thứ tự** gồm hai phần tử khác nhau của V . Trong đơn đồ thị vô hướng sẽ không có cạnh lặp và các cạnh sẽ không quy định chiều.



Hình 4.3: Đơn đồ thị vô hướng

Khi đó:

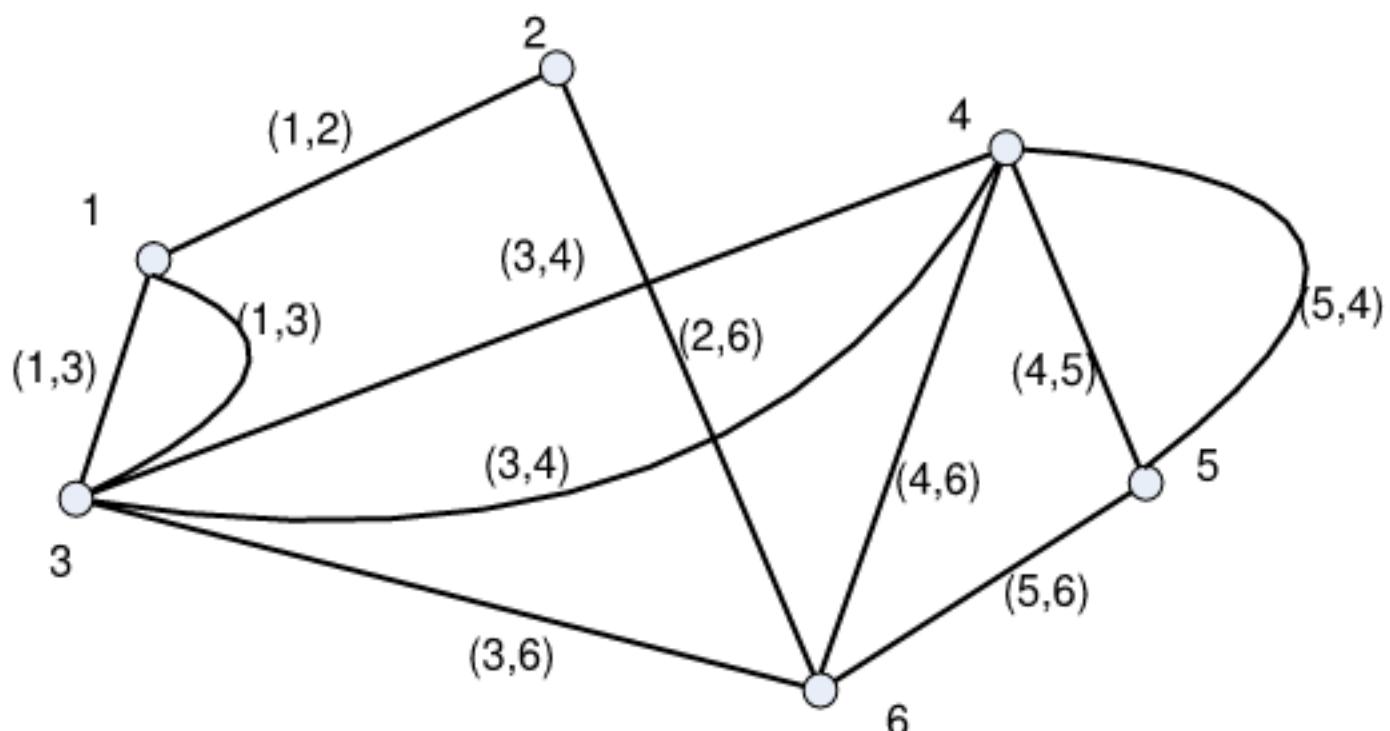
$$V = \{1, 2, 3, 4, 5, 6\}$$

$$E = \{(1, 2), (1, 3), (2, 6), (3, 4), (3, 6), (4, 5), (4, 6), (5, 6)\}$$

• Đa đồ thị vô hướng

Đồ thị $G = (V, E)$ được gọi là đa đồ thị vô hướng nếu tập các cạnh E là **hợ** các cặp không có thứ tự gồm hai phần tử khác nhau của V . Hai cạnh e_1 và e_2 được gọi là **cạnh lặp** nếu chúng cùng tương ứng với một cặp đỉnh.

Trong đa đồ thị vô hướng có thể tồn tại các cạnh lặp và trên tất cả các cạnh cũng vẫn sẽ không quy định chiều.



Hình 4.4: Đa đồ thị vô hướng

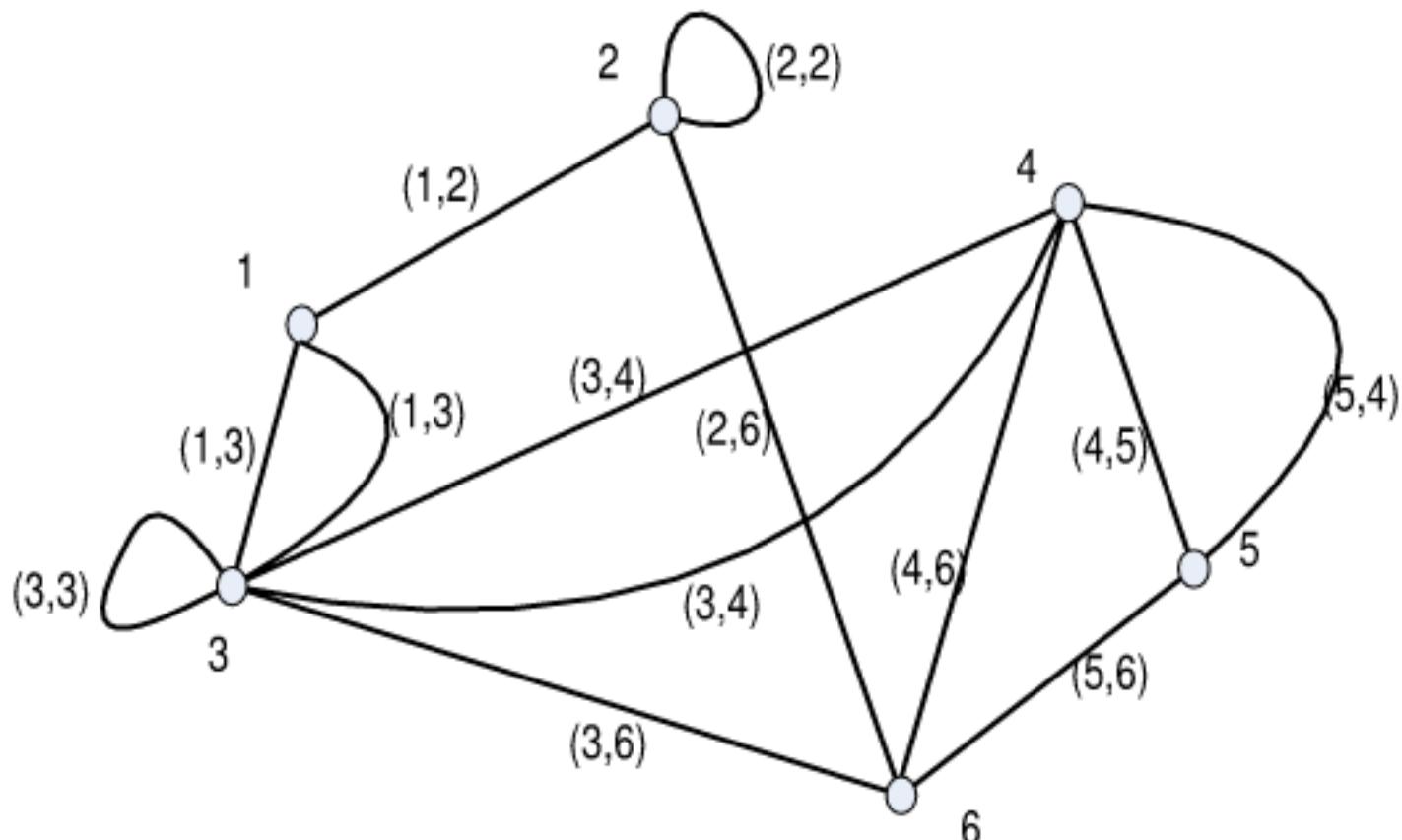
Khi đó:

$$V = \{1, 2, 3, 4, 5, 6\}$$

$$E = \{(1, 2), (1, 3), (2, 6), (3, 4), (3, 6), (4, 5), (4, 6), (5, 6), (1, 3), (3, 4), (5, 4)\}$$

• Giả đồ thị vô hướng

Đồ thị $G = (V, E)$ được gọi là giả đồ thị vô hướng nếu tập các cạnh E là **hợp** các cặp không có thứ tự gồm hai phần tử (không nhất thiết khác nhau) của V . Cạnh e được gọi là **khuyên** nếu nó có dạng $e = (u, u)$.



Hình 4.5: Giả đồ thị vô hướng

Khi đó:

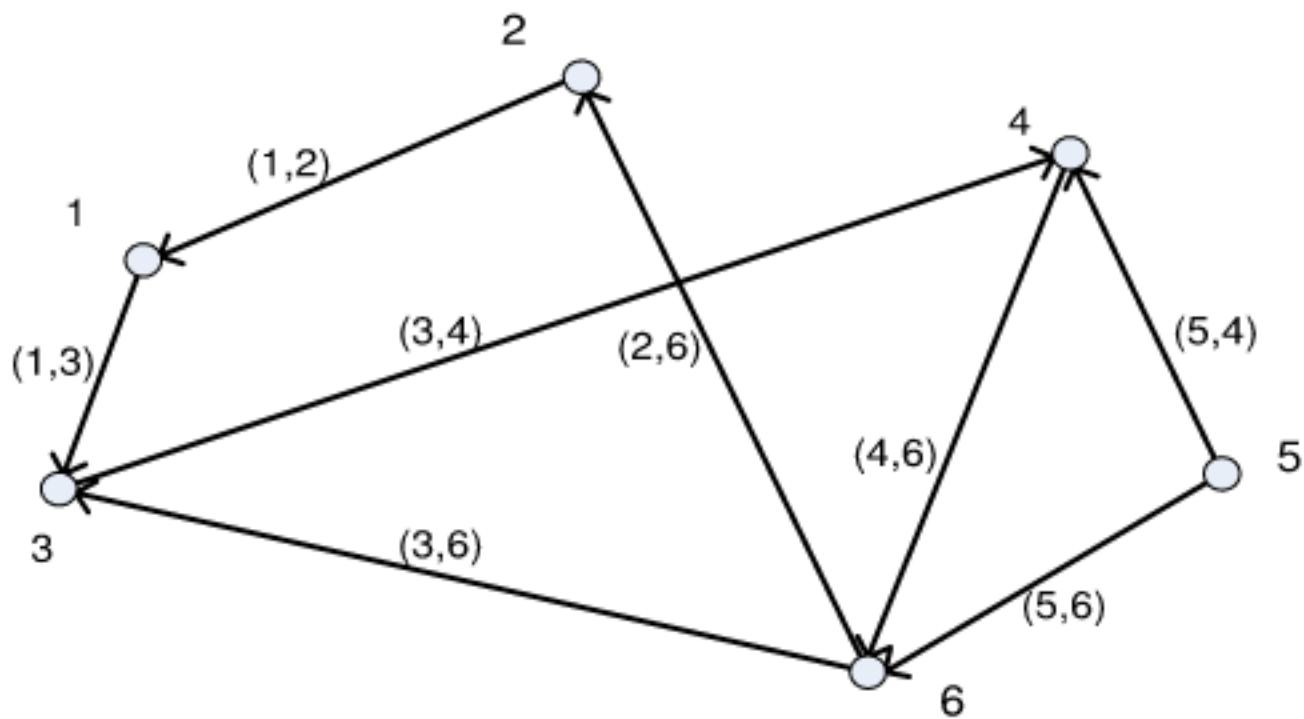
$$V = \{1, 2, 3, 4, 5, 6\}$$

$$E = \{(1, 2), (1, 3), (2, 6), (3, 4), (3, 6), (4, 5), (4, 6), (5, 6), (1, 3), (3, 4), (5, 4), (3, 3)\}$$

b. Đồ thị có hướng

• Đơn đồ thị có hướng

Đồ thị $G = (V, E)$ được gọi là đơn đồ thị có hướng nếu tập các **cung** E là **tập** các cặp **có thứ tự** gồm hai phần tử khác nhau của V .



Hình 4.6: Đơn đồ thị có hướng

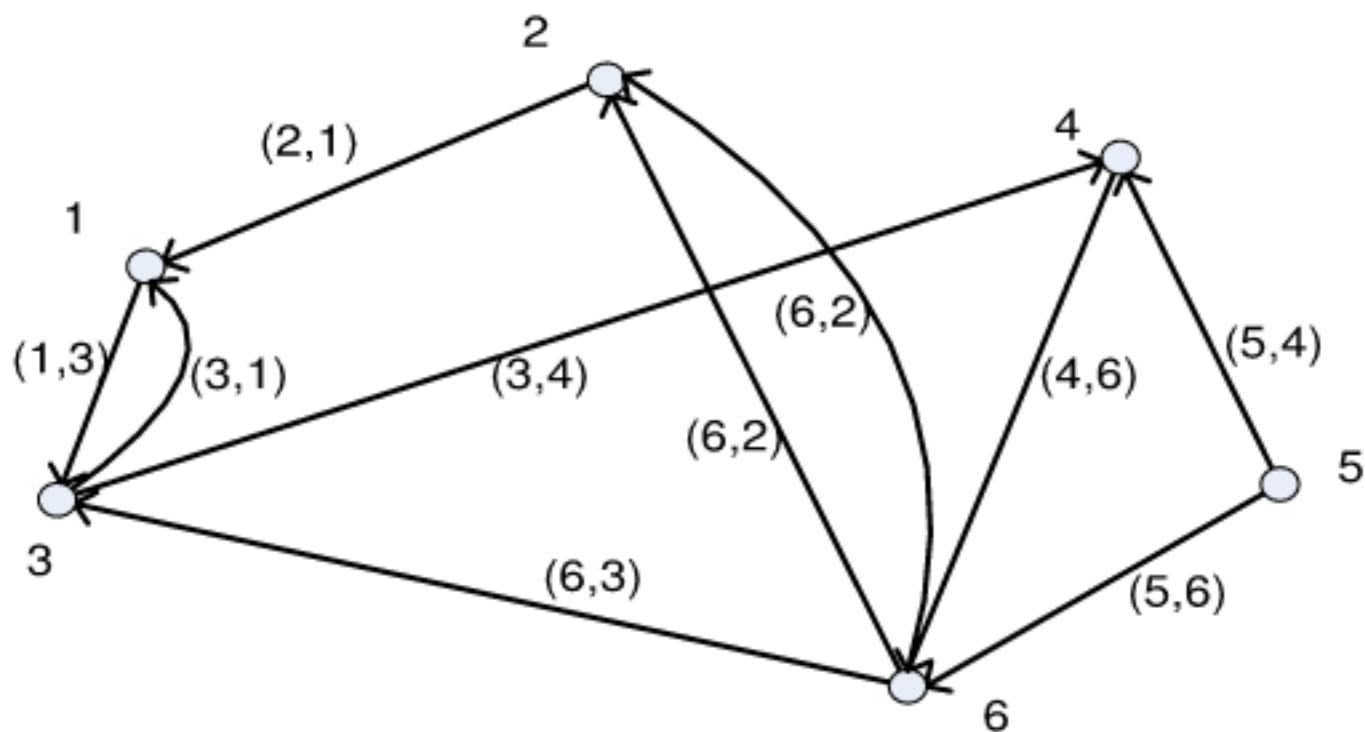
Khi đó:

$$V = \{1, 2, 3, 4, 5, 6\}$$

$$E = \{(2, 1), (1, 3), (6, 2), (3, 4), (6, 3), (4, 6), (5, 4), (5, 6)\}$$

• Đa đồ thị có hướng

Đồ thị $G = (V, E)$ được gọi là đa đồ thị có hướng nếu tập các **cung** E là **họ** các cặp **có thứ tự** gồm hai phần tử khác nhau của V . Hai cung e_1, e_2 được gọi là **cung lặp** nếu chúng cùng tương ứng với một cặp đỉnh.



Hình 4.7: Đa đồ thị có hướng

Khi đó:

$$V = \{1, 2, 3, 4, 5, 6\}$$

$$E = \{(2, 1), (1, 3), (6, 2), (3, 4), (6, 3), (4, 6), (5, 4), (5, 6), (3,1), (6,2)\}$$

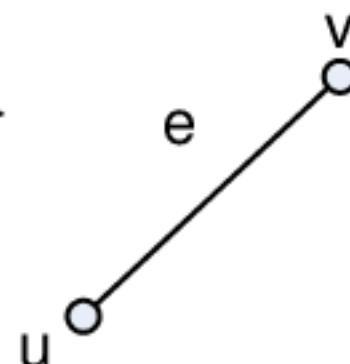
Phần trên đã đưa ra định nghĩa các loại đồ thị. Trong giáo trình này, khi chỉ nói “đồ thị” thì ta ngầm hiểu là “đơn đồ thị vô hướng”. Các loại đồ thị khác như đa đồ thị, giả đồ thị hay đồ thị có hướng khi dùng đến sẽ được nói rõ.

4.3. CÁC THUẬT NGỮ CƠ BẢN

- **Kề và liên thuộc**

Giả sử u và v là hai đỉnh của đồ thị vô hướng G và $e = (u, v)$ là cạnh của đồ thị, khi đó ta nói:

- u và v **kề nhau** và e **liên thuộc** với u và v .
- u và v là các đỉnh đầu của cạnh e

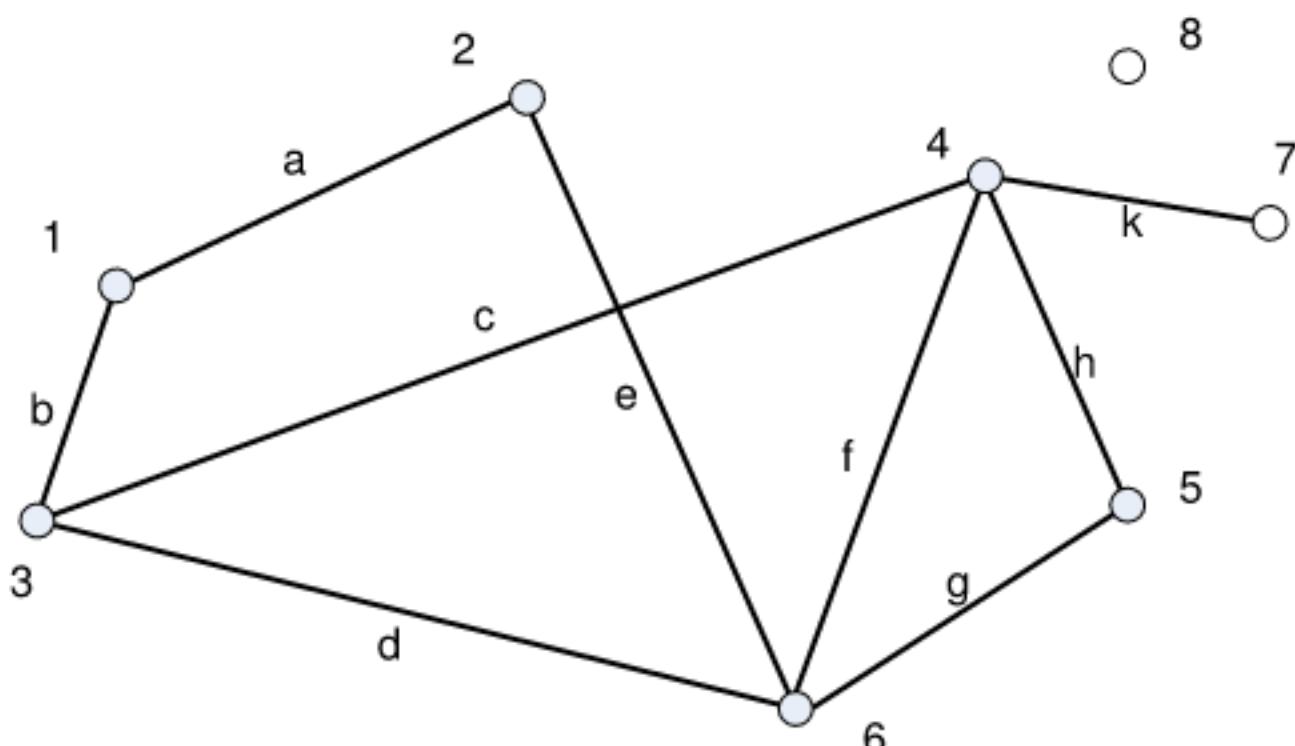


Hình 4.8: Kề và liên thuộc

- **Bậc của đỉnh**

Bậc của đỉnh v trong đồ thị vô hướng là số cạnh liên thuộc với nó. Ký hiệu $\deg(v)$.

Ví dụ:



Hình 4.9: Bậc của đỉnh

Bậc của các đỉnh là:

$\deg(1)= 2$, $\deg(2)= 2$, $\deg(3)= 3$, $\deg(4)= 4$, $\deg(5)= 2$, $\deg(6)= 4$, $\deg(7)= 1$, $\deg(8)= 0$

Đỉnh treo là đỉnh chỉ có duy nhất một cạnh liên thuộc với nó. **Đỉnh cô lập** là đỉnh không có cạnh nào liên thuộc với nó.

Trong ví dụ trên, đỉnh 7 là đỉnh treo, đỉnh 8 là đỉnh cô lập.

✓ **Định lý bắt tay**

Giả sử $G = (V, E)$ là đồ thị vô hướng với m cạnh. Khi đó tổng bậc của tất cả các đỉnh trong V sẽ bằng $2m$.

$$2m = \sum_{v \in V} \deg(v)$$

Chứng minh: Ta thấy, mỗi một cạnh nối với đúng hai đỉnh, vì thế một cạnh đóng góp 2 đơn vị vào tổng các bậc của tất cả các đỉnh. Có nghĩa là tổng các bậc của tất cả các đỉnh gấp đôi số cạnh của đồ thị.

Ví dụ: Trong hình 1.9.

✓ **Hệ quả của định lý bắt tay**

$$\sum_{v \in V} \deg(v) = 18, m = 9$$

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

Chứng minh: Gọi L và C lần lượt là tập các đỉnh bậc lẻ và bậc chẵn của đồ thị vô hướng $G = (V, E)$. Ta có:

$$2m = \sum_{v \in V} \deg(v) = \sum_{v \in L} \deg(v) + \sum_{v \in C} \deg(v)$$

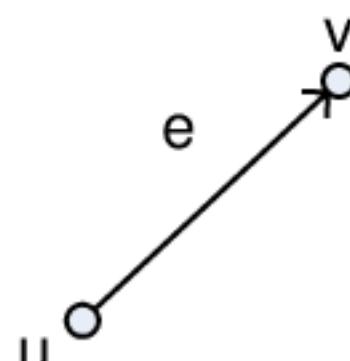
Tổng bậc của tất cả các đỉnh trong G là một số chẵn. Mặt khác, tổng $\sum_{v \in C} \deg(v)$ cũng là một số chẵn vì nó là tổng của các số chẵn. Vì vậy tổng bậc của các đỉnh bậc lẻ $\sum_{v \in L} \deg(v)$ cũng là một số chẵn. Từ đó suy ra số đỉnh bậc lẻ trong đồ thị G là một số chẵn.

• **Kề trong đồ thị có hướng**

Giả sử u và v là hai đỉnh của đồ thị có hướng G và $e = (u, v)$ là một cung của đồ thị, khi đó ta nói:

v kề u nếu cung e đi ra khỏi u và đi vào v .

u là đỉnh đầu, v là đỉnh cuối của cung e .

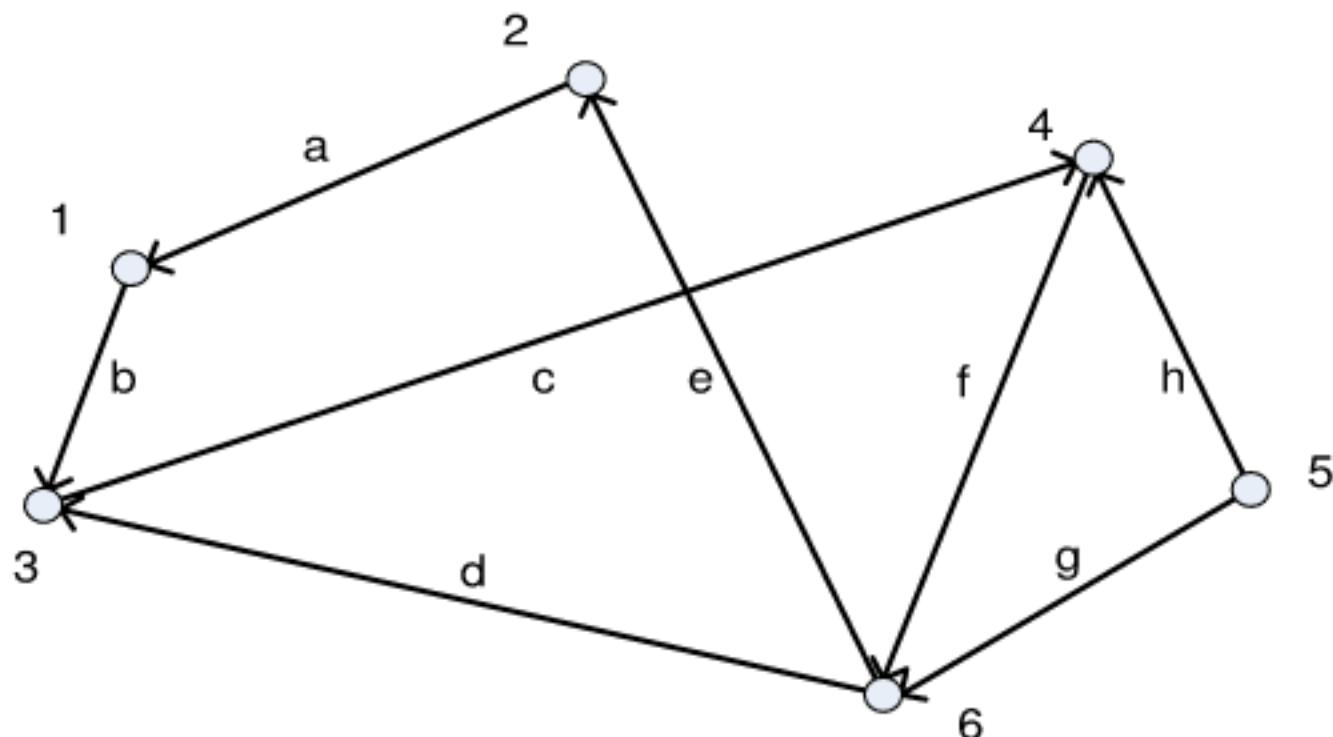


Hình 4.10: Kề trong đồ thị có hướng

• Bán bậc vào và bán bậc ra của đỉnh

Bán bậc vào (bán bậc ra) của đỉnh v trong đồ thị có hướng là số cung đi vào nó (đi ra khỏi nó) và ký hiệu là $\deg^-(v)$ ($\deg^+(v)$).

Ví dụ:



Hình 4.11: Bán bậc vào, bán bậc ra

Bán bậc ra, bán bậc vào của các đỉnh:

$$\deg^+(1) = 1, \deg^-(1) = 1, \deg^+(4) = 1, \deg^-(4) = 2.$$

✓ Định lý

Giả sử $G = (V, E)$ là đồ thị có hướng với m cung, khi đó tổng tất cả các bán bậc ra bằng tổng tất cả các bán bậc vào và bằng m .

$$\sum_{v \in V} \deg^+(v) = \sum_{v \in V} \deg^-(v) = m$$

Chứng minh: Mỗi cung của đồ thị đóng góp một bán bậc ra và một bán bậc vào. Vì thế tổng các bán bậc vào bằng tổng các bán ra và bằng số cung của đồ thị.

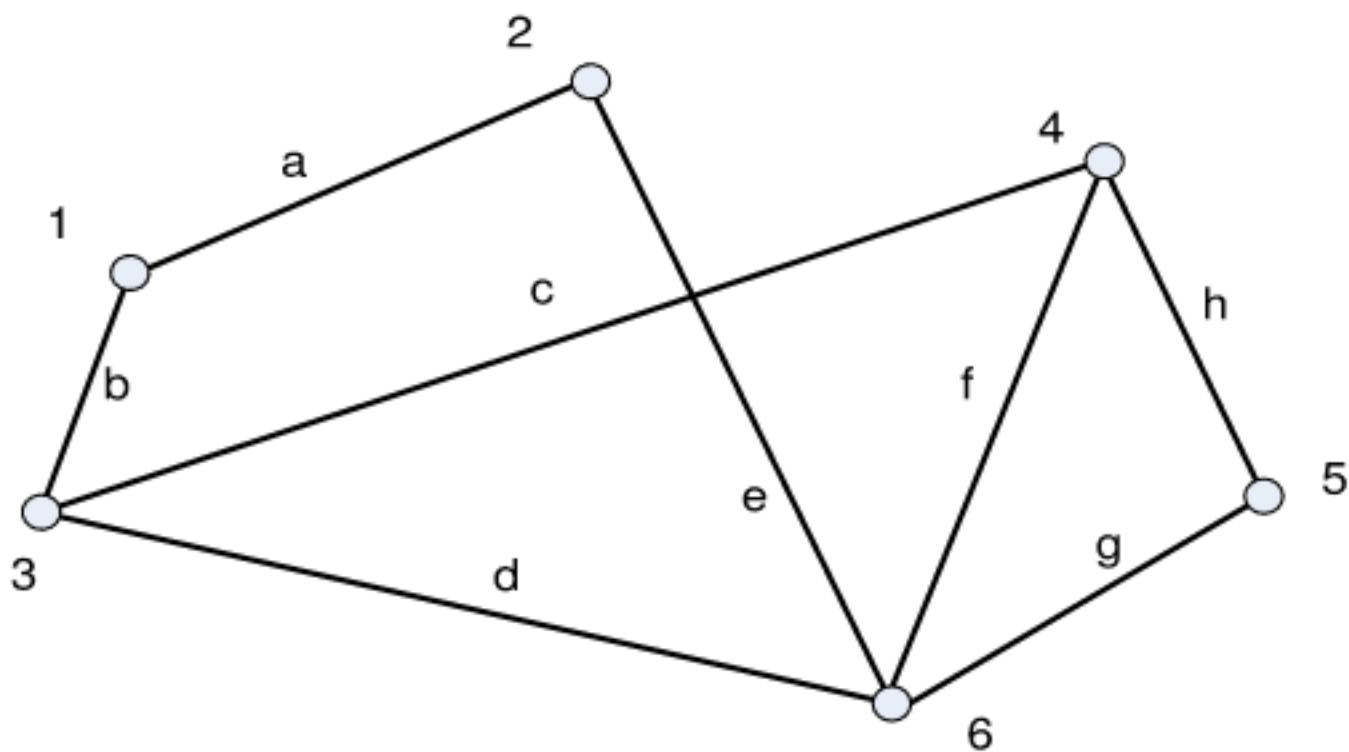
4.4. ĐƯỜNG ĐI, CHU TRÌNH, ĐỒ THỊ LIÊN THÔNG

4.4.1. Đường đi

Đường đi độ dài n ($n \in N^+$) từ đỉnh u đến đỉnh v trên đồ thị vô hướng $G = (V, E)$ là dãy (theo định): $x_0, x_1, x_2, \dots, x_{n-1}, x_n$. Trong đó $u = x_0, v = x_n, (x_i, x_{i+1}) \in E$.

Hay theo cạnh: $(x_0, x_1), (x_1, x_2), \dots, (x_{n-1}, x_n)$.

Khi đó: u gọi là **đỉnh đầu**, v gọi là **đỉnh cuối** của đường đi.



Hình 4.12: Đường đi

Một ví dụ về đường đi từ đỉnh 1 đến đỉnh 6:

Theo đỉnh: (1, 3, 4, 5, 6).

Theo cạnh: (b, c, h, g).

4.4.2. Chu trình

Đường đi có đỉnh đầu và đỉnh cuối trùng nhau gọi là chu trình.

Đường đi (hay chu trình) được gọi là **đơn** nếu nó không đi qua một **cạnh** nào quá một lần.

Ví dụ:

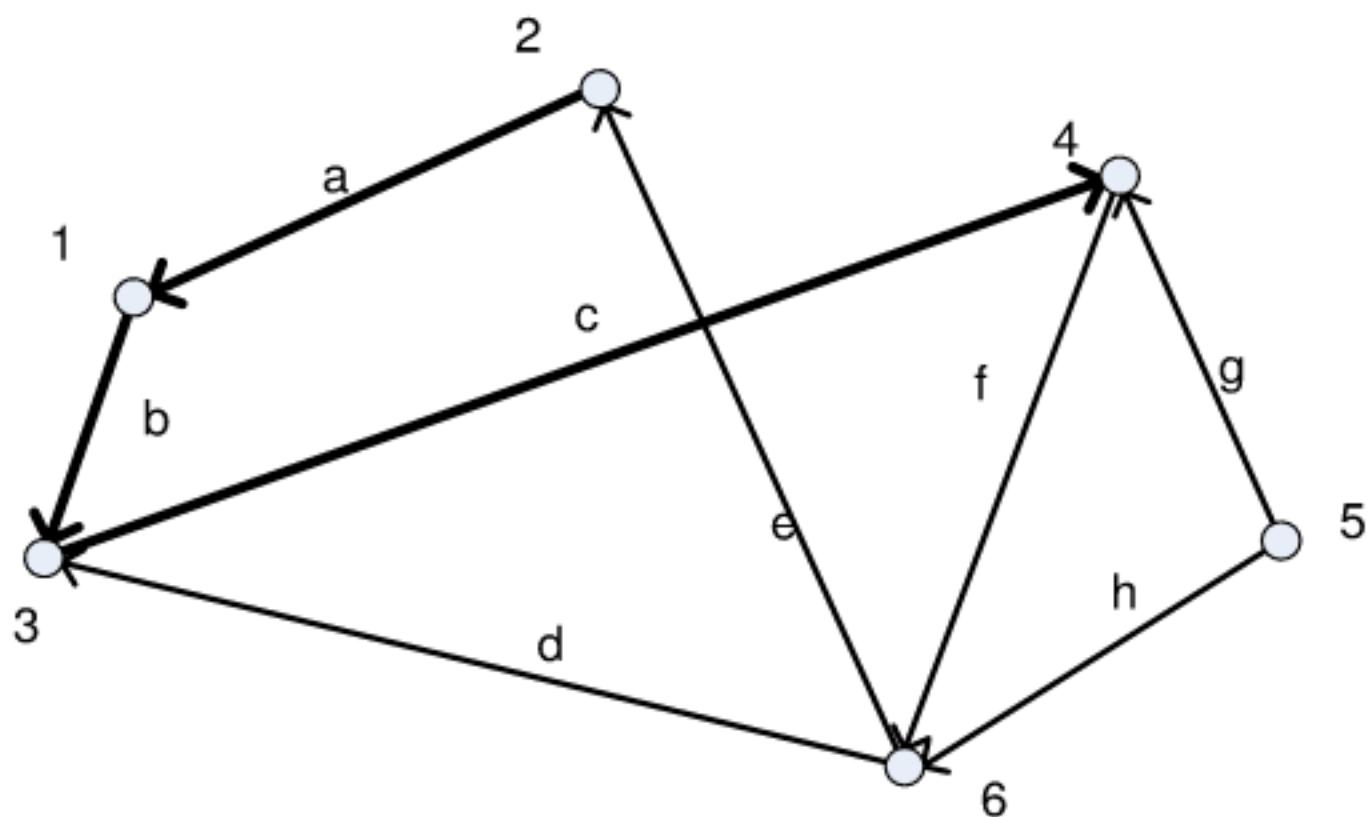
Chu trình đơn: (1, 2, 6, 3, 1).

Chu trình không phải chu trình đơn: (2, 6, 4, 3, 6, 2).

4.4.3. Đường đi và chu trình trong đồ thị có hướng

Đường đi và chu trình trong đồ thị có hướng cũng tương tự như đường đi và chu trình trong đồ thị vô hướng. Khi đó, nếu $x_0, x_1, x_2, \dots, x_{n-1}, x_n$ là một đường đi thì trong đồ thị phải tồn tại các cung $(x_0, x_1), (x_1, x_2), \dots, (x_{n-1}, x_n)$.

Ví dụ: Trong đồ thị phía dưới (2, 1, 3, 4) là một đường đi của đồ thị, nhưng (4, 3, 1, 2) không phải là đường đi vì không tồn tại các cung (4, 3), (3, 1), (1, 2).

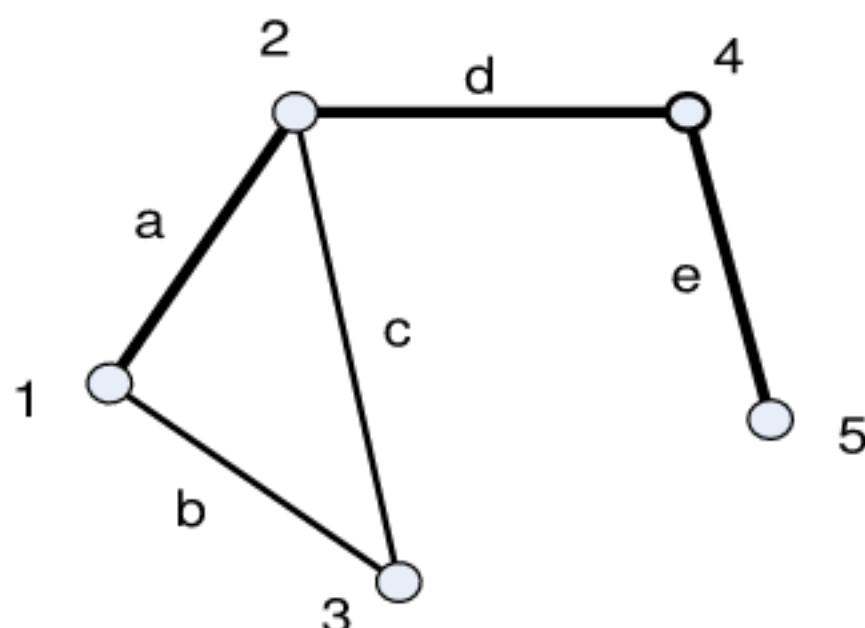


Hình 4.13: Đường đi trong đồ thị có hướng

4.4.4. Đồ thị liên thông

Đồ thị vô hướng $G = (V, E)$ được gọi là **liên thông** nếu luôn tìm được đường đi giữa hai đỉnh bất kỳ của nó.

Đồ thị $H=(W,F)$ được gọi là **đồ thị con** của đồ thị $G=(V,E)$ nếu: $W \subseteq V$ và $F \subseteq E$



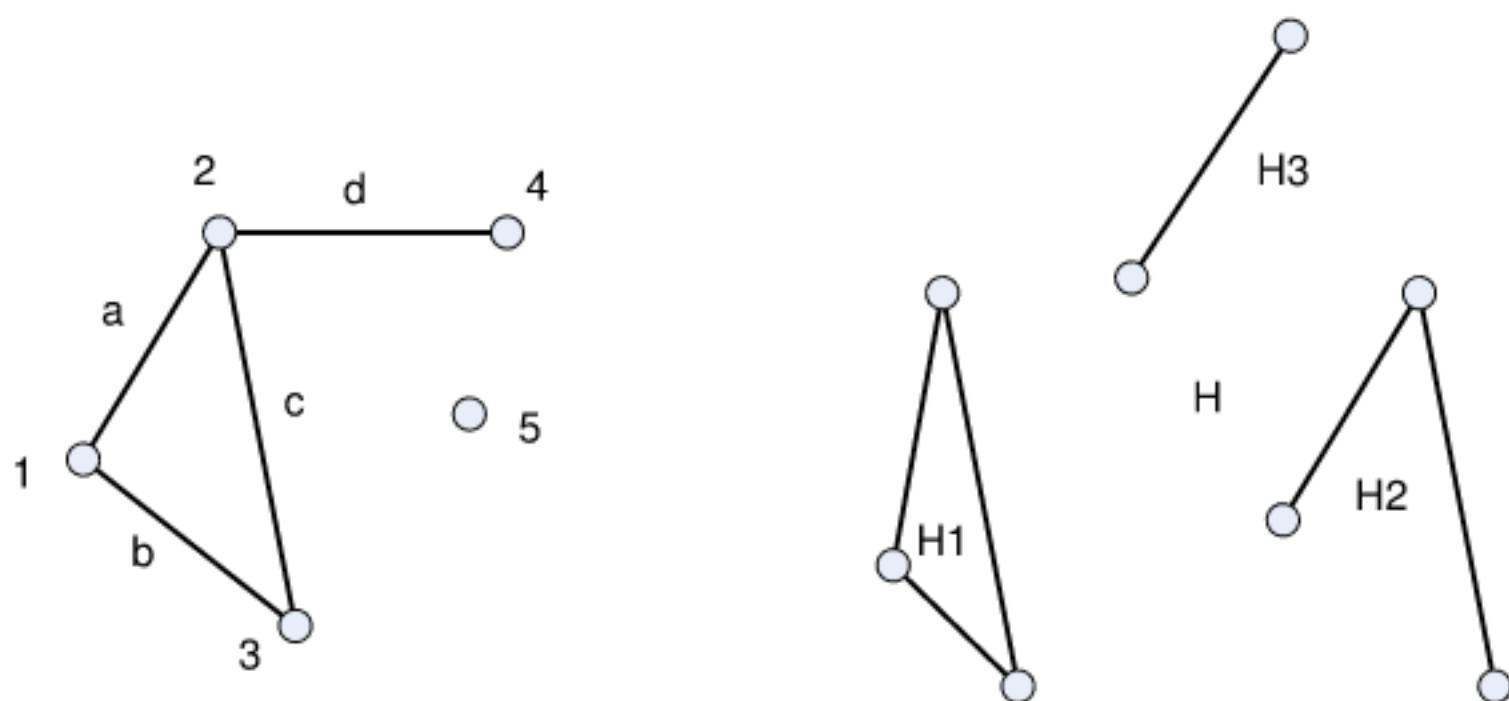
Hình 4.14: Đồ thị con

$$V=\{1, 2, 3, 4, 5\} \quad W=\{1, 2, 4, 5\}$$

$$E=\{a, b, c, d, e\} \quad F=\{a, d, e\}$$

Nhận xét: Một đồ thị bất kỳ sẽ được phân rã thành các **thành phần liên thông**, và mỗi thành phần liên thông này là một đồ thị con của đồ thị ban đầu.

Ví dụ:



Hình 4.15: Thành phần liên thông

Đỉnh v được gọi là **đỉnh rẽ nhánh** nếu việc loại bỏ v cùng các cạnh liên thuộc với nó sẽ làm tăng số thành phần liên thông của đồ thị.

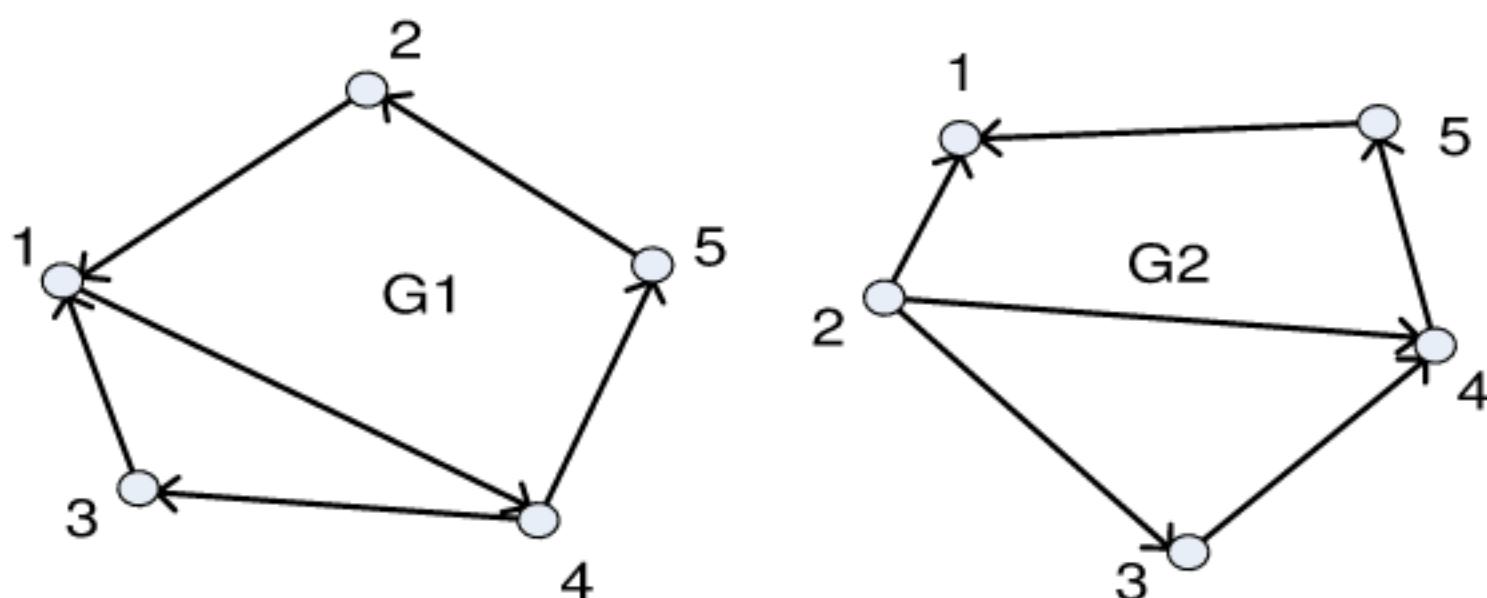
Cạnh e được gọi là **cầu** nếu việc loại bỏ nó sẽ làm tăng số thành phần liên thông của đồ thị.

Ví dụ: Trong hình 1.15 các đỉnh 2, 4 là các đỉnh rẽ nhánh, cạnh d là cầu.

Đồ thị có hướng $G = (V, E)$ được gọi là **liên thông mạnh** nếu luôn tìm được đường đi từ một đỉnh bất kỳ đến một đỉnh bất kỳ khác của nó

Đồ thị có hướng $G = (V, E)$ được gọi là **liên thông yếu** nếu đồ thị vô hướng tương ứng với nó là đồ thị vô hướng liên thông

Ví dụ



Hình 4.16: Liên thông mạnh, yếu

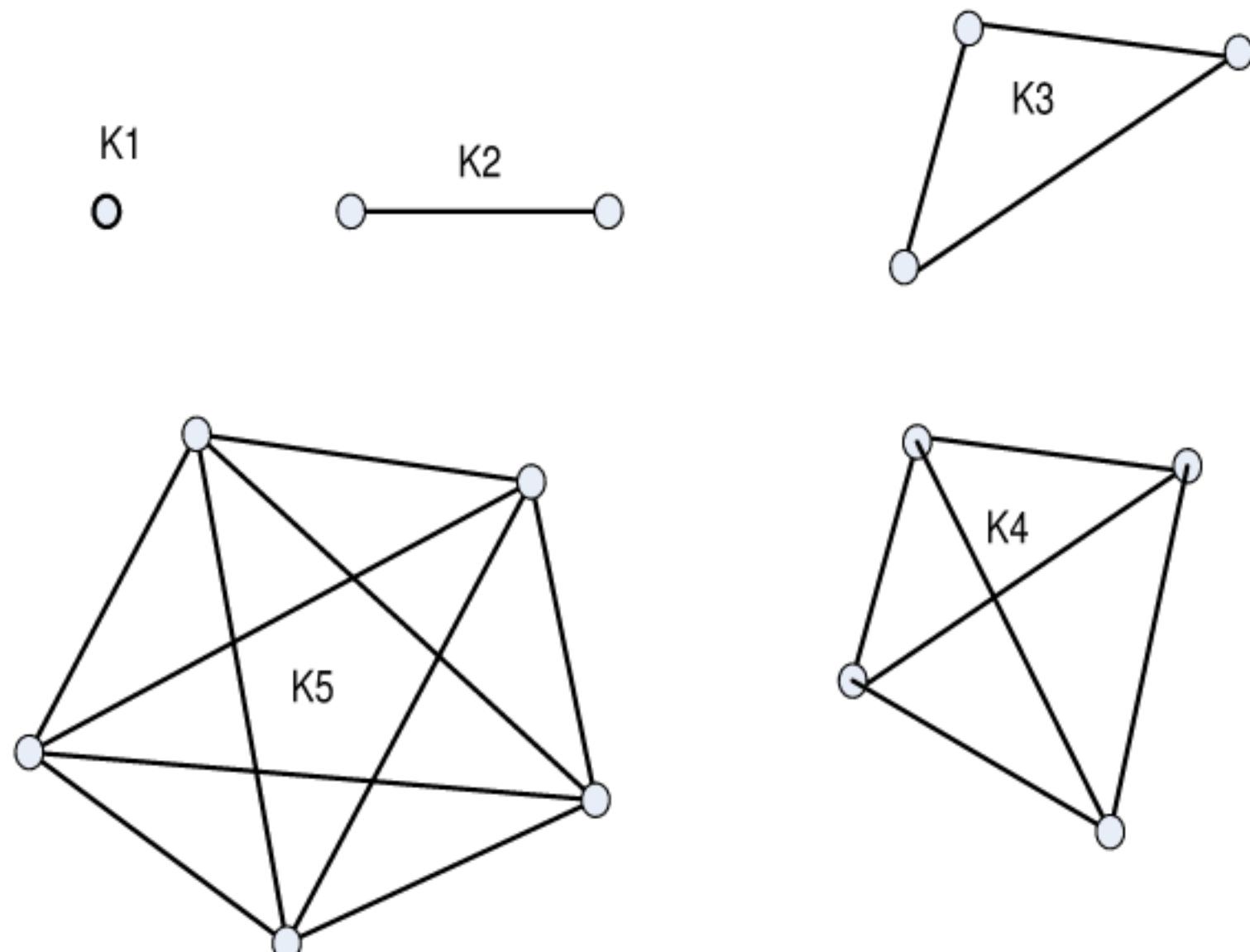
Đồ thị G1 liên thông mạnh, đồ thị G2 không liên thông mạnh vì không có đường đi từ đỉnh 1 đến các đỉnh còn lại. Tuy nhiên, đồ thị G2 là liên thông yếu.

4.5. MỘT SỐ DẠNG ĐỒ THỊ ĐẶC BIỆT

4.5.1. Đồ thị đầy đủ

Đơn đồ thị vô hướng n đỉnh được gọi là đồ thị đầy đủ nếu hai đỉnh bất kỳ đều được nối với nhau bằng một cạnh. Ký hiệu: K_n

Số cạnh của K_n là: $n(n-1)/2$, và là số cạnh tối đa của một đơn đồ thị n đỉnh.

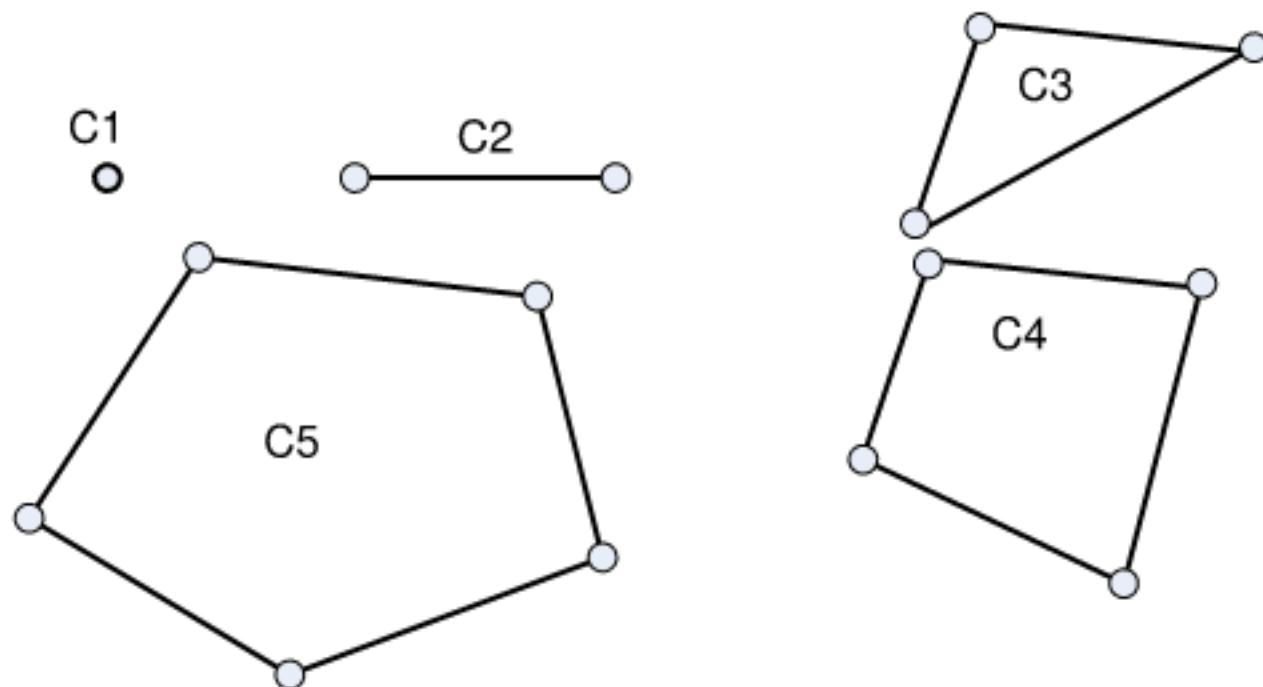


Hình 4.17: Đồ thị đầy đủ

4.5.2. Đồ thị vòng, đồ thị bánh xe

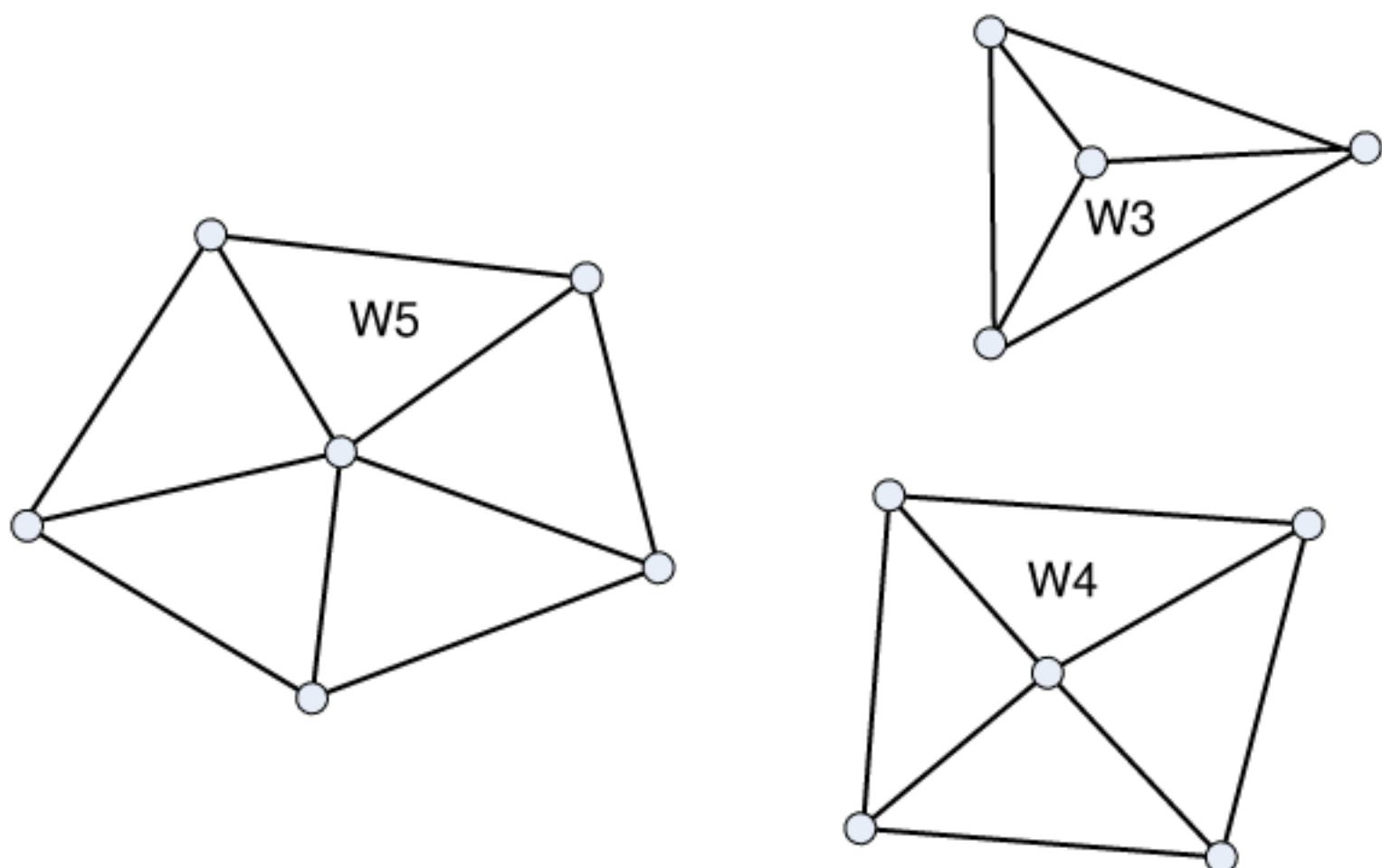
Đơn đồ thị đơn vô hướng n đỉnh được gọi là đồ thị vòng nếu nó có duy nhất một chu trình đơn đi qua tất cả các đỉnh. Ký hiệu C_n .

Với $n \geq 3$, số cạnh của đồ thị vòng là: n .



Hình 4.18: Đồ thị vòng

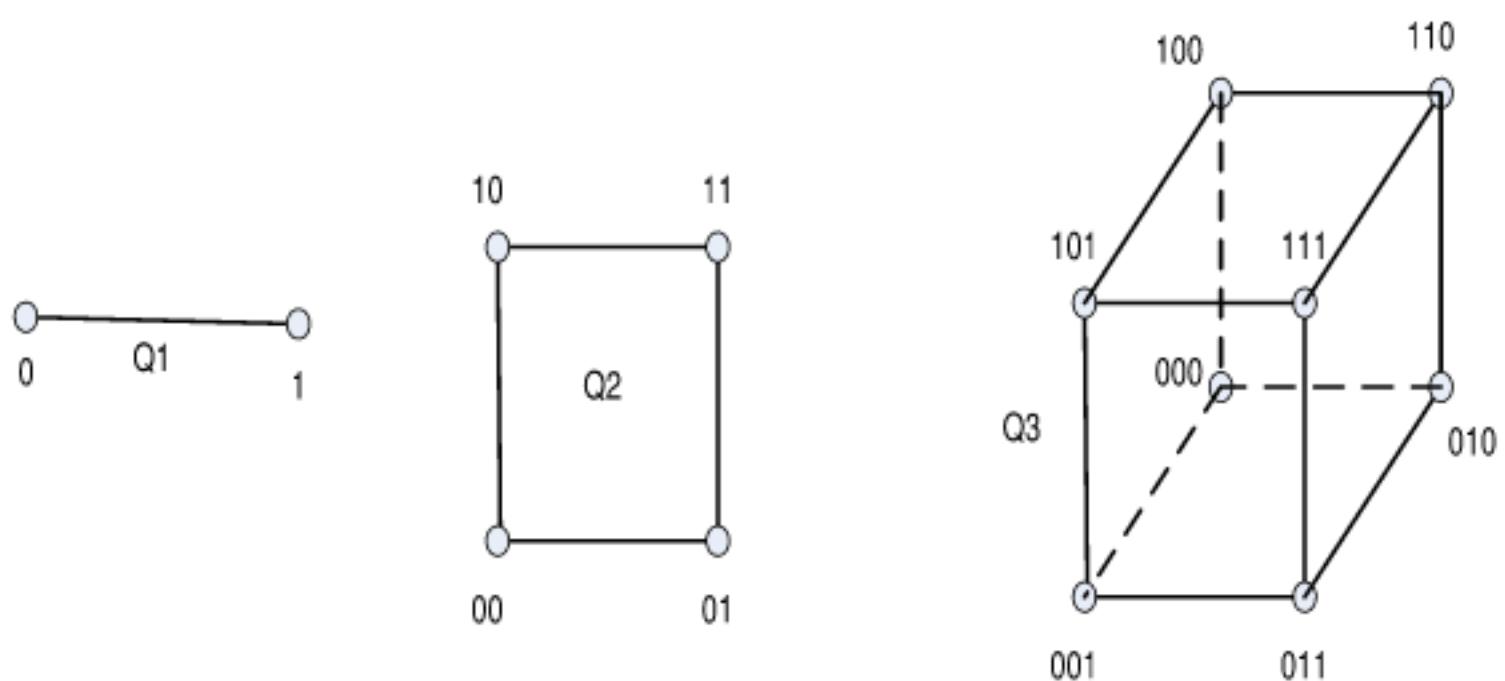
Đồ thị bánh xe $n \geq 3$ đỉnh là đồ thị thu được từ đồ thị C_n bằng cách bổ sung thêm một đỉnh mới nối với tất cả các đỉnh của C_n . Ký hiệu W_n . Số cạnh của đồ thị bánh xe là: $2n$.



Hình 4.19: Đồ thị bánh xe

4.5.3. Đồ thị siêu khối

Đồ thị siêu khối $k=2^n$ đỉnh là đồ thị có các đỉnh được đánh số bằng các chuỗi nhị phân độ dài n . Ký hiệu: Q_n . Hai đỉnh là kề nhau nếu hai chuỗi nhị phân tương ứng chỉ khác nhau 1 bit.



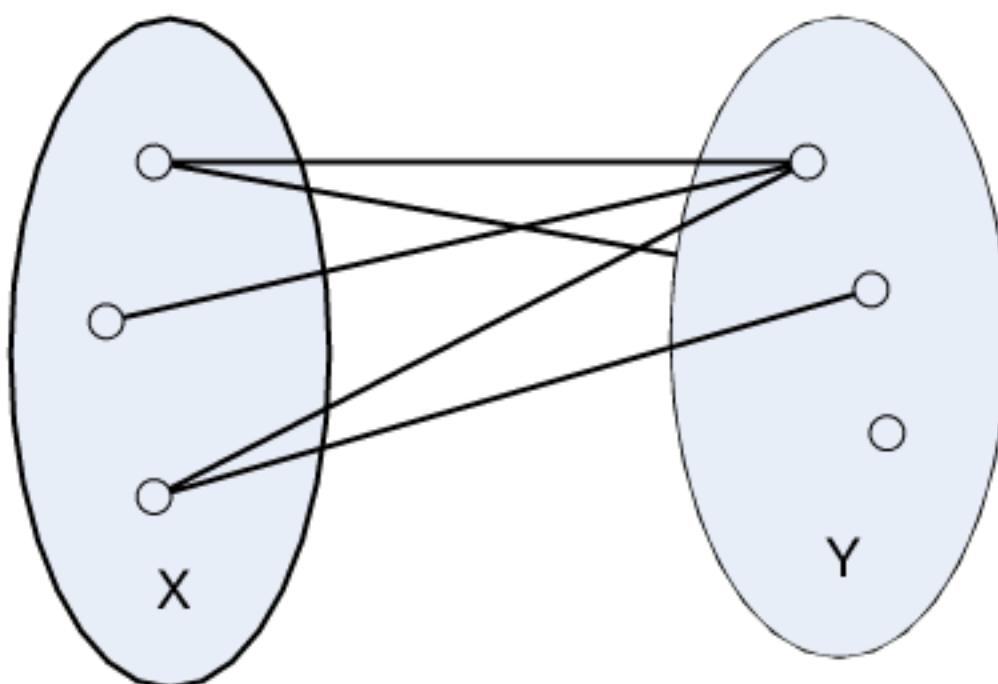
Hình 4.20: Đồ thị siêu khối

4.5.4. Đồ thị hai phía

Đơn đồ thị $G = (V, E)$ gọi là đồ thị hai phía nếu tập các đỉnh V được phân hoạch thành 2 tập X và Y sao cho:

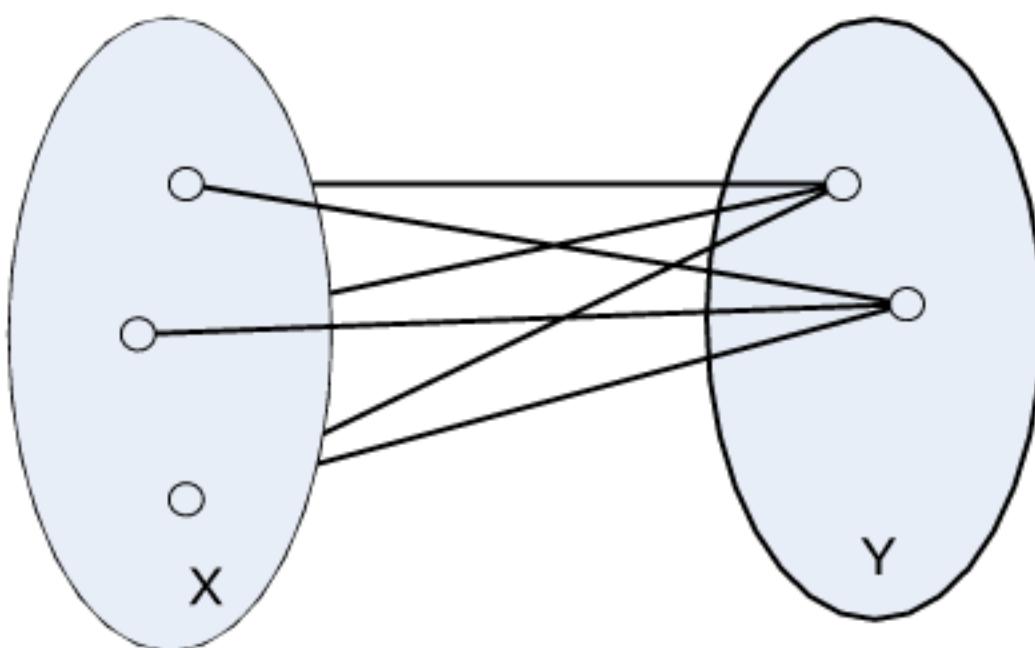
$$V = X \cup Y, X \neq \emptyset, Y \neq \emptyset, X \cap Y = \emptyset.$$

Mỗi cạnh của G sẽ có một đỉnh thuộc X và một đỉnh thuộc Y .



Hình 4.21: Đồ thị hai phía

Đơn đồ thị $G = (X \cup Y, E)$ được gọi là **đồ thị hai phía đầy đủ** nếu: Mọi đỉnh thuộc X sẽ được nối với tất cả các đỉnh thuộc Y và ngược lại. Nếu $|X| = m$ và $|Y| = n$ thì ta sẽ ký hiệu là $K_{m,n}$.



Hình 4.22: Đồ thị hai phía đầy đủ

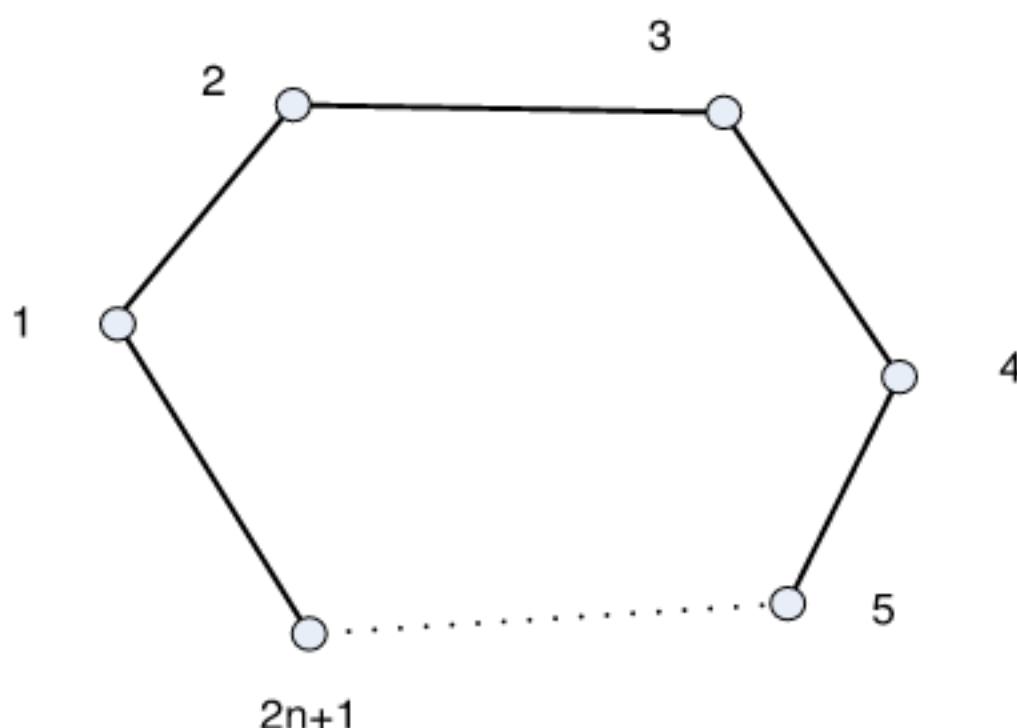
Trong đồ thị hình 1-21: $|X| = 3$, $|Y| = 2$, $K_{2,3}$.

✓ **Định lý**

Đơn đồ thị $G = (V, E)$ là đồ thị hai phía khi và chỉ khi nó không chứa chu trình độ dài lẻ.

Chứng minh: Trước hết ta chứng minh: Mọi đồ thị hai phía không chứa chu trình độ dài lẻ.

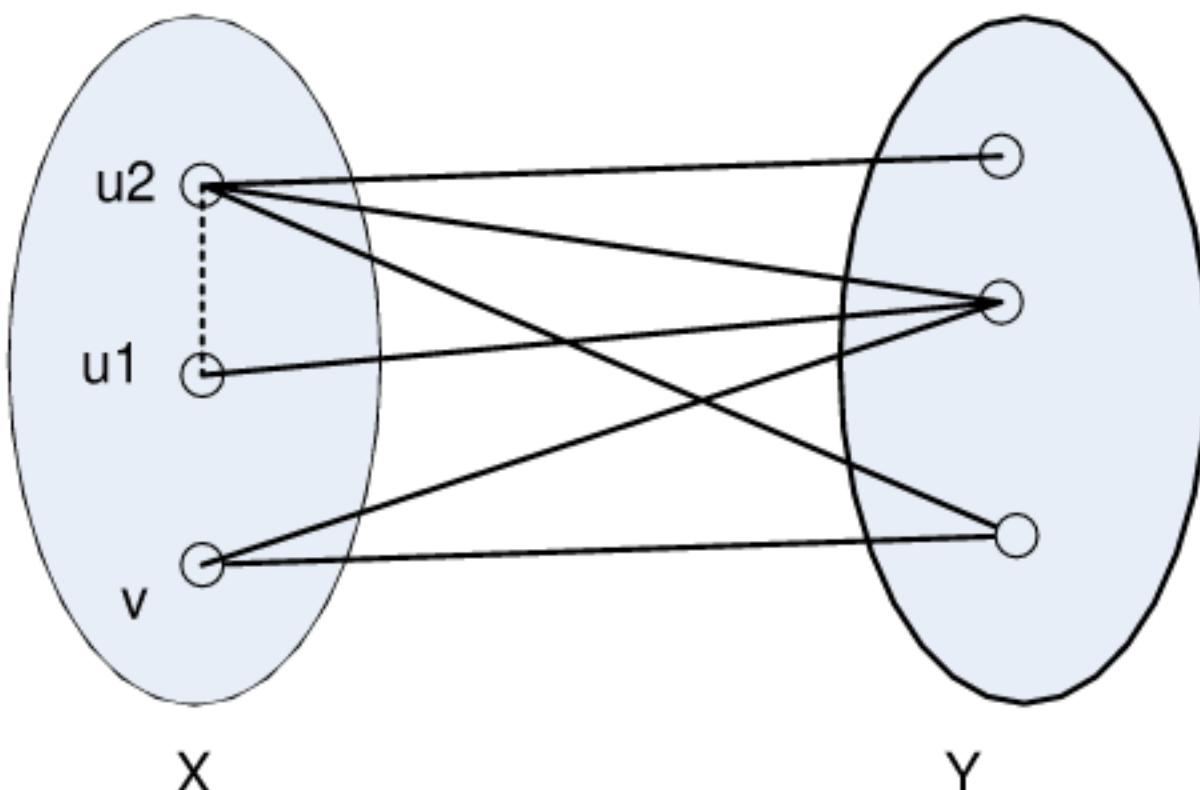
Giả sử $G = (X \cup Y, E)$ là đồ thị hai phía có chứa chu trình độ dài lẻ. Ta đánh số các đỉnh của chu trình 1, 2, 3, ... Nếu các đỉnh lẻ của chu trình thuộc X thì các đỉnh chẵn thuộc Y hay ngược lại. Tức là không tồn tại hai đỉnh cùng chẵn hay cùng lẻ đứng cạnh nhau. Nhưng do chu trình độ dài lẻ nên đỉnh 1 và đỉnh $2n+1$ (đỉnh cuối của chu trình) đứng cạnh nhau. (Vô lý).



Hình 4.23: Hai phía $\rightarrow \exists!$ Chu trình lẻ

Đảo lại, ta chứng minh nếu G không chứa chu trình độ dài lẻ thì nó là đồ thị hai phia.

Đánh số các đỉnh của G theo từng thành phần liên thông bằng các số 0 và 1. Chọn v là đỉnh đầu tiên, đánh số 0. Với mọi đỉnh khác, nếu đường đi ngắn nhất từ nó đến v là một số lẻ thì đánh số 1, ngược lại đánh số 0. Từ đây, ta có tập X là tập các đỉnh được đánh số 0, Y là tập các đỉnh được đánh nhãn 1. Giả sử G không phải là đồ thị hai phia, khi đó tồn tại một cạnh nối các đỉnh có cùng nhãn 0 hoặc 1, ta gọi hai đỉnh đó là u_1 , u_2 . Theo cách đánh số đường đi từ u_1 đến v , và từ u_2 đến v có độ dài là cùng chẵn hoặc cùng lẻ. Như vậy trong đồ thị G tồn tại chu trình $(v, \dots, u_1, u_2, \dots, v)$ có độ dài lẻ (mâu thuẫn với giả thiết). \Rightarrow Điều phải chứng minh (đpcm).



Hình 4.24: $\exists!$ Chu trình lẻ \rightarrow Hai phia

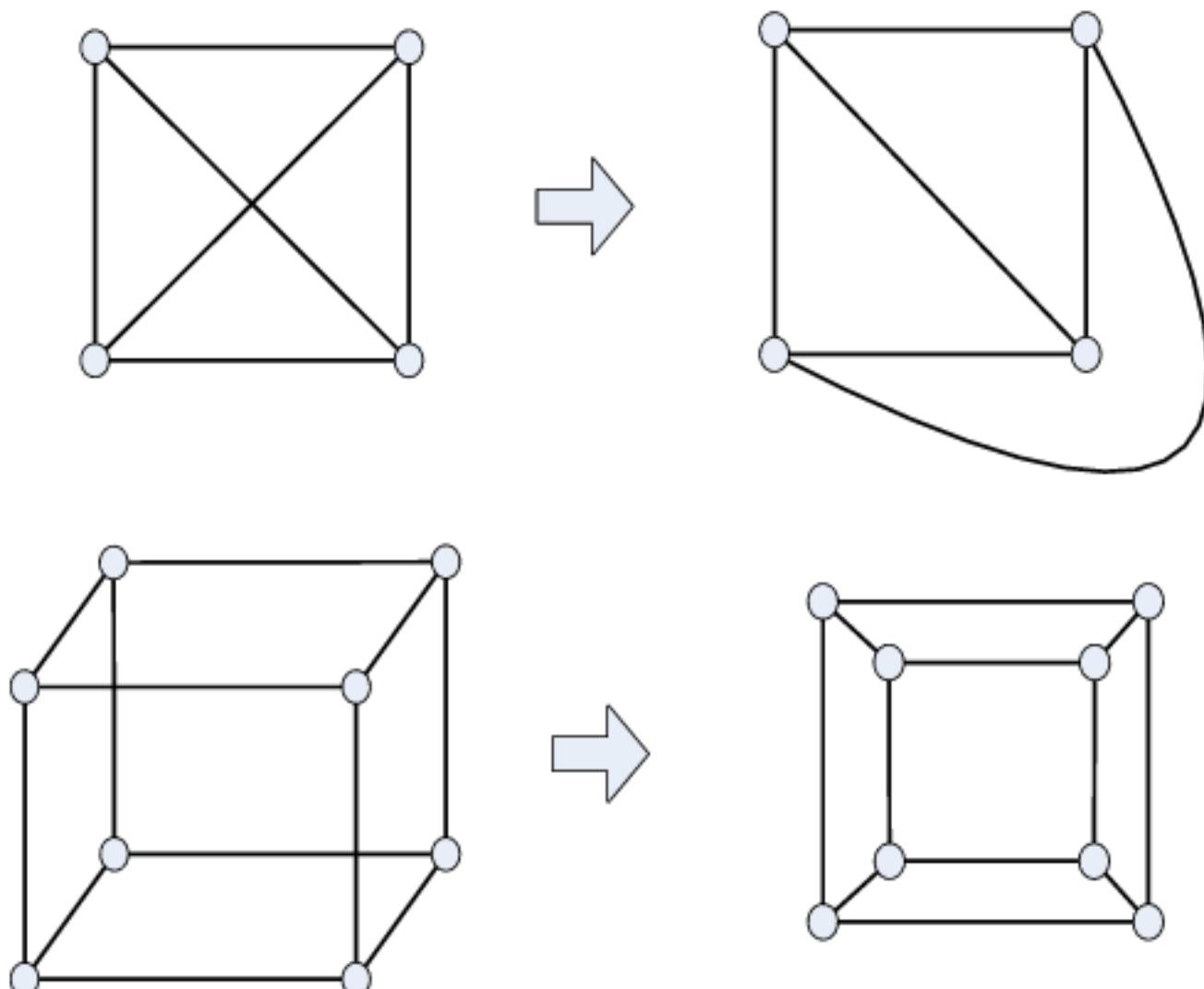
✓ Thuật toán kiểm tra đồ thị hai phia

Sau đây là thuật toán kiểm tra xem một đồ thị G cho trước có là đồ thị hai phia hay không.

1. Chọn v là đỉnh bất kỳ. Đặt $X = \{v\}$.
2. $Y = \{u \mid u \text{ kề với } v, \forall v \in X\}$.
3. Nếu $X \cap Y \neq \emptyset \Rightarrow G$ không là đồ thị hai phia.
4. Ngược lại, đặt $X = Y$ Quay trở lại 2.
5. Nếu tất cả các đỉnh được xét hết mà không xảy ra 3, thì G là đồ thị hai phia. Ngược lại, G không là đồ thị hai phia.

4.5.5. Đồ thị phẳng

Đồ thị được gọi là đồ thị phẳng nếu ta có thể vẽ nó trên một mặt phẳng mà các cạnh không giao nhau.



Hình 4.25: Đồ thị phẳng

✓ Định lý Euler

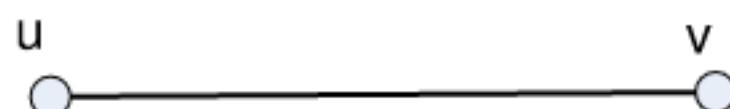
Giả sử $G = (V, E)$ là đồ thị phẳng, liên thông với e cạnh và v đỉnh. Gọi f là số mặt của đồ thị. Khi đó: $f = e - v + 2$.

Chứng minh:

Ta chứng minh định lý bằng phương pháp quy nạp theo số cạnh của đồ thị như sau:

Gọi f_n , e_n , v_n tương ứng là số mặt, số cạnh, số đỉnh của biểu diễn phẳng của đồ thị n cạnh G_n do biểu diễn phẳng của G sinh ra.

- Trường hợp $n = 1$ ta có: $e_1 = 1$, $v_1 = 2$ thì $f_1 = 1 - 2 + 2 = 1$. Điều này đúng, thể hiện qua hình vẽ sau.

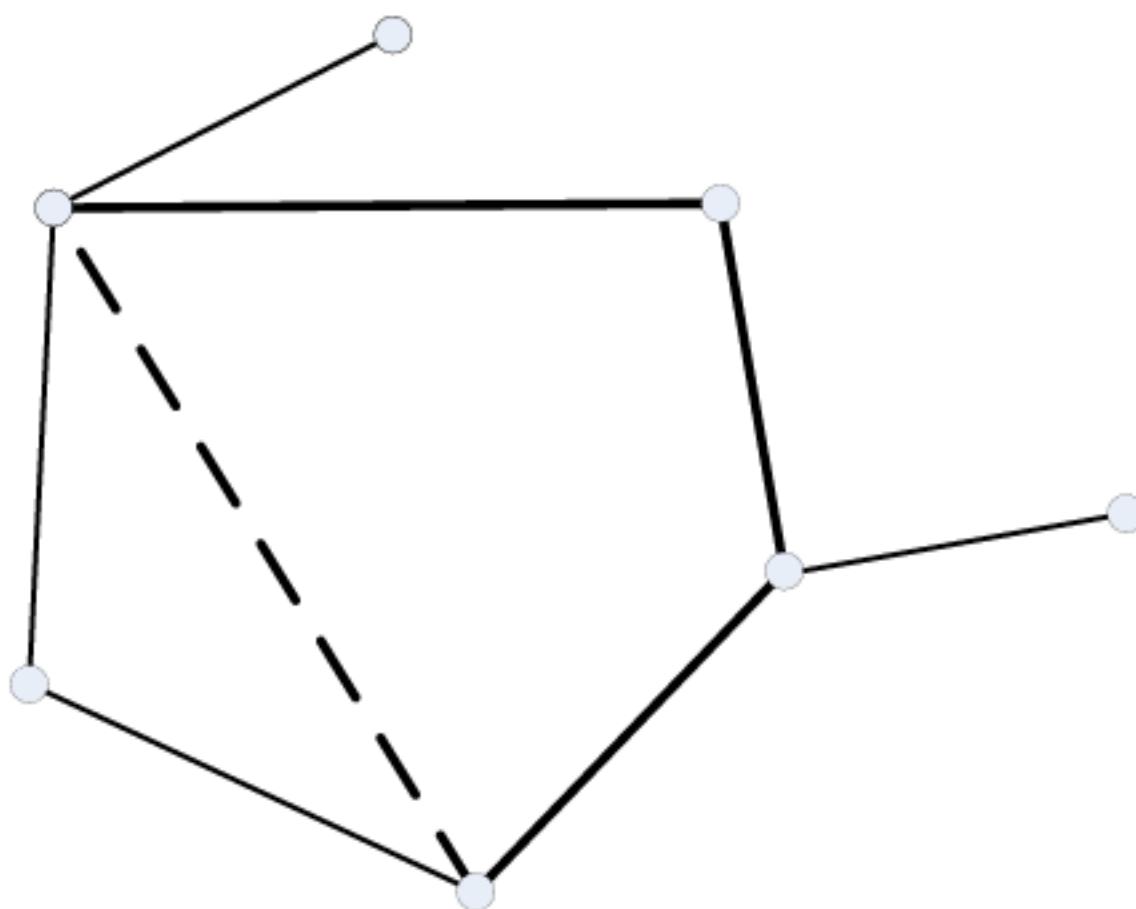


Hình 4.26: Trường hợp $n = 1$

Giả sử định lý đúng cho mọi đồ thị phẳng có n cạnh ($n \geq 1$).

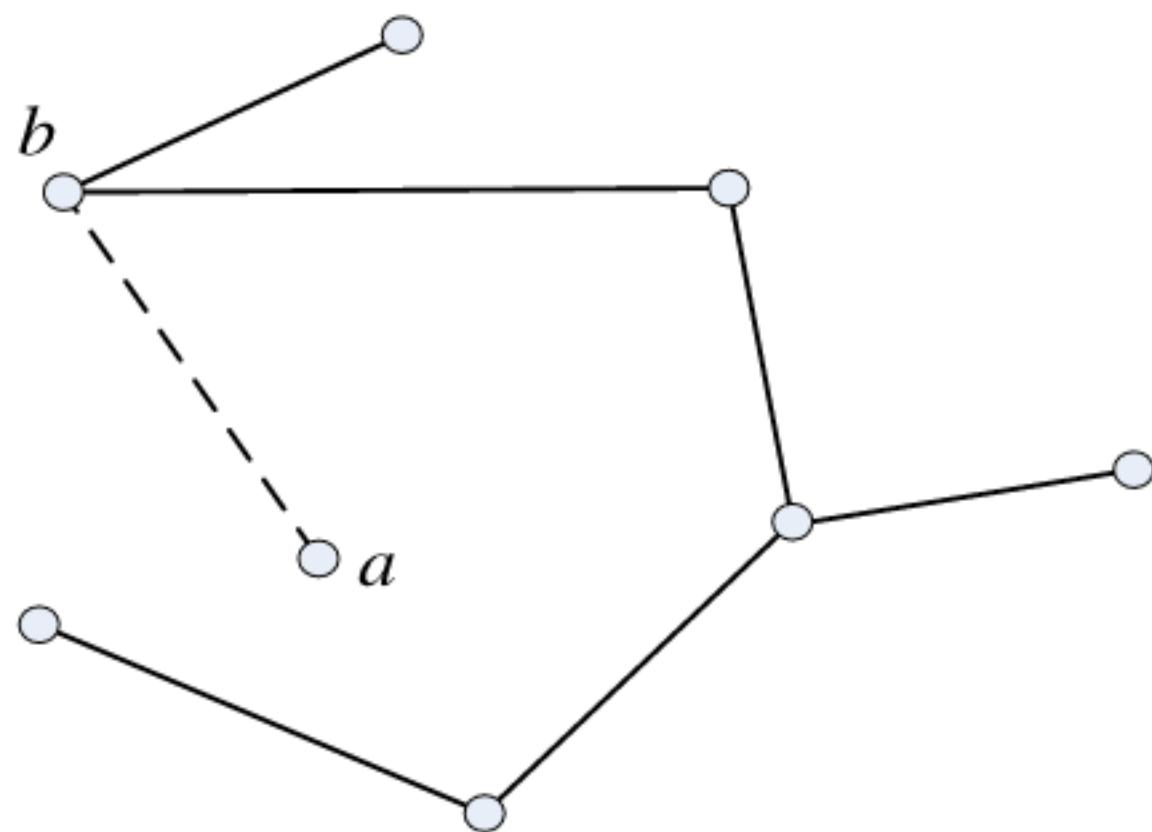
Giả sử G_{n+1} là đồ thị phẳng bất kỳ, liên thông có $n+1$ cạnh, ta phải chứng minh: $f_{n+1} = e_{n+1} - v_{n+1} + 2$. Có hai trường hợp có thể xảy ra:

+ Trong G_{n+1} tồn tại một chu trình: ta có thể bỏ đi một cạnh của chu trình đó và nhận được đồ thị phẳng, liên thông n cạnh G_n . Việc bỏ đi cạnh này sẽ làm số miền của G_{n+1} giảm đi 1 và số đỉnh thì không đổi. Tức là: $f_n = f_{n+1} - 1$, $e_n = e_{n+1} - 1$, $v_n = v_{n+1}$. Vì thế từ: $f_n = e_n - v_n + 2 \Rightarrow (f_{n+1} - 1) = (e_{n+1} - 1) - v_{n+1} + 2$
 $\Rightarrow f_{n+1} = e_{n+1} - v_{n+1} + 2$ (đpcm).



Hình 4.27: Trường hợp G_{n+1} có chu trình

+ Trong G_{n+1} không tồn tại chu trình: Khi đó sẽ tồn tại đỉnh treo a cùng với cạnh (a, b) thuộc G_{n+1} . (Có thể chọn a là đỉnh đầu của đường đi dài nhất trong G_{n+1}). Ta có thể bỏ đi cạnh (a, b) này cùng với đỉnh treo a và nhận được đồ thị phẳng, liên thông n cạnh G_n . Việc bỏ đi cạnh này sẽ làm số miền của G_{n+1} không thay đổi nhưng số đỉnh giảm đi 1. Tức là: $f_n = f_{n+1}$, $e_n = e_{n+1} - 1$, $v_n = v_{n+1} - 1$. Vì thế từ: $f_n = e_n - v_n + 2 \Rightarrow f_{n+1} = (e_{n+1} - 1) - (v_{n+1} - 1) + 2 \Rightarrow f_{n+1} = e_{n+1} - v_{n+1} + 2$ (đpcm).

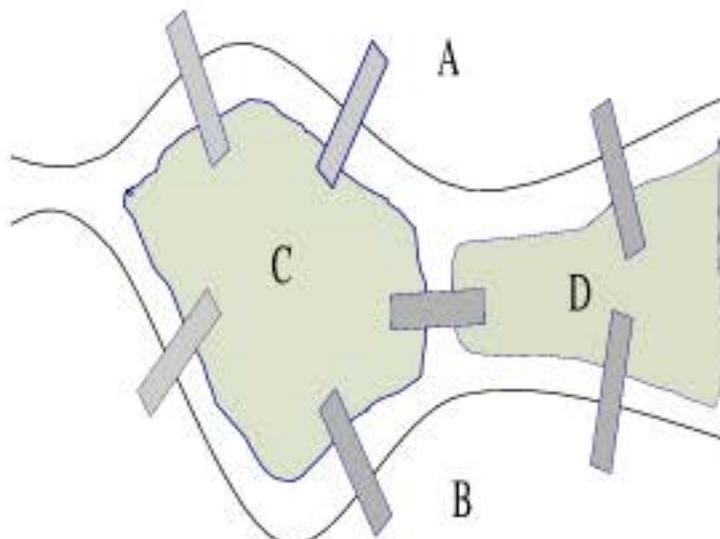


Hình 4.28: Trường hợp G_{n+1} không có chu trình

BÀI TẬP CHƯƠNG 4

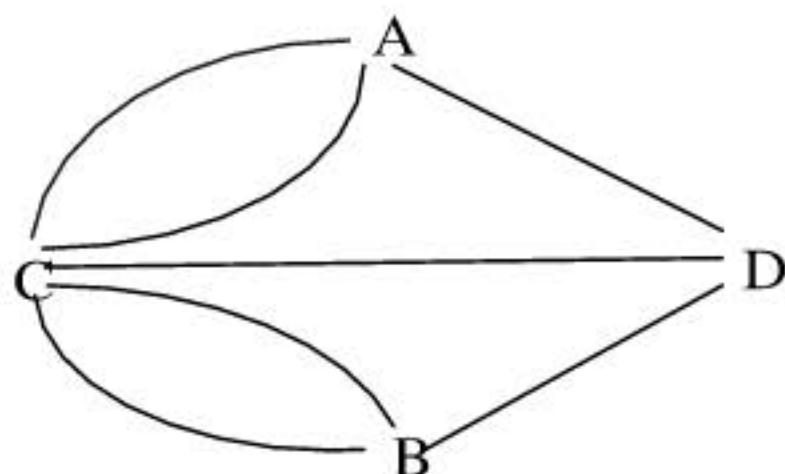
1. Hãy tìm số đỉnh, số cạnh và số bậc của mỗi đỉnh trong các đồ thị dưới đây:

a)



Hình 4.29

b)



Hình 4.30

2. Có thể tồn tại đồ thị đơn có 15 đỉnh, mỗi đỉnh có bậc bằng 5 không?

3. Hãy vẽ các đồ thị sau đây:

a) K_9

b) $K_{1,7}$

c) C_7

d) W_9

4. Tìm số cạnh trong các đồ thị

a) K_8

b) C_{12}

c) W_{15}

d) $K_{3,4}$

5. Hãy vẽ tất cả các đồ thị con có ít nhất một đỉnh của đồ thị K_3

6. Cho G là đồ thị có v đỉnh, e cạnh. Gọi M, m tương ứng là bậc lớn nhất và nhỏ nhất của các đỉnh của G . Chứng minh rằng: $v.m \leq 2.e \leq v.M$

7. Trong đồ thị G có chứa đúng hai đỉnh bậc lẻ (các đỉnh còn lại nếu có đều là bậc chẵn). Chứng minh rằng có một đường đi nối hai đỉnh bậc lẻ đó với nhau.

8. Cho đồ thị hai phía đầy đủ p đỉnh và q cạnh. Chứng minh:

$$q \leq \frac{p^2}{4}$$

Dấu “=” xảy ra khi nào?

9. Có bao nhiêu đồ thị (đơn, vô hướng) gồm 6 đỉnh và có 5 hoặc 7 cạnh?
10. Chứng minh rằng trong một đồ thị đơn vô hướng n đỉnh, luôn tồn tại hai đỉnh có bậc bằng nhau.
11. Trong một giải thi đấu thể thao có n đội tham dự và đã có $n+1$ trận đấu được tiến hành. Chứng minh rằng có một đội đã thi đấu ít nhất 3 trận.

Chương 5

BIỂU DIỄN ĐỒ THỊ TRÊN MÁY TÍNH

Một đồ thị có thể có nhiều cách biểu diễn khác nhau trên máy tính. Tùy theo tính chất của đồ thị và yêu cầu của bài toán mà ta sẽ chọn cách biểu diễn thích hợp nhất. Sau đây là một số cách biểu diễn thông dụng.

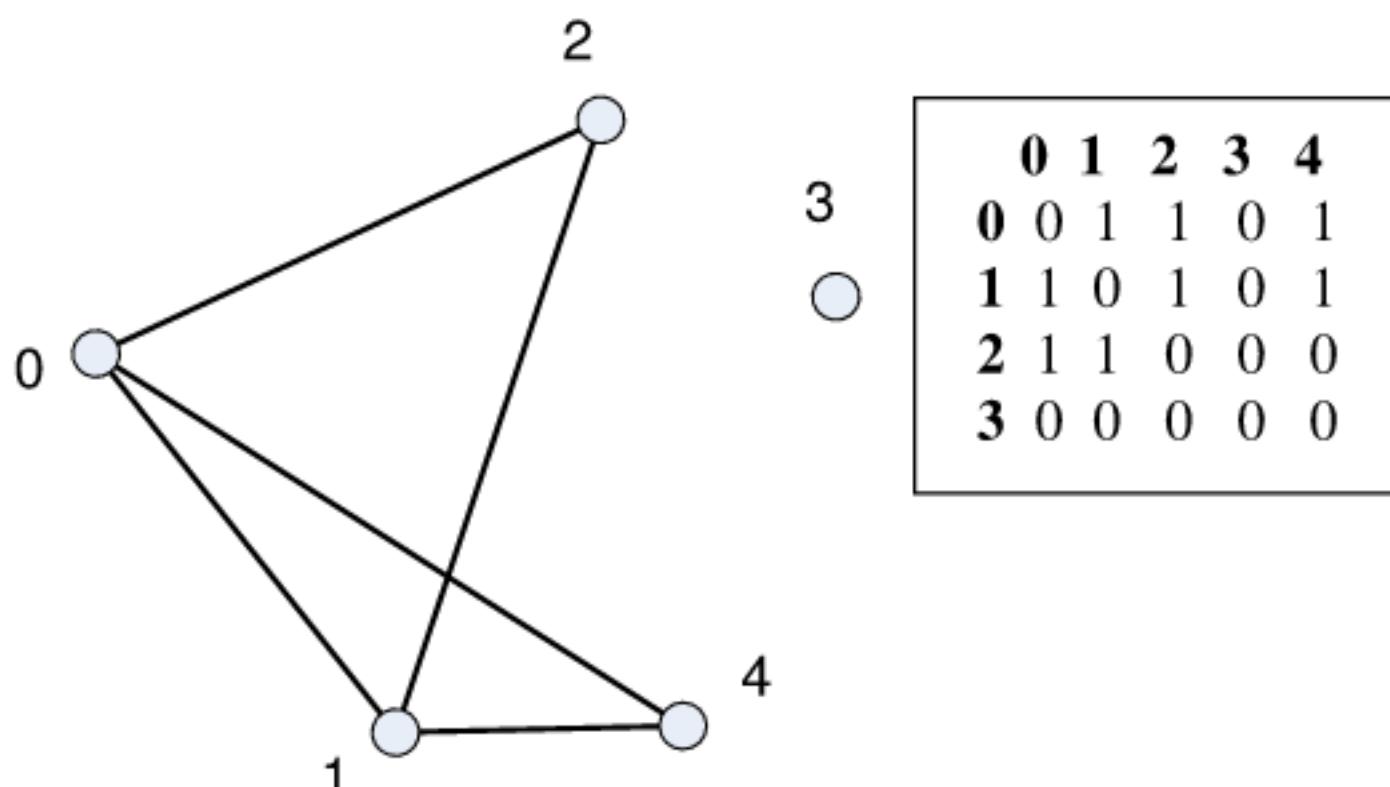
5.1. MA TRẬN KÈ, MA TRẬN TRỌNG SỐ

5.1.1. Ma trận kè

Giả sử $G = (V, E)$ là một đơn đồ thị vô hướng với tập đỉnh $V=\{0, 1, 2, \dots, n-1\}$, tập cạnh $E=\{e_0, e_1, e_2, \dots, e_{m-1}\}$. Ta gọi ma trận kè của G là: $A=\{a_{i,j}, i, j=0..n-1\}$. Trong đó:

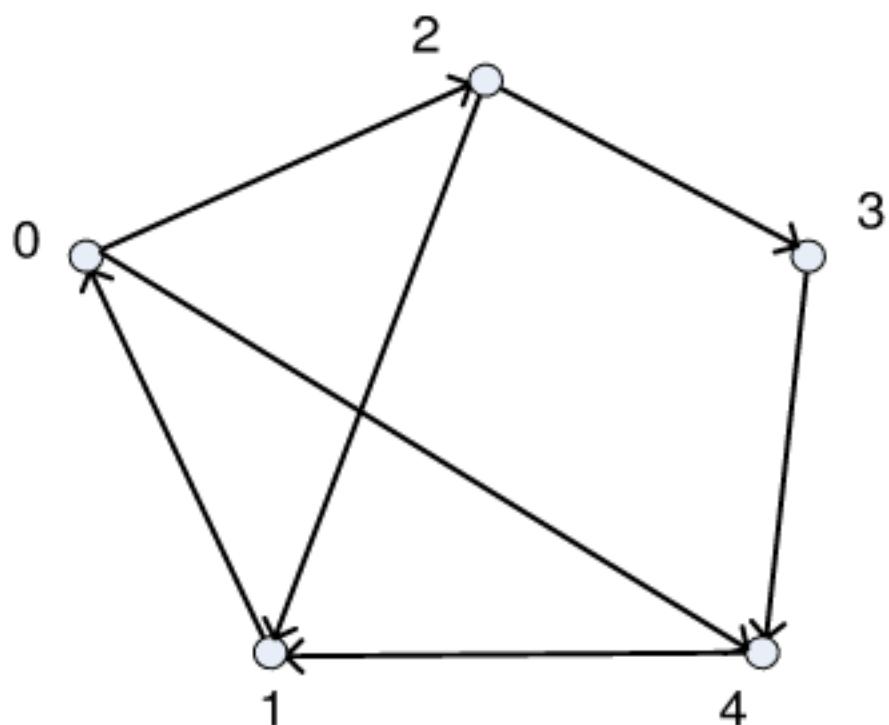
- $a_{i,j} = 0$ nếu $(i, j) \notin E$
- $a_{i,j} = 1$ nếu $(i, j) \in E$

Ví dụ:



Hình 5.1

Biểu diễn **đơn đồ thị có hướng** cũng giống như vậy với E là tập cung của đồ thị.

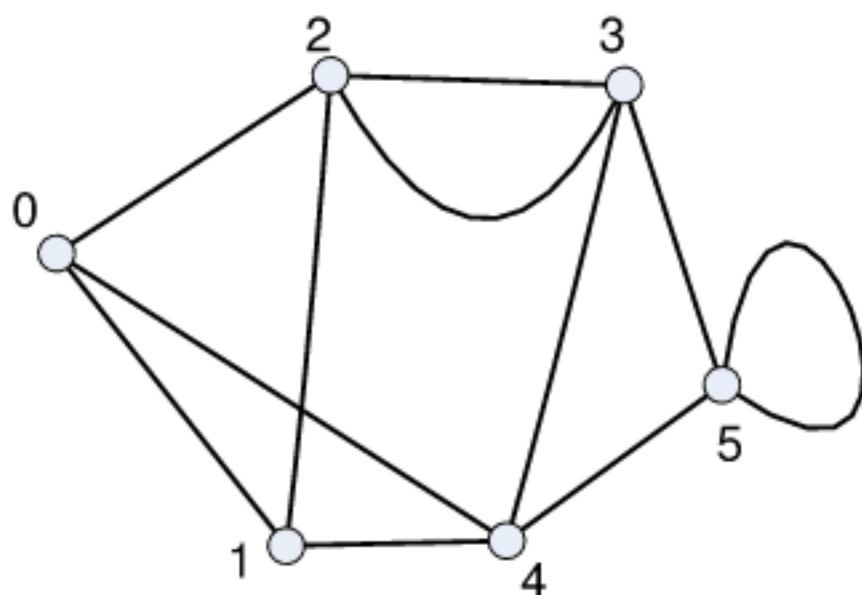


0	1	2	3	4	
0	0	0	1	0	1
1	1	1	0	0	0
2	0	1	0	1	0
3	0	0	0	0	1
4	0	1	0	0	0

Hình 5.2

Đối với **đa đồ thị** thì việc biểu diễn tương tự như đơn đồ thị và $a_{i,j}$ lưu số cạnh nối giữa hai đỉnh i và j.

Ví dụ:



0	1	2	3	4	5	
0	0	1	1	0	1	0
1	1	0	1	0	1	0
2	1	1	0	2	0	0
3	0	0	2	0	1	1
4	1	1	0	1	0	1
5	0	0	0	1	1	1

Hình 5.3

• Các tính chất của ma trận kè

Ma trận kè của một đồ thị vô hướng là đối xứng, $a[i, j] = a[j, i]$.

Nếu đồ thị vô hướng:

- Tổng dòng thứ i = Tổng cột thứ i = $\deg(i)$

Nếu đồ thị có hướng:

- Tổng dòng thứ i = $\deg^+(i)$

- Tổng cột thứ i = $\deg^-(i)$

Sử dụng ma trận kề để biểu diễn đồ thị có ưu điểm là đơn giản, trực quan, dễ cài đặt. Hạn chế của nó là đối với các đồ thị có số cạnh tương đối ít (đồ thị thưa, chẳng hạn $m < 6n$) thì việc biểu diễn này lại gây lãng phí bộ nhớ và tốn chi phí không cần thiết khi duyệt đồ thị.

5.1.2. Ma trận trọng số

Giả sử $G = (V, E)$ là một đơn đồ thị với tập đỉnh $V=\{0, 1, 2, \dots, n-1\}$, tập cạnh $E=\{e_0, e_1, e_2, \dots, e_{m-1}\}$. Ta gọi ma trận kề trọng số (gọi tắt là ma trận trọng số) của G là: $A=\{a_{i,j}, i, j=0..n-1\}$. Trong đó:

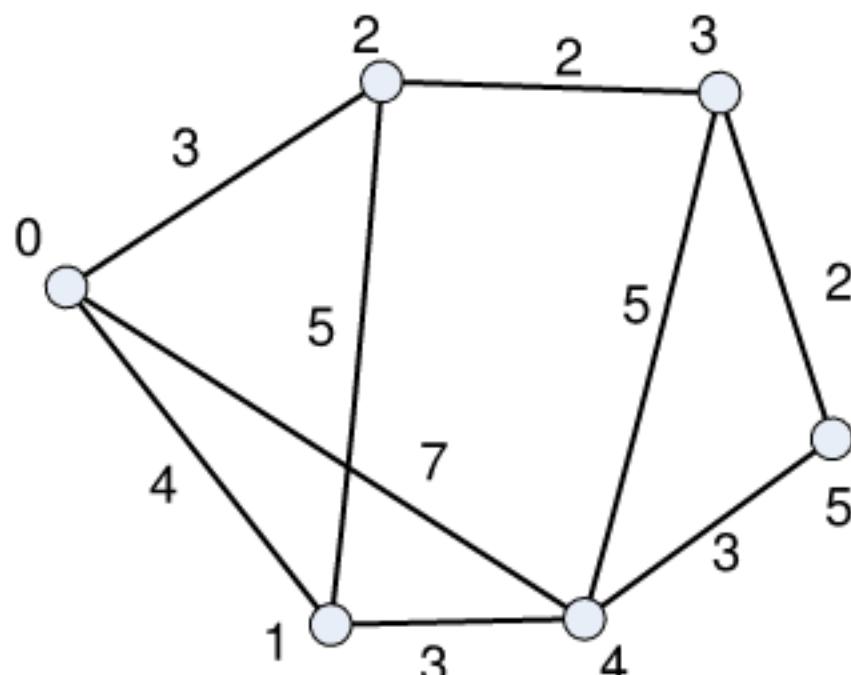
- $a_{i,j} = b$ nếu $(i, j) \notin E$
- $a_{i,j} = c_{ij}$ nếu $(i, j) \in E$

Trong đó:

c_{ij} : trọng số của cạnh nối hai đỉnh i và j .

b: giá trị đặc biệt (tùy theo từng trường hợp ứng dụng của đồ thị, ví dụ: 0, ∞ , $-\infty$).

Ví dụ:



0	1	2	3	4	5	
0	0	4	3	0	7	0
1	4	0	5	0	3	0
2	3	5	0	2	0	0
3	0	0	2	0	5	2
4	7	3	0	5	0	3
5	0	0	0	2	3	0

Hình 5.4

5.1.3. Cài đặt

Phần này hướng dẫn cài đặt chương trình cho phép nhập xuất đồ thị dưới dạng ma trận kề (hay ma trận trọng số) bằng ngôn ngữ lập trình C. Ta định nghĩa một cấu trúc GRAPH chứa các thông tin liên quan đến đồ thị (số đỉnh, ma trận trọng số hay ma trận kề của đồ thị). Chương trình gồm hai hàm chính: Hàm NhậpĐoThi() cho phép đọc một ma trận biểu

diễn của đồ thị trên tập tin, hàm XuatDoThi() cho phép in số đỉnh và ma trận lên màn hình.

Cấu trúc của tập tin đầu vào như sau:

- + Dòng đầu tiên chứa số nguyên n, cho biết số đỉnh của đồ thị
- + n dòng tiếp theo, mỗi dòng chứa n số nguyên (hoặc số thực) ứng với các phần tử trong ma trận kề (ma trận trọng số).

Ví dụ: Đối với đồ thị hình 5.4, tập tin lưu trữ như sau:

6
0 4 3 0 7 0
4 0 5 0 3 0
3 5 0 2 0 0
0 0 2 0 5 2
7 3 0 5 0 3
0 0 0 2 3 0

Chương trình cài đặt nhập xuất đồ thị bằng ngôn ngữ lập trình C:

```
#include<stdio.h>
#include<conio.h>
#define MAX 100
typedef struct
{
    int      n;
    int      a[MAX][MAX];
}GRAPH;
void NhapDoThi (GRAPH &g)
{
    FILE* f;
    f = fopen("đường dẫn đầy đủ đến tập tin", "rt");
    if (f == NULL)
```

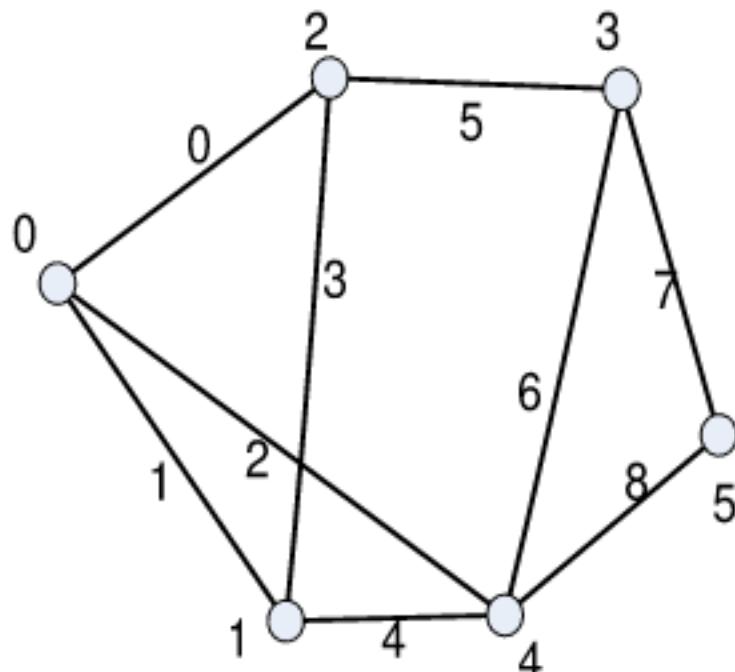
```
{  
    printf("Khong mo duoc file\n");  
    return;  
}  
  
fscanf(f, "%d", &g.n);  
for (int i=0; i<g.n; i++)  
for (int j=0; j<g.n; j++)  
    fscanf(f, "%d", &g.a[i][j]);  
fclose(f);  
}  
  
void XuatDoThi (GRAPH &g)  
{  
    printf("%d\n", g.n);  
    int i, j;  
    for (i=0; i<g.n; i++)  
    {  
        for (j=0; j<g.n; j++)  
            printf("%5d", g.a[i][j]);  
        printf("\n");  
    }  
}  
  
void main()  
{  
    GRAPH g;  
    NhapDoThi(g);  
    XuatDoThi (g);  
}
```

5.2. DANH SÁCH CẠNH (CUNG)

Đồ thị G với n đỉnh, m cạnh có thể được biểu diễn dưới dạng danh sách cạnh bằng cách lưu các cạnh $e = (u, v)$ của đồ thị trong một danh sách. Danh sách được lưu trong bộ nhớ có thể là một mảng hoặc là một danh sách liên kết.

Đối với các đồ thị thưa thì biểu diễn bằng danh sách cạnh giúp tiết kiệm không gian lưu trữ.

Ví dụ:



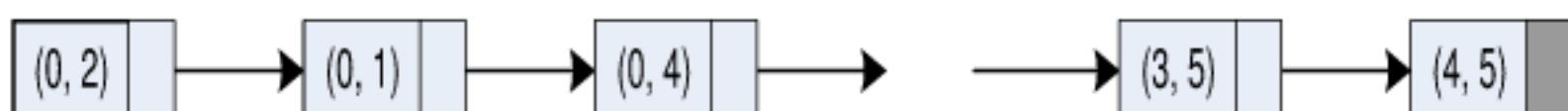
Cạnh	Đầu 1	Đầu 2
0	0	2
1	0	1
2	0	4
3	1	2
4	1	4
5	2	3
6	3	4
7	3	5
8	4	5

Hình 5.5

Cài đặt bằng mảng:

0	1	2	3	4	5	6	7	8
(0,2)	(0,1)	(0,4)	(1,2)	(1,4)	(2,3)	(3,4)	(3,5)	(4,5)

Cài đặt bằng danh sách liên kết

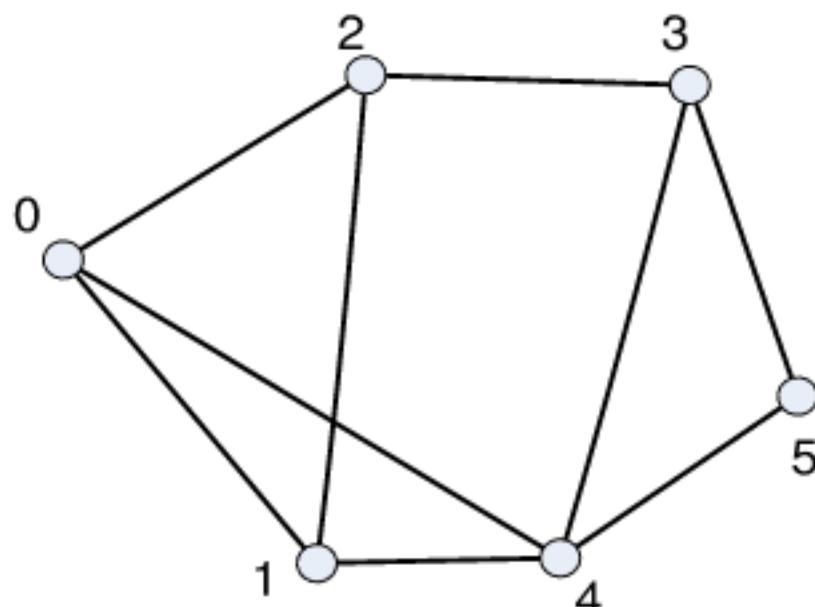


Trong trường hợp đồ thị có hướng thì mỗi phần tử của danh sách (gọi là **danh sách cung**) là một cung $e = (u, v)$. Trong đó u là đỉnh đầu, v là đỉnh cuối của cung.

5.3. DANH SÁCH KÈ

Cách biểu diễn đồ thị $G = (V, E)$ theo danh sách kè là với mỗi đỉnh V của đồ thị, ta có tương ứng một danh sách để lưu các đỉnh kè với nó. Có thể cài đặt danh sách cạnh bằng cách dùng mảng hoặc danh sách liên kết.

Ví dụ:



Đỉnh V	Các cạnh kè
0	1, 2, 4
1	0, 2, 4
2	0, 1, 3
3	2, 4, 5
4	0, 1, 3, 5
5	3, 4

Hình 5.6

• Sử dụng mảng

Ta sử dụng một mảng tên $Ke[]$ để lưu các danh sách đỉnh kè của các đỉnh trong đồ thị. Mảng $ViTri[]$ dùng để lưu vị trí bắt đầu của các danh sách đỉnh kè trong mảng $Ke[]$. Giá trị $ViTri[i]$ là vị trí bắt đầu danh sách đỉnh kè với đỉnh thứ i trong mảng $Ke[]$.

Trong ví dụ trên ta có:

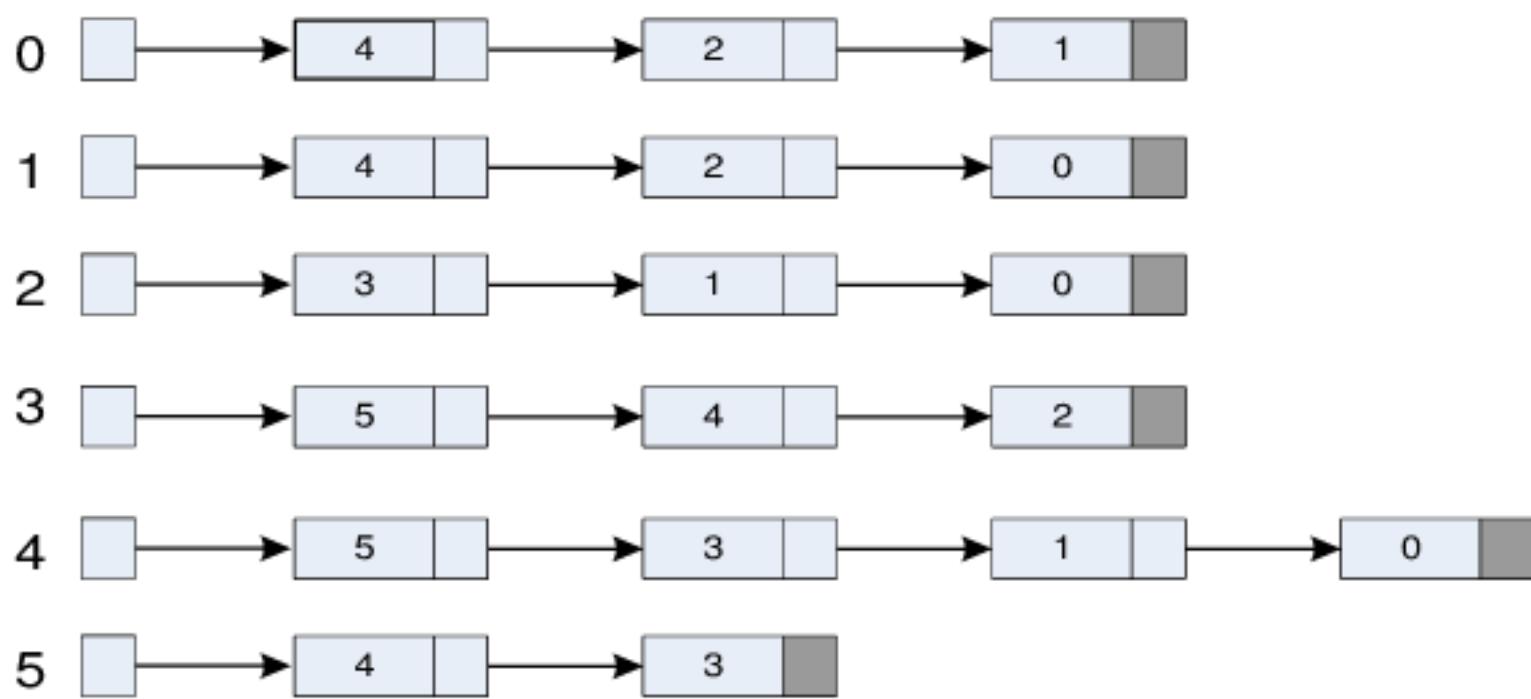
$$Ke[] = \{1, 2, 4, 0, 2, 4, 0, 1, 3, 2, 4, 5, 0, 1, 3, 5, 3, 4\}$$

$$ViTri[] = \{0, 3, 6, 9, 12, 16\}$$

Dễ thấy, số phần tử của mảng $Ke[]$ là $2m$ (với m là số cạnh của đồ thị).

• Sử dụng danh sách liên kết

Ngoài việc sử dụng mảng, người ta có thể sử dụng danh sách liên kết để biểu diễn danh sách kè. Ví dụ, với đồ thị hình 5.6 ta có thể biểu diễn như sau:



Chương trình cài đặt tạo một danh sách kề liên kết được viết bằng ngôn ngữ C++.

```
# include <iostream.h>
#include <stdlib.h>
const maxV = 99;
typedef struct Node {
    int v;
    struct Node*next;
}node;
int j, x, y, m, n, v;
node *p, *ke[maxV];
int main()
{
    cout<<"Cho so canh va so dinh cua do thi: ";
    cin>>m>>n;
    for(j=0;j<n;j++)
        ke[j]=NULL;
    for(j=1;j<=m;j++)
    {
        cout<<"Cho dinh dau, dinh cuoi cua canh "<<j<<":";
        cin>>x>>y;
        p=(node*)malloc(sizeof(node));
        p->v=y;
        p->next=ke[x];
        ke[x]=p;
    }
}
```

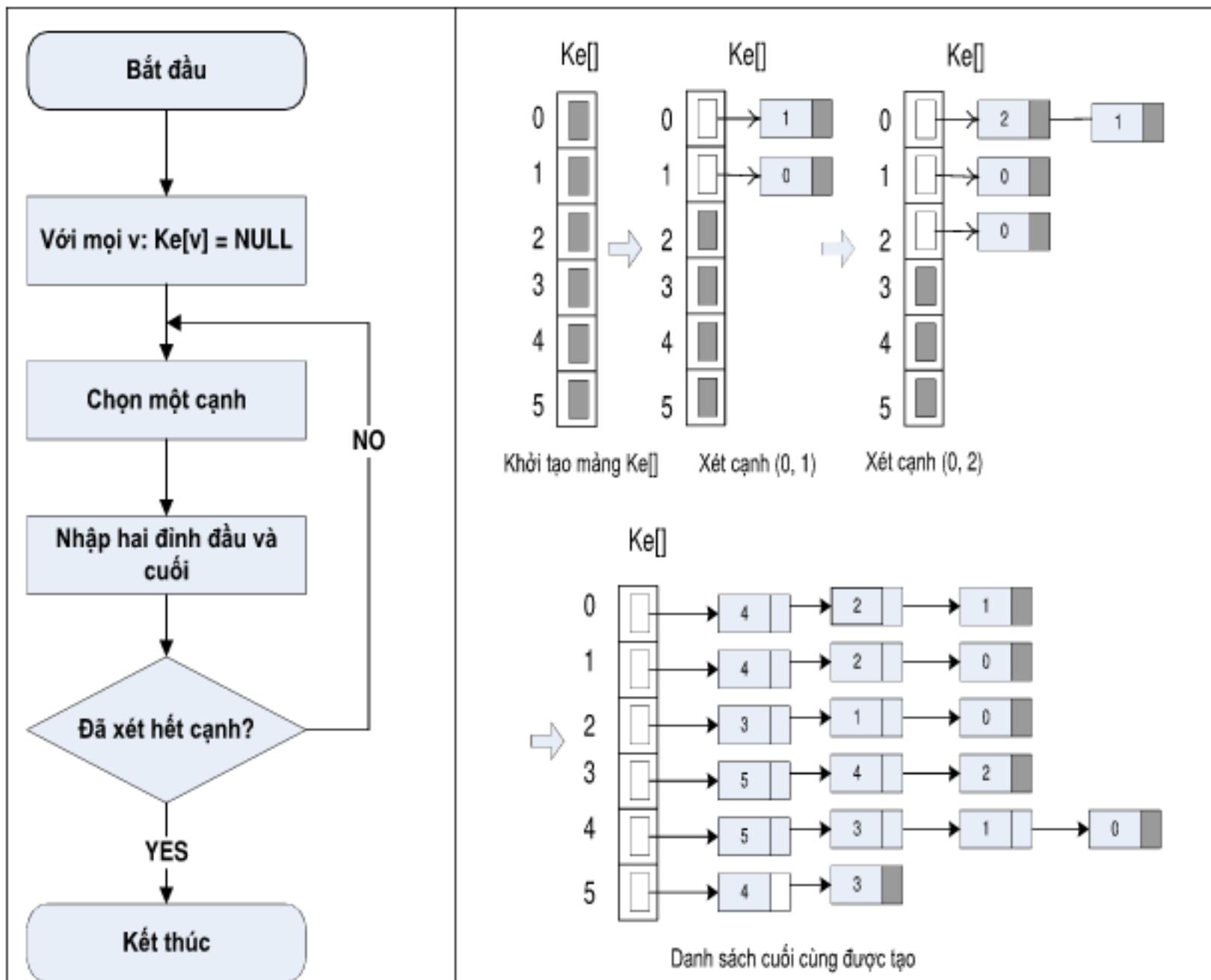
```

    cin>>x>>y;
    p = (node*)malloc(sizeof(node));
    p->v = x;
    p->next = ke[y];
    ke[y]=p;
    p = (node*)malloc(sizeof(node));
    p->v = y;
    p->next = ke[x];
    ke[x]=p;
}

cout<<"Xuat danh sach ke"; //Xuat danh sach ke da tao
for(int i=0;i<n;i++)
{
    node* q=ke[i];
    cout<<"\nDanh sach thu "<<i;
    while(q != NULL)
    {
        cout<<"\n"<<q->v;
        q=q->next;
    }
}
return 0;
}

```

Thuật toán được mô phỏng và chạy từng bước trong hình dưới đây theo đồ thị hình 5.6. Với đồ thị có n đỉnh, mảng Ke[] lưu trữ địa chỉ phần tử đầu của n danh sách liên kết. Danh sách liên kết thứ i ($0 \leq i \leq n-1$) lưu trữ k phần tử tương ứng với k đỉnh kề của nó.

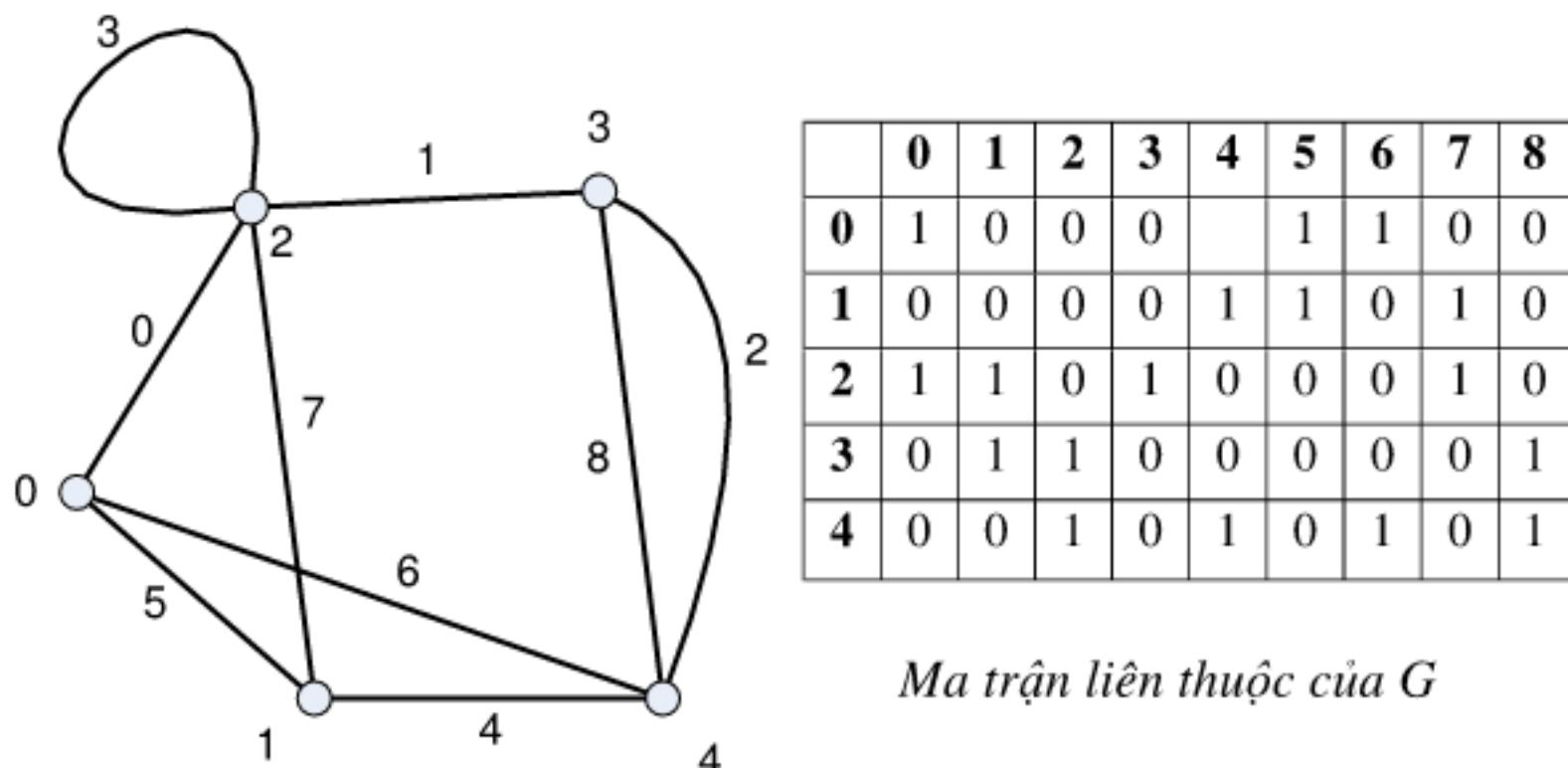


5.4. MA TRẬN LIÊN THUỘC

Đồ thị vô hướng $G = (V, E)$ với tập đỉnh $V = \{0, 1, \dots, n-1\}$, tập cạnh $E = \{e_0, e_1, e_2, \dots, e_{m-1}\}$. Ta gọi ma trận liên thuộc của G là: $B = \{b_{i,j}, i, j = \overline{0..n-1}\}$. Trong đó:

- $b_{i,j} = 1$ nếu đỉnh i liên thuộc cạnh e_j
- $b_{i,j} = 0$ nếu đỉnh i không liên thuộc e_j

Ví dụ:



Hình 5.7

- **Tính chất của ma trận liên thuộc**

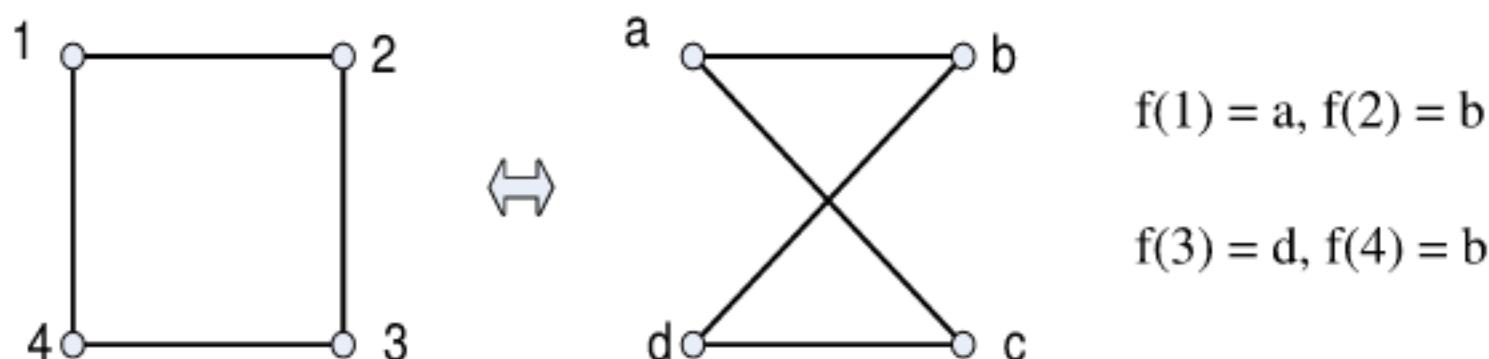
- + Mỗi cột chứa đúng hai số 1 chỉ hai đỉnh đầu của cạnh tương ứng với cột đó. Cột ứng với khuyên chứa đúng một số 1.
- + Các cột ứng với các cạnh lặp thì giống nhau.
- + Nếu đồ thị không có khuyên thì tổng hàng i là bậc của đỉnh i.

5.5. SỰ ĐĂNG CẨU CỦA ĐỒ THỊ

5.5.1. Định nghĩa

Các đồ thị đơn $G_1 = (V_1, E_1)$ và $G_2 = (V_2, E_2)$ là đăng cầu nếu có hàm song ánh: $f: V_1 \rightarrow V_2$ sao cho \forall đỉnh a và b liền kề trong $V_1 \Leftrightarrow f(a)$ và $f(b)$ liền kề trong V_2 .

Ví dụ:



Hình 5.8

5.5.2. Tính bất biến

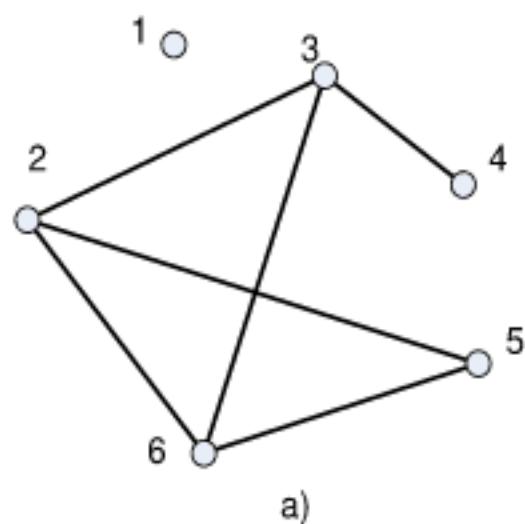
Hai đồ thị đẳng cấu bất kỳ có tính chất giống nhau (số đỉnh, số cạnh, bậc của một đỉnh,...). Người ta gọi đó là tính bất biến trong các đồ thị đẳng cấu.

Trong thực tế, các tính chất của một đồ thị có thể mở rộng ra cho mọi đồ thị đẳng cấu với nó.

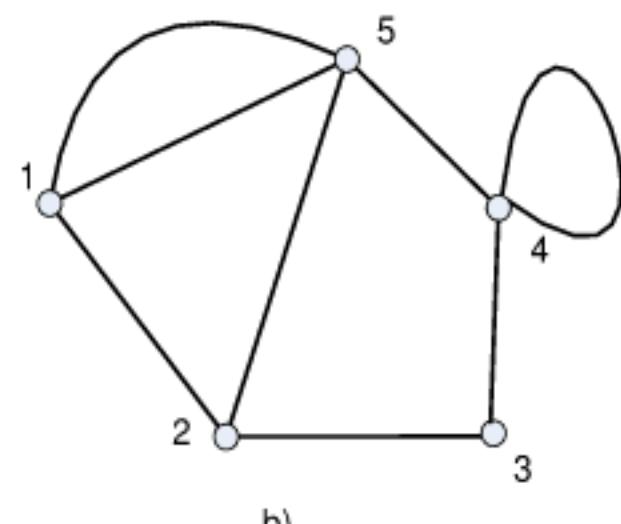
Nhận xét: Hai đồ thị đẳng cấu được biểu diễn hoàn toàn giống nhau trên máy tính.

BÀI TẬP CHƯƠNG 5

1. Xây dựng ma trận kề cho các đồ thị sau.



a)

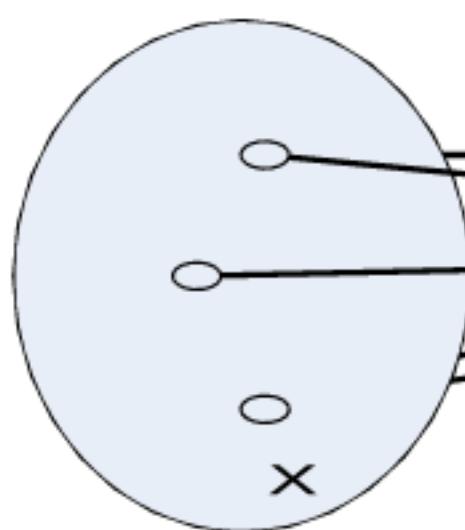


b)

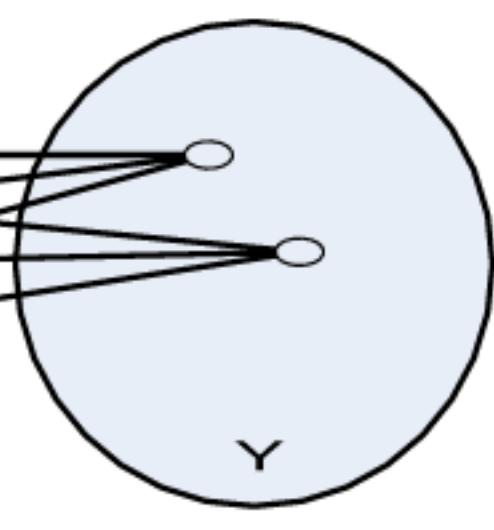
Hình 5.9

Hình 5.10

2. Xây dựng danh sách cạnh, danh sách kề cho các đồ thị sau.

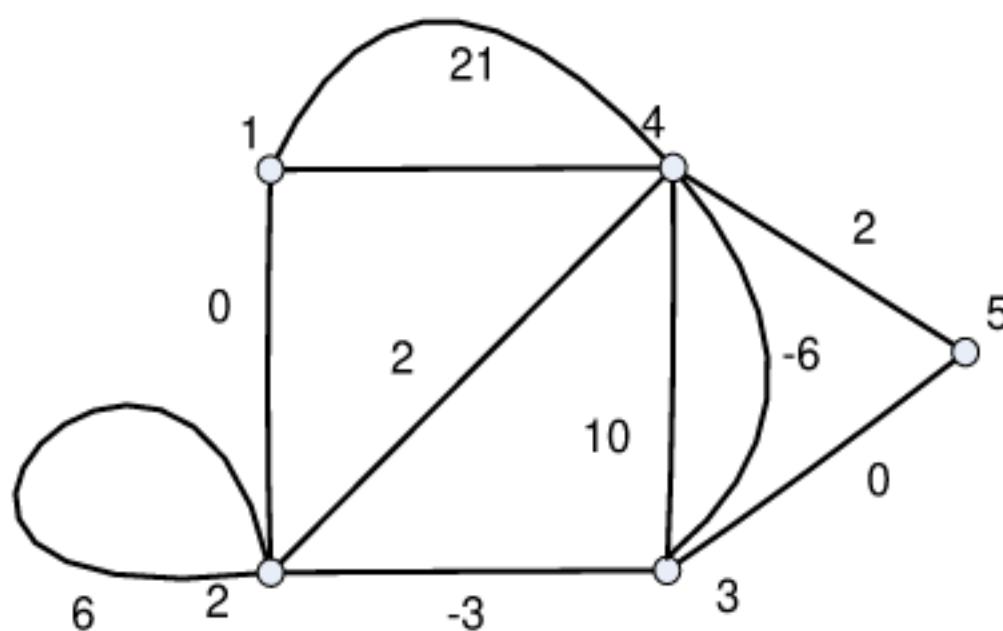


Hình 5.11



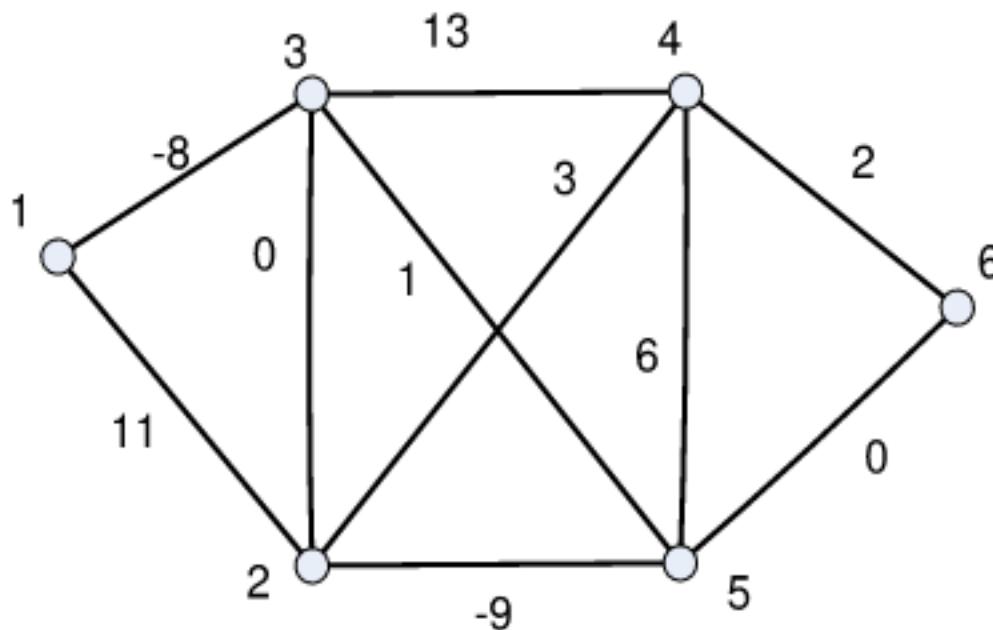
Hình 5.12

3. Xây dựng ma trận liên thuộc cho đồ thị sau.



Hình 5.13

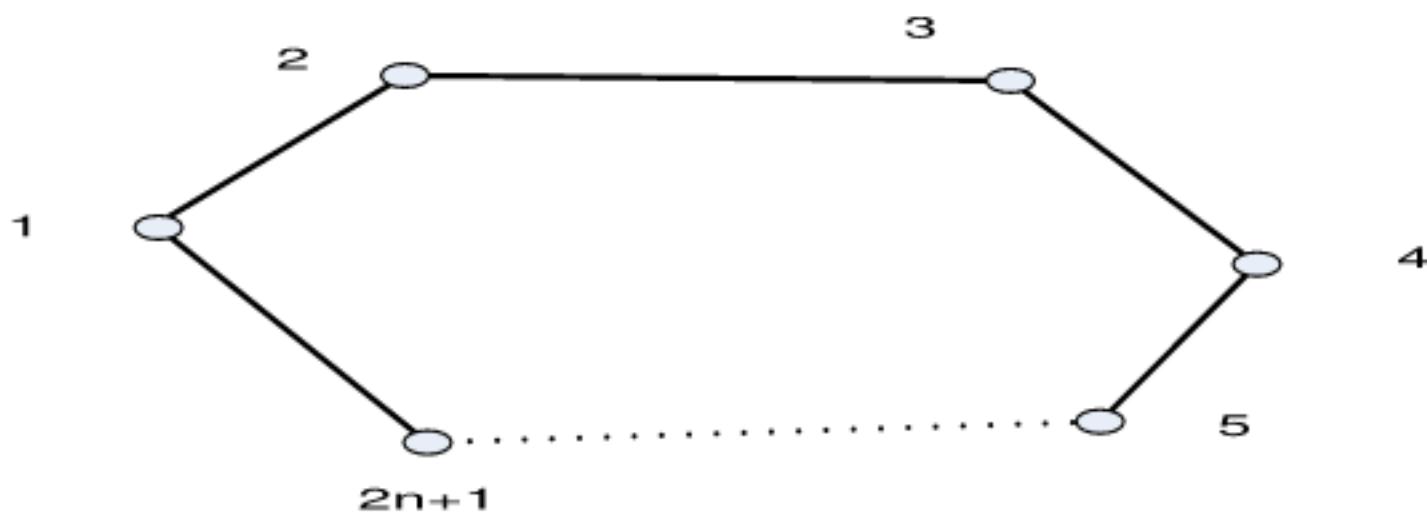
4. Xây dựng ma trận trọng số cho đồ thị sau.



Hình 5.12

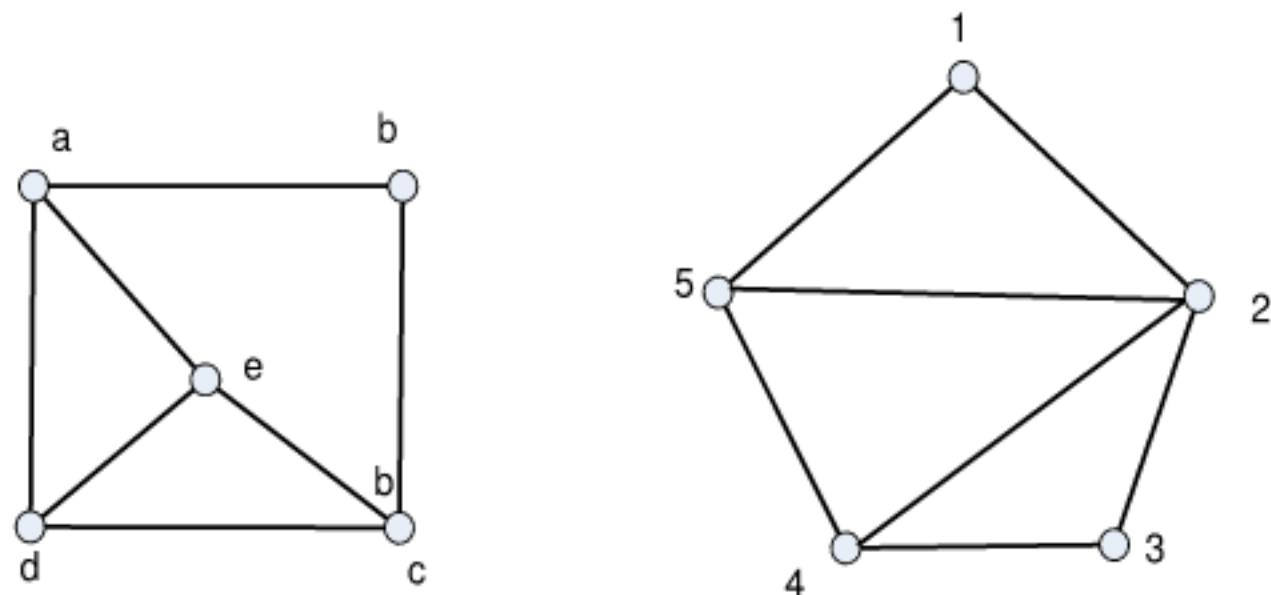
5. Hãy xác định các đồ thị sau có đằng cầu hay không?

a)



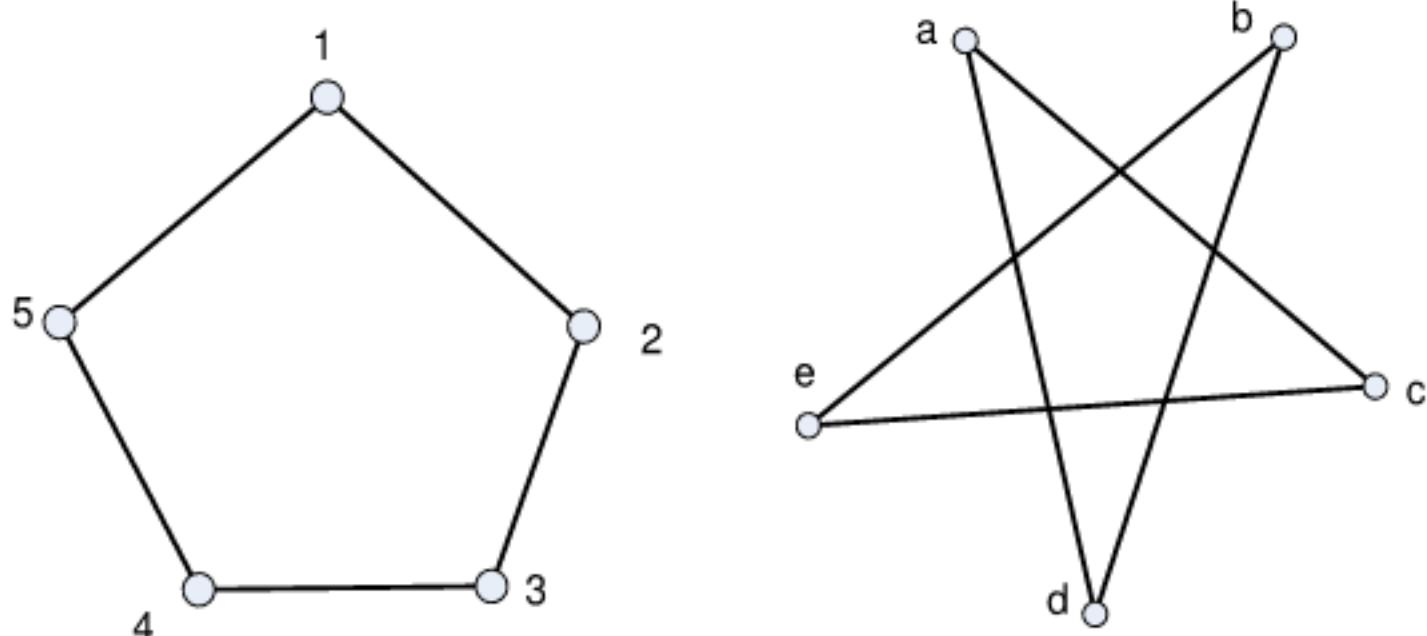
Hình 5.15

b)



Hình 5.16

c)



Hình 5.17

6. Viết chương trình nhập và xuất đồ thị theo các cấu trúc dữ liệu sau:
- Ma trận kề
 - Danh sách cạnh (cung)
 - Danh sách kề
 - Ma trận liên thuộc
 - Ma trận trọng số.
7. Viết chương trình chuyển đổi các cách biểu diễn đồ thị:
- Từ danh sách cạnh sang ma trận kề và ngược lại.
 - Từ ma trận kề sang ma trận liên thuộc và ngược lại.
 - Từ ma trận liên thuộc sang danh sách cạnh và ngược lại.
8. Viết chương trình nhập xuất đồ thị từ tập tin dưới dạng danh sách kề. Thực hiện các chức năng sau:
- Tìm đỉnh có bậc cao nhất.
 - Tìm bậc nhỏ nhất của các đỉnh.
 - Tính tổng bậc các đỉnh của đồ thị.
 - Đếm số đỉnh chẵn và đỉnh lẻ của đồ thị.
9. Viết chương trình nhập xuất đồ thị từ tập tin dưới dạng ma trận kề (ma trận trọng số). Thực hiện các chức năng sau:
- Kiểm tra đồ thị là vô hướng hay có hướng.
 - Nếu đồ thị vô hướng, xuất ra bậc của các đỉnh. Nếu đồ thị có hướng, xuất ra bán bậc vào/bán bậc ra của các đỉnh.

Chương 6

DUYỆT ĐỒ THỊ

Trong chương 6 sẽ khảo sát các phương pháp duyệt đồ thị. Các phương pháp này sẽ được ứng dụng để giải quyết các bài toán khác nhau trên đồ thị. Những bài toán được xem xét trực tiếp ở đây là bài toán tìm đường đi và kiểm tra tính liên thông của đồ thị.

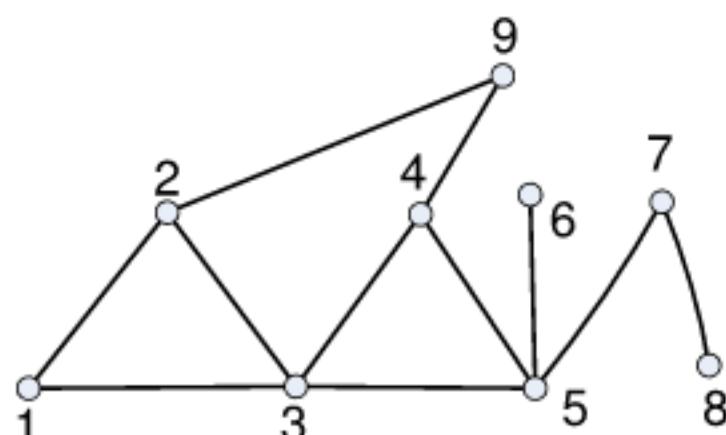
Duyệt đồ thị là quá trình đi qua tất cả các đỉnh của đồ thị sao cho mỗi đỉnh của nó được viếng thăm đúng một lần. Nội dung của chương này sẽ đề cập hai phương pháp duyệt đồ thị: duyệt theo chiều sâu (Depth First Search) và duyệt theo chiều rộng (Breadth First Search).

6.1. DUYỆT ĐỒ THỊ THEO CHIỀU SÂU

6.1.1. Phương pháp duyệt

Bắt đầu duyệt từ một đỉnh v nào đó của đồ thị. Tiếp theo, chọn u là một đỉnh tùy ý kề với v (với đồ thị có hướng thì u là đỉnh cuối, v là đỉnh đầu của cung uv) và lặp lại quá trình này với u cho đến khi không tìm được đỉnh kề (chưa được duyệt) tiếp theo nữa thì quay trở lại đỉnh ngay trước đó để tìm qua nhánh khác.

Ví dụ: Thực hiện duyệt chiều sâu đồ thị sau:



Hình 6.1

Nếu bắt đầu từ đỉnh 1 thì thứ tự các đỉnh được duyệt theo chiều sâu như sau:

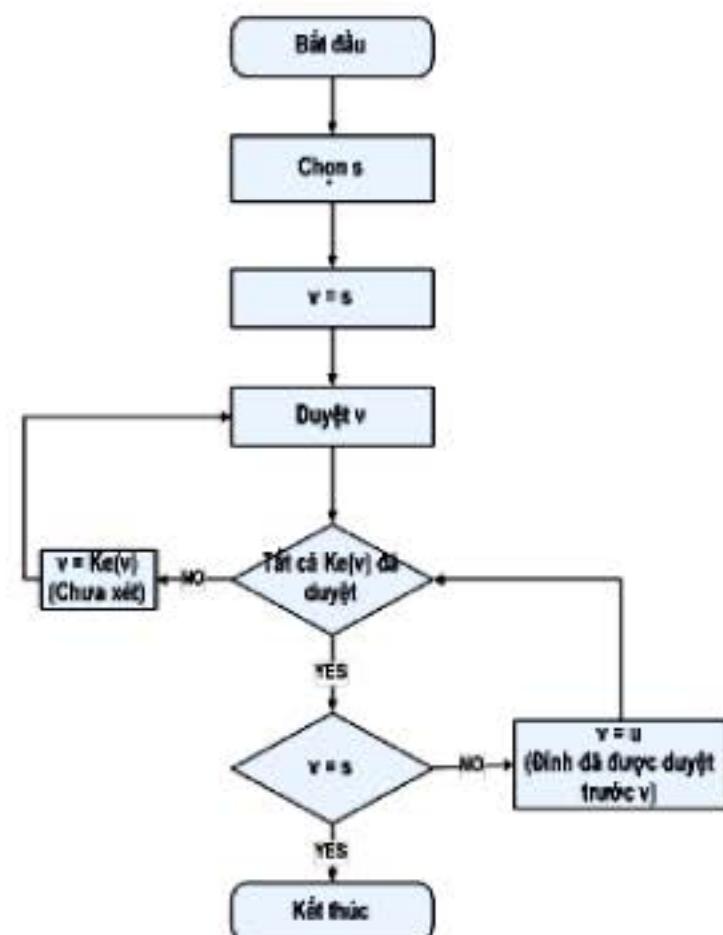
$1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6$, lùi lại $5 \rightarrow 7 \rightarrow 8$ lùi lại $4 \rightarrow 9$

6.1.2. Cài đặt thuật toán

Với nguyên lý duyệt đồ thị như vậy, ta có thể cài đặt theo hai cách: đệ quy và không đệ quy (sử dụng cấu trúc dữ liệu ngăn xếp – Stack).

6.1.2.1. Thuật toán đệ quy

1. Lấy s là một đỉnh của đồ thị.
2. Đặt $v = s$.
3. Duyệt đỉnh v .
4. Nếu \forall đỉnh kề của v đều được duyệt, đặt $v =$ đỉnh đã được duyệt trước đỉnh v . Nếu $v = s$ thì đi đến 6, ngược lại trở lại 4.
5. Chọn u là đỉnh kề chưa được duyệt của v , đặt $v = u$, trở lại 3.
6. Kết thúc.

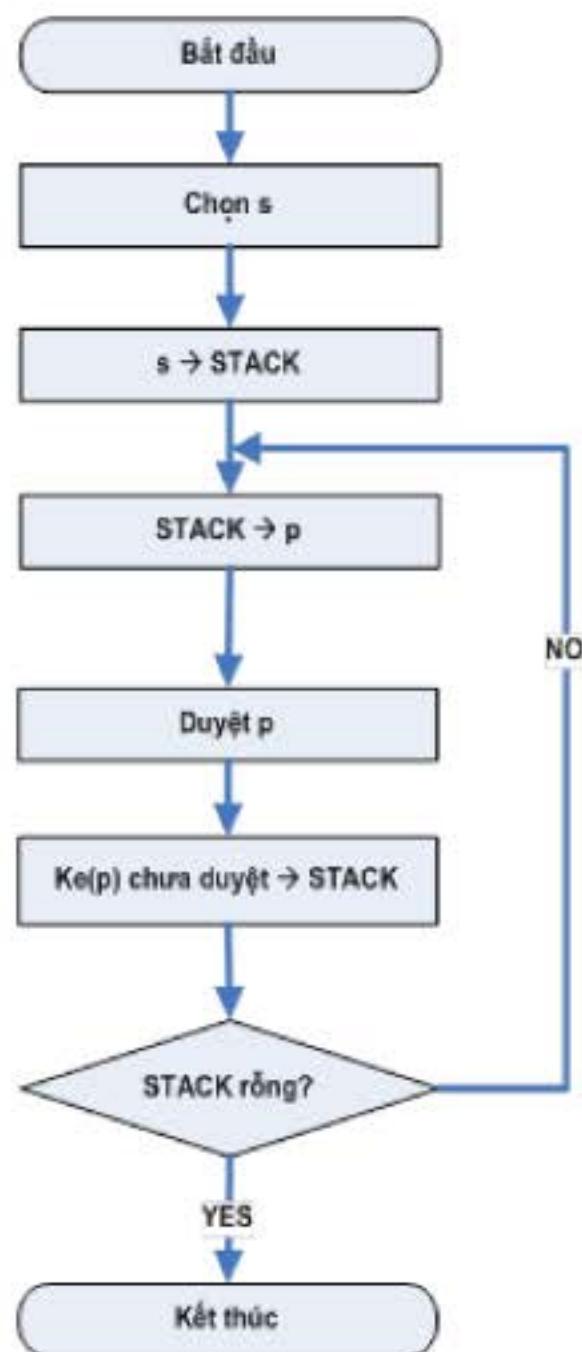


Dưới đây là giải thuật duyệt đồ thị theo chiều sâu bằng phương pháp đệ quy được viết bằng mã giả. Trong giải thuật, ta khai báo một mảng $Ke[]$ để lưu danh sách kề của các đỉnh, và mảng $ChuaXet[]$ để đánh dấu một đỉnh nào đó đã được duyệt hay chưa.

```
/* Khai báo các biến ChuaXet, Ke */
DFS(v)
{
    Duyệt đỉnh (v);
    ChuaXet[v] = 0; /*Dánh dấu đã xét đỉnh v*/
    for (u ∈ Ke(v))
        if (ChuaXet[u]) DFS(u);
}
void main()
{
    /* Nhập đồ thị, tạo mảng Ke */
    for (v ∈ V) ChuaXet[v] = 1; /*Khởi tạo cờ cho đỉnh*/
    for (v ∈ V)
        if (ChuaXet[v]) DFS(v);
}
```

6.1.2.2. Thuật toán không đệ quy (Sử dụng ngăn xếp - Stack)

1. Lấy s là một đỉnh của đồ thị.
2. Đặt s vào STACK.
3. Nếu STACK rỗng đi đến 7.
4. Lấy đỉnh p từ STACK.
5. Duyệt đỉnh p.
6. Đặt các đỉnh kề của p chưa được xét (*chưa từng có mặt trong STACK*) vào STACK, trở lại 3.
7. Kết thúc.



Cũng theo ví dụ hình 6.1, quá trình thực thi của giải thuật được minh họa dưới đây.

Bắt đầu duyệt từ đỉnh 1. Nội dung ngăn xếp sẽ thay đổi như sau:

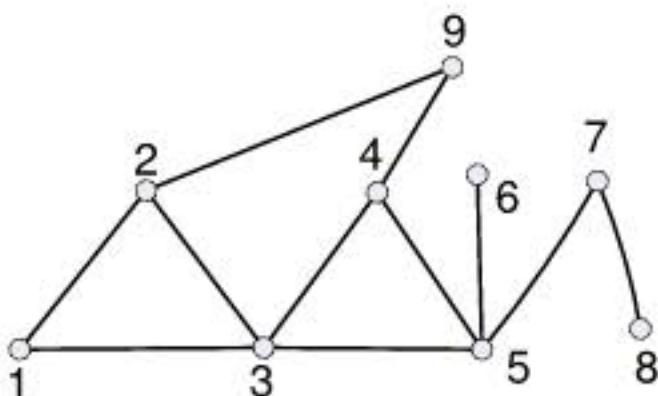
Begin	STACK									End
\emptyset	1	2	9	4	5	7	8	6	3	\emptyset
	3	3	3	3	6	6	3			
					3	3				

6.2. DUYỆT ĐỒ THỊ THEO CHIỀU RỘNG

6.2.1. Phương pháp duyệt

Bắt đầu từ một đỉnh v bất kỳ, duyệt đỉnh v sau đó lưu tất cả những đỉnh kề với v vào hàng đợi. Tiếp theo, lấy đỉnh u ở đầu hàng đợi, duyệt u và làm tương tự giống như với v.

Ví dụ:



Hình 6.2

Nếu bắt đầu duyệt từ đỉnh 1 thì ta có cách duyệt theo chiều rộng như sau:

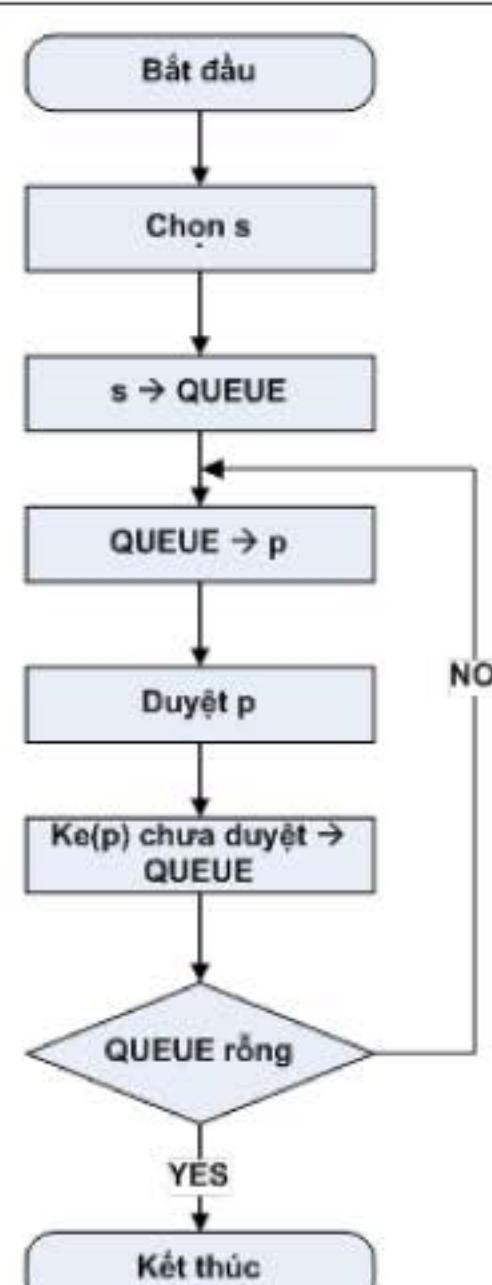
1 2 3 9 4 5 6 7 8

6.2.2. Cài đặt thuật toán

Tương tự như phương pháp duyệt đồ thị theo chiều sâu, phương pháp duyệt theo chiều rộng có thể được cài đặt theo hai cách: đệ quy và không đệ quy (sử dụng cấu trúc dữ liệu hàng đợi - Queue)

6.2.2.1. Thuật toán không đệ quy (Sử dụng hàng đợi – Queue)

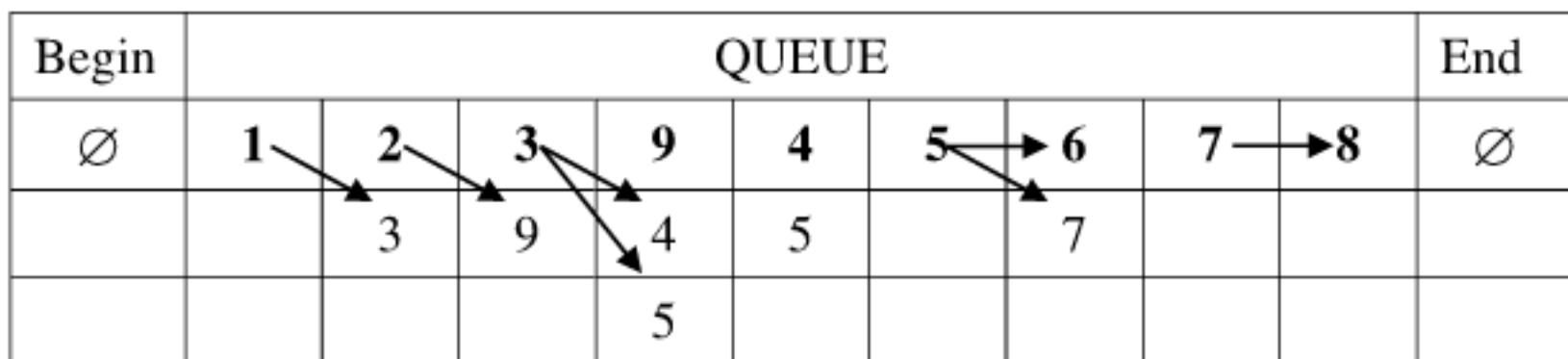
1. Lấy s là một đỉnh của đồ thị.
2. Đặt s vào QUEUE.
3. Lặp nếu QUEUE chưa rỗng.
4. Lấy đỉnh p từ QUEUE.
5. Duyệt đỉnh p .
6. Đặt các đỉnh kề của p chưa được xét (*chưa từng có mặt trong QUEUE*) vào QUEUE.
7. Kết thúc.



Chúng ta dễ dàng nhận thấy, giải thuật duyệt đồ thị theo chiều rộng không đệ quy hoàn toàn tương tự với giải thuật duyệt đồ thị theo chiều sâu không đệ quy. Điểm khác biệt là ta thay thế kiểu dữ liệu ngăn xếp bằng kiểu dữ liệu hàng đợi.

Dưới đây là minh họa quá trình thực thi của giải thuật cho ví dụ ở hình 6.2.

Bắt đầu duyệt từ đỉnh 1, nội dung hàng đợi sẽ thay đổi như sau:



Dưới đây là phần cài đặt bằng mã giả cho thuật toán. Tương tự như thuật toán duyệt theo chiều sâu, ta cần khai báo một mảng $Ke[]$ và mảng $ChuaXet[]$. Ngoài ra, ta cần xây dựng một cấu trúc QUEUE để sử dụng cho thuật toán.

```

/* Khai báo các biến ChuaXet, Ke */
BFS(v)
{
    QUEUE =  $\emptyset$ ;
    QUEUE  $\Leftarrow$  v;
    ChuaXet[v] = 0; /*Đánh dấu đã xét đỉnh v*/
    while (QUEUE  $\neq$   $\emptyset$ )
    {
        p  $\Leftarrow$  QUEUE;
        Duyệt đỉnh p;
        for (u  $\in$  Ke(p))
            if (ChuaXet[u])
            {
                QUEUE  $\Leftarrow$  u;
                ChuaXet[u] = 1;
            }
    }
}
```

```

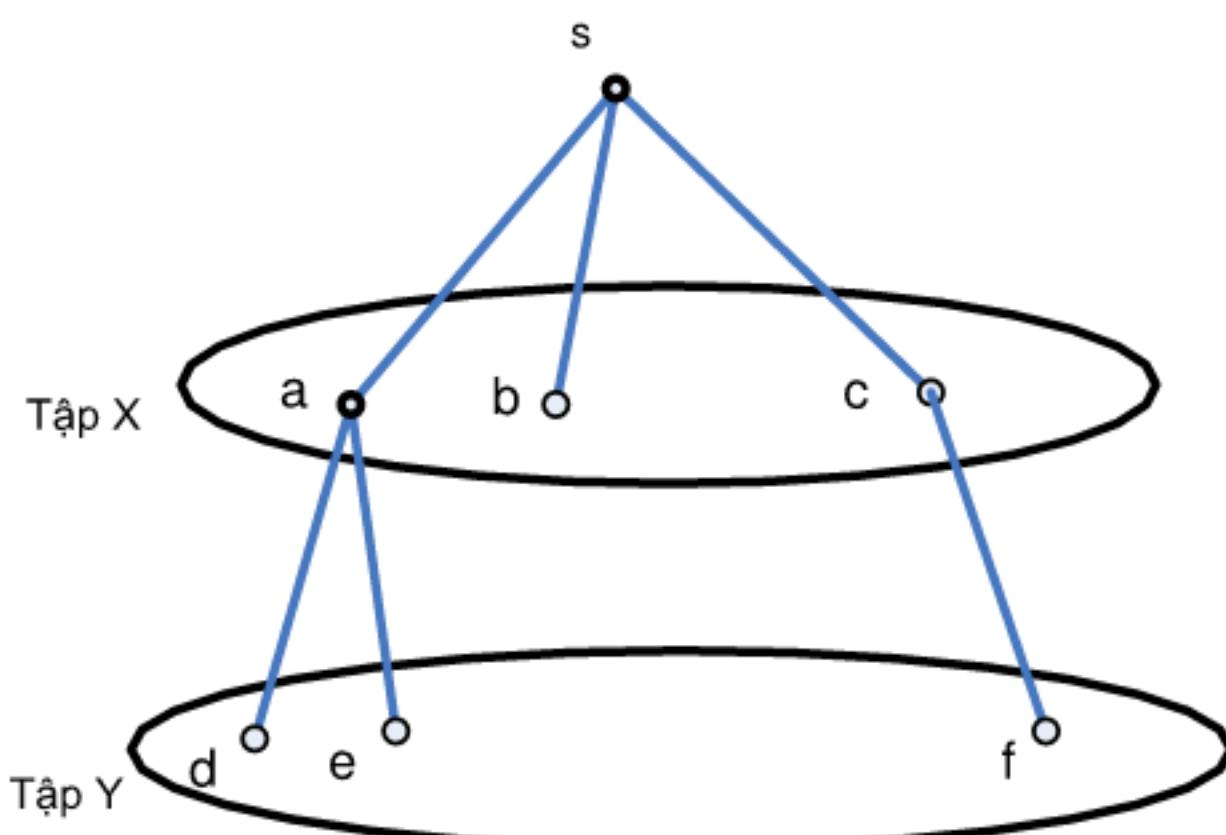
ChuaXet[u] = 0; /*Đánh dấu đã xét đỉnh */
}
}
}

main()
/* Nhập đồ thị, tạo biến Ke */
{
    for (v ∈ V) ChuaXet[v] = 1; /* Khởi tạo cờ cho đỉnh */
    for (v ∈ V)
        if (ChuaXet[v]) BFS(v);
}

```

6.2.2.2. Thuật toán đê quy

Thuật toán duyệt đồ thị theo chiều rộng đê quy còn gọi là thuật toán loang. Ta sử dụng hai tập hợp X, Y để lưu trữ thông tin các đỉnh được duyệt theo cùng cấp. Chẳng hạn, theo hình bên dưới, sau khi đỉnh s được duyệt, các đỉnh tiếp theo được duyệt (xem là cùng cấp) là a, b, c. Tương tự, các đỉnh được duyệt kế tiếp là d, e, f.



Hình 6.3

Thuật toán được trình bày bằng mã giả:

Bước 1: Khởi tạo

- Bắt đầu từ đỉnh s. Đánh dấu đỉnh s, các đỉnh khác s đều chưa bị đánh dấu;
- $X = \{s\}$, $Y = \emptyset$.

Bước 2: Lặp lại cho đến khi $X = \emptyset$

- Gán $Y = \emptyset$;
- Với mọi đỉnh $u \in X$.

Xét tất cả các đỉnh v kề với u mà chưa bị đánh dấu. Với mỗi đỉnh đó:

- Đánh dấu v;
- Lưu đường đi, đỉnh liền trước v trong đường đi từ s $\rightarrow v$ là u;
- Đưa v vào tập Y.

- Gán $X = Y$.

6.3. TÌM ĐƯỜNG ĐI VÀ KIỂM TRA TÍNH LIÊN THÔNG

Các thuật toán duyệt đồ thị ở trên có thể áp dụng để giải quyết bài toán tìm đường đi giữa hai đỉnh và kiểm tra tính liên thông của đồ thị.

6.3.1. Bài toán tìm đường đi

Cho đồ thị G, s và t là hai đỉnh tùy ý của đồ thị. Hãy tìm đường đi từ s đến t.

• **Nhận xét**

Như ta đã biết, các hàm duyệt đồ thị DFS(v) và BFS(v) cho phép thăm các đỉnh của đồ thị G. Do đó, nếu ta bắt đầu duyệt từ đỉnh s và sau khi kết thúc thuật toán mà giá trị ChuaXet[t] = 1 thì điều đó có nghĩa là *không có đường đi từ s đến t*, còn nếu ChuaXet[t] = 0 thì có nghĩa là *có đường đi từ s đến t*. Ngoài ra, ta sẽ thêm biến Truoc[] để lưu vết. Sau đây là các giải thuật viết bằng mã giả để tìm đường đi bằng cách duyệt theo chiều sâu, và chiều rộng.

• **Thuật toán tìm đường đi theo chiều sâu**

```
/* Khai báo các biến ChuaXet, Ke */
```

```
DFS(v)
```

```
{
```

```

Duyệt đỉnh (v);
ChuaXet[v] = 0;
for (u ∈ Ke(v))
    if (ChuaXet[u])
    {
        Truoc[u] = v; /* Lưu vết*/
        DFS(u);
    }
}
main()
{ /* Nhập đồ thị, tạo biến Ke */
    for (v ∈ V) ChuaXet[v] = 1; /*Khởi tạo cờ cho đỉnh*/
    /* Nhập các đỉnh s, t */
    DFS(s);
    if(ChuaXet[t])
        /*Xuất “có đường đi từ s đến t”*/
    else
        /* Xuất “Không có đường đi từ s đến t”*/
}

```

- **Thuật toán tìm đường đi theo chiều rộng**

```

/* Khai báo các biến ChuaXet, Ke, QUEUE */
BFS(v);
{
    QUEUE = ∅;
    QUEUE ← v;
    ChuaXet[v] = 0;
    while (QUEUE ≠ ∅)
    {
        p ← QUEUE;

```

```

Duyệt đỉnh p;
    for (u ∈ Ke(p))
        if (ChuaXet[u])
            {
                QUEUE ← u;
                ChuaXet[u] = 0;
                Truoc[u] = p; /* Lưu vết*/
            }
    }
main()
{
    /* Nhập đồ thị, tạo biến Ke */
    for (v ∈ V) ChuaXet[v] = 1; /* Khởi tạo cờ cho đỉnh */
    /* Nhập các đỉnh s, t */
    BFS(s);
    if(ChuaXet[t])
        /* Xuất “có đường đi từ s đến t”*/
    else
        /* Xuất “Không có đường đi từ s đến t“*/
}

```

6.3.2. Bài toán kiểm tra tính liên thông

Bài toán này được sử dụng để tính số thành phần liên thông của đồ thị và xác định những đỉnh thuộc cùng một thành phần liên thông.

- **Nhận xét**

Các hàm DFS(s) và BFS(s) cho phép ta duyệt tất cả các đỉnh thuộc cùng một thành phần liên thông với s. Vì vậy, số thành phần liên thông bằng số lần gọi đến hàm này. Trong thuật toán kiểm tra tính liên thông của đồ thị dưới đây, ta thêm biến inconnect để đếm số thành phần liên thông của đồ thị, và biến index[v] để lưu chỉ số của thành phần liên thông chứa đỉnh v. Mỗi khi gọi đến hàm DFS(v) hay BFS(v) sẽ gán index[v] = inconnect.

- Thuật toán kiểm tra tính liên thông theo chiều sâu

```
/* Khai báo các biến ChuaXet, Ke, index*/
DFS(v);
{
    Duyệt đỉnh (v);
    index[v] = inconnect;
    ChuaXet[v] = 0;
    for (u ∈ Ke(v))
        if (ChuaXet[u]) DFS(u);
}
main()
{ /* Nhập đồ thị, tạo biến Ke */
    for (v ∈ V) ChuaXet[v] = 1; /* Khởi tạo cờ cho đỉnh */
    inconnect = 0;
    for (v ∈ V)
        if (ChuaXet[v])
        {
            inconnect++;
            DFS(v);
        }
}
```

- Thuật toán kiểm tra tính liên thông theo chiều rộng

```
/* Khai báo các biến toàn cục ChuaXet, Ke, QUEUE, index */
BFS(v);
{
    QUEUE = 0;
    QUEUE ← v;
    ChuaXet[v] = 0;
```

```

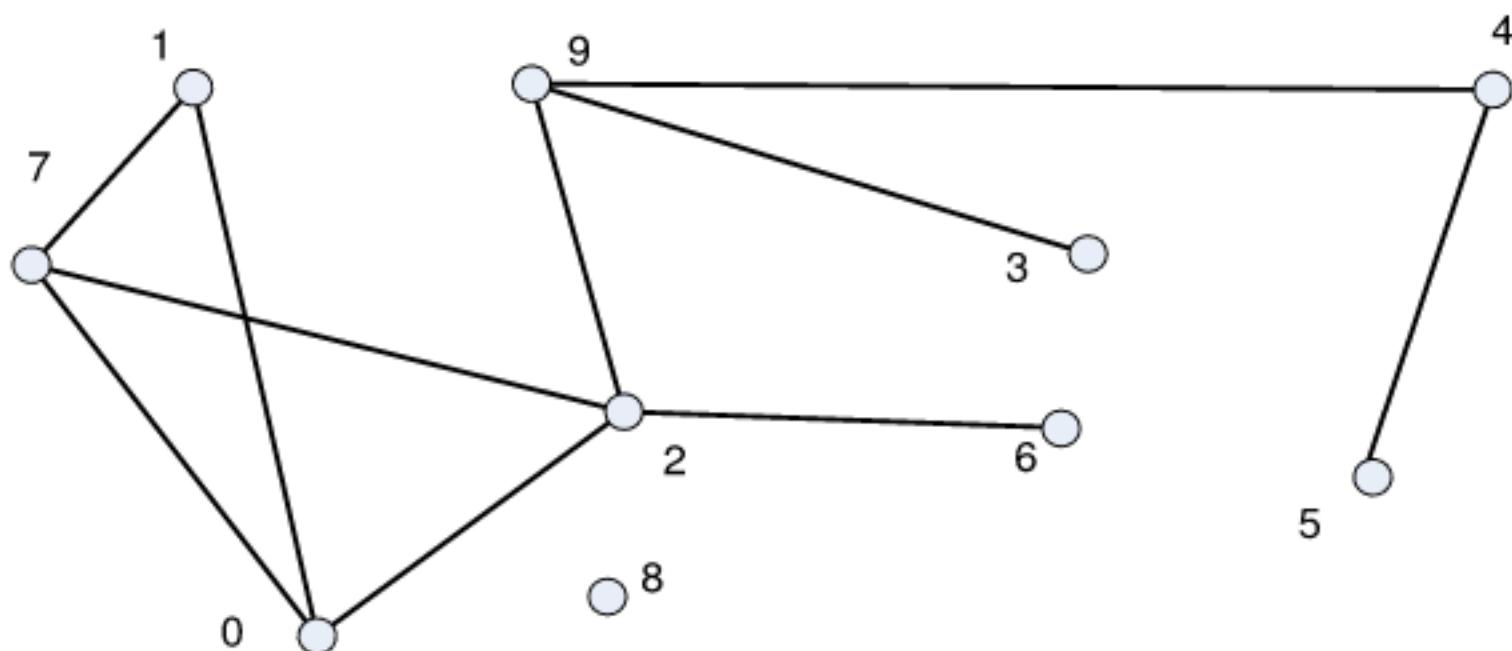
while (QUEUE ≠ 0) {
    p ← QUEUE;
    Duyệt đỉnh p;
    index[p] = inconnect;
    for (u ∈ Ke(p))
        if (ChuaXet[u])
            {
                QUEUE ← u;
                ChuaXet[u] = 0;
            }
    }
main()
{
    for (v ∈ V) ChuaXet[v] = 1;
    inconnect = 0;
    for (v ∈ V)
        if (ChuaXet[v])
            {
                inconnect ++;
                BFS(v);
            }
}

```

BÀI TẬP CHƯƠNG 6

1. Cho đồ thị vô hướng, sử dụng thuật toán duyệt đồ thị, hãy trình bày các bước duyệt tất cả các đỉnh của đồ thị sau:

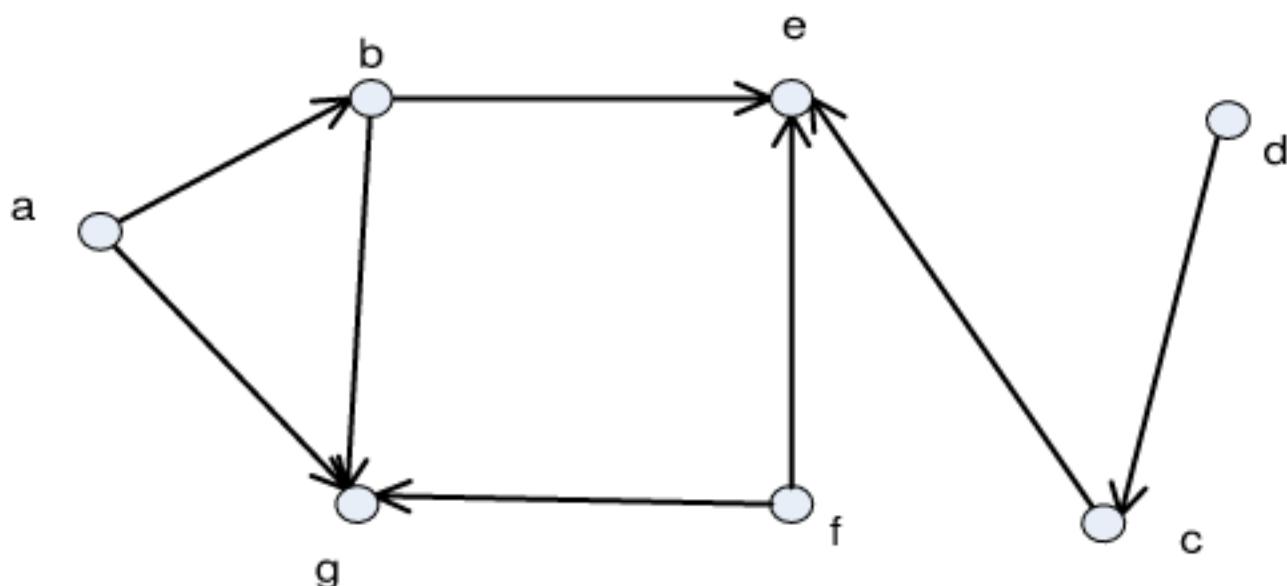
- a) Duyệt theo chiều sâu.
- b) Duyệt theo chiều rộng.



Hình 6.4

2. Cho đồ thị có hướng, sử dụng thuật toán duyệt đồ thị, hãy trình bày các bước duyệt tất cả các đỉnh của đồ thị sau:

- a) Duyệt theo chiều sâu.
- b) Duyệt theo chiều rộng.



Hình 6.5

3. Viết chương trình thực hiện duyệt đồ thị theo chiều sâu bằng phương pháp đệ quy và không đệ quy.

4. Viết chương trình thực hiện duyệt đồ thị theo chiều rộng bằng phương pháp đệ quy và không đệ quy.
5. Áp dụng hàm duyệt đồ thị theo chiều sâu, hãy tìm tất cả các cạnh là cầu trên đồ thị vô hướng $G = (V, E)$.
6. Áp dụng hàm duyệt đồ thị theo chiều rộng, viết chương trình tìm đường đi giữa hai đỉnh của đồ thị.
7. Áp dụng hàm duyệt đồ thị theo chiều sâu, viết chương trình kiểm tra tính liên thông của đồ thị. Cho biết các đỉnh trong cùng một thành phần liên thông.
8. Sử dụng ngôn ngữ lập trình C, C++ hoặc C#, viết chương trình kiểm tra xem một đồ thị đơn vô hướng có phải là đồ thị vòng hay không.
9. Cho một bảng hình chữ nhật chia thành $M \times N$ ô vuông. Mỗi ô vuông ghi một số nguyên dương (trong khoảng từ 1 đến 255). Một miền của bảng là tập hợp tất cả các ô có cùng giá trị và có thể đi được sang nhau bằng cách đi qua các ô có chung cạnh. Diện tích của mỗi miền là số ô thuộc miền đó. Hãy viết chương trình đếm số miền của bảng và cho biết diện tích của miền lớn nhất.

Ví dụ:

Bảng kích thước 5×4 . Có 6 miền.

Diện tích miền lớn nhất là: 5.

1	1	3	3
2	1	3	1
2	2	3	1
2	1	1	3
1	1	1	3

10. Một nhóm gồm N lập trình viên được đánh số từ 0 đến $N-1$. Một số người trong họ có địa chỉ email của nhau. Khi biết một thông tin nào mới, họ có thể gửi mail để trao đổi thông tin với nhau. Giả sử bạn có email của tất cả N lập trình viên này và biết rõ mối quan hệ giữa họ. Bạn có một thông tin quan trọng cần báo cho tất cả N lập trình viên này biết. Viết chương trình chỉ ra một số ít nhất các lập trình viên mà bạn cần cho họ biết thông tin, sao cho những người đó có thể thông báo cho tất cả những người còn lại thông tin của bạn.

Yêu cầu:

Dữ liệu đầu vào của chương trình dưới dạng file text có cấu trúc như sau:

- + Dòng đầu tiên chứa số lập trình viên N ($N < 500$).
- + Dòng thứ i trong N dòng tiếp theo chứa danh sách các lập trình viên mà người thứ i biết địa chỉ email của họ.

Dữ liệu đầu ra của dưới dạng file text có cấu trúc như sau:

- + Dòng đầu ghi số người ít nhất mà bạn cần cho họ biết thông tin.
- + Dòng thứ 2 ghi chỉ số của những người đó.

Ví dụ:

FileInput.inp	FileChooser.out
6	3
2 3	1 4 6
1	
5	
4	
1	

Chương 7

ĐỒ THỊ EULER VÀ ĐỒ THỊ HAMILTON

Chương này sẽ khảo sát hai dạng đồ thị được ứng dụng phổ biến trong việc giải quyết các bài toán bằng lý thuyết đồ thị.

7.1. ĐỒ THỊ EULER

Ở chương 4, chúng ta đã biết đến bài toán 7 chiếc cầu ở thành phố Konigsber. Lời giải mà nhà toán học Leonhard Euler đưa ra năm 1736 có thể xem là một ứng dụng đầu tiên của lý thuyết đồ thị. Từ bài toán này, người ta đã đưa ra các khái niệm như chu trình Euler, đường đi Euler và đồ thị Euler.

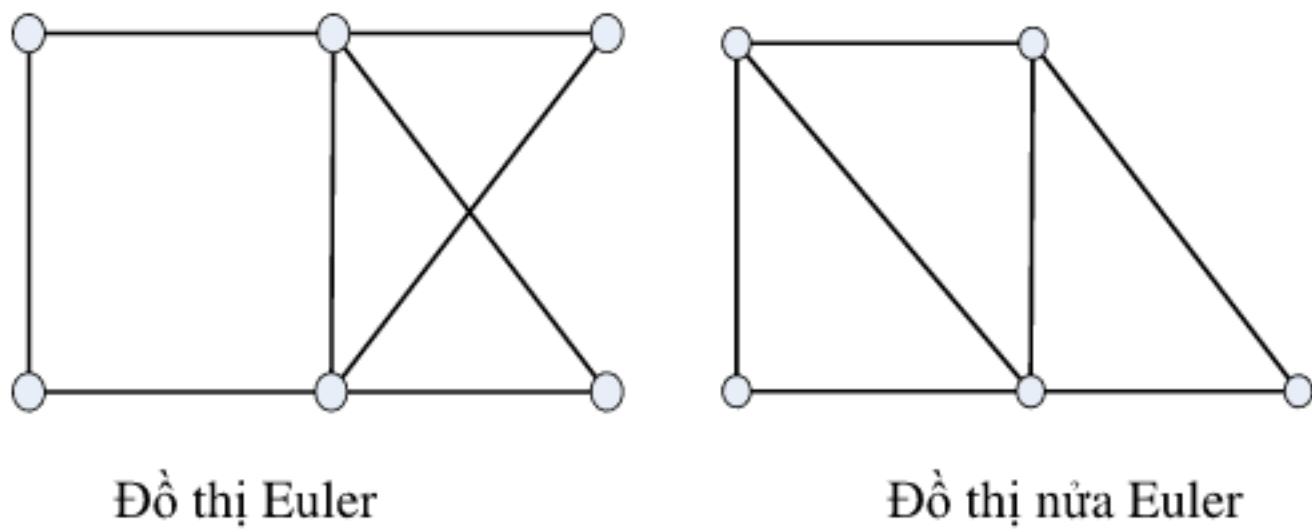
7.1.1. Định nghĩa

Giả sử G là đơn (đa) đồ thị vô (có) hướng, ta có các định nghĩa sau:

Chu trình Euler trong G là chu trình đơn đi qua tất cả các cạnh của đồ thị. Nếu G có chu trình Euler thì G được gọi là **đồ thị Euler**.

Đường đi Euler trong G là đường đi đơn qua tất cả các cạnh của đồ thị. Nếu G có đường đi Euler thì G được gọi là **đồ thị nửa Euler**.

Ví dụ:



Hình 7.1

7.1.2. Định lý Euler

✓ **Định lý 1**

Đồ thị vô hướng, liên thông $G = (V, E)$ có chu trình Euler khi và chỉ khi mọi đỉnh của G đều có bậc chẵn.

Chứng minh

- Trước hết ta chứng minh bở đề sau: “*Cho đồ thị $G = (V, E)$, nếu mọi đỉnh $u \in V$ có $\deg(u) \geq 2$ thì G có chu trình*”

Thực vậy, xét đường đi từ một đỉnh v_0 bất kỳ: $v_0 \rightarrow v_1 \rightarrow v_2 \rightarrow \dots$. Vì bậc của tất cả các đỉnh đều không bé hơn 2 nên nếu v_i là một đỉnh mới của đường đi này thì khi có đường đi vào v_i thì nhất định có đường đi ra khỏi v_i . Do số đỉnh của đồ thị là một số hữu hạn, nên đường đi này sẽ quay về một đỉnh cũ v_k nào đó, khi đó ta sẽ có ít nhất một chu trình (đpcm).

- Chứng minh định lý
 - *Chứng minh điều kiện cần: Nếu G có chu trình Euler thì mọi đỉnh của G đều có bậc chẵn.*

Đầu tiên ta gán trọng số của tất cả các đỉnh là 0. Xét chu trình Euler của đồ thị, mỗi khi đi qua một đỉnh thì tăng trọng số đỉnh đó lên 2. Do chu trình sau khi đi qua tất cả các đỉnh sẽ quay lại đỉnh xuất phát nên trọng số (bậc) của tất cả các đỉnh sẽ là chẵn.

- *Chứng minh điều kiện đủ: Nếu mọi đỉnh của G đều có bậc chẵn thì G có chu trình Euler.*

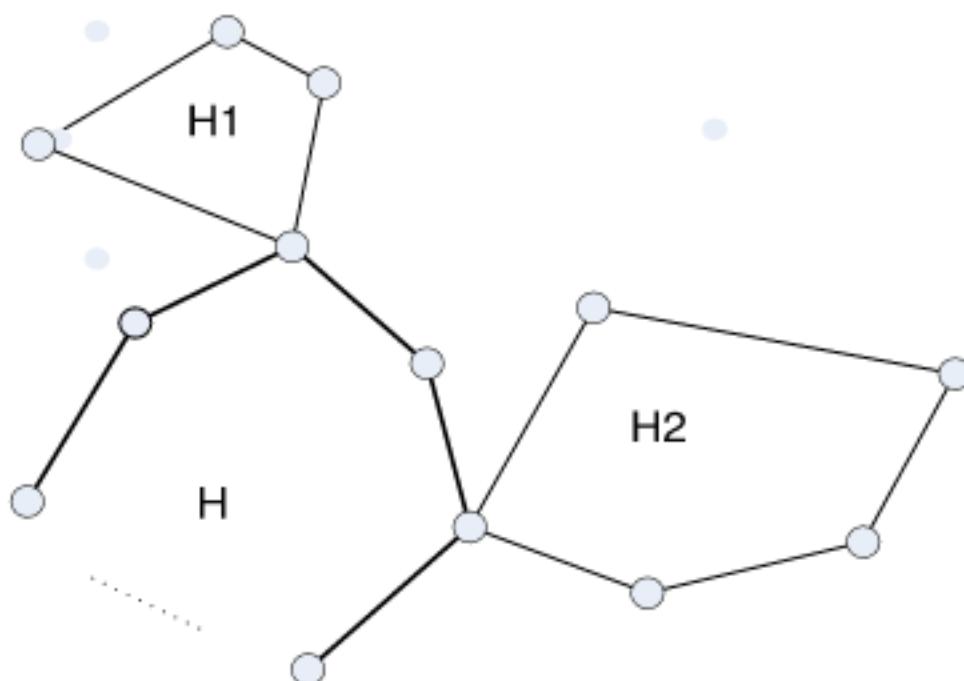
Ta chứng minh quy nạp theo số cạnh của đồ thị. Do G liên thông và $\forall u \in V$ đều có bậc chẵn nên ta có thể suy ra $\deg(u) \geq 2 \ \forall u$. Theo bở đề trên suy ra G có chu trình H nào đó. Xét hai trường hợp:

- + Nếu H đi qua tất cả các cạnh của G thì H là chu trình Euler;
- + Trường hợp ngược lại. Xét đồ thị $G' = G \setminus H$ gồm các thành phần liên thông thỏa mãn điều kiện của định lý Euler với số đỉnh ít hơn. Suy ra trong mỗi thành phần liên thông của G' có chu trình Euler tương ứng H_1, H_2, \dots . Từ các chu trình này, ta xây dựng được chu trình Euler cho G như sau:

Đi theo chu trình H ;

Tại các đỉnh thuộc H mà không có lặp trong G' , đi theo chu trình H_i ;

Quay trở lại H đi tiếp.



Hình 7.2

Cứ tiếp tục như vậy, ta sẽ thu được chu trình Euler cho đồ thị G (đpcm).

✓ **Định lý 2**

Đồ thị vô hướng, liên thông $G = (V, E)$ có đường đi Euler mà không có chu trình Euler khi và chỉ khi G có đúng hai đỉnh bậc lẻ.

✓ **Định lý 3**

Đồ thị có hướng, liên thông yếu $G = (V, E)$ có chu trình Euler khi và chỉ khi mọi đỉnh của G có bán bậc vào bằng bán bậc ra: $\deg^+(v) = \deg^-(v), \forall v \in V$.

⇒ Khi G (có hướng) có chu trình Euler thì nó liên thông mạnh.

✓ **Định lý 4**

Đồ thị có hướng, liên thông yếu $G = (V, E)$ có đường đi Euler nhưng không có chu trình Euler khi và chỉ khi G tồn tại duy nhất hai đỉnh $u, v \in V$ sao cho $\deg^+(u) - \deg^-(u) = \deg^-(v) - \deg^+(v) = 1$, và tất cả các đỉnh còn lại có bán bậc vào bằng bán bậc ra.

Việc chứng minh các định lý này dành cho người đọc xem như là bài tập.

7.1.3. Giải thuật xây dựng chu trình Euler

Đầu vào: Đồ thị G được cho bằng danh sách kề (Mảng Ke()).

Kết quả trả về: Chu trình Euler CE.

u là đỉnh xuất phát, dùng một biến cài đặt theo cấu trúc STACK để lưu tạm thời các đỉnh trong quá trình duyệt đồ thị.

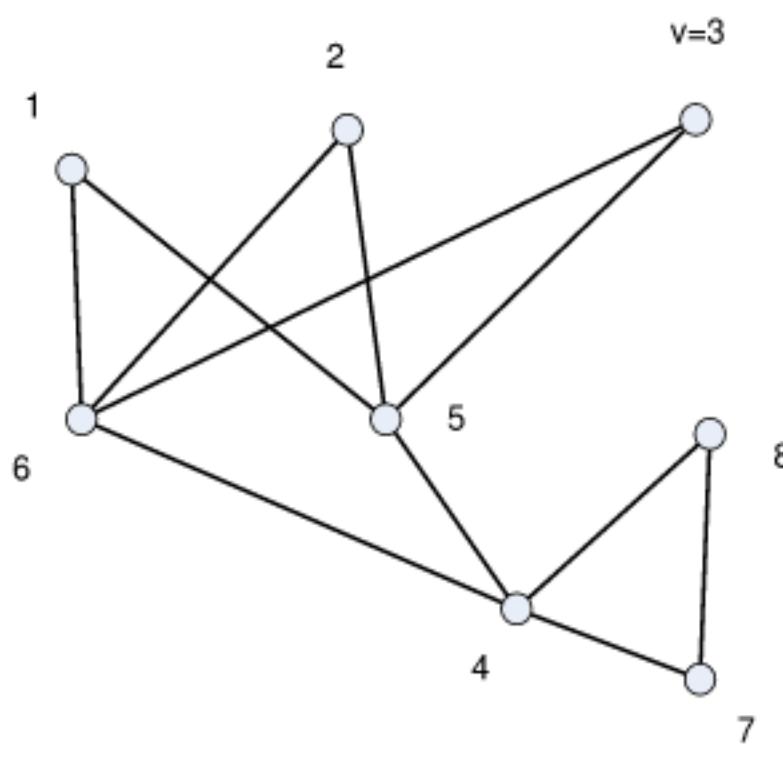
```

// Chuong trinh Euler_Cycle;
main()
{
    STACK = ∅;
    CE = ∅; /* CE - Chu trình Euler */
    Chọn u là 1 đỉnh bất kỳ của đồ thị;
    STACK ← u;
    while (STACK != ∅)
    {
        x = top(STACK);
        if (Ke(x) != ∅)
        {
            y = Đỉnh đầu trong danh sách Ke(x);
            STACK ← y;
            Ke(x) = Ke(x) \ {y};
            Ke(y) = Ke(y) \ {x}; /* Bỏ cạnh (x,y) */
        }
        else
        {
            x ← STACK;
            CE ← x;
        }
    }
}

```

Ví dụ:

Tìm chu trình Euler cho đồ thị sau:



Định v	Ke(v)
1	6, 5
2	5, 6
3	6, 5
4	6, 5, 7, 8
5	4, 3, 2, 1
6	4, 3, 2, 1
7	4, 8
8	4, 7

Hình 7.3

Áp dụng thuật toán ta có:

Input: Ke(v)

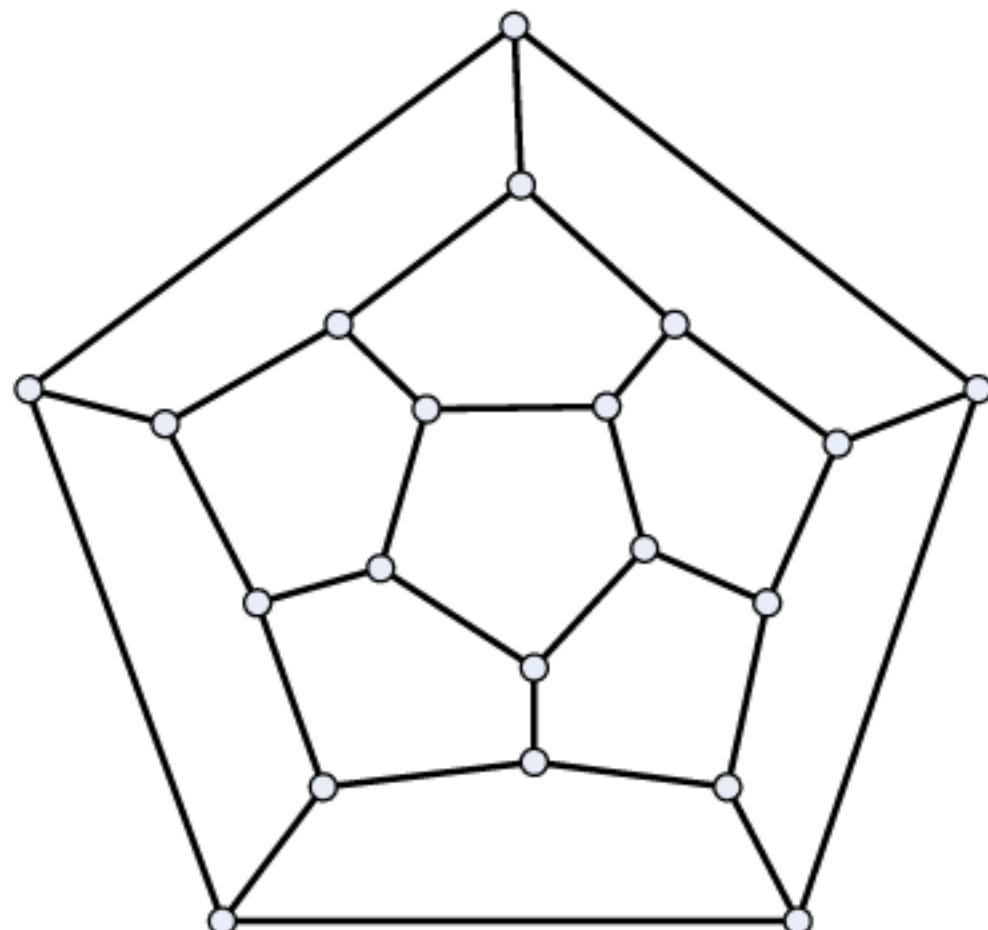
Output: CE – Chu trình Euler của đồ thị

Sau đây là kết quả chạy từng bước của thuật toán

STACK	CE
3, 6	\emptyset
3, 6, 4	\emptyset
3, 6, 4, 5	\emptyset
3, 6, 4, 5, 3	\emptyset
3, 6, 4, 5	3
3, 6, 4, 5, 2	3
3, 6, 4, 5, 2, 6	3
3, 6, 4, 5, 2, 6, 1	3
3, 6, 4, 5, 2, 6, 1, 5	3
3, 6, 4	3, 5, 1, 6, 2, 5
3, 6, 4, 7	3, 5, 1, 6, 2, 5
3, 6, 4, 7, 8	3, 5, 1, 6, 2, 5
3, 6, 4, 7, 8, 4	3, 5, 1, 6, 2, 5
\emptyset	3, 5, 1, 6, 2, 5, 4, 8, 7, 4, 6, 3

7.2. ĐỒ THỊ HAMILTON

Năm 1857, nhà toán học người Ailen, William Rowan Hamilton, đã đưa ra bài toán đố như sau: “Giả sử ta có một khối 12 mặt, mỗi mặt là một hình ngũ giác đều. Mỗi đỉnh trong 20 đỉnh của khối này được đặt bằng tên của một thành phố. Hãy tìm một đường xuất phát từ một thành phố, đi dọc theo các cạnh của khối, ghé thăm mỗi một trong 19 thành phố còn lại đúng một lần, cuối cùng trở lại thành phố ban đầu”.



Hình 7.4: Khối được biểu diễn trên mặt phẳng

Bài toán được phát biểu lại dưới dạng đồ thị như sau: Trong đồ thị hình trên có hay không một chu trình đi qua tất cả các đỉnh của đồ thị, mỗi đỉnh đúng một lần?

Từ bài toán này, người ta đã đưa ra các khái niệm chu trình Hamilton, đường đi Hamilton và đồ thị Hamilton.

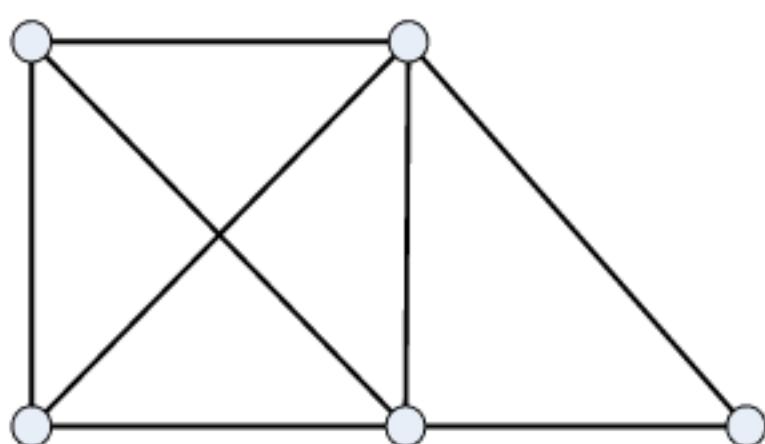
7.2.1. Định nghĩa

Giả sử G là đơn đồ thị vô (có) hướng, ta có các định nghĩa sau:

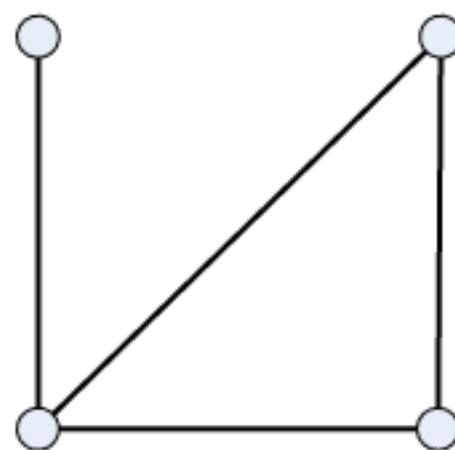
Chu trình Hamilton là chu trình đi qua tất cả các đỉnh của đồ thị, mỗi đỉnh đúng một lần. Nếu G có chu trình Hamilton thì G được gọi là *đồ thị Hamilton*.

Đường đi Hamilton là đường đi qua tất cả các đỉnh của đồ thị, mỗi đỉnh đúng một lần. Nếu G có đường đi Hamilton thì G được gọi là *đồ thị nửa Hamilton*.

Ví dụ:



Đồ thị Hamilton



Đồ thị nửa Hamilton

Hình 7.5

Hiện tại, chưa có một tiêu chuẩn rõ ràng để nhận biết một đồ thị có phải là đồ thị Hamilton hay không. Hầu hết các kết quả nghiên cứu thu được chỉ dừng lại ở điều kiện đủ mà thôi. Tức là, khi một đồ thị đạt một tiêu chuẩn nào đó (ví dụ: số cạnh đủ lớn) thì đồ thị đó là đồ thị Hamilton, nhưng chiều ngược lại thì có thể không đúng.

7.2.2. Định lý Dirac

Cho đồ thị vô hướng $G = (V, E)$ có n đỉnh ($n \geq 3$). Nếu mọi đỉnh $v \in V$ đều có $\deg(v) \geq n/2$ thì G có chu trình Hamilton.

✓ Định lý Dirac cho đồ thị có hướng

Cho đồ thị có hướng, liên thông mạnh $G = (V, E)$ và có n đỉnh. Nếu mọi đỉnh $v \in V$ đều có $\deg^+(v) \geq n/2$ và $\deg^-(v) \geq n/2$ thì G có chu trình Hamilton.

7.2.3. Giải thuật xây dựng chu trình Hamilton

Tiếp theo đây chúng ta sẽ sử dụng phương pháp quay lui để tìm và liệt kê tất cả các chu trình Hamilton. Đây là phương pháp cho kết quả chính xác tuyệt đối nhưng lại tốn nhiều thời gian xử lý. Hiện nay, người ta cũng đã đưa ra một số phương pháp khác để cải thiện thời gian xử lý so với phương pháp quay lui, nhưng chưa có một phương pháp nào mang lại kết quả chính xác tuyệt đối.

Giải thuật tìm các chu trình Hamilton được trình bày như sau:

- Bắt đầu từ một đỉnh, đi theo con đường dài nhất có thể được (depth – first);
- Nếu đường đó chứa mọi đỉnh và có thể nối hai đỉnh đầu và cuối bằng một cạnh thì đó là chu trình Hamilton;

- Nếu trái lại ta lùi lại một đỉnh để mở con đường theo chiều sâu khác;
- Cứ tiếp tục quá trình trên cho đến khi duyệt hết các đỉnh của đồ thị.

• Giải thuật tìm chu trình Hamilton cài đặt bằng mã giả

Đầu vào: Đồ thị $G = (V, E)$ lưu bằng danh sách kề (Mảng Ke())

Kết quả trả về: Danh sách các chu trình Hamilton của đồ thị.

Mảng Chuaxet[] để đánh dấu các đỉnh đã xét hay chưa.

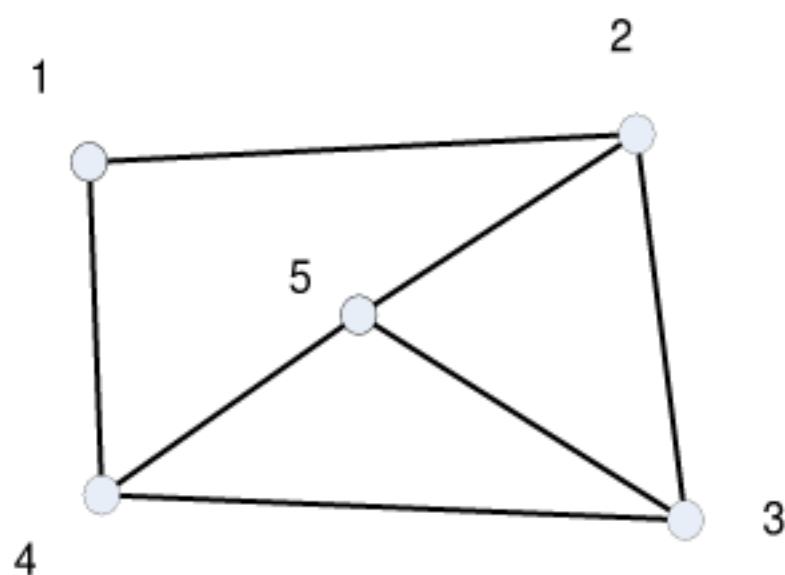
```

void hamilton(k);
/*Phát triển dãy X1,X2,...,Xk-1
G=(V,E) được cho bởi Danh Sách kề: Ke(v), v ∈ V */
{
    for (y ∈ Ke(Xk-1))
        if ((k == n+1) && (y == v0))
            Xuất(X1,...Xn,v0);
        else if (Chuaxet[y])
        {
            Xk = y;
            Chuaxet[y] = 0;
            Hamilton(k+1);
            Chuaxet[y] = 1; //Quay lui
        };
    }
main()
{
    for (v ∈ V)
        Chuaxet[v] = 1;
    X1 = v0;
    Chuaxet[v0] = 0;
    Hamilton(2);
}

```

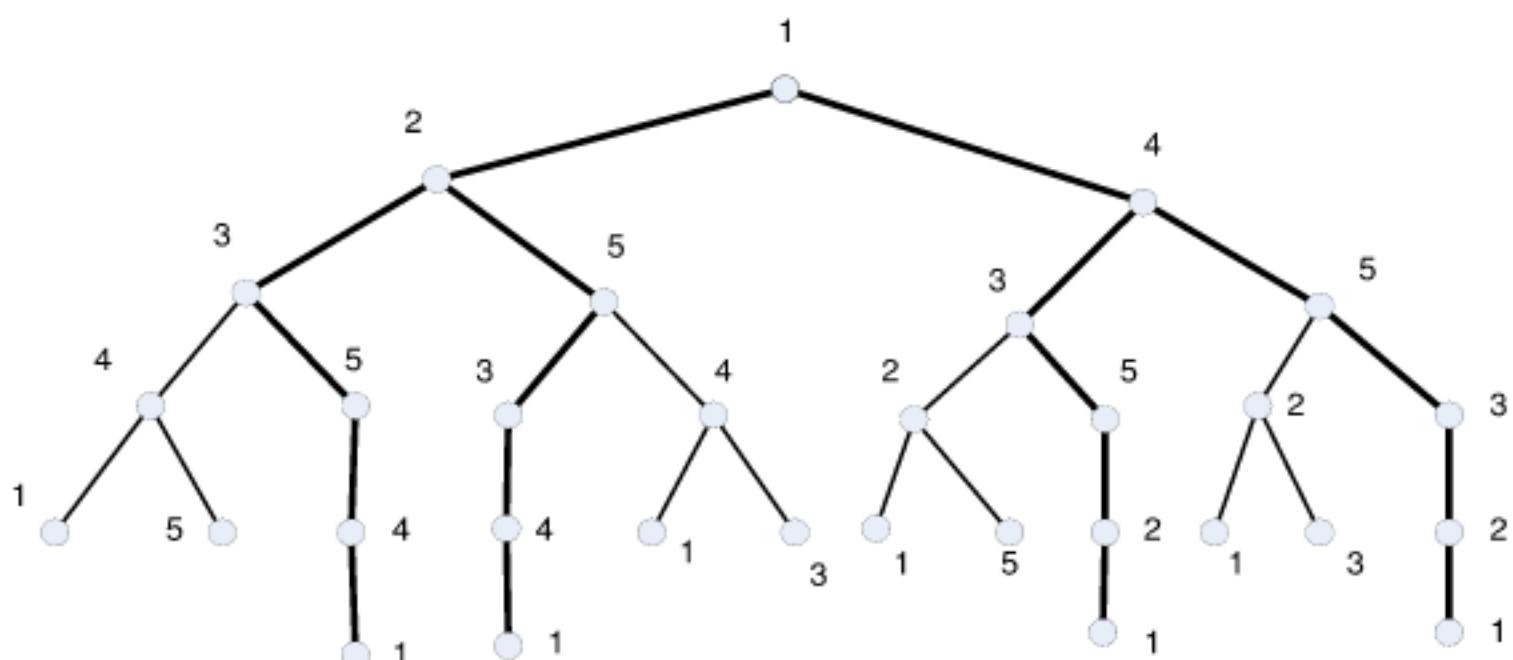
Ví dụ:

Tìm các chu trình Hamilton của đồ thị sau



Hình 7.6

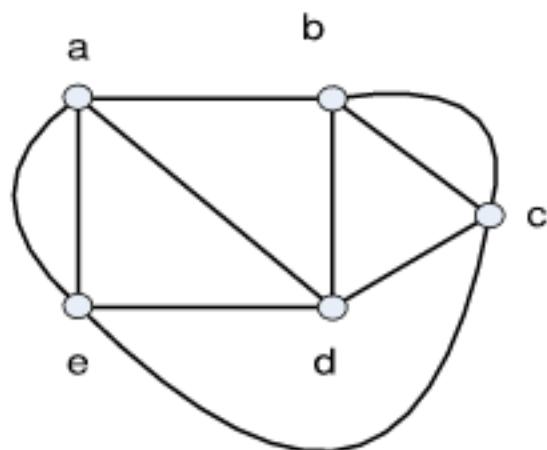
Áp dụng thuật toán trên, thứ tự duyệt các đỉnh của đồ thị được minh họa qua hình dạng cây sau:



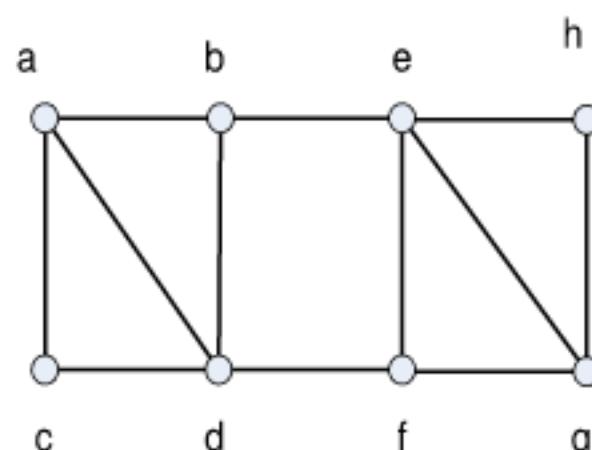
Hình 7.7

BÀI TẬP CHƯƠNG 7

1. Hãy xác định xem các đồ thị sau có chu trình Euler hay không. Nếu có, áp dụng giải thuật đã học để tìm chúng.

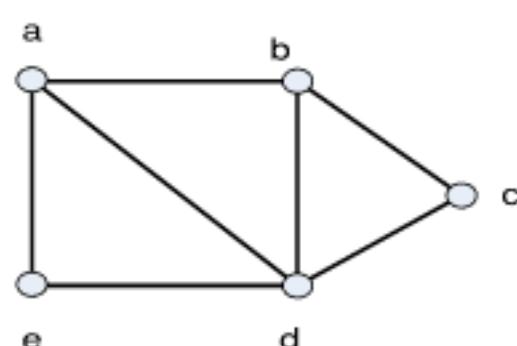


Hình 7.8

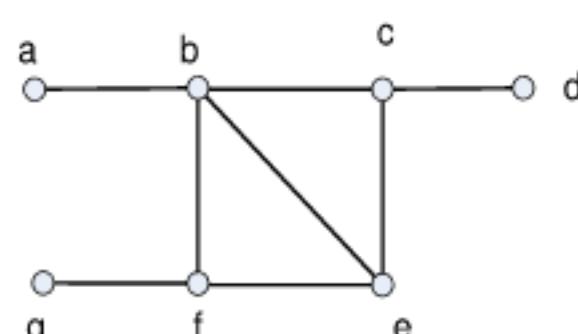


Hình 7.9

2. Hãy xác định xem các đồ thị sau có đường đi Hamilton hay không, nếu có hãy tìm đường đi đó. Nếu không, hãy giải thích vì sao không có.



Hình 7.10



Hình 7.11

3. Đồ thị cho bởi danh sách cạnh, hãy viết chương trình xác định xem nó có chu trình hay đường đi Euler hay không? Nếu có thì tìm chúng (xét trường hợp đồ thị vô hướng và có hướng).
4. Đồ thị cho bởi danh sách cạnh, hãy viết chương trình tìm đường đi Hamilton cho đồ thị nếu có.
5. Đồ thị cho bởi danh sách cạnh, hãy viết chương trình tìm chu trình Hamilton cho đồ thị nếu có.
6. Với giá trị nào của m và n thì đồ thị hai phía đầy đủ $K_{m,n}$ là:
- Đồ thị Euler.
 - Đồ thị nửa Euler.
7. Chứng minh rằng nếu G là đồ thị Hamilton thì G không chứa đỉnh rẽ nhánh nào.

8. Chứng minh rằng nếu G là đồ thị Euler thì G không chứa cạnh là cầu nào.

9. Bài toán mã đi tuần

Hãy cài đặt chương trình xác định lộ trình của quân mã trên bàn cờ 8×8 ô, bắt đầu từ ô $[i, j]$ đi qua tất cả các ô của bàn cờ và mỗi ô chỉ đi qua một lần duy nhất. Mở rộng với trường hợp bàn cờ kích thước $N \times N$.

Chương 8

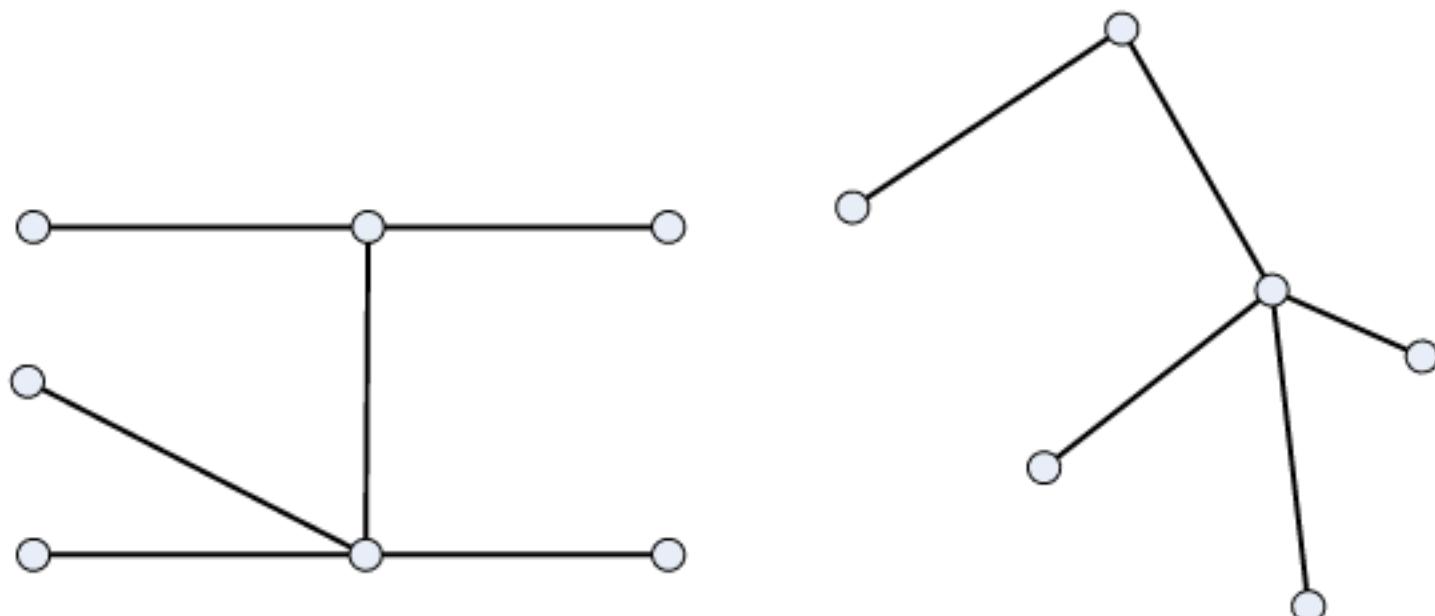
CÂY

Chương này trình bày một dạng đồ thị đặc biệt, có nhiều ứng dụng trong tin học, đó là cây. Người đầu tiên sử dụng khái niệm cây để biểu diễn là nhà toán học người Anh tên là Arthur Cayley. Năm 1857, ông dùng cây để xác định những dạng khác nhau của hợp chất hóa học. Sau đây sẽ trình bày các khái niệm, tính chất và những thuật toán duyệt cây.

8.1. ĐỊNH NGHĨA VÀ CÁC TÍNH CHẤT CƠ BẢN CỦA CÂY

Cây là một đồ thị vô hướng, liên thông, và không có chu trình.

Ví dụ:



Hình 8.1: Cây

Rừng là một đồ thị vô hướng và không có chu trình. Như vậy, rừng có thể có nhiều thành phần liên thông và mỗi thành phần liên thông là một cây.

Các tính chất cơ bản của cây

Các tính chất cơ bản của cây thể hiện trong định lý sau:

✓ **Định lý nhận biết cây**

Cho $T = (V, E)$ là đồ thị vô hướng n đỉnh. Các mệnh đề sau đây là tương đương:

MĐ1: T là cây (T liên thông và không chứa chu trình).

MĐ2: T không chứa chu trình và có $n - 1$ cạnh.

MĐ3: T liên thông và có $n - 1$ cạnh.

MĐ4: T liên thông và mỗi cạnh của nó đều là cầu.

MĐ5: Hai đỉnh bất kỳ của T được nối với nhau bởi đúng một đường đi đơn.

MĐ6: T không chứa chu trình nhưng hổn cứ thêm vào nó một cạnh ta thu được đúng một chu trình.

Chứng minh:

Ta sẽ chứng minh định lý trên theo sơ đồ sau:

$$\text{MĐ1} \Rightarrow \text{MĐ2} \Rightarrow \text{MĐ3} \Rightarrow \text{MĐ4} \Rightarrow \text{MĐ5} \Rightarrow \text{MĐ6} \Rightarrow \text{MĐ1}$$

Chứng minh MĐ1 \Rightarrow MĐ2: Nếu T là cây n đỉnh thì T không có chu trình và có $n - 1$ cạnh.

T không có chu trình là hiển nhiên, do đó, chỉ cần chứng minh rằng T có $n - 1$ cạnh.

Ta chứng minh bằng phương pháp quy nạp theo số đỉnh của cây

+ Với $n = 1$ thì đồ thị có số cạnh là: $n - 1 = 1 - 1 = 0$ (Đúng)

+ Giả sử khẳng định đúng \forall cây có $k \geq 1$ đỉnh.

T là cây có $k+1$ đỉnh. Cần phải chỉ ra T có k cạnh.

Chọn đường đi dài nhất trong T là $P = (v_1, v_2, \dots, v_m)$. Rõ ràng v_1 là đỉnh treo vì:

+ v_1 không thể kề với các đỉnh v_3, \dots, v_m vì T không có chu trình;

+ v_1 không thể được nối với các đỉnh khác vì P là đường đi dài nhất.

Xét cây T' có được từ T sau khi bỏ đi đỉnh v_1 và cạnh (v_1, v_2) . Rõ ràng T' cũng là cây và có k đỉnh, và theo giả thiết quy nạp T' có $k - 1$ cạnh $\Rightarrow T$ có k cạnh (đpcm).

Chứng minh MĐ2 \Rightarrow MĐ3: Nếu T không chứa chu trình và có $n - 1$ cạnh thì T liên thông.

Ta chứng minh bằng phương pháp phản chứng. Giả sử T không liên thông, khi đó T được phân rã thành $k > 1$ thành phần liên thông T_1, T_2, \dots, T_k . Vì T không chứa chu trình (theo giả thiết) nên các T_i cũng vậy, suy ra T_i là cây.

Gọi $v(T)$ và $e(T)$ tương ứng là số đỉnh và cạnh của T . Theo phần trước $MĐ1 \Rightarrow MĐ2$ ta có: $e(T_i) = v(T_i) - 1$. Suy ra:

$$\sum e(T_i) = \sum (v(T_i) - 1) = \sum v(T_i) - k$$

$$\Leftrightarrow e(T) = v(T) - k ; \text{ hay } n - 1 = n - k. \text{ Vô lý với } k > 1 \text{ (đpcm).}$$

Chứng minh $MĐ3 \Rightarrow MĐ4$: Nếu T liên thông và có $n - 1$ cạnh thì mỗi cạnh của T là cầu.

Xóa một cạnh bất kỳ của T , khi đó ta nhận được cây T' có n đỉnh và $n - 2$ cạnh. Nếu ta chỉ ra được T' không liên thông thì cạnh đó là cầu của T .

Ta sẽ chứng minh rằng đồ thị với n đỉnh và $n - 2$ cạnh sẽ không liên thông. Chứng minh bằng quy nạp theo n và phản chứng giống $MĐ1 \Rightarrow MĐ2$. Chọn đường đi dài nhất và xoá bỏ cạnh treo cùng đỉnh (đpcm).

Chứng minh $MĐ4 \Rightarrow MĐ5$: Nếu T liên thông và mỗi cạnh của T là cầu thì hai đỉnh bất kỳ của T được nối với nhau đúng bởi một đường đơn.

T liên thông nên mọi cặp đỉnh của T tồn tại đường nối giữa chúng. Đường nối này là duy nhất vì trái lại T sẽ có chu trình và các cạnh trên chu trình đó sẽ không thể là cầu. (Đpcm)

Chứng minh $MĐ5 \Rightarrow MĐ6$: Nếu hai đỉnh bất kỳ của T được nối với nhau đúng bởi một đường đơn thì T không chứa chu trình, nhưng nếu thêm vào nó một cạnh ta thu được đúng một chu trình.

+ T không chứa chu trình vì nếu T có chu trình thì sẽ có cặp đỉnh được nối với nhau bởi ít nhất 2 đường đơn.

+ Giả sử cạnh (u,v) được thêm vào ta sẽ nhận được chu trình gồm đường đơn nối u với v và cạnh (u,v) mới.

+ Do đường đơn nói trên là duy nhất nên chu trình nhận được cũng là duy nhất. (Đpcm)

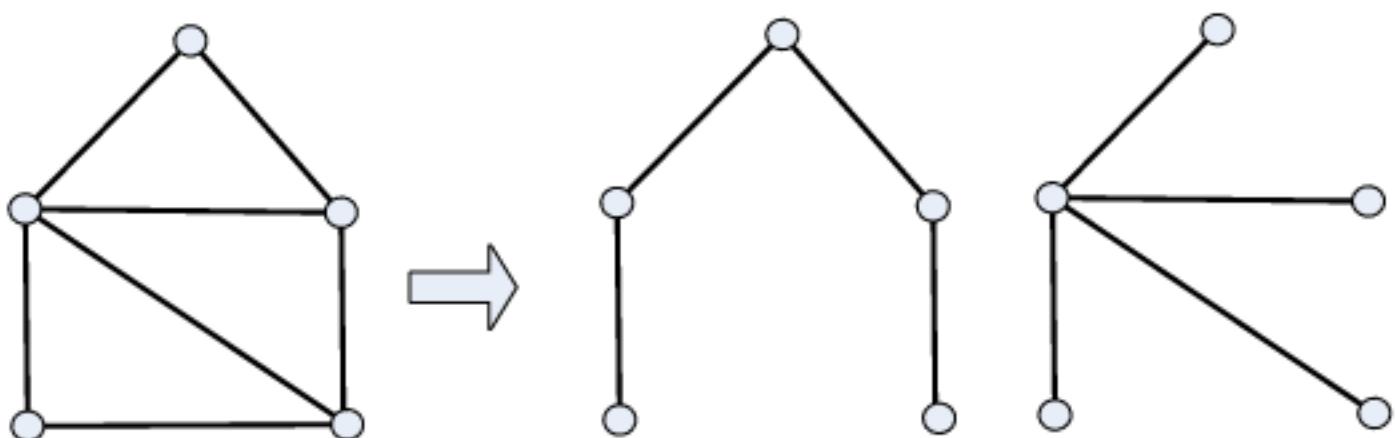
8.2. CÂY KHUNG CỦA ĐỒ THỊ

8.2.1. Định nghĩa

Cho đồ thị $G = (V, E)$ vô hướng, liên thông. Một cây $T = (V, F)$ được xây dựng từ G với $F \subset E$ (T chứa tất cả các đỉnh của G và tập cạnh F là con của tập cạnh E) được gọi là **cây khung** của đồ thị G .

Người ta còn gọi cây khung là: cây bao trùm hay cây tối đại.

Ví dụ:



Hình 8.2: Cây khung

8.2.2. Thuật toán xây dựng cây khung

Phần này trình bày phương pháp xây dựng cây khung dựa trên các thuật toán duyệt đồ thị theo chiều sâu và theo chiều rộng.

Đầu vào của các thuật toán: Đồ thị G lưu dưới dạng danh sách kề - Mảng Ke[]

Kết quả trả về: Cây khung T của đồ thị

Mảng ChuaXet[] dùng để đánh dấu các đỉnh đã được xét hay chưa.

8.2.2.1. Xây dựng cây khung theo chiều sâu

```
/* Khai báo các biến toàn cục ChuaXet, Ke, T */
```

```
void Tree_DFS(v);
{
    ChuaXet[v] = 0;
    for (u ∈ Ke(v))
        if (ChuaXet[u])
            {
                T = T ∪ (v,u);
                Tree_DFS(u);
            };
}
```

```
main()
```

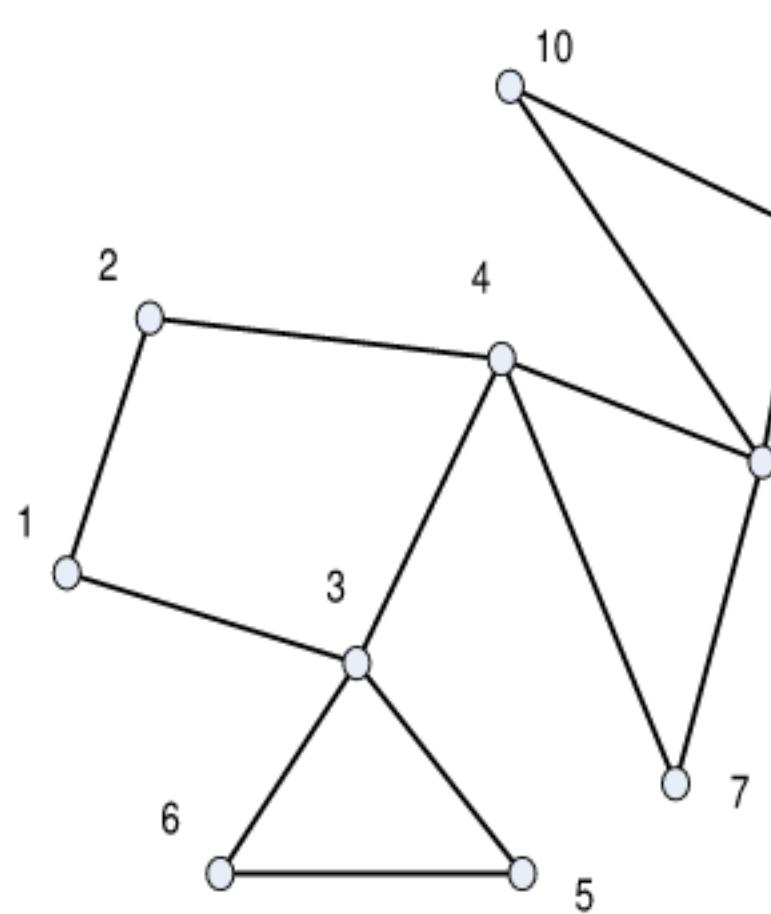
```

/* Nhập đồ thị, tạo biến Ke */
for (v ∈ V)
    ChuaXet[v] = 1; /* Khởi tạo cờ cho đỉnh */
    T = ∅; /* T là tập cạnh cây khung */
    Tree_DFS(root); /* root là đỉnh nào đó của đồ thị */
}

```

Ví dụ:

Áp dụng giải thuật trên, tìm cây khung cho đồ thị G sau:



Định v	Ke(v)
1	2, 3
2	4, 1
3	1, 6, 5, 4
4	2, 3, 7, 8
5	6, 3
6	3, 5
7	4, 8
8	7, 4, 10, 9
9	8, 10
10	8, 9

Hình 8.3: Tìm cây khung theo chiều sâu

Áp dụng thuật toán xây dựng cây khung theo chiều sâu, bắt đầu từ đỉnh 1, thứ tự các đỉnh được duyệt như sau:

1 → 2 → 4 → 3 → 6 → 5
 ↓
 7 → 8 → 10 → 9

Khi đó cây khung của G là: {(1, 2), (2, 4), (4, 3), (3, 6), (6, 5), (4, 7), (7, 8), (8, 10), (10, 9)}

8.2.2.2. Xây dựng cây khung theo chiều rộng

```
/* Khai báo các biến toàn cục ChuaXet, Ke, QUEUE */
void Tree_BFS(r);
{
    QUEUE = ∅;
    QUEUE ← r;
    ChuaXet[r] = 0;
    while (QUEUE != ∅)
    {
        v ← QUEUE;
        for (u ∈ Ke(v))
            if (ChuaXet[u])
            {
                QUEUE ← u;
                ChuaXet[u] = 0;
                T = T ∪ (v,u);
            };
    }
}

main() /* Nhập đồ thị, tạo biến Ke */
{
    for (v ∈ V)
        ChuaXet[v] = 1; /* Khởi tạo cờ cho đỉnh */
    T = ∅; /* T là tập cạnh cây khung */
    Tree_DFS(root); /* root là đỉnh nào đó của đồ thị */
}
```

Ví dụ:

Áp dụng thuật toán xây dựng cây khung theo chiều rộng cho đồ thị hình 8.3.

Bắt đầu duyệt từ đỉnh 1. Thứ tự duyệt các đỉnh như sau:

$1 \rightarrow 2 \rightarrow 3$

$4 \rightarrow 6 \rightarrow 5$

$7 \rightarrow 8$

$10 \rightarrow 9$

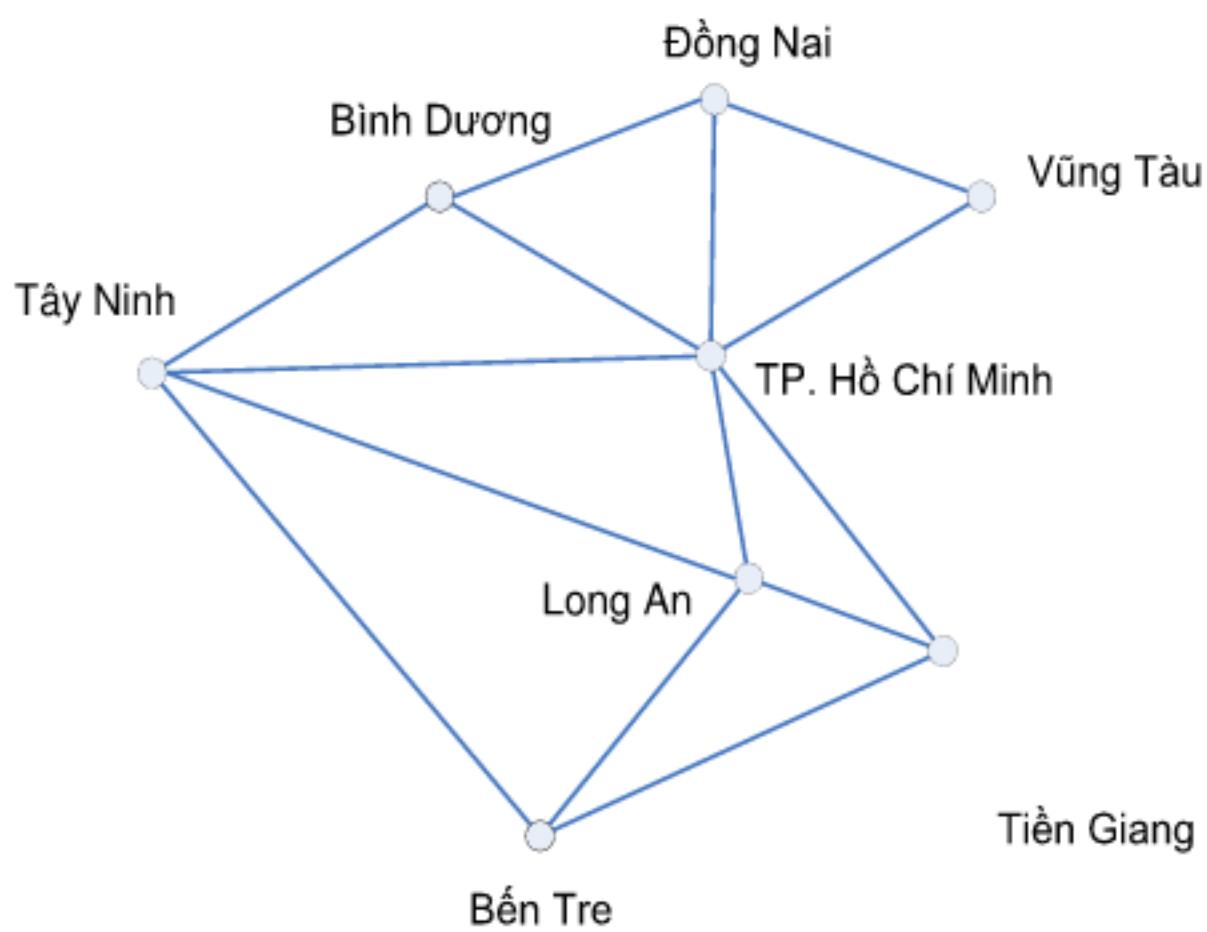
Khi đó, cây khung của G là: $\{(1, 2), (2, 3), (2, 4), (4, 6), (6, 5), (4, 7), (7, 8), (8, 10), (10, 9)\}$

8.3. BÀI TOÁN CÂY KHUNG NHỎ NHẤT

Ở các phần trên, chúng ta đã đề cập đến khái niệm cây khung và các thuật toán tìm cây khung cho một đồ thị vô hướng, liên thông. Trong phần này, chúng ta sẽ nghiên cứu các thuật toán tìm cây khung có tổng trọng số các cạnh là nhỏ nhất. Trước hết, chúng ta cần nắm một số khái niệm.

Cho đồ thị G vô hướng, liên thông, có trọng số. Giả sử $T = (V, F)$ là cây khung của G . Tổng trọng số các cạnh của T (hay được gọi là trọng số của cây T) được định nghĩa: $w(T) = \sum_{e \in F} w(e)$. Cây khung có trọng số nhỏ nhất trong số các cây khung của G được gọi là cây khung nhỏ nhất.

Bài toán tìm cây khung nhỏ nhất có thể được ứng dụng trong việc giải quyết các vấn đề thực tế như: bài toán xây dựng hệ thống đường sắt sao cho chi phí xây dựng nhỏ nhất nhưng vẫn bảo đảm nhu cầu giao lưu giữa các thành phố với nhau; bài toán nối mạng máy tính sao cho chi phí lắp đặt là nhỏ nhất; xây dựng hệ thống giao thông để nối liền các vùng, sao cho tổng quãng đường xây dựng là ngắn nhất (hình 8.4), v.v.



Hình 8.4

Sau đây là các thuật toán tìm cây khung nhỏ nhất thông dụng.

8.3.1. Thuật toán Kruskal

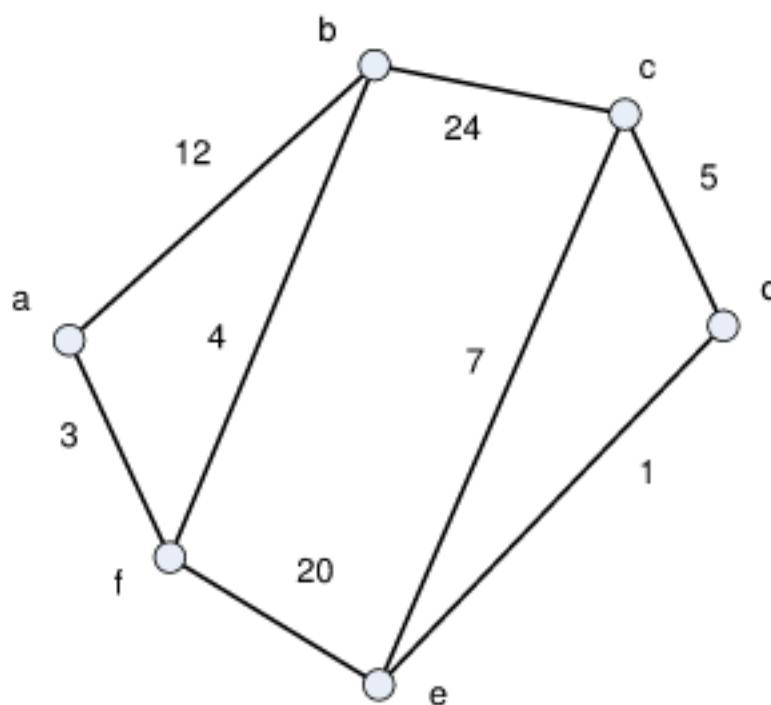
Nội dung của thuật toán

Xây dựng tập cạnh F của $T = (V, F)$ theo từng bước:

1. Sắp xếp các cạnh của G theo thứ tự trọng số (độ dài) tăng dần;
2. Bắt đầu với $F = \emptyset$ bổ sung dần các cạnh của G vào F với điều kiện không tạo nên chu trình trong T ;
3. Thuật toán dừng lại khi có $n-1$ cạnh được chọn.

Ví dụ:

Áp dụng thuật toán Kruskal, tìm cây khung nhỏ nhất cho đồ thị sau:



Hình 8.5: Ví dụ thuật toán Kruskal

Danh sách các cạnh theo thứ tự trọng số tăng dần:

(d, e)	(a, f)	(b, f)	(c, d)	(c, e)	(a, b)	(f, e)	(b, c)
1	3	4	5	7	12	20	24

Cây khung nhỏ nhất của đồ thị:

(d, e)	(a, f)	(b, f)	(c, d)	(f, e)
1	3	4	5	20

Sau đây là cài đặt thuật toán Kruskal bằng mã giả:

```

void Kruskal;
{
    F = ∅;
    while ((|F| < n-1) && (E != ∅))
    {
        Chọn e = min ∈ E;
        E = E \ {e};
        if (F ∪ {e} không chứa chu trình)
            F = F ∪ {e};
    }
    if (|F| < n-1) cout << "Đồ thị không liên thông";
}

```

Trong thực tế, việc kiểm tra đồ thị không chứa chu trình không hề đơn giản, mà trong thuật toán ta phải kiểm tra tính chất này trong mỗi vòng lặp. Do vậy, có thể tiếp cận bằng cách xây dựng “dần dần” cây khung nhỏ nhất theo cách sau:

- Coi đồ thị ban đầu có n đỉnh và không có cạnh. Có thể coi đây như là một rừng gồm n cây;

- Mỗi lần chúng ta sẽ thêm vào một cạnh theo thứ tự tăng dần theo độ dài sao cho hai đầu mút của cạnh mới được thêm phải nằm ở hai cây khác nhau;

- Thuật toán sẽ dừng lại khi có $n - 1$ cạnh được thêm

Áp dụng vào đồ thị trên hình 8.5 ta có:

Danh sách các cạnh theo thứ tự trọng số tăng dần:

(d, e)	(a, f)	(b, f)	(c, d)	(c, e)	(a, b)	(f, e)	(b, c)
1	3	4	5	7	12	20	24

Thực hiện theo các bước của thuật toán:

Chọn cạnh	CÁC CÂY						
	A	b	c	d	e	f	
(d, e)=1	A	b	c	d,e		f	
(a, f)=3	a,f	b	c	d,e			
(b, f)=4	a,f,b		c	d,e			
(c, d)=5	a,f,b			c,d,e			
(c, e)=7							
(a, b)=12							
(f, e)=20			a,f,b,c,d,e				
(b, c)=24							

Các cạnh đã được chọn cho cây khung nhỏ nhất:

(d, e)	(a, f)	(b, f)	(c, d)	(f, e)
1	3	4	5	20

Sau đây là cài đặt thuật toán Kruskal trên thực tế:

```
Kruskal()
{
    T = ∅;
    for (v ∈ V)
        MakeSet(v); // Mỗi đỉnh có một nhãn khác nhau
    xếp thứ tự các cạnh trong E tăng dần theo trọng số w
    for (u,v) ∈ E (theo thứ tự đã sắp xếp)
        if (FindSet(u) != FindSet(v)) // Kiểm tra hai đỉnh u và v có khác
nhãn ?
            T = T ∪ { {u,v} }; // Chọn cạnh (u,v)
            Union(FindSet(u), FindSet(v)); // Đóng nhất nhãn của u và v
    }
}
```

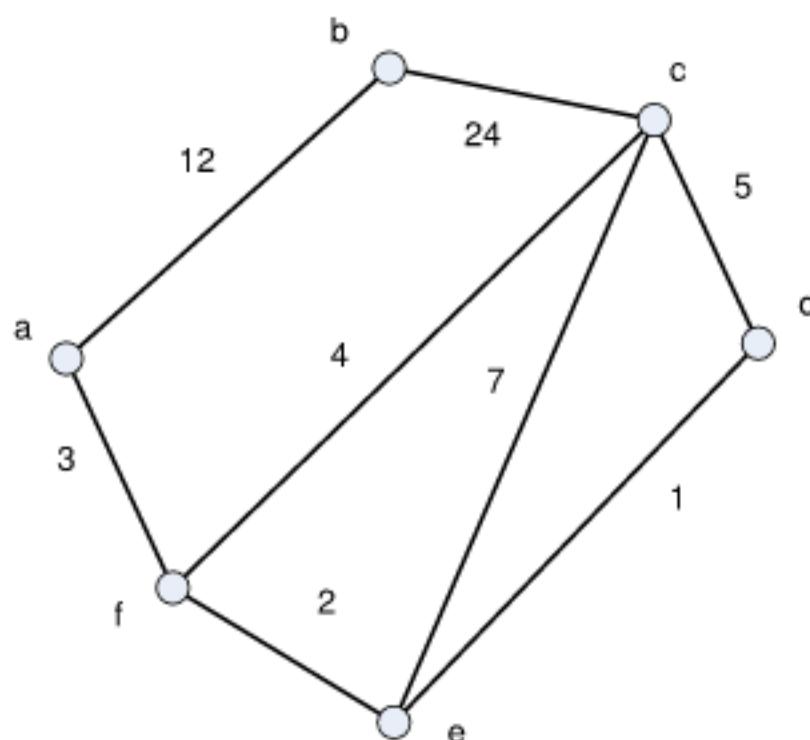
8.3.2. Thuật toán Prim

Nội dung của thuật toán như sau:

Xây dựng tập đỉnh V_T và tập cạnh F của cây khung $T=(V_T, F)$ theo từng bước:

1. Bắt đầu với $V_T = \{s\}$, một đỉnh bất kỳ và $F=\emptyset$;
2. Trong các cạnh có 1 đỉnh $\notin V_T$ và 1 đỉnh $\in V_T$ chọn cạnh có trọng số nhỏ nhất;
3. Bổ sung cạnh đó vào F và đỉnh tương ứng vào V_T ;
4. Thuật toán dừng lại khi có $n - 1$ cạnh được chọn (hoặc $V_T = V$).

Ví dụ: Áp dụng thuật toán Prim, tìm cây khung nhỏ nhất cho đồ thị sau:



Hình 8.6: Thuật toán Prim

Để hiện thực thuật toán ta sẽ gán nhãn các đỉnh: Mỗi đỉnh $v \notin V_T$ sẽ có hai nhãn

- $d[v] = \min \{ w(v,u); u \in V_T \}$
- $\text{near}[v] = \{ u; \text{với } w(v,u) \text{ min} \}$

Thuật toán Prim sẽ được thực hiện theo các bước sau (đỉnh bắt đầu là a):

V_T	b	c	d	e	f
a	12, a	∞ , a	∞ , a	∞ , a	3, a
a, f	12, a	4, f	∞ , a	2, f	
a, f, e	12, a	4, f	1, e		
a, f, e, d	12, a	4, f			
a, f, e, d, c	12, a				
a, f, e, d, c, b	12, a	4, f	1, e	2, f	3, a

Dưới đây là cài đặt thuật toán Prim bằng mã giả:

```

void Prim()
{
    F = ∅; V_T = {u};
    while (|F| < n-1)
    {
        ...
    }
}

```

```

Chọn e = { min w(u,v) (u ∈ VT) & (v ∉ VT) };

F = F ∪ {e};

VT = VT ∪ {v};

}

}

```

Nhận xét

- Thuật toán Kruskal và Prim đều là dạng thuật toán tham lam. Thuật toán tham lam là thuật toán thực hiện một lựa chọn tối ưu ở mỗi giai đoạn. Việc tối ưu hóa ở mỗi giai đoạn của thuật toán không đảm bảo tạo ra lời giải tối ưu toàn cục. *Nhưng hai thuật toán này đều là các thuật toán tham lam tạo các lời giải tối ưu toàn cục.*
- Thuật toán Kruskal làm việc kém hiệu quả với các đồ thị dày ($m \geq n(n-1)/2$, m là số cạnh, n là số đỉnh đồ thị).
- Từ bài toán tìm cây khung nhỏ nhất, ta có thể chuyển qua bài toán tìm cây khung lớn nhất bằng cách đổi dấu các trọng số, sau đó áp dụng các thuật toán tìm cây khung nhỏ nhất.

8.4. CÂY CÓ GỐC

8.4.1. Các định nghĩa

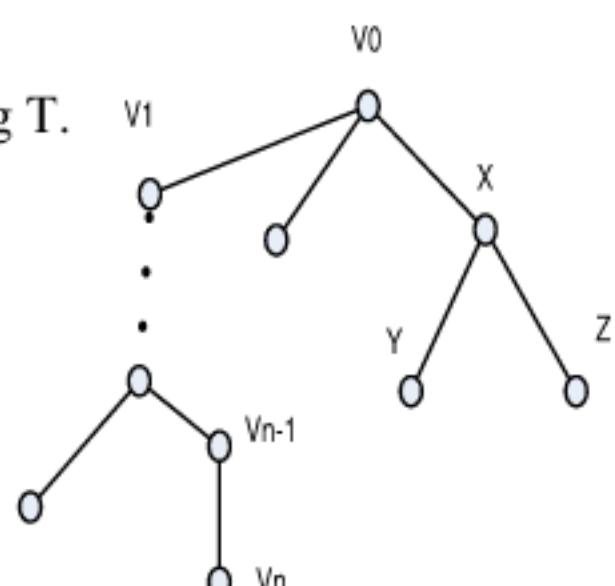
Cho T là một cây với một đỉnh v_0 được chọn làm gốc.

x, y, z là các đỉnh trong T.

(v_0, v_1, \dots, v_n) là một đường đi đơn trong T.

Ta định nghĩa các khái niệm sau:

- Đỉnh v_{n-1} là **cha** của v_n .
- v_n là **con** của v_{n-1} .
- v_0, v_1, \dots, v_{n-1} là các **tiền bối** của v_n .



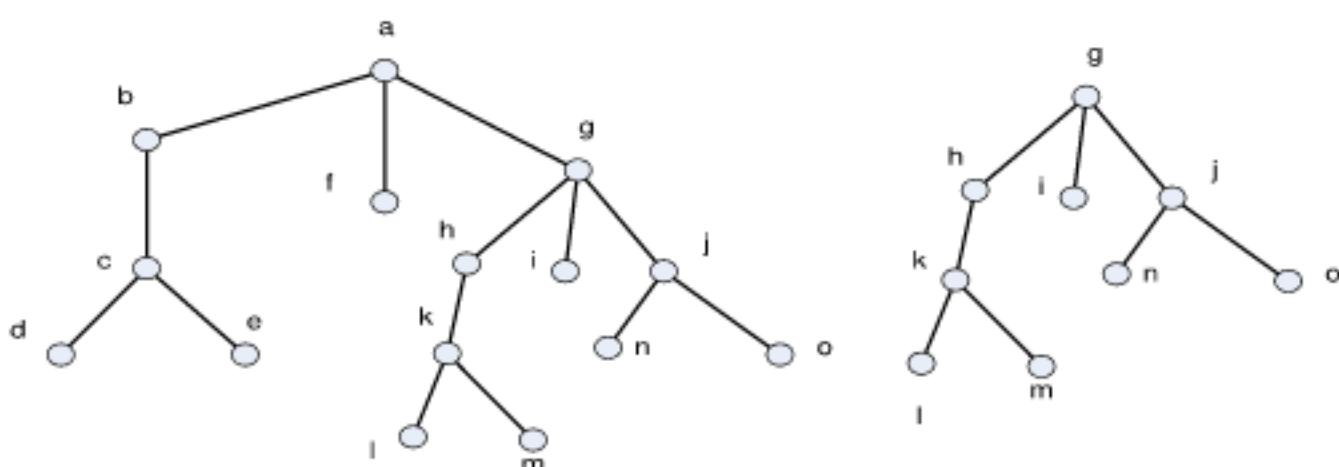
Hình 8.7

- Nếu x là tiền bối của y thì y là **hậu duệ** của x.
- Nếu y, z cùng là con của x thì y, z là **anh em** của nhau.
- Nếu x không có con thì x là **lá**. Ngược lại x là **đỉnh trong**.

- **Mức** của đỉnh x là chiều dài (số cạnh) của đường đi đơn từ gốc v_0 tới x.

- Như vậy, mức của v_0 là: $\text{level}(v_0) = 0$.
- **Chiều cao** của một cây là mức lớn nhất của các đỉnh trong cây.
- **Cây con** của T gốc tại x là đồ thị con của T mà:
 - + Tập đỉnh gồm x và các hậu duệ của x.
 - + Tập cạnh gồm mọi cạnh trong T nối từ x tới các hậu duệ của x.

Ví dụ:



Hình 8.8

Cha của c là b

Con của g là h, i, j

Các **tiền bối** của e là c, b, a

Các **hậu duệ** của b là c, d, e

Các **đỉnh trong**: a, b, c, g, h, j, k

Các **lá**: d, e, f, l, m, i, n, o

Mức của c là 2, của k là 3

Chiều cao của cây là 4

Cây con gốc g.

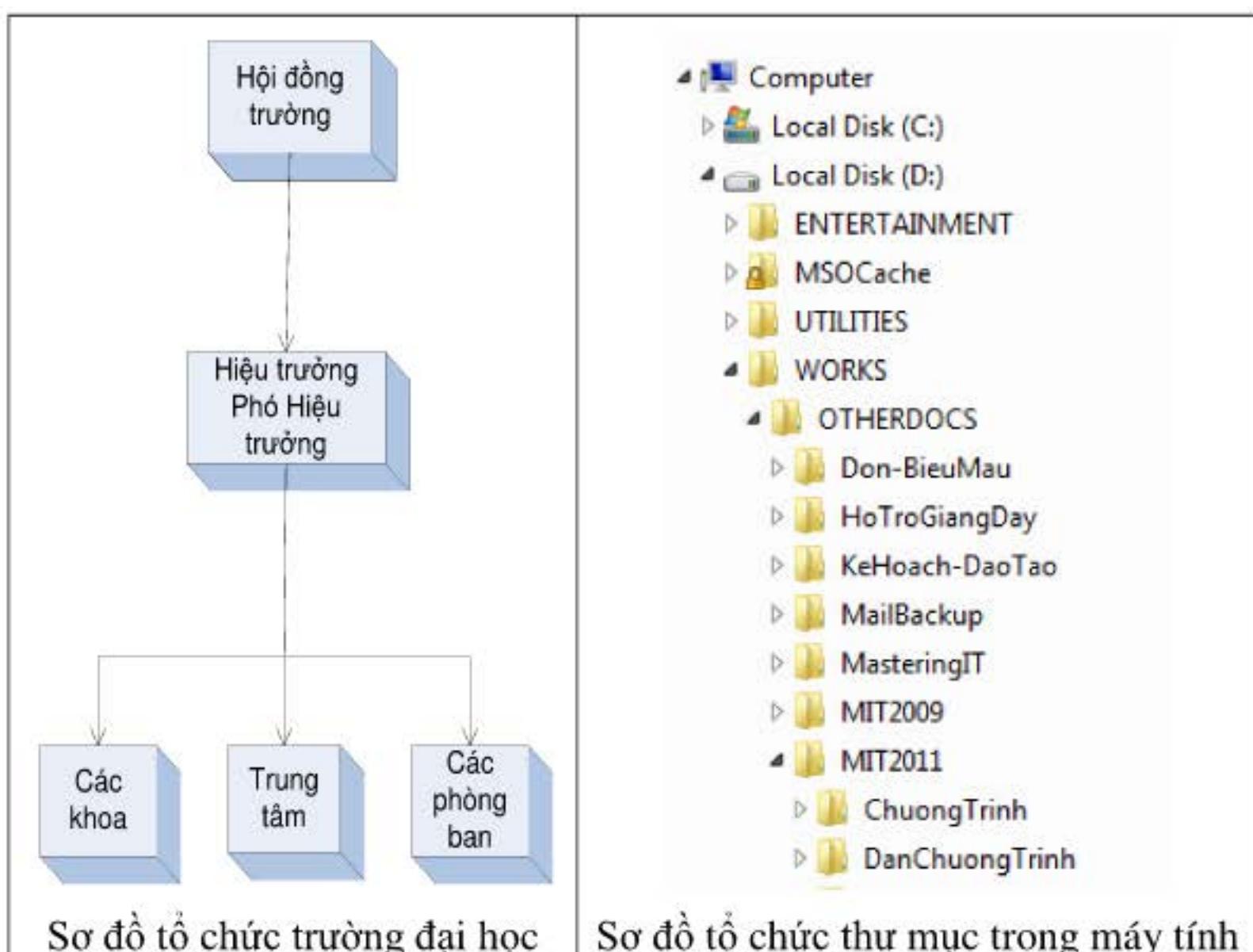
Một cây có gốc gọi là:

- **m-cây**: nếu mỗi đỉnh trong có không quá m con.
- **m-cây đầy**: nếu mỗi đỉnh trong có đúng m con.
- **Cây nhị phân**: nếu mỗi đỉnh trong không quá 2 con.
- **Cây có gốc thứ tự**: nếu các con của mỗi đỉnh trong được xếp thứ tự (từ trái qua phải).

- Với **Cây nhị phân có thứ tự**: nếu mỗi đỉnh trong đú 2 con thì:
 - + Con thứ nhất là **con bên trái**.
 - + Con thứ hai là **con bên phải**.
 - Một m-cây với chiều cao h được gọi là **cây thăng bằng** (**Cây cân bằng**) nếu tất cả các lá đều ở mức h hay h-1.

Một số ví dụ về cây có gốc

Trong thực tế, có nhiều mô hình sử dụng cây có gốc như: mô hình gia phả trong một dòng họ; mô hình tổ chức của một cơ quan; mô hình các thư mục, tập tin trong máy tính.



Hình 8.9: Ví dụ cây có gốc

8.4.2. Cây nhị phân tìm kiếm

Định nghĩa

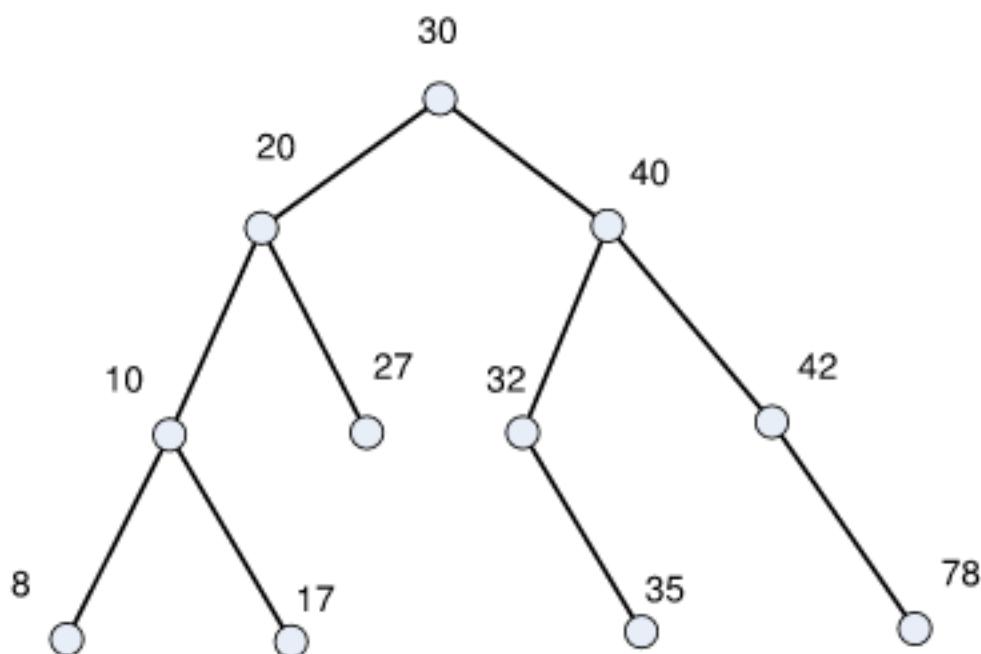
Một cây nhị phân tìm kiếm là một cây nhị phân T trong đó mỗi đỉnh được gán cho một nhãn, các nhãn của đỉnh có thể so sánh được với nhau và với mọi đỉnh $v \in T$, các nhãn trong cây con bên trái của v đều nhỏ hơn nhãn của v và các nhãn trong cây con bên phải của v đều lớn hơn nhãn của v .

Ví dụ:

Tạo cây nhị phân tìm kiếm cho các phần tử có nhãn như sau: 30, 20, 10, 40, 32, 27, 17, 8, 42, 78, 35.

Gốc của cây là phần tử có nhãn 30.

Ta có cây nhị phân tìm kiếm:



Hình 8.10

Thuật toán tìm kiếm trên cây nhị phân tìm kiếm

```
void TK(Cây NPTK T, phần tử x); // Tìm x trên cây NPTK T
{
    v = gốc của T;
    ketqua = "Không tìm thấy x";
    while ((v != NULL) && (label(v) != x))
    {
        if (x == label(v)) ketqua = "Tìm được x";
        if (x < label(v))
            if (con bên trái v != NULL) v = con bên trái v;
        if (x > label(v))
            if (con bên phải v != NULL) v = con bên phải v;
    }
    cout << ketqua;
}
```

8.4.3. Cây quyết định

Định nghĩa

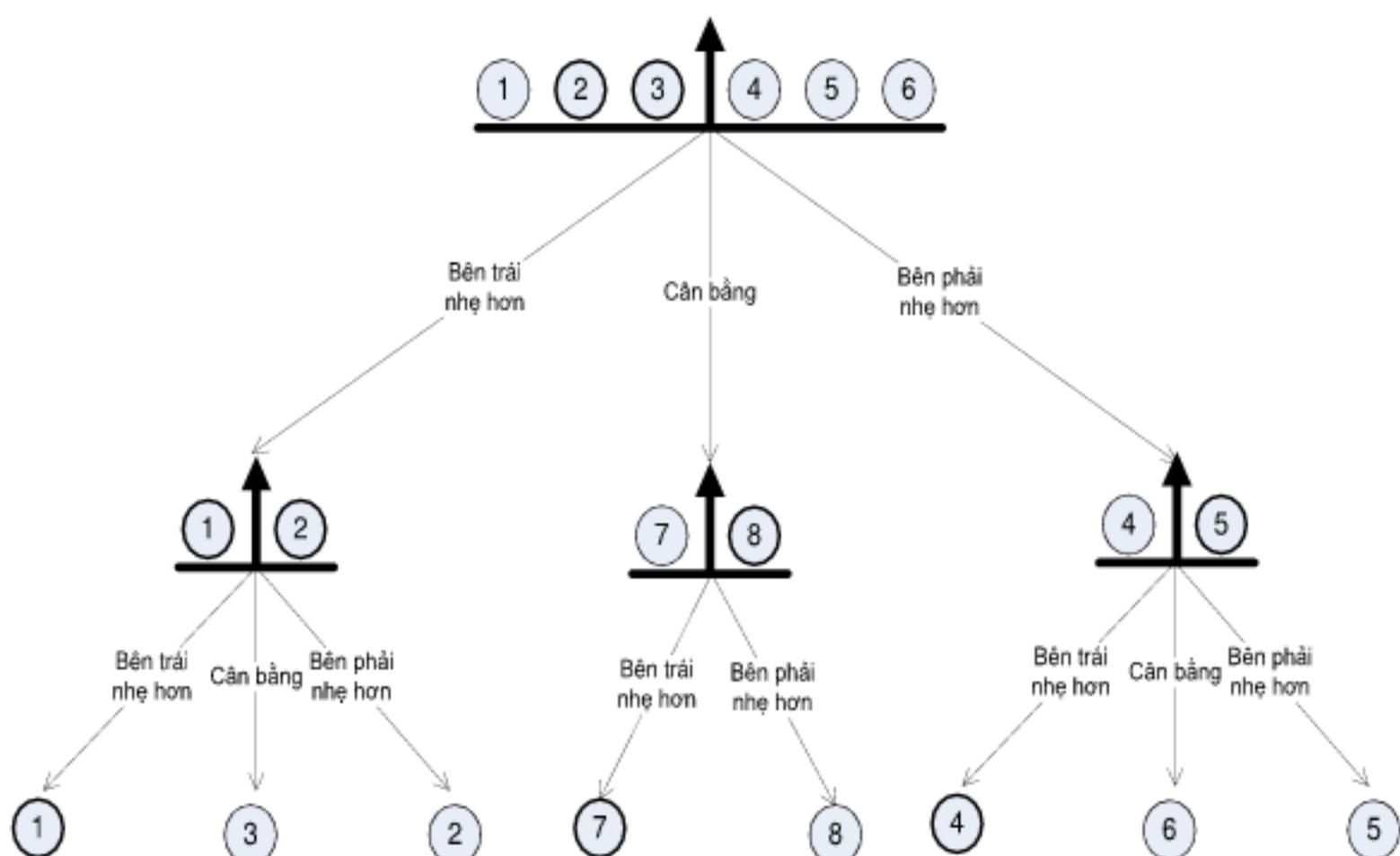
Cây quyết định là cây có gốc trong đó mỗi đỉnh tương ứng với một quyết định và mỗi cây con tại các đỉnh này ứng với mỗi kết cục có thể của quyết định. Một lời giải là một đường đi từ gốc đến lá.

Ví dụ

Bài toán 8 đồng xu: Cho 8 đồng xu nhìn bè ngoài rất giống nhau, trong đó có 7 đồng xu thật nặng như nhau, và một đồng xu giả nhẹ hơn 7 đồng còn lại. Bằng một cái cân thăng bằng, hãy xác định đồng xu giả.

Giải:

Ta xây dựng cây quyết định từ ngữ cảnh của bài toán như sau: Mỗi lần cân sẽ có 3 trạng thái: bên trái nặng hơn, bên phải nặng hơn, và thăng bằng. Vậy cây quyết định mà chúng ta xây dựng là cây tam phân. Cây có ít nhất 8 lá vì có 8 kết cục có thể có và mỗi quyết định cần biểu diễn bằng 1 lá. Khi đó, số lần cân nhiều nhất để xác định đồng xu giả là chiều cao h của cây. Ta có $h \geq \lceil \log_3 8 \rceil = 2$. Sau đây là một cách cân và cây quyết định tương ứng:



Hình 8.11: Bài toán 8 đồng xu

8.4.4. Các phương pháp duyệt cây

Thuật toán viếng thăm mọi đỉnh của một cây có gốc có thứ tự đúng một lần một cách có hệ thống gọi là thuật toán duyệt cây. Dưới đây sẽ

giới thiệu ba thuật toán được sử dụng thường xuyên nhất: Duyệt tiền tự (Preorder traversal), Duyệt trung tự (Inorder traversal), Duyệt hậu tự (Postorder traversal).

Thuật toán duyệt tiền tự

Kiểu duyệt này trước tiên là thăm nút gốc, sau đó thăm các nút của các cây con từ trái qua phải.

```
void Preorder(cây thứ tự có gốc T);
{
    r = gốc của T;
    Thăm r;
    for (Mỗi cây con c của r từ trái sang phải)
    {
        T(c) = Cây con với gốc c;
        Preorder(T(c));
    }
}
```

Thuật toán duyệt trung tự

Kiểu duyệt này trước tiên là thăm các nút của cây con bên trái nhất, sau đó thăm nút gốc rồi đến các nút của các cây con còn lại theo thứ tự từ trái qua phải.

```
void Inorder(cây thứ tự có gốc T)
{
    r = gốc của T;
    if (r là lá) Thăm r;
    else
    {
        s = con đầu tiên từ trái sang phải của r;
        T(s) = Cây con với gốc s;
        Inorder(T(s));
        Thăm r;
        for (Mỗi cây con c của r từ trái sang phải trừ s)
```

```
T(c) = Cây con với gốc c;
```

```
Inorder(T(c));
```

```
}
```

```
}
```

Thuật toán duyệt hậu tự

Kiểu duyệt này trước tiên là thăm các nút của các cây con từ trái qua phải. Cuối cùng là thăm nút gốc.

```
void Postorder(cây thứ tự có gốc T);
```

```
{
```

```
r = gốc của T;
```

```
for (Mỗi cây con c của r từ trái sang phải)
```

```
{
```

```
T(c) = Cây con với gốc c;
```

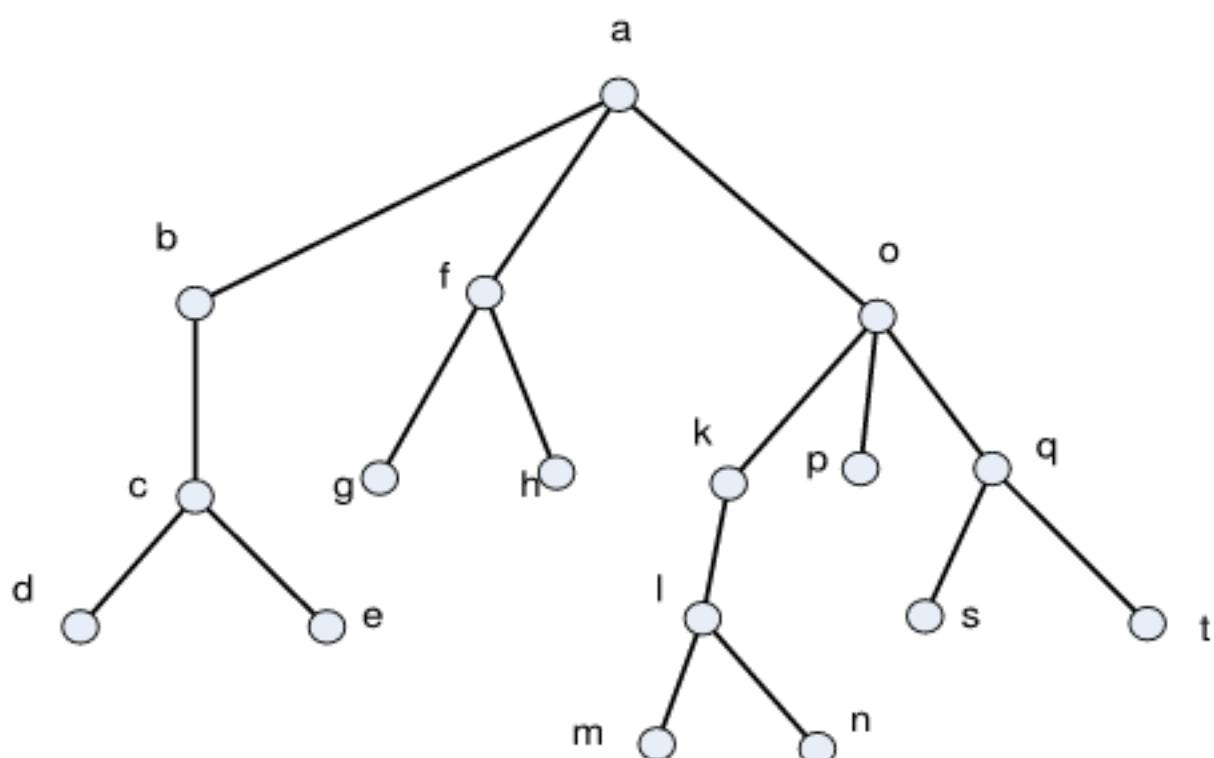
```
Postorder(T(c));
```

```
}
```

```
Thăm r;
```

```
}
```

Ví dụ:



Hình 8.12: Các phương pháp duyệt cây

Thực hiện các phương pháp duyệt khác nhau:

- + Duyệt tiền tự: a, b, c, d, e, f, g, h, o, k, l, m, n, p, q, s, t
- + Duyệt trung tự: d, c, e, b, a, g, f, h, m, l, n, k, o, p, s, q, t
- + Duyệt hậu tự: d, e, c, b, g, h, f, m, n, l, k, p, s, t, q, o, a

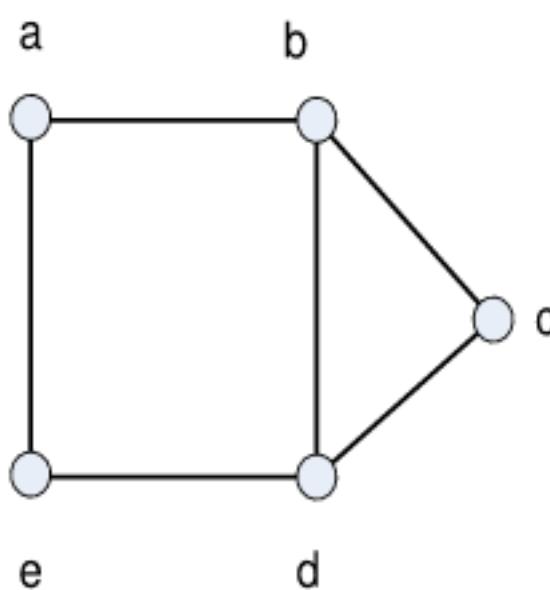
Chú ý:

Nếu cho phép duyệt các cây con từ phải sang trái ta sẽ nhận được thêm 3 thuật toán duyệt cây nữa:

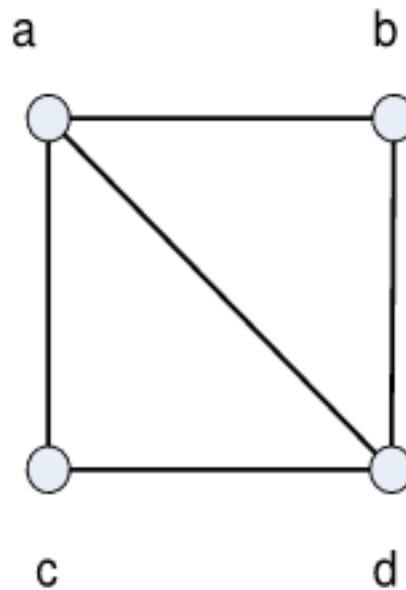
- + Duyệt tiền tự: a, o, q, t, s, p, k, l, n, m, f, h, g, b, c, e, d
- + Duyệt trung tự: t, q, s, o, p, n, l, m, k, a, h, f, g, e, c, d, b
- + Duyệt hậu tự: t, s, q, p, n, m, l, k, o, h, g, f, e, d, c, b, a

BÀI TẬP CHƯƠNG 8

1. Cho T là một cây 5-phân đầy đủ có 156 đỉnh. Hỏi T có bao nhiêu lá?
2. Cho G là đồ thị đơn vô hướng liên thông có n đỉnh và m cạnh. Hỏi cần phải xóa đi bao nhiêu cạnh trong G để G trở thành 1 cây?
3. Khi nào thì một cạnh của đồ thị G luôn luôn nằm trong tất cả các cây khung của nó?
4. Hãy vẽ tất cả các cây khung của đồ thị sau

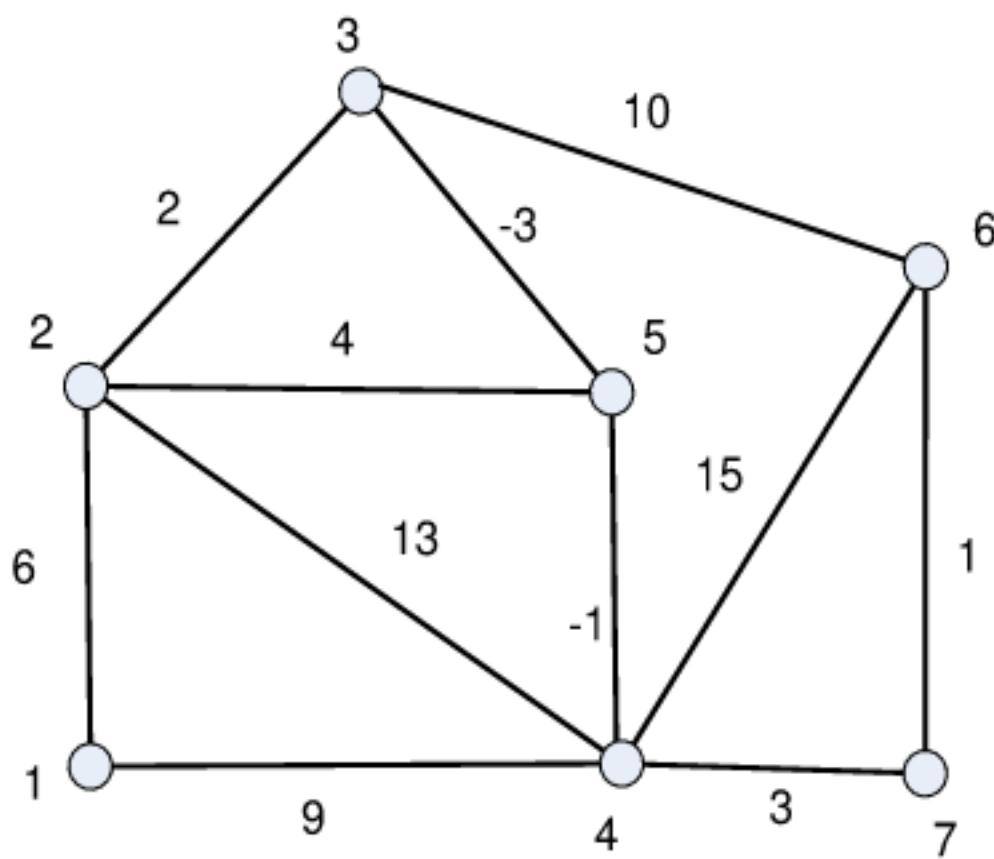


Hình 8.13



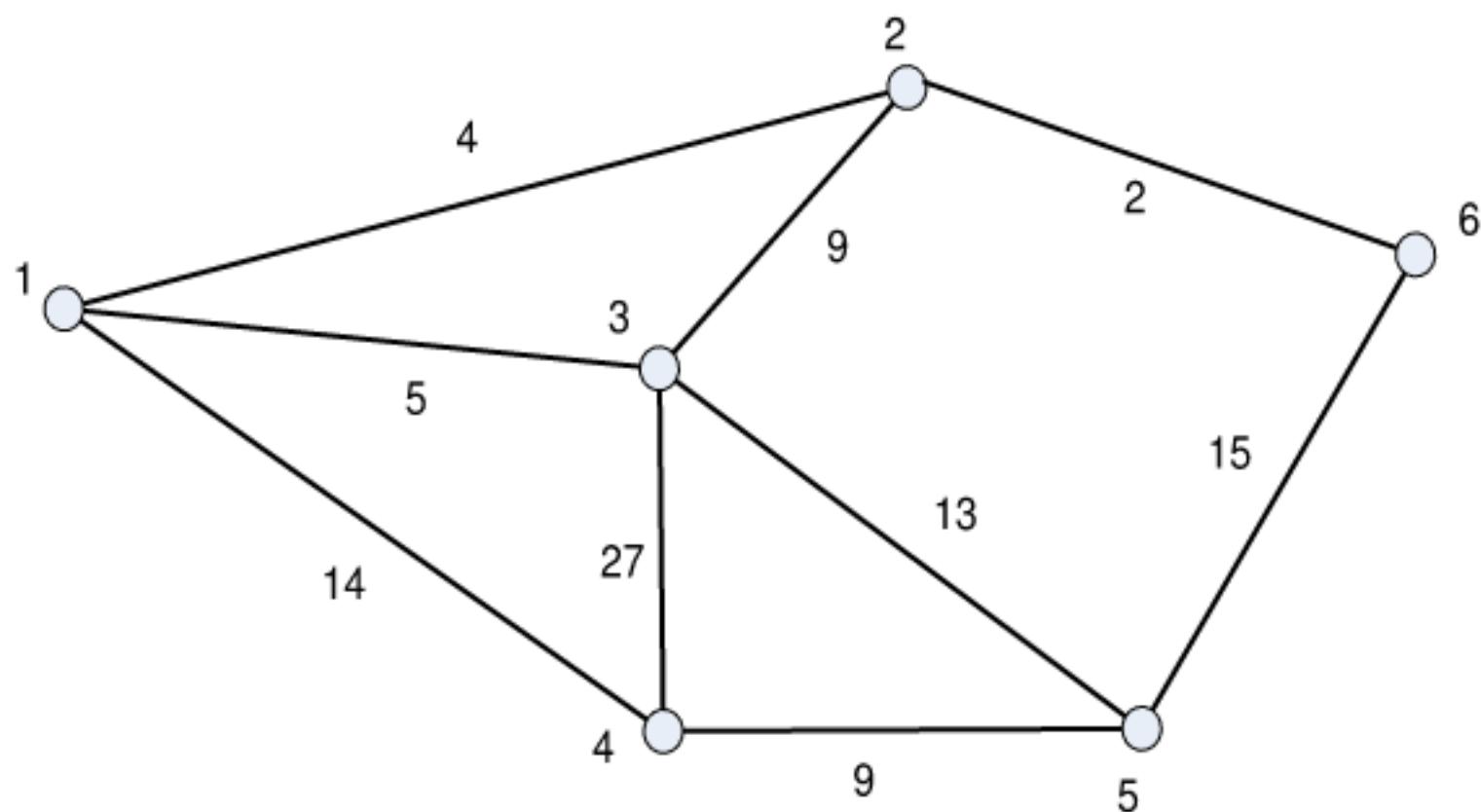
Hình 8.14

5. Áp dụng thuật toán Kruskal, tìm cây khung nhỏ nhất cho đồ thị sau



Hình 8.15

6. Áp dụng thuật toán Prim tìm cây khung nhỏ nhất cho đồ thị sau. Đỉnh chọn đầu tiên là đỉnh 3.



Hình 8.16

7. Cho ma trận kề của một đồ thị vô hướng. Hãy viết chương trình kiểm tra xem nó có phải là cây hay không.
8. Cho ma trận kề của một cây có gốc và một đỉnh của cây. Hãy viết chương trình tìm chiều cao của cây.
9. Cho danh sách các phần tử là các số nguyên. Hãy viết chương trình xây dựng cây tìm kiếm nhị phân chứa các phần tử này.
10. Cho một đồ thị vô hướng được biểu diễn dưới dạng ma trận kề. Hãy viết chương trình cho phép tìm cây khung nhỏ nhất của đồ thị.
- Sử dụng thuật toán Kruskal
 - Sử dụng thuật toán Prim

Chương 9

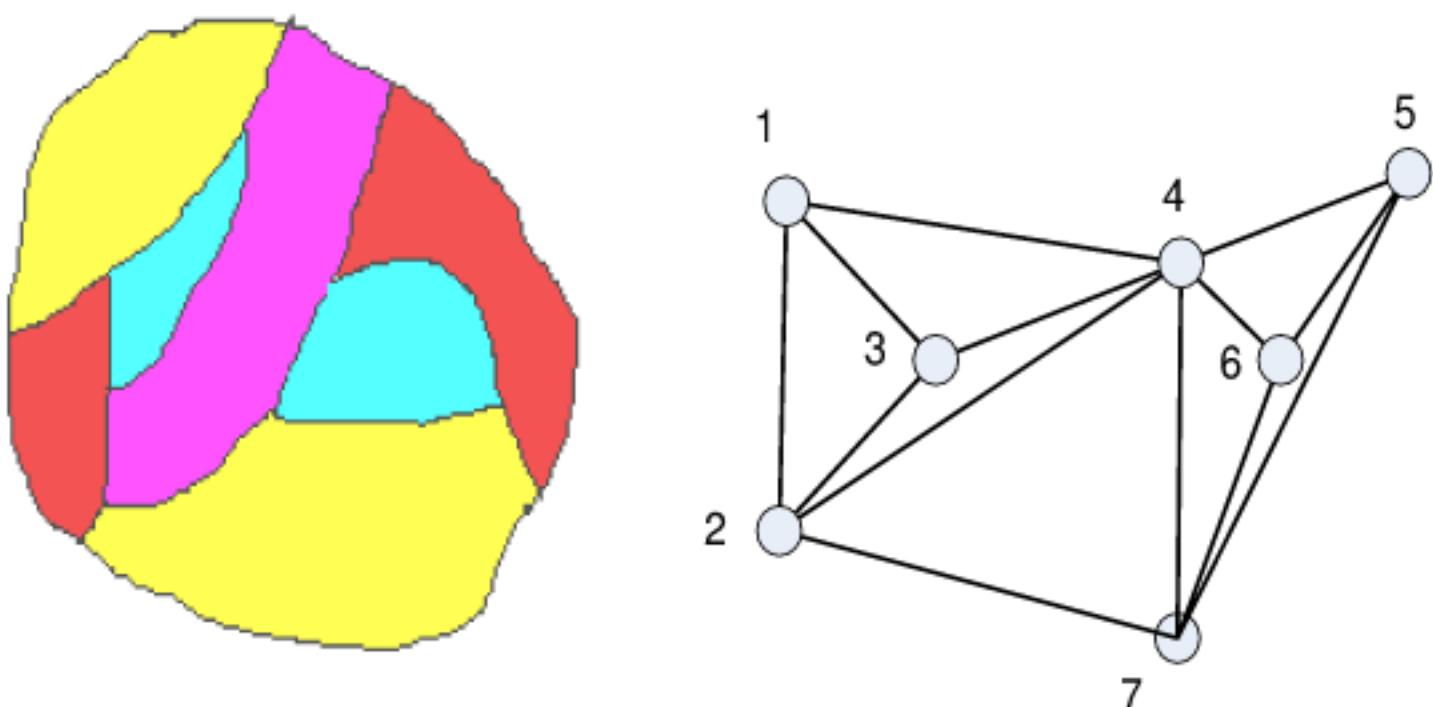
TÔ MÀU ĐỒ THỊ

Chương này trình bày phương pháp tô màu đồ thị (graph coloring) và ứng dụng của nó trong thực tế. Tô màu đồ thị là trường hợp đặc biệt của phương pháp gán nhãn cho đồ thị mà trong đó mỗi đỉnh, mỗi cạnh hay mỗi miền của đồ thị có thể được gán bởi một màu hay một tập hợp màu nào đó. Trong phạm vi của giáo trình này, ta chỉ xét đến vấn đề tô màu cho đỉnh của đồ thị và vận dụng nó để giải quyết các bài toán khác nhau.

9.1. MỞ ĐẦU

Giả sử có một bản đồ cần được tô màu. Để có thể phân biệt các miền với nhau thì hai miền chung biên giới phải được tô bằng hai màu khác nhau. Bài toán đặt ra là: *Hãy tính số màu tối thiểu cần thiết để tô màu cho mọi bản đồ!*

Từ bài toán thực tế này, ta sẽ tìm cách chuyển nó thành bài toán trên đồ thị. Xem mỗi miền của bản đồ là một đỉnh của đồ thị và mỗi đường biên giới giữa hai miền là một cạnh nối hai đỉnh tương ứng với hai miền đó. Bài toán tô màu bản đồ quy về bài toán tô màu các đỉnh của đồ thị.



Hình 9.1

9.1.1. Định nghĩa 1

Tô màu một đơn đồ thị là sự gán màu cho các đỉnh của nó sao cho hai đỉnh kề nhau được gán màu khác nhau.

9.1.2. Định nghĩa 2

Số màu của một đồ thị là số tối thiểu các màu cần thiết để tô màu đồ thị này.

9.2. ĐỊNH LÝ BỐN MÀU

✓ Định lý

Số màu của một đồ thị phẳng là không lớn hơn 4.

Định lý bốn màu được đưa ra lần đầu tiên vào năm 1850, nhưng mãi cho đến năm 1976 mới được chứng minh bởi hai nhà toán học người Mỹ là Kenneth Appel và Wolfgang Haken. Định lý bốn màu chỉ có thể áp dụng cho đồ thị phẳng. Các đồ thị không phẳng có thể có số màu lớn hơn.

Khi chứng minh một đồ thị có thể tô ít nhất n màu, chúng ta phải thực hiện hai việc. Đầu tiên chúng ta phải chứng tỏ rằng đồ thị có thể được tô bằng n màu bằng cách thực hiện công việc tô màu đồ thị. Sau đó, chúng ta phải chứng tỏ không thể tô màu đồ thị đó với ít hơn n màu.

9.3. ĐỒ THỊ HAI MÀU

Cách để nhận biết một đồ thị có thể được tô bởi ít nhất hai màu (đồ thị 2-màu) được thể hiện qua định lý sau:

✓ Định lý

Một đồ thị G là 2-màu khi và chỉ khi G không chứa một chu trình lẻ nào.

Chứng minh:

Giả sử G là đồ thị 2-màu, ta sẽ chứng minh rằng G không chứa chu trình lẻ.

Thật vậy nếu G có chu trình lẻ $C=(v_1, v_2, \dots, v_{2n+1}, v_1)$, do C chỉ được tô bởi 2 màu, vì thế các đỉnh lẻ sẽ được tô bằng cùng 1 màu. Nhưng lúc đó v_1 và v_{2n+1} là 2 đỉnh kề nhau có cùng màu. Vô lý !!! (đpcm).

Giả sử G không chứa chu trình lẻ nào, ta sẽ chứng minh rằng G là đồ thị 2 - màu.

Chọn 1 đỉnh r làm gốc và tô nó màu đỏ. $\forall x \in V$ sẽ được tô màu đỏ nếu đường đi ngắn nhất từ x tới r có độ dài chẵn. Trái lại, tô x màu xanh.

Ta sẽ chứng minh rằng đỉnh x, y của cạnh (x,y) bất kỳ được tô hai màu khác nhau. Trái lại giả sử x và y là 2 đỉnh của cạnh (x,y) nào đó được tô cùng màu. Gọi P_x và P_y là các đường đi ngắn nhất từ r tới x và y tương ứng. Rõ ràng $|P_x|$ và $|P_y|$ là cùng chẵn hay cùng lẻ vì x và y cùng màu. Xét 2 trường hợp có thể xảy ra:

+ Trường hợp 1: P_x và P_y không có chung cạnh. Ta có $P_x + (x,y) + P_y$ là chu trình có số cạnh lẻ. (Mâu thuẫn giả thiết).

+ Trường hợp 2: P_x và P_y có chung k cạnh từ đỉnh a tới đỉnh b . Ta sẽ nhận được hai chu trình C_a, C_b và k cạnh chung. Ta có $P_x + (x,y) + P_y$ có số lẻ cạnh mà:

$$|P_x + (x,y) + P_y| = |C_a| + |C_b| + 2k$$

Do đó, một trong hai chu trình C_a hoặc C_b sẽ có số cạnh lẻ (mâu thuẫn giả thiết). Suy ra đầu mút cạnh (x, y) bất kỳ của đồ thị phải được tô hai màu khác nhau $\Rightarrow G$ là đồ thị 2-màu (đpcm).

9.4. THUẬT TOÁN SEQUENTIALCOLOR

Số màu của đồ thị phẳng bất kỳ là không quá 4. Vậy đối với đồ thị không phẳng sẽ có số màu là bao nhiêu? Việc nhận biết đồ thị 2-màu đã được giải quyết. Tuy vậy, việc nhận biết đồ thị k -màu với $k > 2$ vẫn chưa có lời giải.

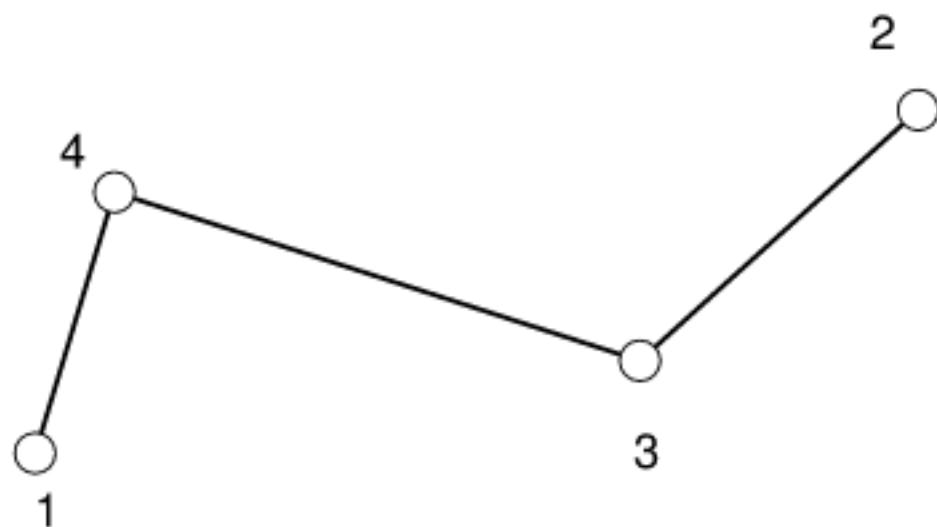
Thuật toán SequentialColor cho phép tô màu các đồ thị (có thể không phẳng) với số màu ít nhất có thể, từ đó tính được số màu của đồ thị.

Ý tưởng thực hiện thuật toán như sau: Sắp xếp các đỉnh theo thứ tự bất kỳ từ 1 đến $|V|$. Tại mỗi đỉnh $v \in V$, gán màu đầu tiên có sẵn mà chưa được gán cho một đỉnh nào kề v .

Sau đây là các bước thực hiện của thuật toán

1. Xếp các đỉnh theo thứ tự bất kỳ $1, 2, \dots, n$.
2. Tạo tập L_i - tập các màu có thể gán cho đỉnh i .
3. Bắt đầu tô từ đỉnh 1.
4. Lần lượt với $k \in \{1, \dots, n\}$ tô màu đầu tiên của L_k cho k .
5. $\forall j > k$ và j kề k loại bỏ trong L_j màu đã được tô cho k .
6. Giải thuật dừng lại khi tất cả các đỉnh đã được tô. Số màu đã sử dụng chính là chỉ số lớn nhất của màu đã dùng.

Ví dụ:



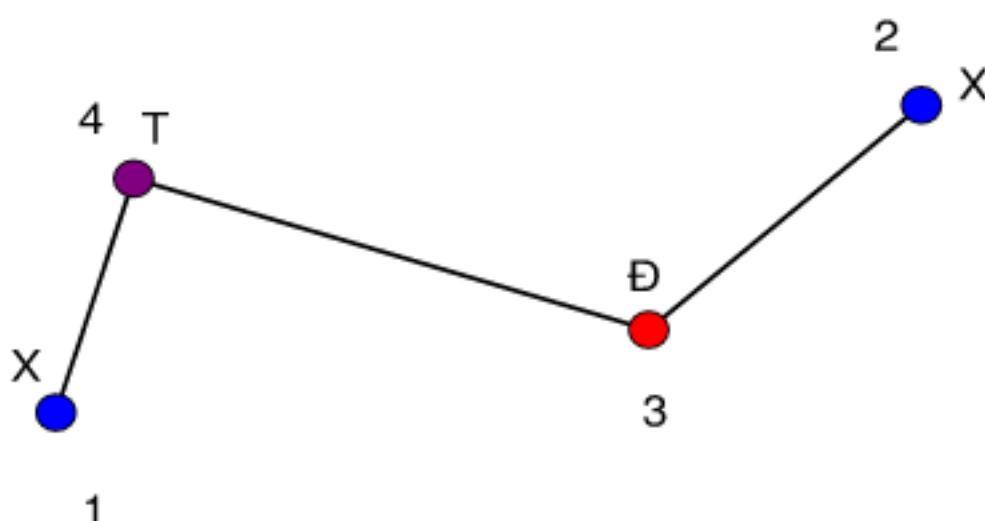
Hình 9.2

Áp dụng thuật toán tô màu cho đồ thị trên, vì đồ thị có 4 đỉnh nên số màu tối đa cần dùng là 4. Ta lấy 4 màu: Xanh (X), Đỏ (Đ), Tím (T), Vàng (V) để tô màu cho đồ thị. L₁, L₂, L₃, L₄ lần lượt là tập các màu có thể tô cho các đỉnh 1, 2, 3, 4. Kết quả của thuật toán phụ thuộc vào thứ tự tô các đỉnh. Sau đây ta sẽ tô đồ thị theo các thứ tự khác nhau:

- Tô màu đồ thị theo thứ tự: 1, 2, 3, 4

Các bước	1	2	3	4	Màu tô
Khởi tạo	X, Đ, T, V				
B1	X	X, Đ, T, V	X, Đ, T, V	Đ, T, V	1 - Xanh
B2		X	Đ, T, V	Đ, T, V	2 - Xanh
B3			Đ	T, V	3 - Đỏ
B4				T	4 - Tím

Như vậy, đồ thị có thể được tô bởi 3 màu

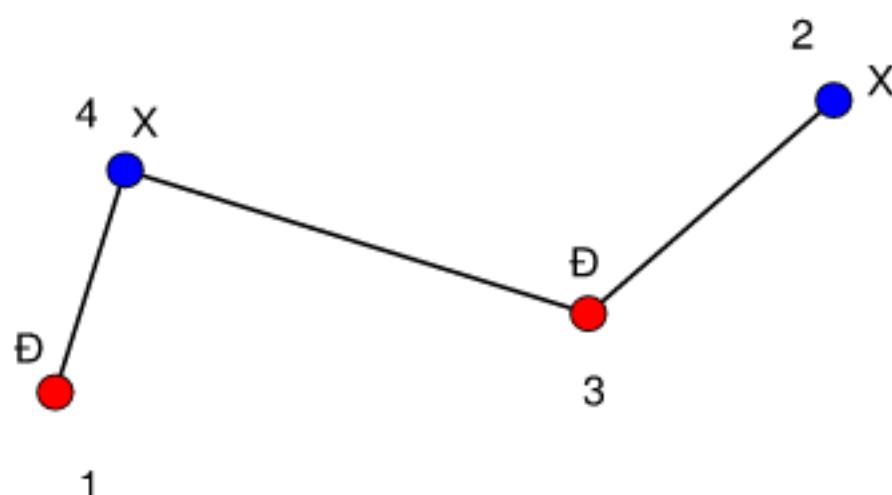


Hình 9.3

- Tô màu đồ thị theo thứ tự: 4, 3, 1, 2

Các bước	4	3	1	2	Màu tô
Khởi tạo	X, Đ, T, V				
B1	X	Đ, T, V	Đ, T, V	X, Đ, T, V	4 - Xanh
B2		Đ	Đ, T, V	X, T, V	3 - Đỏ
B3			Đ	X, T, V	1 - Đỏ
B4				X	2 - Xanh

Trường hợp này đồ thị được tô bởi 2 màu



Hình 9.4

Nhận xét

Thuật toán SequentialColor là dạng thuật toán tham lam nên lời giải tìm được chưa chắc đã tối ưu, như ta đã thấy ở ví dụ trên. Độ phức tạp của thuật toán là $O(n^2)$, với n là số đỉnh của đồ thị.

Người ta đã chứng minh được rằng: *Luôn tồn tại một cách sắp xếp thứ tự các đỉnh của đồ thị sao cho khi áp dụng thuật toán SequentialColor theo thứ tự đó sẽ cho số màu tối thiểu cần thiết để tô màu đồ thị.*

Việc sắp xếp các đỉnh của đồ thị theo thứ tự giảm dần của bậc đỉnh sẽ cho chúng ta kết quả khá tốt. Đây là cách mà trong thực tế thường được sử dụng.

9.5. NHỮNG ỨNG DỤNG CỦA BÀI TOÁN TÔ MÀU ĐỒ THỊ

Bài toán tô màu đồ thị có nhiều ứng dụng trong thực tế như: bài toán lập lịch thi, bài toán phân chia tần số, lập kế hoạch tối ưu cho các lĩnh vực khác nhau, phân bổ thanh ghi trong máy tính (register

allocation), đối sách mẫu (pattern matching). Sau đây là một vài bài toán áp dụng kỹ thuật tô màu đồ thị.

9.5.1. Bài toán lập lịch thi

Hãy lập lịch thi trong trường đại học sao cho không có sinh viên nào có 2 môn thi cùng một thời gian.

Giải:

Ta giải bài toán bằng lý thuyết đồ thị như sau. Xem các môn thi là các đỉnh. Hai đỉnh nào đó sẽ được nối với nhau nếu có ít nhất 1 sinh viên thi cả hai môn tương ứng với các đỉnh này. Thời gian thi sẽ được thể hiện bởi các màu khác nhau. Khi đó việc lập lịch tương ứng với việc tô màu đồ thị này.

Ví dụ: Có 7 môn thi: Toán (t), Anh Văn (a), Lý (l), Pascal (p), Tin học đại cương (h), Tiếng Việt thực hành (v), Visual Basic (b).

Các cặp môn thi có chung sinh viên là: (t,a), (t,l), (t,p), (t,b),(a,l), (a,p), (a,h), (a,b), (l,p), (l,b), (p,h), (p,v), (h,b), (v,b).

Thứ tự duyệt các đỉnh như sau: a, l, p, h, v, b, t.

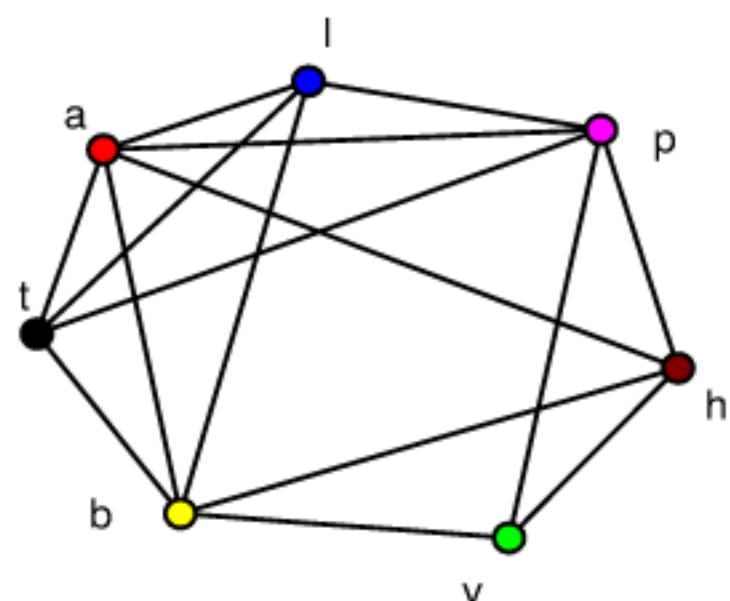
Vì đồ thị có 7 đỉnh nên ta xét 7 màu để tô.

Áp dụng thuật toán SequentialColor ta có kết quả tô màu đồ thị như sau:

a	l	p	h	v	B	t
Đỏ	Xanh	Tím	Nâu	Lá cây	Vàng	Đen

Từ đó ta có kết quả xếp lịch thi:

Đợt thi	Môn thi
1	Anh Văn
2	Lý, Tin học đại cương
3	Pascal, Visual Basic
4	Tiếng Việt thực hành, Toán



Hình 9.5

9.5.2. Bài toán phân chia tần số

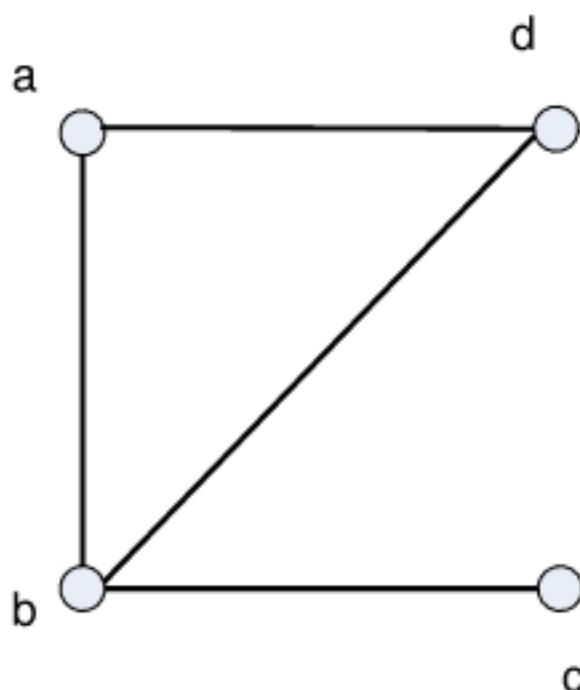
Các kênh truyền hình từ số 2 đến 13 được phân chia cho các đài truyền hình sao cho 2 đài gần nhau dưới 150km có 2 kênh khác nhau.

Giải:

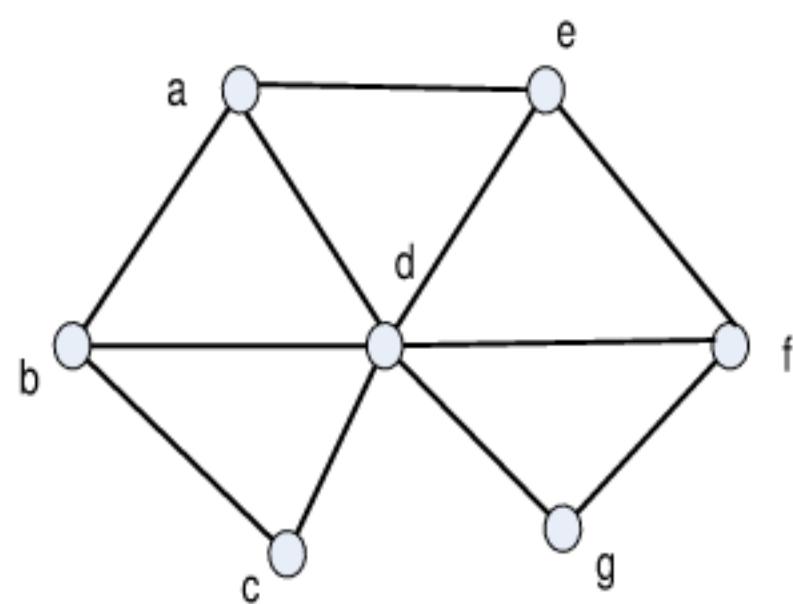
Tương tự bài toán lập lịch thi, ta chuyển bài toán từ thực tế về bài toán trong lý thuyết đồ thị. Coi mỗi đài phát là 1 đỉnh. Hai đài gần nhau dưới 150km là hai đỉnh được nối với nhau bởi 1 cạnh. Việc phân chia kênh được chuyển về công việc tô màu cho đồ thị. Trong đó, mỗi màu biểu thị cho một kênh. Từ đây, ta có thể áp dụng thuật toán SequentialColor để tô màu đồ thị và suy ra kết quả của bài toán.

BÀI TẬP CHƯƠNG 9

- Chứng minh rằng, nếu đồ thị G có chứa một chu trình có độ dài lẻ thì số màu của G ít nhất là 3.
- Tìm số màu của các đồ thị sau:

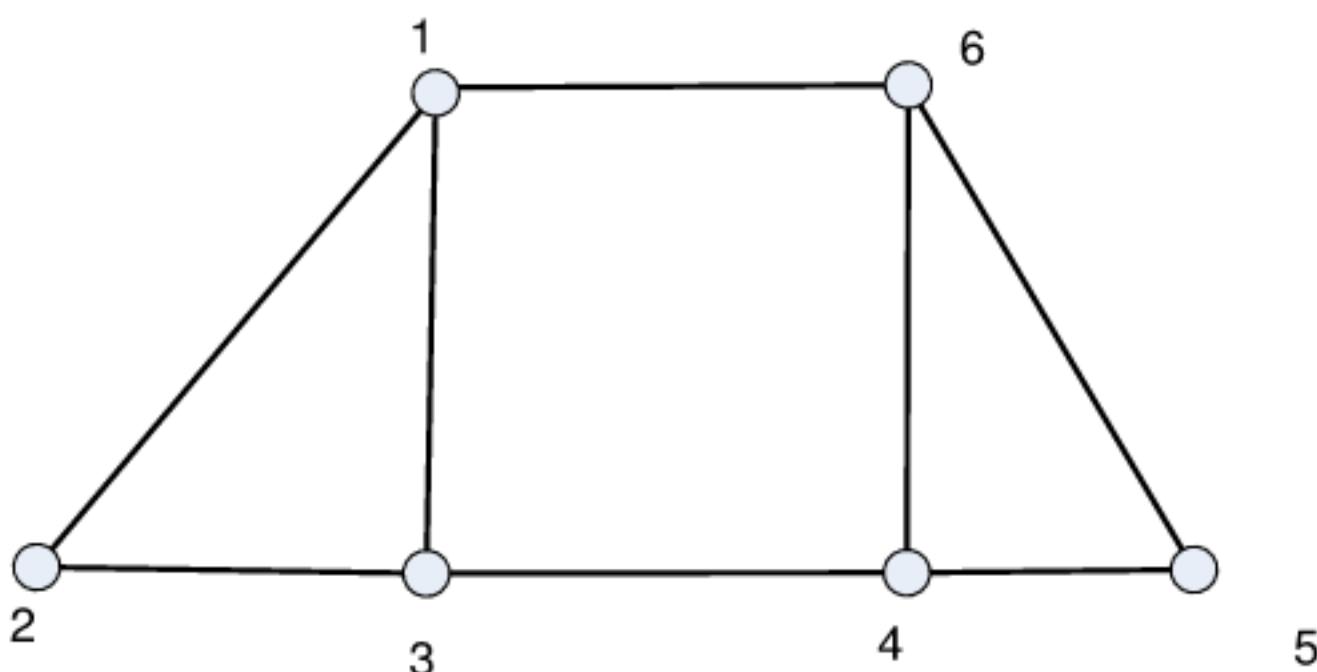


Hình 9.6



Hình 9.7

- Tính số màu của đồ thị W_n .
- Hãy chỉ ra rằng một đơn đồ thị với số màu bằng 2 là đồ thị phân đôi.
- Áp dụng thuật toán SequentialColor tô màu cho đồ thị sau



Hình 9.8

- Cho danh sách các cạnh của một đồ thị vô hướng. Hãy viết chương trình tô màu đồ thị theo thuật toán SequentialColor.

7. Ở một đơn vị thường xuyên có nhiều cuộc họp được tổ chức. Trong thời gian tới, công ty có n cuộc họp A_1, A_2, \dots, A_n với các chủ đề là C_1, C_2, \dots, C_m . Số người đăng ký cho cuộc họp thứ i là m_i . Cụ thể như sau:

Chủ đề	C_1	C_1	C_1	C_2	C_2	C_2	C_2	C_2
Tên cuộc họp	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
Người đăng ký	10	15	30	35	43	12	53	10

Nếu hai cuộc họp có tổng số người đăng ký lớn hơn 30 và có cùng một chủ đề thì không được phép tổ chức cùng một ngày. Hãy viết chương trình lập lịch sao cho số ngày họp là ít nhất.

Chương 10

ĐƯỜNG ĐI NGẮN NHẤT

Trong thực tế, ta thường gặp những bài toán như: chọn một hành trình tiết kiệm nhất (về chi phí, thời gian,...) trên một mạng giao thông đường bộ, đường thủy, hay hàng không; Bài toán lập lịch thi công các công đoạn trong một công trình lớn sao cho tiết kiệm chi phí và thời gian nhất; Bài toán chọn lựa đường truyền tin tối ưu trong mạng thông tin, v.v. Để giải quyết hiệu quả các bài toán này, người ta quy về dạng bài toán tìm đường đi ngắn nhất trong đồ thị. Trong chương này, chúng ta sẽ xem xét một số thuật toán tìm đường đi ngắn nhất thông dụng trong đồ thị.

10.1. ĐỊNH NGHĨA

Cho $G = (V, E)$ là một đồ thị có trọng số. Ký hiệu $a(u, v) \in \mathbb{R}$ là trọng số (độ dài) của cạnh (u, v) . Nếu dãy $v_0, v_1, v_2, \dots, v_p$ là một đường đi trên G thì độ dài của nó bằng tổng trọng số của các cạnh trong đường đi đó.

$$\text{Độ dài của đường đi } v_0, v_1, v_2, \dots, v_p = \sum_{i=1}^p a(v_{i-1}, v_i).$$

Giả sử có nhiều đường đi từ v_0 đến v_p , đường đi ngắn nhất là đường đi có độ dài nhỏ nhất.

Điều kiện tiên quyết cho bài toán tìm đường đi ngắn nhất giữa 2 đỉnh trong đồ thị có lời giải là đồ thị phải không có chu trình âm. Thật vậy, nếu đồ thị có chu trình âm thì ta có thể hướng đường đi vào chu trình này và sau đó có thể đi một số lần đủ lớn vòng quanh chu trình âm để được một đường đi với độ dài đủ nhỏ (!). Lúc đó sẽ không thể có đường đi ngắn nhất. Do vậy ta mặc nhiên coi tất cả các đồ thị được xem xét trong chương này đều không có chu trình âm.

10.2. THUẬT TOÁN DIJKSTRA

Thuật toán Dijkstra dùng để tìm đường đi ngắn nhất từ đỉnh s đến các đỉnh còn lại trong đồ thị. Thuật toán được áp dụng cho đồ thị có trọng số các cung không âm.

10.2.1. Ý tưởng thuật toán

- Đầu vào

- Đồ thị có hướng $G=(V, E)$ với n đỉnh.
- $s \in V$ là đỉnh xuất phát.
- $a[u, v] \geq 0$ với $u, v \in V$ là ma trận trọng số

- Đầu ra

- Khoảng cách từ s đến tất cả các đỉnh còn lại $d[v]$, $v \in V$.
- $\text{Truoc}[v]$, $v \in V$ là đỉnh nằm trước v trong đường đi ngắn nhất từ s đến v .

Các bước thuật toán như sau:

Bước 1: Đặt nhãn cho tất cả các đỉnh trong tập $T = V \setminus \{s\}$.

Bước 2: Nếu $T = \emptyset$ thì đi đến Bước 5.

Trái lại chọn $u \in T$ gần s nhất ($d(u)$: min).

Bước 3: Loại u ra khỏi T : $T = T \setminus \{u\}$; Cố định nhãn của u .

Bước 4: Gán lại nhãn các đỉnh của T (theo đường đi ngắn nhất từ s qua u).

Quay lại Bước 2

Bước 5: Xuất các nhãn cố định (chính là khoảng cách ngắn nhất).

10.2.2. Cài đặt thuật toán

Sau đây là thuật toán được cài đặt bằng mã giả, người đọc có thể tham khảo mã nguồn cài đặt bằng ngôn ngữ lập trình C ở phần phụ lục của giáo trình.

```
void Dijkstra;  
{  
    for (v ∈ V) /* Khởi tạo d và Truoc */  
    {  
        d[v] = a[s,v];  
        Truoc[v] = s;  
    }  
    d[s] = 0;  
    T = V \ {s};
```

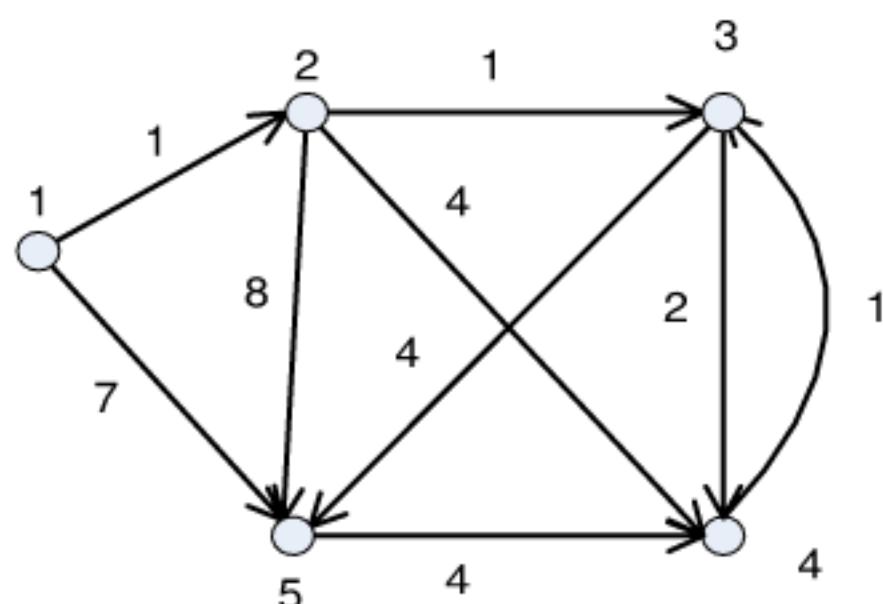
```

while ( $T \neq \emptyset$ )
{
    Tìm  $u \in T$  sao cho  $d(u) = \min \{ d(z): z \in T \}$ 
     $T = T \setminus \{u\}$ ; /* Cố định nhãn của  $u$  */
    for ( $v \in T$ ) do
        if ( $d[v] > d[u] + a[u,v]$ ) then
            {
                 $d[v] = d[u] + a[u,v];$ 
                Truoc[v] = u;
            }
    }
} /* Độ phức tạp của thuật toán là  $O(n^2)$  */

```

Ví dụ:

Sử dụng thuật toán Dijkstra, tìm đường đi ngắn nhất từ đỉnh 1 đến tất cả các đỉnh còn lại trong đồ thị sau:



	1	2	3	4	5
1	∞	1	∞	∞	7
2	∞	∞	1	4	8
3	∞	∞	∞	2	4
4	∞	∞	1	∞	∞
5	∞	∞	∞	4	∞

Hình 10.1

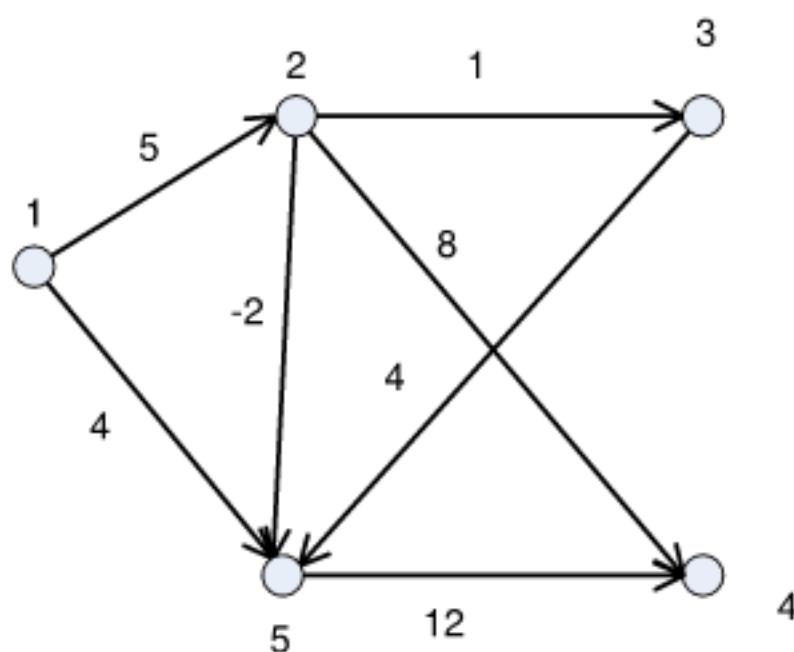
Các bước thực hiện thuật toán được minh họa qua bảng sau:

T	$d[2]$, Truoc[2]	$d[3]$, Truoc[3]	$d[4]$, Truoc[4]	$d[5]$, Truoc[5]
2, 3, 4, 5	1, 1	∞ , 1	∞ , 1	7, 1
3, 4, 5		2, 2	5, 2	7, 1
4, 5			4, 3	6, 3
5				6, 3
\emptyset	1, 1	2, 2	4, 3	6, 3

Từ kết quả thực hiện của giải thuật, ta có thể biết được độ dài và đường đi từ đỉnh 1 đến các đỉnh còn lại. Chẳng hạn, đường đi từ đỉnh 1 đến đỉnh 4 có độ dài là 4. Danh sách các đỉnh trên đường đi có thể biết được bằng cách truy xuất mảng Truoc[]: $\text{Truoc}[4]=3$, $\text{Truoc}[3]=2$, $\text{Truoc}[2]=1$. Như vậy, đường đi là: $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$.

Một điểm lưu ý là, thuật toán Dijkstra không áp dụng cho đồ thị có cung có trọng số âm vì trong một số trường hợp, cung có trọng số âm khiến giải thuật thực hiện không cho kết quả chính xác.

Chẳng hạn đối với đồ thị dưới đây, khi áp dụng thuật toán tìm đường đi ngắn nhất từ 1 đến 5 sẽ có kết quả là 4, trong khi kết quả chính xác là: $5-2=3$.



Hình 10.2

Nhận xét

Thuật toán Dijkstra được đánh giá là thực hiện nhanh, tốn ít chi phí tính toán. Lý do: vì sau mỗi bước lặp của giải thuật, ta sẽ nhận được kết quả đường đi ngắn nhất đến một đỉnh nào đó. Đỉnh này sẽ được loại bỏ, không tham gia tính toán trong các bước sau. Chính vì thế, số phép tính toán giảm dần theo thời gian. Tuy nhiên, đó cũng là nguyên nhân khiến thuật toán có khả năng cho kết quả sai khi đồ thị có cung trọng số âm như trong ví dụ hình 10.2. Một giải thuật khác cũng cho phép tìm đường đi ngắn nhất từ 1 đỉnh đến các đỉnh còn lại của đồ thị nhưng cho phép đồ thị có cung trọng số âm là thuật toán Ford-Bellman.

10.3. THUẬT TOÁN FORD-BELLMAN

Thuật toán Ford-Bellman dùng để tìm đường đi ngắn nhất từ một đỉnh s đến tất cả các đỉnh còn lại của đồ thị.

10.3.1. Ý tưởng thuật toán

- Đầu vào:

- Đồ thị có hướng $G = (V, E)$ với n đỉnh.

- $s \in V$ là đỉnh xuất phát.

- $a[u,v]$ với $u, v \in V$ là ma trận trọng số

- Đầu ra:

- Khoảng cách từ s đến tất cả các đỉnh còn lại $d[v], \forall v \in V$.

- $Truoc[v], \forall v \in V$ là đỉnh nằm trước v trong đường đi ngắn nhất từ s đến v .

Cho đồ thị có hướng, có trọng số $G = (V, E)$. Trọng số các cung của G được tính như sau:

- $TrongSo(u, v) = \infty$ nếu $(u, v) \notin E$.

- $TrongSo(u, v) = a(u, v)$ nếu cung $(u, v) \in E$.

Thuật toán tìm đường đi ngắn nhất $d(v)$ từ đỉnh $s, \forall v \in V$:

- + Xét $u \in V$. Nếu $d(u) + TrongSo(u, v) < d(v)$ thì ta thay $d(v) = d(u) + TrongSo(u, v)$.

- + Quá trình này sẽ được lặp lại với tất cả các đỉnh u có thể, với tất cả các đỉnh v có thể cho đến khi không thể làm tốt hơn các giá trị $d(v)$.

10.3.2. Cài đặt thuật toán

Sau đây ta cài đặt thuật toán bằng mã giả.

```
void Ford_Bellman()
{
    for (v ∈ V) /* Khởi tạo d và Truoc */
    {
        d[v] = a[s,v];
        Truoc[v] = s;
    }
    d[s] = 0;
    for (k = 1; k < n-1; k++) /* Thực hiện n - 2 lần */
    /**
     * for (v ∈ V \ {s})
```

```

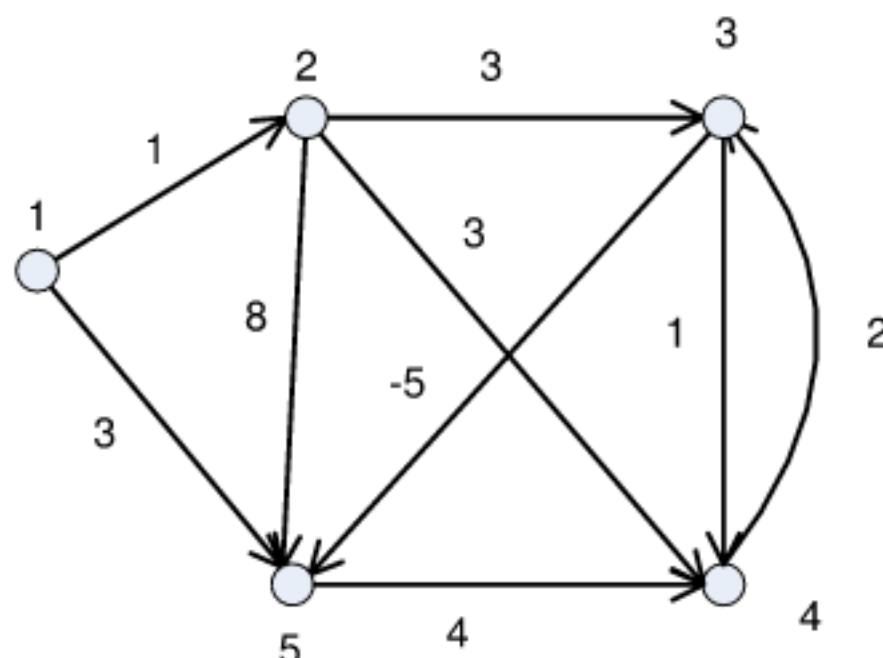
for (u ∈ V)
    if (d[v] > d[u] + a[u,v])
    {
        d[v] = d[u] + a[u,v];
        Truoc[v] = u;
    }
} /* Độ phức tạp của thuật toán là  $O(n^3)$  */

```

Thuật toán trên sẽ thực hiện vòng lặp $***/ n - 2$ lần. Trong thực tế số lần lặp có thể ít hơn và thuật toán có thể dừng lại ngay nếu các giá trị $d[v]$ không thay đổi nữa.

Ví dụ:

Áp dụng thuật toán Ford-Bellman, tìm đường đi ngắn nhất từ đỉnh 1 đến tất cả các đỉnh còn lại của đồ thị sau:



	1	2	3	4	5
1	∞	1	∞	∞	3
2	∞	∞	3	3	8
3	∞	∞	∞	1	-5
4	∞	∞	2	∞	∞
5	∞	∞	∞	4	∞

Hình 10.3

Các bước chạy thuật toán được thể hiện trong bảng sau. Thuật toán DÙNG khi hai dòng kết quả cuối cùng giống nhau (khi đó giá trị $d[v]$ của các đỉnh không được cải tiến nữa).

k	d[5], Truoc[5]	d[4], Truoc[4]	d[3], Truoc[3]	d[2], Truoc[2]
1	3, 1	∞ , 1	∞ , 1	1, 1
2	3, 1	4, 2	4, 2	1, 1
3	-1, 3	4, 2	4, 2	1, 1
4	-1, 3	3, 5	4, 2	1, 1
5	-1, 3	3, 5	4, 2	1, 1

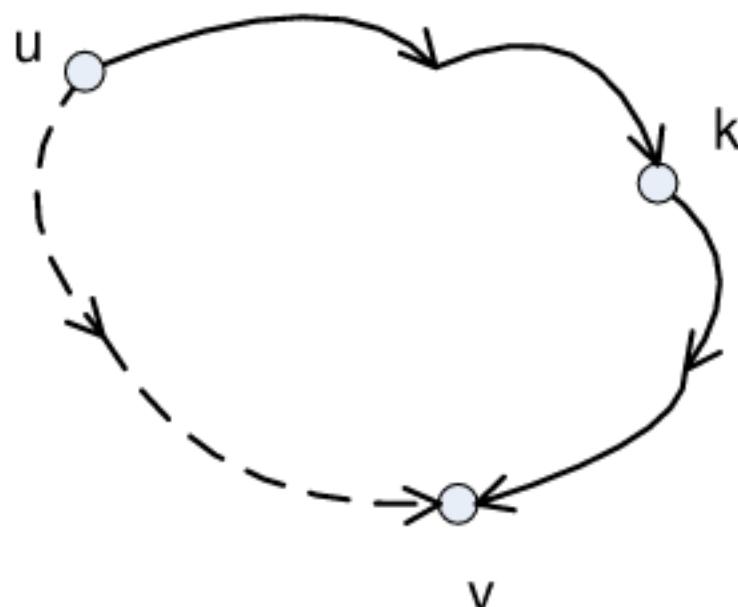
10.4. THUẬT TOÁN FLOYD

Thuật toán Floyd cho phép tìm đường đi ngắn nhất giữa tất cả các cặp đỉnh trong đồ thị.

10.4.1. Ý tưởng thuật toán

Giả sử k là một đỉnh nào đó của đồ thị. Xét các cặp đỉnh u, v. Ta sẽ tìm đường đi ngắn nhất từ u đến v mà có đi qua k theo công thức:

$$a(u, v) = \min (a(u, v), a(u, k) + a(k, v))$$



Hình 10.4

Nếu đường đi hiện có từ u đến v dài hơn đường đi từ u đến k cộng với đường đi từ k đến v thì đường đi từ u đến v hiện tại sẽ được thay thế bởi đường đi mới qua đỉnh k. Đồng thời ta phải cập nhật thông tin lưu vết của các đường đi.

10.4.2. Cài đặt thuật toán

- Đầu vào

- Đồ thị cho bởi ma trận trọng số: $a[i, j]$, $i, j = \overline{1..n}$

- Đầu ra: Hai ma trận

- Ma trận đường đi ngắn nhất giữa các cặp đỉnh:

$$d[i, j], i, j = \overline{1..n}$$

$d[i, j]$ là độ dài đường đi ngắn nhất từ i đến j

- Ma trận ghi nhận đường đi:

$$p[i, j], i, j = \overline{1..n}$$

$p[i, j]$ là đỉnh nằm trước đỉnh j trong đường đi ngắn nhất từ i đến j .

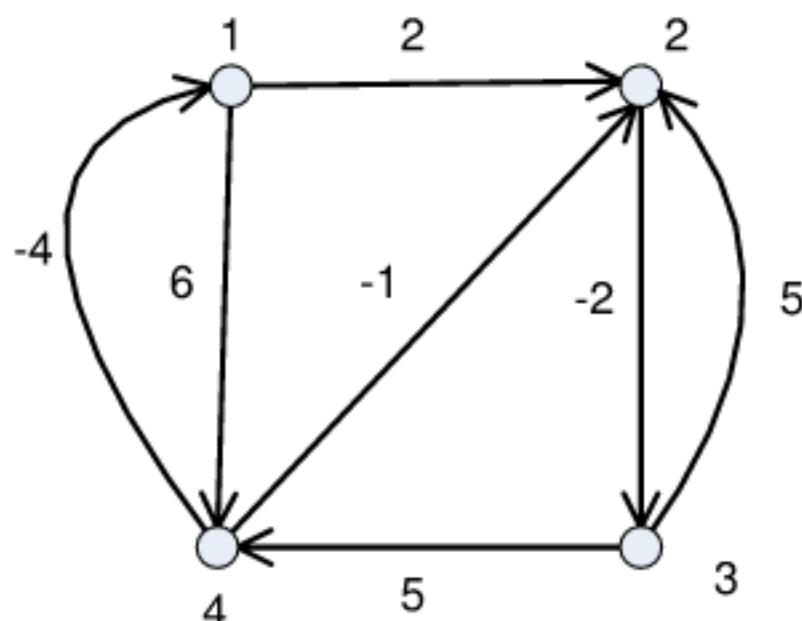
```

void Floyd;
{
    for (i = 1; i <= n; i++)
        for (j = 1; j <= n; j++)
    {
        d[i,j] = a[i,j];
        p[i,j] = i;
    }
    for (k = 1; k <= n; k++)
        for (i = 1; i <= n; i++)
            for (j = 1; j <= n; j++)
                if (d[i,j] > d[i,k] + d[k,j])
    {
        d[i,j] = d[i,k] + d[k,j];
        p[i,j] = p[k,j];
    }
}

```

Ví dụ:

Áp dụng thuật toán Floyd, tìm đường đi ngắn nhất giữa tất cả các cặp đỉnh trong đồ thị sau:



Hình 10.5

- Khởi tạo hai ma trận:

D: Ma trận lưu giá trị đường đi.

P: Ma trận lưu vết.

D₀				
1	2	3	4	
1	∞	2	∞	6
2	∞	∞	-2	∞
3	∞	5	∞	5
4	-4	-1	∞	∞

P₀				
1	2	3	4	
1	1	1	1	1
2	2	2	2	2
3	3	3	3	3
4	4	4	4	4

- Kết quả các bước lặp của thuật toán:

k=1

D₁				
1	2	3	4	
1	∞	2	∞	6
2	∞	∞	-2	∞
3	∞	5	∞	5
4	-4	-2	∞	2

P₁

P₁				
1	2	3	4	
1	1	1	1	1
2	2	2	2	2
3	3	3	3	3
4	4	1	4	1

k=2

D₂				
1	2	3	4	
1	∞	2	0	6
2	∞	∞	-2	∞
3	∞	5	3	5
4	-4	-2	-4	2

P₂

P₂				
1	2	3	4	
1	1	1	2	1
2	2	2	2	2
3	3	3	2	3
4	4	1	2	1

k=3

D₃

1	2	3	4	
1	∞	2	0	5
2	∞	3	-2	3
3	∞	5	3	5
4	-4	-2	-4	1

P₃

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

k=4

D₄

1	2	3	4	
1	1			
2	0	5		
2	-1	1	-2	3
3	1	3	1	5

P₄

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

Kết quả cuối cùng của thuật toán được lưu trong hai ma trận D và P. Độ dài đường đi ngắn nhất từ đỉnh s đến đỉnh t sẽ là: D[s, t]. Đường đi cụ thể sẽ được tính từ ma trận lưu vết P.

Ví dụ: Tìm đường đi ngắn nhất từ đỉnh 3 đến đỉnh 2:

Độ dài đường đi: D[3, 2] = 3.

P[3, 2] = 1. Đỉnh 1 là đỉnh trước của đỉnh 2. P[3, 1] = 4. Đỉnh 4 trước đỉnh 1.

P[3, 4] = 3. Đỉnh 3 trước đỉnh 4.

Vậy đường đi ngắn nhất từ đỉnh 3 đến đỉnh 2 là: 3 \rightarrow 4 \rightarrow 1 \rightarrow 2. Với độ dài = 3.

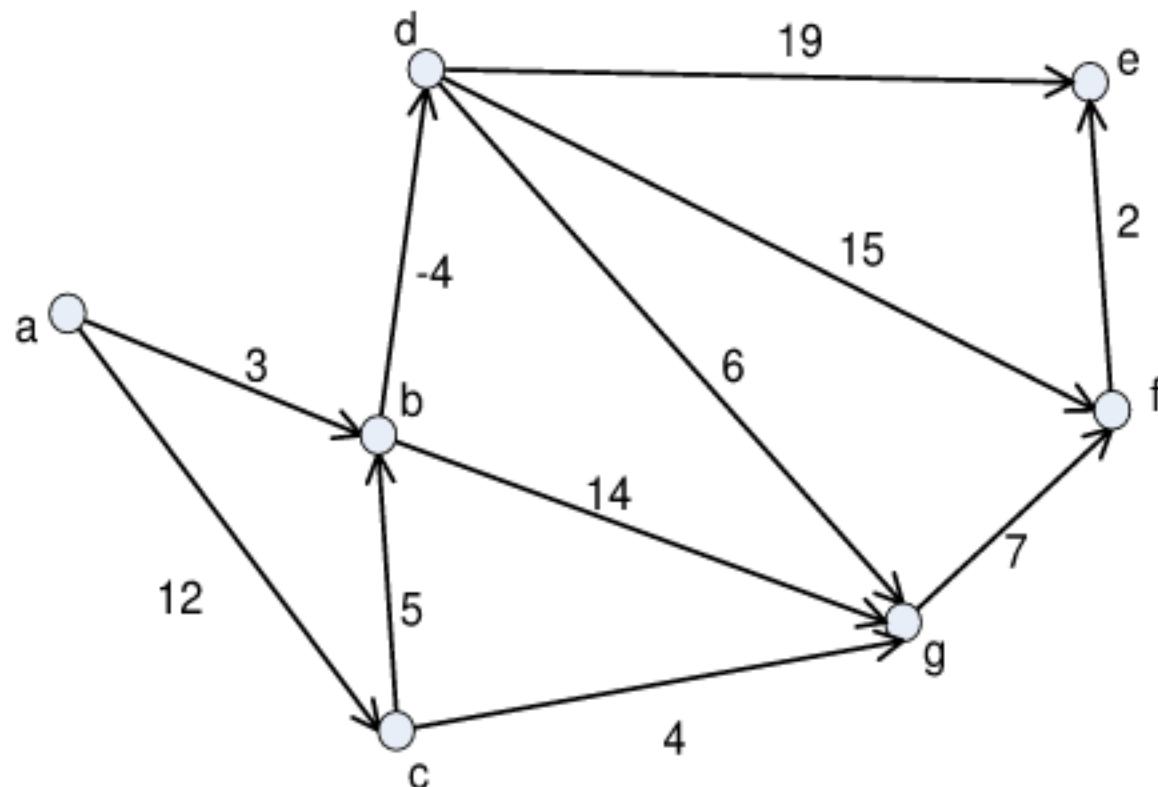
Một số kết quả có thể suy ra từ các ma trận D và P:

Nếu trên đường chéo chính của ma trận D có giá trị nào đó khác ∞ thì đồ thị sẽ có chu trình.

Nếu tất cả các giá trị của D (không kể đường chéo chính) đều khác ∞ thì đồ thị liên thông mạnh.

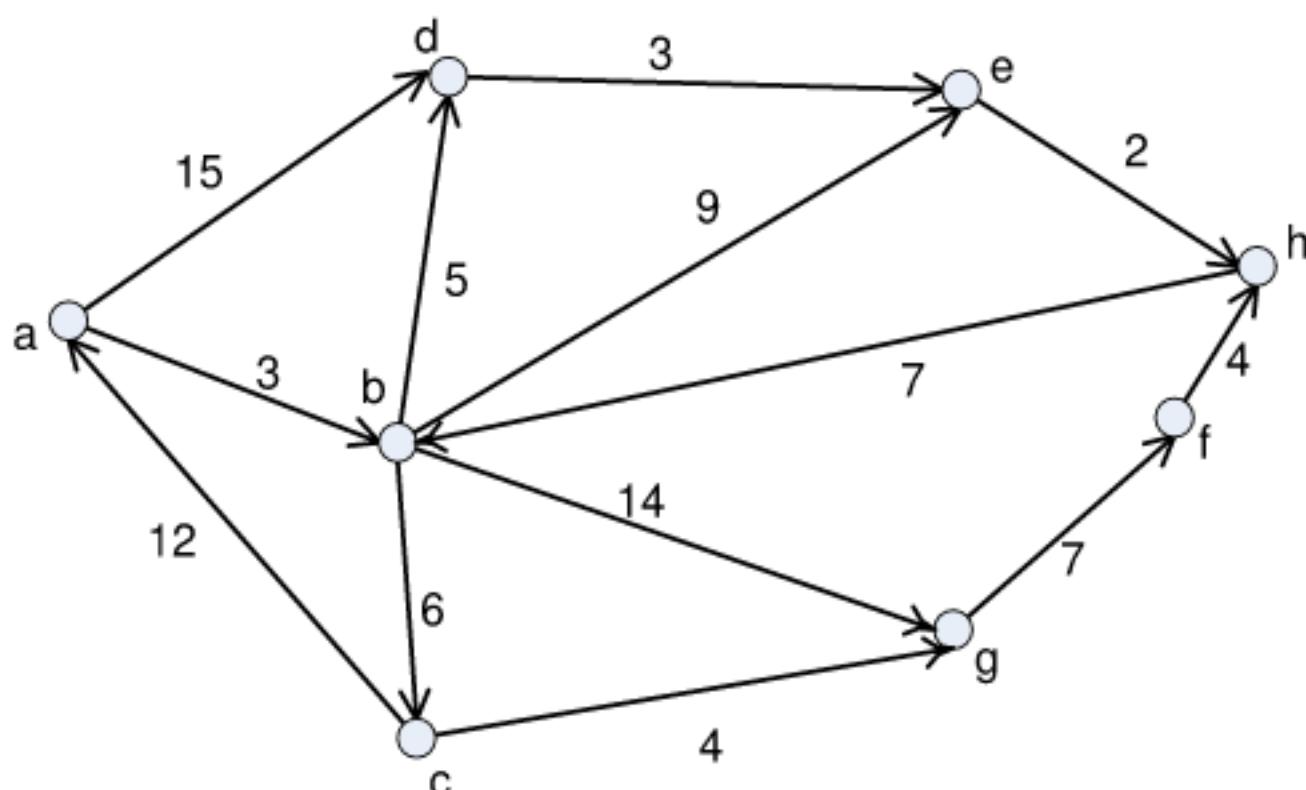
BÀI TẬP CHƯƠNG 10

1. Áp dụng thuật toán Ford-Bellman, tìm đường đi ngắn nhất từ đỉnh a đến tất cả các đỉnh trong đồ thị sau:



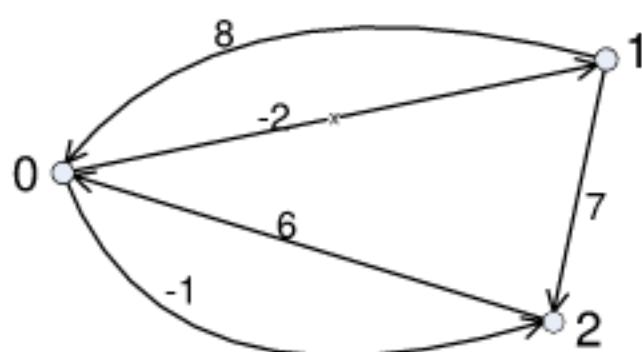
Hình 10.6

2. Áp dụng thuật toán Dijkstra, tìm đường đi ngắn nhất từ đỉnh a đến tất cả các đỉnh trong đồ thị sau:



Hình 10.7

3. Áp dụng thuật toán Floyd, tìm đường đi ngắn nhất giữa các cặp đỉnh trong đồ thị sau:

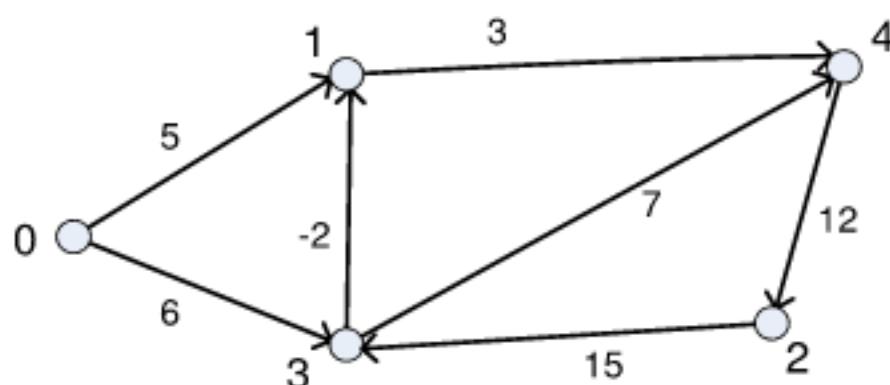


Hình 10.8

4. Viết chương trình cài đặt các thuật toán sau trên ngôn ngữ C++:

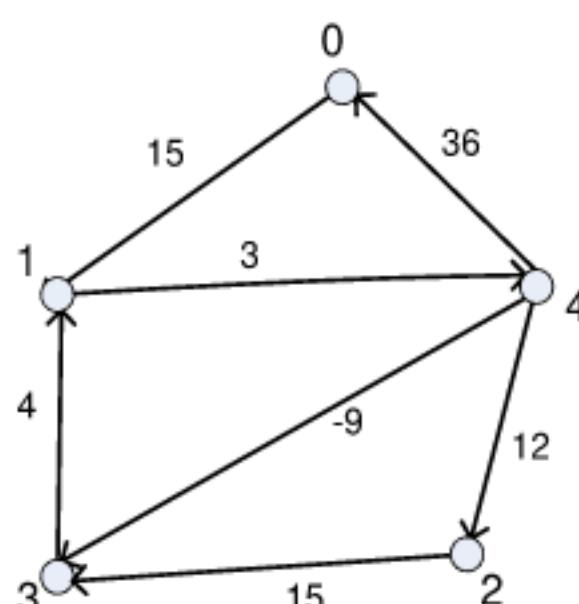
- a) Thuật toán Ford-Bellman.
- b) Thuật toán Dijkstra.
- c) Thuật toán Floyd.

5. Có thể áp dụng giải thuật Dijkstra để tìm đường đi ngắn nhất từ đỉnh 0 đến các đỉnh còn lại cho đồ thị sau được không? Giải thích lý do.



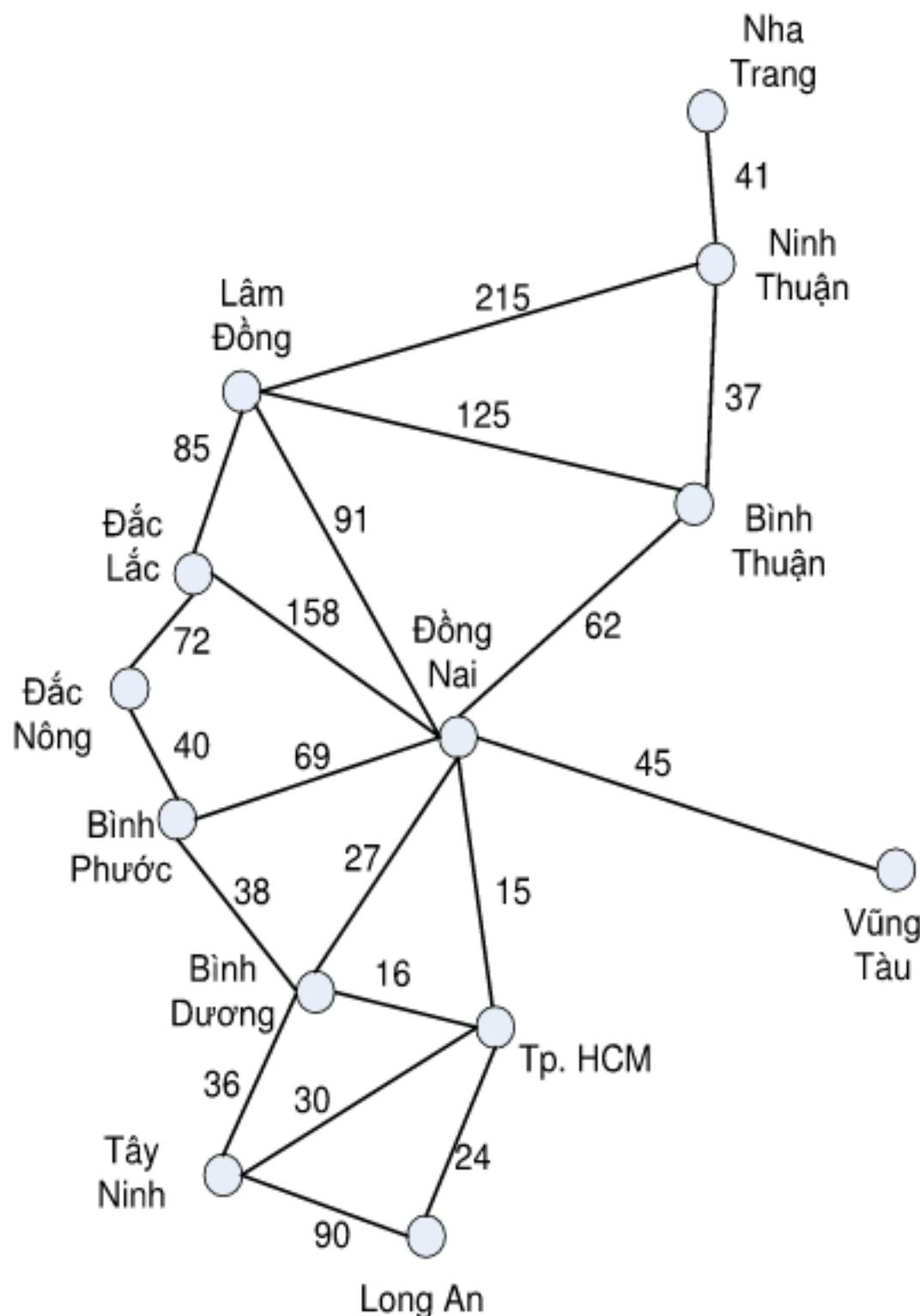
Hình 10.9

6. Minh họa từng bước quá trình tìm đường đi ngắn nhất từ đỉnh 0 đến các đỉnh còn lại của đồ thị. Giải thích lý do vì sao không tìm được đường đi ngắn nhất.



Hình 10.10

7. Cho sơ đồ các thành phố và chi phí (tính theo đơn vị nghìn đồng) để di chuyển (bằng ô tô) giữa các thành phố như sau:



Hình 10.11

Giả sử một công ty du lịch đặt tại TP Hồ Chí Minh muốn tổ chức các tour du lịch từ TP Hồ Chí Minh đến các tỉnh, thành phố khác. Hãy chỉ đường cho công ty này sao cho các tour là ít tốn kém về tiền vé nhất.

Chương 11

LUỒNG TRONG MẠNG

Bài toán luồng cực đại là một dạng bài toán tối ưu trong đồ thị. Nó xuất phát từ vấn đề được đặt ra năm 1954 bởi T.E. Harris nhằm tính toán khả năng vận chuyển lớn nhất từ một thành phố này đến một thành phố khác trong hệ thống các đường xe lửa nối liền các thành phố với nhau. Năm 1955, hai nhà toán học Lester R. Ford và Delbert R. Fulkerson đề xuất thuật toán Ford-Fulkerson để giải quyết vấn đề này. Ngày nay, thuật toán tìm luồng cực đại được vận dụng để giải rất nhiều bài toán khác nhau một cách hiệu quả, trong đó có các bài toán tổ hợp. Chương này trình bày các khái niệm, nội dung thuật toán và các ứng dụng của nó.

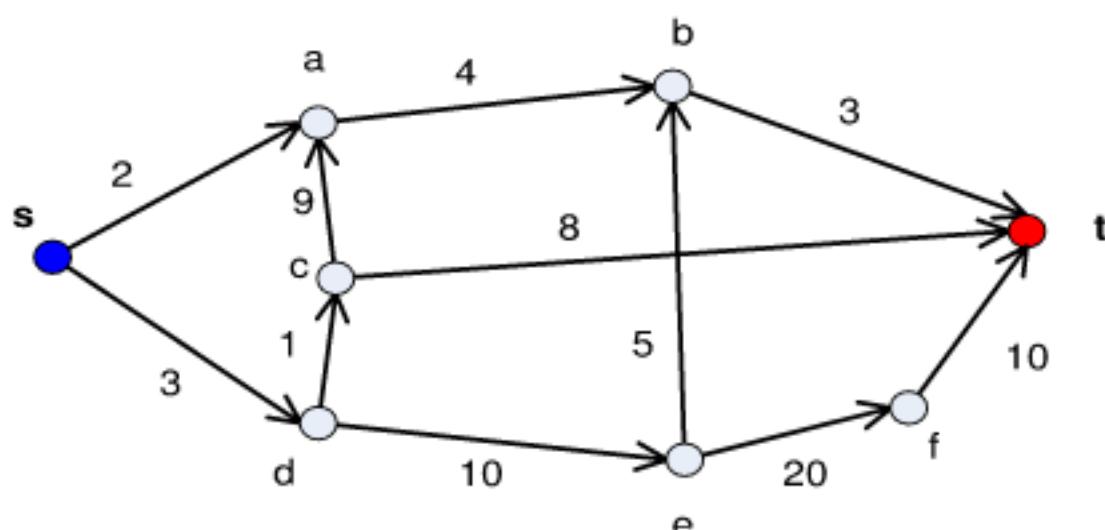
11.1. MỘT SỐ KHÁI NIỆM CƠ BẢN

11.1.1. Mạng

Mạng là một đồ thị có hướng $G = (V, E)$ trong đó có duy nhất một đỉnh s không có cung nào đi vào gọi là **điểm phát**; duy nhất một đỉnh t không có cung nào đi ra gọi là **điểm thu**, và mỗi cung $e = (u, v)$ được gán với một số không âm $c(e) = c(u, v)$ được gọi là **khả năng thông qua** của cung e .

Quy ước: Trong phạm vi của giáo trình này, ta quy ước nếu không có cung (u, v) thì khả năng thông qua của $c(u, v)$ của nó được gán bằng 0.

Ví dụ:



Hình 11.1

- + s là điểm phát, t là điểm thu;
- + Khả năng thông qua của cung (a, b) là: $c(a, b)=4$.

11.1.2. Luồng trong mạng

Cho mạng $G = (V, E)$, ta gọi luồng f trong mạng G là một ánh xạ $f: E \rightarrow R^*$, với mọi cung $e=(u, v) \in E$ được gán với một số không âm $f(e) = f(u, v) \geq 0$ gọi là **luồng trên cung e**, thỏa mãn các điều kiện sau:

- Luồng trên mỗi cung $e \in E$ không vượt quá khả năng thông qua của nó: $0 \leq f(e) \leq c(e)$;
- Với mọi đỉnh u không trùng với đỉnh phát s và đỉnh thu t , tổng luồng trên các cung đi vào u bằng tổng luồng trên các cung đi ra khỏi u .

$$Div_f(u) = \sum_{v \in \Gamma^-(u)} f(v, u) - \sum_{v \in \Gamma^+(u)} f(u, v) = 0$$

Trong đó:

$$\Gamma^-(u) = \{v \in V \mid (v, u) \in E\}$$

$$\Gamma^+(u) = \{v \in V \mid (u, v) \in E\}$$

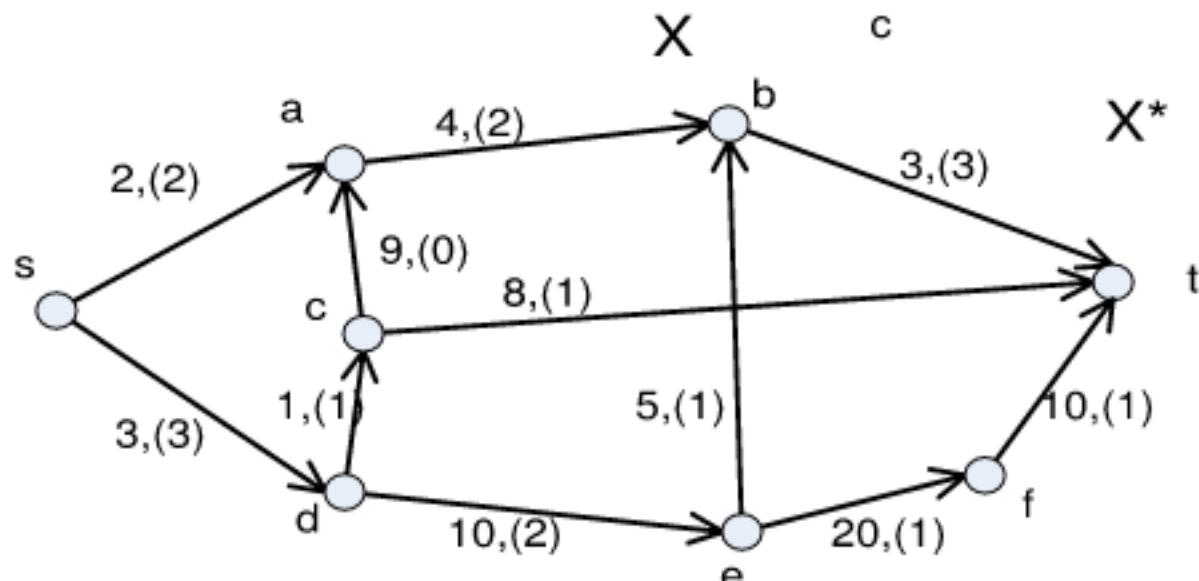
Tính chất này được gọi là **điều kiện cân bằng luồng**.

- **Giá trị của luồng f** là tổng luồng trên các cung đi ra khỏi đỉnh phát (bằng tổng luồng trên các cung đi vào đỉnh thu).

$$val(f) = \sum_{v \in \Gamma^+(s)} f(s, v) = \sum_{v \in \Gamma^-(t)} f(v, t)$$

Ví dụ:

Hình sau biểu diễn một luồng trong mạng. Giá trị thứ nhất là khả năng thông qua của mỗi cung và giá trị trong ngoặc đơn là luồng trên mỗi cung. Giá trị luồng là: $val(f) = 5$.



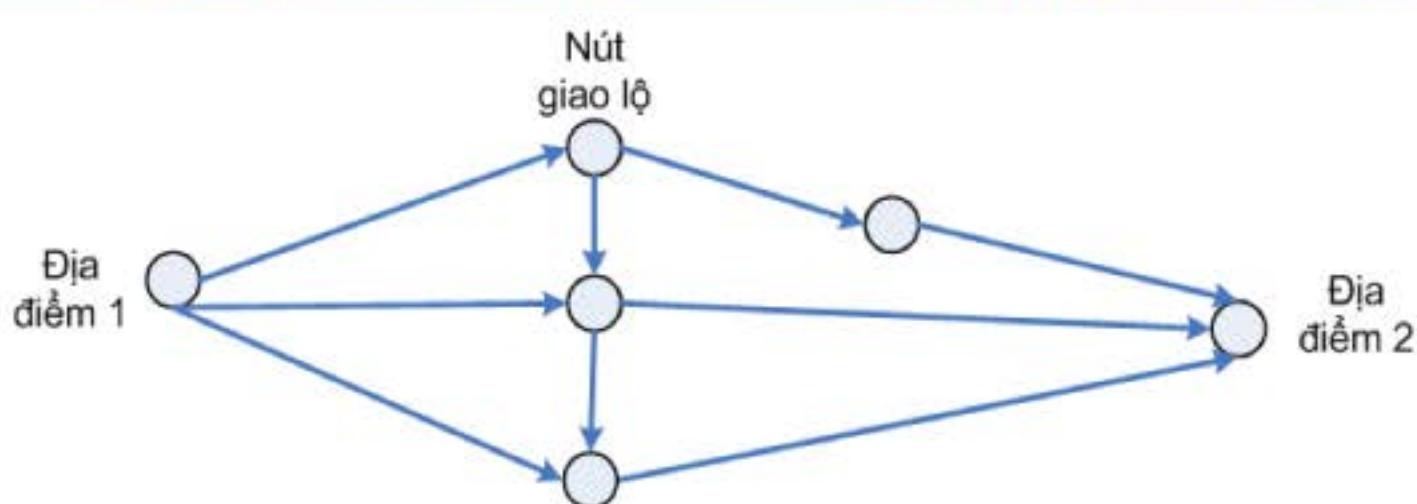
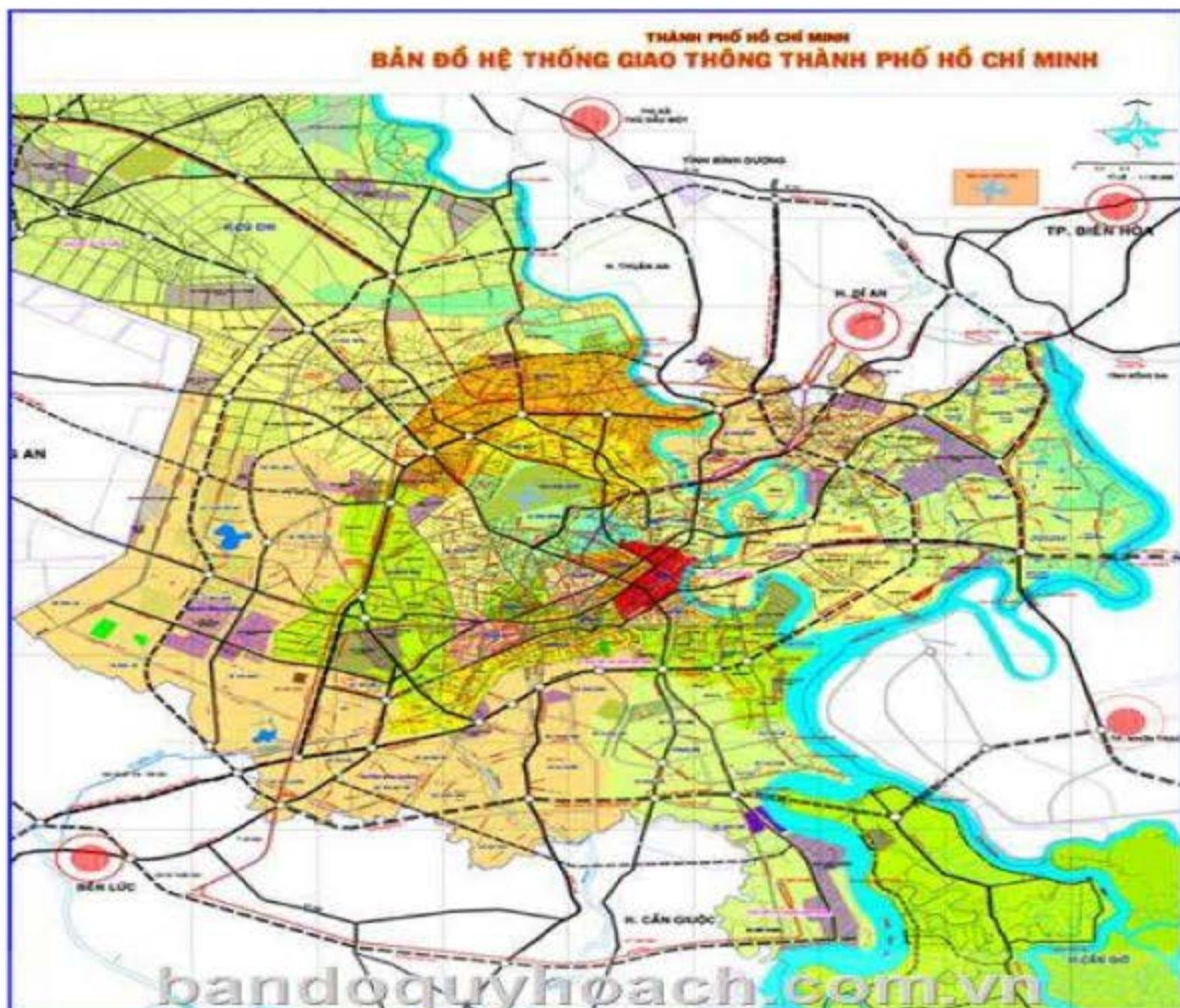
Hình 11.2

• Bài toán luồng cực đại

Cho mạng $G = (V, E)$, hãy tìm luồng f trong mạng sao cho giá trị luồng là lớn nhất.

Luồng như vậy gọi là **luồng cực đại**. Bài toán này được gọi là **bài toán luồng cực đại trong mạng**.

Bài toán luồng cực đại có nhiều ứng dụng trong thực tế như: bài toán lập bản đồ giao thông trong thành phố, bài toán đám cưới vùng quê, v.v.



Hình 11.3

Xác định cường độ dòng vận tải giữa hai nút giao thông?

11.2. ĐỊNH LÝ FORD-FULKERSON

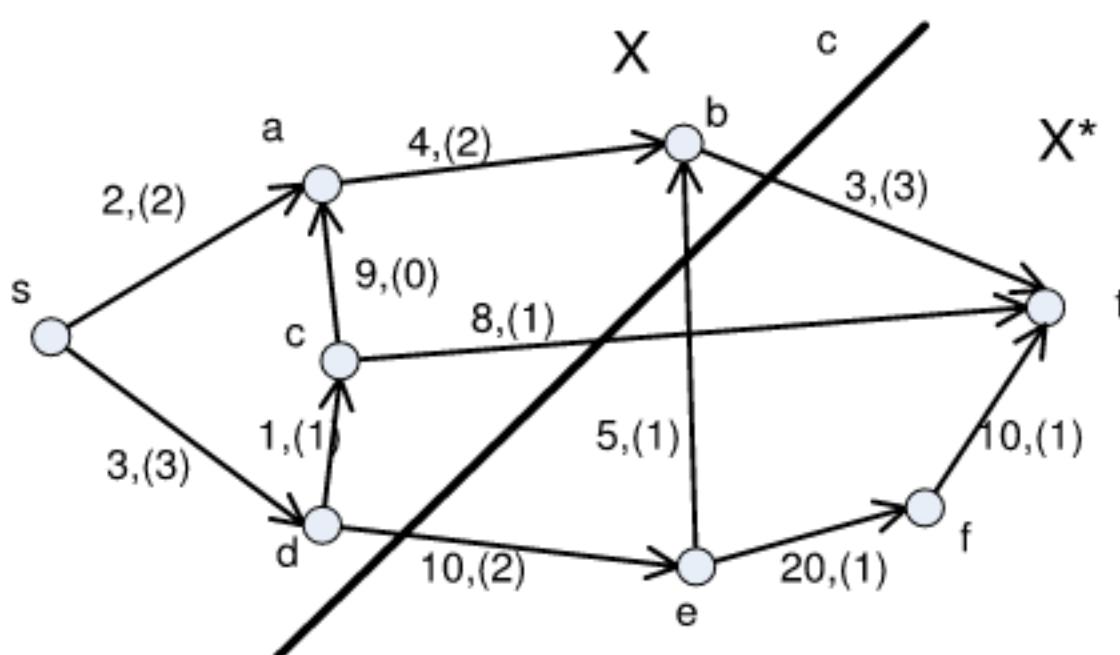
Hai nhà toán học Ford và Fulkerson đã tìm ra lời giải cho bài toán luồng cực đại khi đưa ra định lý Ford-Fulkerson. Trước khi tìm hiểu định lý, ta xem xét các khái niệm và tính chất liên quan.

- **Lát cắt**

Cho mạng $G = (V, E)$. Lát cắt (X, X^*) là một phân hoạch tập đỉnh V của mạng thành hai tập X và X^* với điểm phát $s \in X$ và điểm thu $t \in X^*$.

Khả năng thông qua của lát cắt (X, X^*) là tổng tất cả các khả năng thông qua của các cung (u, v) có $u \in X$ và $v \in X^*$. Lát cắt với khả năng thông qua nhỏ nhất được gọi là **lát cắt hẹp nhất**.

Ví dụ:



Hình 11.4

Khả năng thông qua của lát cắt (X, X^*) là: $3 + 8 + 10 = 21$.

✓ **Định lý 1**

Giá trị của mọi luồng f trong mạng không lớn hơn khả năng thông qua của lát cắt bất kỳ (X, X^) .*

$$val(f) \leq c(X, X^*)$$

Chứng minh:

Với mọi $u \in V$, ta cộng các điều kiện cân bằng luồng trên các đỉnh thuộc X :

$$\sum_{u \in X} \left(\sum_{w \in \Gamma^-(u)} f(v, u) - \sum_{w \in \Gamma^+(u)} f(u, v) \right) = \text{Div}(s) = -\text{val}(f)$$

Tổng này gồm các số hạng dạng $f(u,v)$ với dấu + và dấu - mà có ít nhất u hoặc $v \in X$. Nếu cả u và v đều $\in X$ thì $f(u,v)$ sẽ xuất hiện với dấu + trong $\text{Div}(v)$ và dấu - trong $\text{Div}(u)$ nên chúng triệt tiêu lẫn nhau. Ta thu được:

$$\begin{aligned} & - \sum_{u \in X, v \in X^*} f(u, v) + \sum_{u \in X^*, v \in X} f(u, v) = -\text{val}(f) \\ \Leftrightarrow & \text{val}(f) = \sum_{u \in X, v \in X^*} f(u, v) - \sum_{u \in X^*, v \in X} f(u, v) \leq \sum_{u \in X, v \in X^*} c(u, v) \\ \Leftrightarrow & \text{val}(f) \leq c(X, X^*) \text{ (đpcm).} \end{aligned}$$

Ví dụ:

Trên hình 11.4, lát cắt (X, X^*) có khả năng thông qua là $3+8+10=21$. Giá trị của luồng f : $\text{val}(f)=5<21$.

✓ Hệ quả

Giá trị luồng cực đại trong mạng không vượt quá khả năng thông qua của lát cắt hẹp nhất trong mạng.

• Định lý 2 (Định lý Ford-Fulkerson)

Giá trị luồng cực đại trên mạng đúng bằng khả năng thông qua của lát cắt hẹp nhất.

Để chứng minh định lý Ford-Fulkerson và xây dựng thuật toán tìm luồng cực đại, ta cần tiếp cận các khái niệm đồ thị tăng luồng, đường tăng luồng.

• Đồ thị tăng luồng, đường tăng luồng

Giả sử f là một luồng trong mạng $G = (V, E)$. Từ mạng G ta xây dựng đồ thị có trọng số $G_f = (V, E_f)$ như sau:

Xét các cạnh $e = (u, v) \in E$:

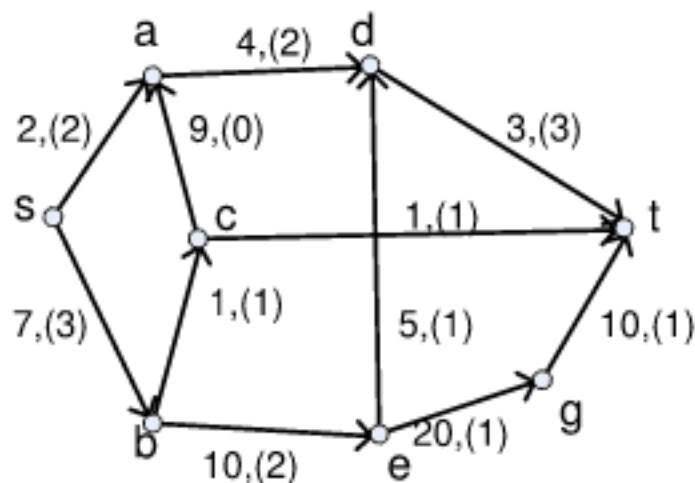
+ Nếu $f(u, v) = 0$ thì ta thêm một cung (u, v) có trọng số là $c(u, v)$ vào G_f .

+ Nếu $f(u, v) = c(u, v)$ thì ta thêm một cung (v, u) có trọng số $c(u, v)$ vào G_f .

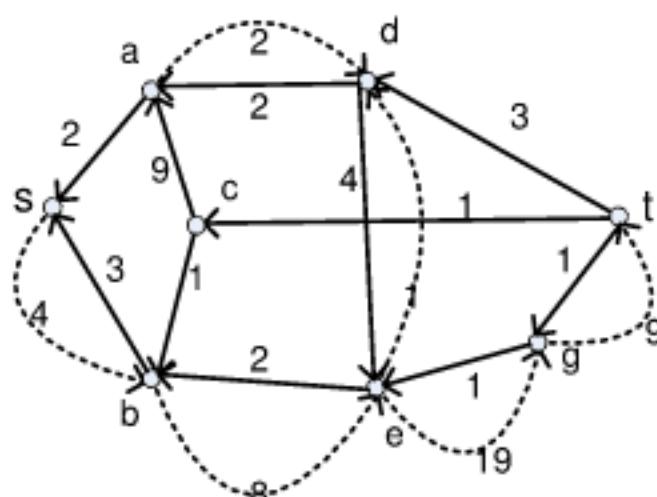
+ Nếu $0 < f(u, v) < c(u, v)$ thì ta thêm một cung (u, v) có trọng số $c(u, v) - f(u, v)$, và một cung (v, u) có trọng số $f(u, v)$ vào G_f .

Các cung của G_f đồng thời cũng là cung của G được gọi là **cung thuận**, các cung còn lại được gọi là **cung nghịch**. Đồ thị G_f được gọi là **đồ thị tăng luồng**.

Ví dụ:



Hình 11.5: Mạng $G = (V, E)$



Hình 11.6: Đồ thị tăng luồng $G_f = (V, E_f)$

Giả sử $P = (s, v_1, \dots, v_k, t)$ là một đường đi từ s đến t trên đồ thị tăng luồng G_f . Gọi d là trọng số nhỏ nhất trong các trọng số của các cung trên đường đi P. Từ luồng f, xây dựng luồng f' trên mạng G như sau:

- + Nếu $(u, v) \in P$ là cung thuận thì $f'(u, v) = f(u, v) + d$.
- + Nếu $(u, v) \in P$ là cung nghịch thì $f'(u, v) = f(u, v) - d$.
- + Nếu $(u, v) \notin P$ thì $f'(u, v) = f(u, v)$.

Khi đó ta được luồng f' là luồng trong mạng G và giá trị của luồng f' tăng thêm d so với giá trị của luồng f. Đường đi P được gọi là **đường tăng luồng**. Ta có định nghĩa đường tăng luồng như sau:

Đường tăng luồng f là mọi đường đi từ điểm phát s đến điểm thu t trên đồ thị tăng luồng G_f .

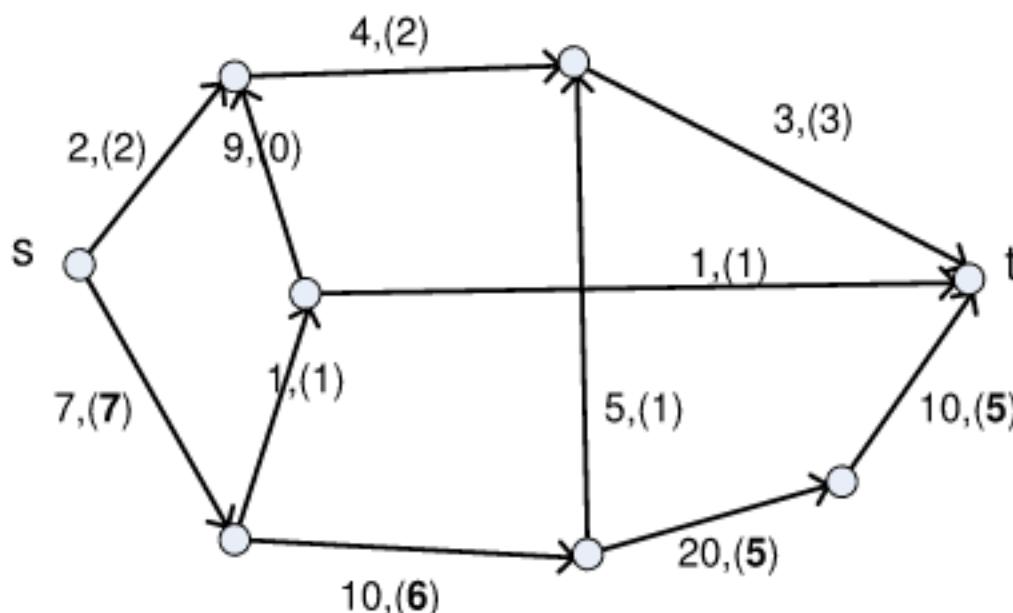
Ví dụ:

Từ đồ thị tăng luồng G_f ở hình 11.6, ta xét đường tăng luồng $P=(s, b, e, g, t)$, trọng số nhỏ nhất trong các trọng số của các cung trên đường đi P là: $d = 4$. Ta có:

$$f'(s, b) = f(s, b) + 4 = 3 + 4 = 7, f'(b, e) = f(b, e) + 4 = 2 + 4 = 6.$$

$$f'(e, g) = f(e, g) + 4 = 1 + 4 = 5, f'(g, t) = f(g, t) = 1 + 4 = 5.$$

Khi đó ta sẽ có luồng f' tăng thêm $d = 4$ so với luồng f trên mạng G : Giá trị luồng của f' : $\text{val}(f') = 9$.



Hình 11.7

✓ Định lý 3

Cho mạng $G = (V, E)$ và f là một luồng trong mạng G . Các mệnh đề sau là tương đương:

- f là luồng cực đại trong mạng.
- Không tìm được đường tăng luồng f .
- $\text{val}(f) = c(X, X^*)$, với (X, X^*) là một lát cắt nào đó của mạng.

Chứng minh:

Ta sẽ chứng minh theo thứ tự sau: a \Rightarrow b \Rightarrow c \Rightarrow a

- **a \Rightarrow b:** Nếu f là luồng cực đại trong mạng thì không tìm được đường tăng luồng f :

Thật vậy, nếu tìm được đường tăng luồng f thì có thể sẽ tìm được một luồng mới có giá trị luồng tăng hơn so với luồng cực đại (vô lý). (Đpcm).

- **b \Rightarrow c:** Nếu không tìm được đường tăng luồng f thì $\text{val}(f) = c(X, X^*)$:

Ta có:

$$val(f) = \sum_{u \in X, v \in X^*} f(u, v) - \sum_{u \in X^*, v \in X} f(u, v)$$

- + Với $u \in X^*$, $v \in X$, $f(u, v) = 0$. (Vì trong trường hợp ngược lại sẽ có cung $(v, u) \in G_f$).
- + Với $u \in X$, $v \in X^*$, vì $(u, v) \notin G_f \Rightarrow (u, v) \in G$ với $f(u, v) = c(u, v)$.

Suy ra:

$$val(f) = \sum_{u \in X, v \in X^*} c(u, v) = c(X, X^*)$$

- **c => a:** Theo định lý 1, $val(f) \leq c(X, X^*)$. Mà ta có $val(f) = c(X, X^*)$ nên ta có f là luồng cực đại trong mạng.

11.3. THUẬT TOÁN TÌM LUỒNG CỰC ĐẠI TRONG MẠNG

Tiếp theo, ta nghiên cứu thuật toán tìm luồng cực đại trong mạng, được gọi là thuật toán Ford-Fulkerson. Quy trình thuật toán Ford-Fulkerson như sau:

- + Đặt luồng ban đầu bằng 0 (luồng không). Vì một mạng bất kỳ đều có ít nhất một luồng là luồng không.
- + Lặp lại hai quá trình tìm đường tăng luồng và tăng luồng cho mạng theo đường tăng luồng đó. Vòng lặp kết thúc khi không tìm được đường tăng luồng nữa.
- + Khi đã có luồng cực đại, xây dựng lát cắt hẹp nhất của mạng.

• Thuật toán Ford-Fulkerson

Như chúng ta đã tìm hiểu ở trên, để tìm luồng cực đại trong mạng, việc cần làm là xây dựng đồ thị tăng luồng, từ đó tìm ra đường tăng luồng và tăng giá trị luồng trong mạng. Tuy nhiên, khi thực hiện giải thuật cài đặt trên máy tính, ta sử dụng kỹ thuật gán nhãn. Với kỹ thuật này, hai giai đoạn tìm đồ thị tăng luồng và tìm đường tăng luồng được ghép lại thành một (được thực hiện cùng lúc). Hay nói một cách khác, ta không tìm đồ thị tăng luồng một cách tường minh, kết quả trả về là đường tăng luồng mà không cần biết đến đồ thị tăng luồng.

Kỹ thuật gán nhãn như sau: Mỗi đỉnh sẽ có ba trạng thái: chưa có nhãn, có nhãn chưa xét và có nhãn đã xét.

Nhân gán cho mỗi đỉnh v có hai dạng: $[+p(v), \varepsilon(v)]$ và $[-p(v), \varepsilon(v)]$. Trong đó $+p(v)$ và $-p(v)$ cho biết ta có thể tăng (giảm) luồng theo cung $(p(v), v)$ ((v, p(v)) với giá trị là $\varepsilon(v)$.

Việc gán nhân bắt đầu từ điểm phát s.

Đầu vào: Mạng G = (V, E).

Đầu ra: Luồng cực đại f, Lát cắt hẹp nhất (X, X*)

Bước 1: Đặt $f(e)=0$, với mọi cung $e \in E$

Bước 2: Gán nhân cho s:

$p[s]=[-, \varepsilon(s)]; \varepsilon(s)=\infty;$

Đặt $u=s$;

Bước 3:

a) $\forall v \in K^+(u)$, Nếu v chưa có nhân và $s(u,v)=c(u,v) - f(u,v)>0$ thì:

Đặt $\varepsilon(v) = \min(\varepsilon(u), s(u,v))$;

Gán nhân $p[v] = [+u, \varepsilon(v)]$;

b) $\forall v \in K^-(u)$, Nếu v chưa có nhân và $f(v,u)>0$ thì:

Đặt $\varepsilon(v) = \min (\varepsilon(u), f(v,u))$;

Gán nhân $p[v] = [-u, \varepsilon(v)]$;

Bước 4: Nếu t đã có nhân ($v=t$) Đến **Bước 5**.

Ngược lại:

Nếu Mọi đỉnh có nhân đã xét: Đến **Bước 6**.

Ngược lại: đặt $u=v$, Đến **Bước 3**.

Bước 5: Dùng $p[t]$ để tìm đường tăng luồng P bằng cách đi ngược từ t đến s.

Đặt $f = f + \varepsilon(t) \quad \forall$ cạnh $e \in P$. Đến **Bước 2**.

Bước 6: $X = \{\text{Các đỉnh có nhân đã xét}\}, X^* = V \setminus X$.

Lát cắt (X, X^*) là cực tiểu.

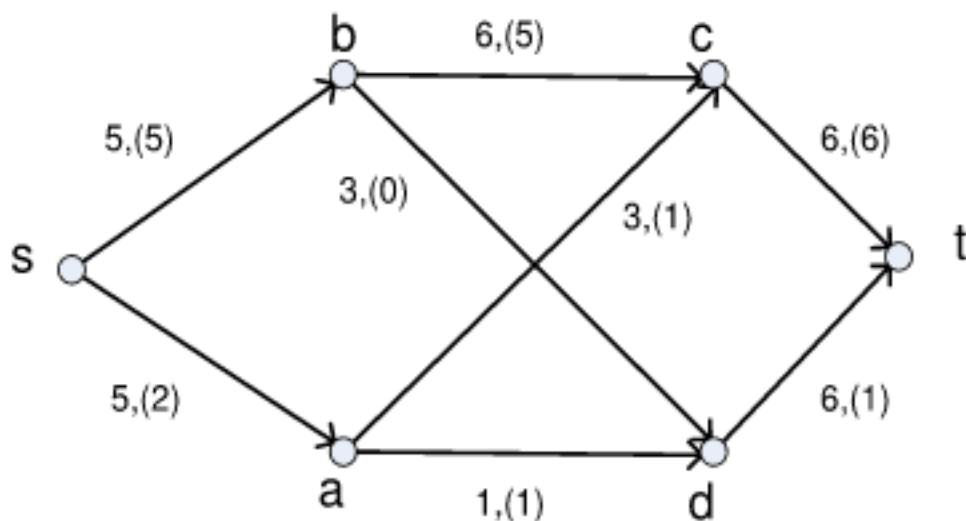
Giải thuật có thể được giải thích ngắn gọn như sau:

Đầu tiên, gán nhân cho s và đặt nó là chưa xét. Tiếp tục ta gán nhân cho các đỉnh kề của s và s trở thành đỉnh đã xét. Làm tương tự cho các đỉnh kề với s và đã được gán nhân. Thuật toán dừng lại nếu:

- Đỉnh t được gán nhãn. Khi đó ta tìm được đường tăng luồng.
- Hoặc t chưa có nhãn mà tất cả các đỉnh có nhãn khác đã được xét. Khi đó luồng đang xét là cực đại, không tìm được đường tăng luồng.

Ví dụ:

Tìm đường tăng luồng cho mạng $G = (V, E)$ với luồng hiện tại $f = 7$ như sau:



Hình 11.8

Minh họa từng bước khi áp dụng thuật toán:

Lần 1:

+ Gán nhãn cho s: $p[s] = [-\infty]$, $\varepsilon(s) = \infty$, $u = s$

+ Xét s:

Các đỉnh $\in K_{e+}(s) = \{a, b\}$

- Đỉnh a: $s(s, a) = c(s, a) - f(s, a) = 5 - 2 = 3 > 0$, $\varepsilon(a) = \min(\infty, 3) = 3$, gán nhãn cho a: $p[a] = [+s, 3]$.

- Đỉnh b: $s(s, b) = c(s, b) - f(s, b) = 5 - 5 = 0$. Không gán nhãn cho b

Các đỉnh $\in K_{e-}(s) = \emptyset$.

Đỉnh t chưa được gán nhãn. Trong số các đỉnh đã được gán nhãn, còn đỉnh a chưa xét.

+ Xét a:

Các đỉnh $\in K_{e+}(a) = \{c, d\}$.

- Đỉnh c: $s(a, c) = c(a, c) - f(a, c) = 3 - 1 = 2 > 0$, $\varepsilon(c) = \min(3, 2) = 2$, gán nhãn cho c: $p[c] = [+a, 2]$.

- Đỉnh d: $s(a, d) = c(a, d) - f(a, d) = 1 - 1 = 0$. Không gán nhãn cho d.

Các đỉnh $\in K_{e-}(s) = \{s\}$.

- Đỉnh s: đã có nhãn (và cũng đã xét).

Đỉnh t chưa được gán nhãn. Trong số các đỉnh đã được gán nhãn, còn đỉnh c chưa xét.

- + Xét c:

Các đỉnh $\in \text{Ke}^+(c) = \{t\}$

- Đỉnh t: $s(c, t) = c(c, t) - f(c, t) = 6 - 6 = 0$. Không gán nhãn cho t

Các đỉnh $\in \text{Ke}^-(c) = \{a, b\}$.

- Đỉnh a: đã có nhãn (và cũng đã xét).

Đỉnh b chưa được gán nhãn. Trong số các đỉnh đã được gán nhãn, còn đỉnh b chưa xét.

- + Xét b:

Các đỉnh $\in \text{Ke}^+(b) = \{c, d\}$

- Đỉnh c: đã có nhãn (và cũng đã xét).

Đỉnh d: $s(b, d) = c(b, d) - f(b, d) = 3 - 0 = 3 > 0$, $\varepsilon(d) = \min(2, 3) = 2$, gán nhãn cho d: $p[d] = [+b, 2]$.

Các đỉnh $\in \text{Ke}^-(b) = \{s\}$.

- Đỉnh s: đã có nhãn (và cũng đã xét).

Đỉnh t chưa được gán nhãn. Trong số các đỉnh đã được gán nhãn, còn đỉnh d chưa xét.

- + Xét d:

Các đỉnh $\in \text{Ke}^+(d) = \{t\}$

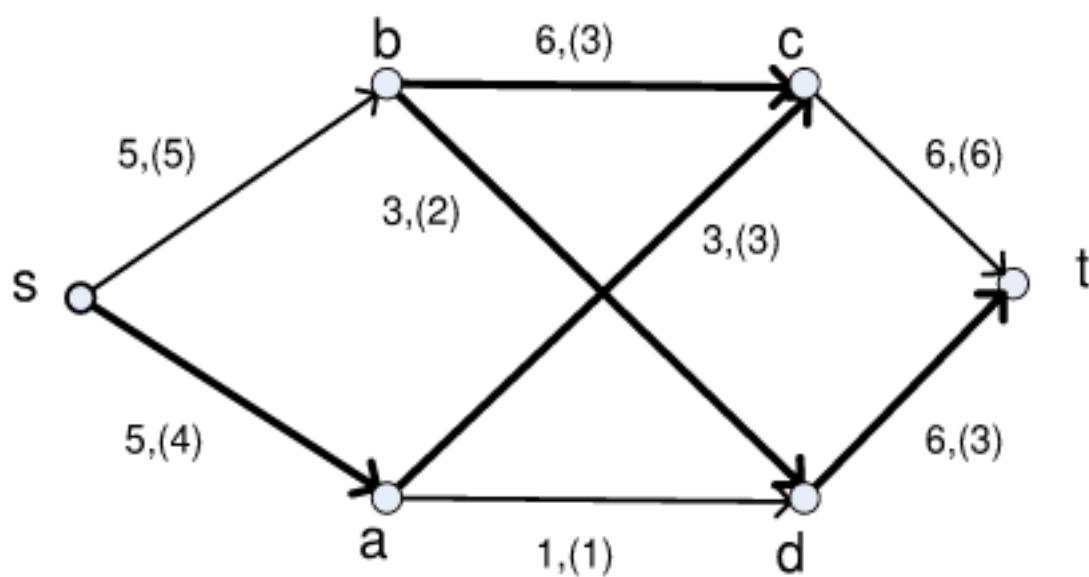
Đỉnh t: $s(d, t) = c(d, t) - f(d, t) = 6 - 1 = 5 > 0$, $\varepsilon(t) = \min(2, 5) = 2$, gán nhãn cho t: $p[t] = [+d, 2]$.

Các đỉnh $\in \text{Ke}^-(d) = \{a, b\}$.

- Đỉnh a: đã có nhãn (và cũng đã xét).

- Đỉnh b: đã có nhãn (và cũng đã xét).

Đỉnh t đã được gán nhãn, với $\varepsilon(t) = 2$, ta có đường tăng luồng: $s \rightarrow a \rightarrow c \rightarrow b \rightarrow d \rightarrow t$. Giá trị mới của luồng f: $f = f + 2 = 7 + 2 = 9$.



Hình 11.9

Lần 2:

+ Gán nhãn cho s : $p[s] = [-\infty, \infty]$, $\varepsilon(s) = \infty$, $u = s$;

+ Xét s :

Các đỉnh $\in K_{e+}(s) = \{a, b\}$.

- Đỉnh a : $s(s, a) = c(s, a) - f(s, a) = 5 - 4 = 1 > 0$, $\varepsilon(a) = \min(\infty, 1) = 1$, gán nhãn cho a : $p[a] = [+s, 1]$.

- Đỉnh b : $s(s, b) = c(s, b) - f(s, b) = 5 - 5 = 0$. Không gán nhãn cho b .

Các đỉnh $\in K_{e-}(s) = \emptyset$.

Đỉnh t chưa được gán nhãn. Trong số các đỉnh đã được gán nhãn, còn đỉnh a chưa xét.

+ Xét a :

Các đỉnh $\in K_{e+}(a) = \{c, d\}$.

- Đỉnh c : $s(a, c) = c(a, c) - f(a, c) = 3 - 3 = 0$. Không gán nhãn cho c ;

- Đỉnh d : $s(a, d) = c(a, d) - f(a, d) = 1 - 1 = 0$. Không gán nhãn cho d .

Các đỉnh $\in K_{e-}(a) = \{s\}$.

- Đỉnh s : đã có nhãn (và cũng đã xét).

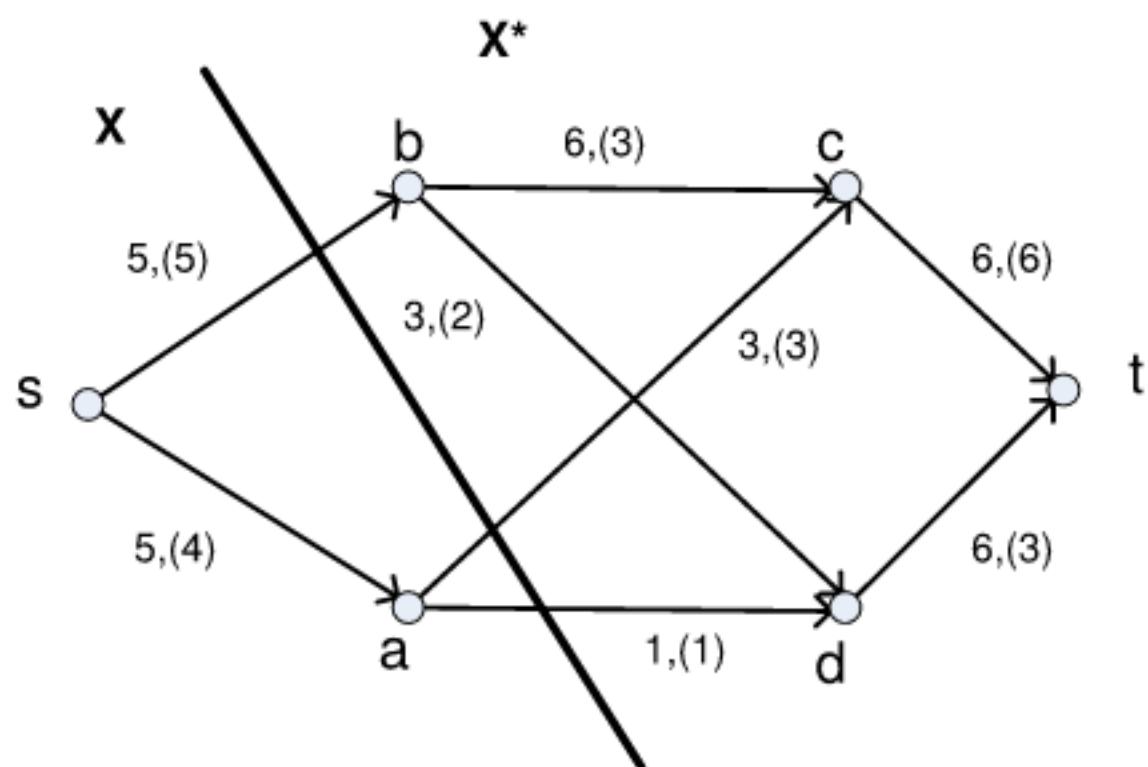
Đỉnh t chưa được gán nhãn. Tất cả các đỉnh có nhãn đã xét. Thuật toán DUNG.

Luồng $f = 9$ là luồng cực đại của mạng.

Lát cắt hẹp nhất (X, X^*) :

$X = \{\text{Các đỉnh có nhãn đã xét}\} = \{s, a\}$.

$X^* = V \setminus X = \{b, c, d, t\}$.

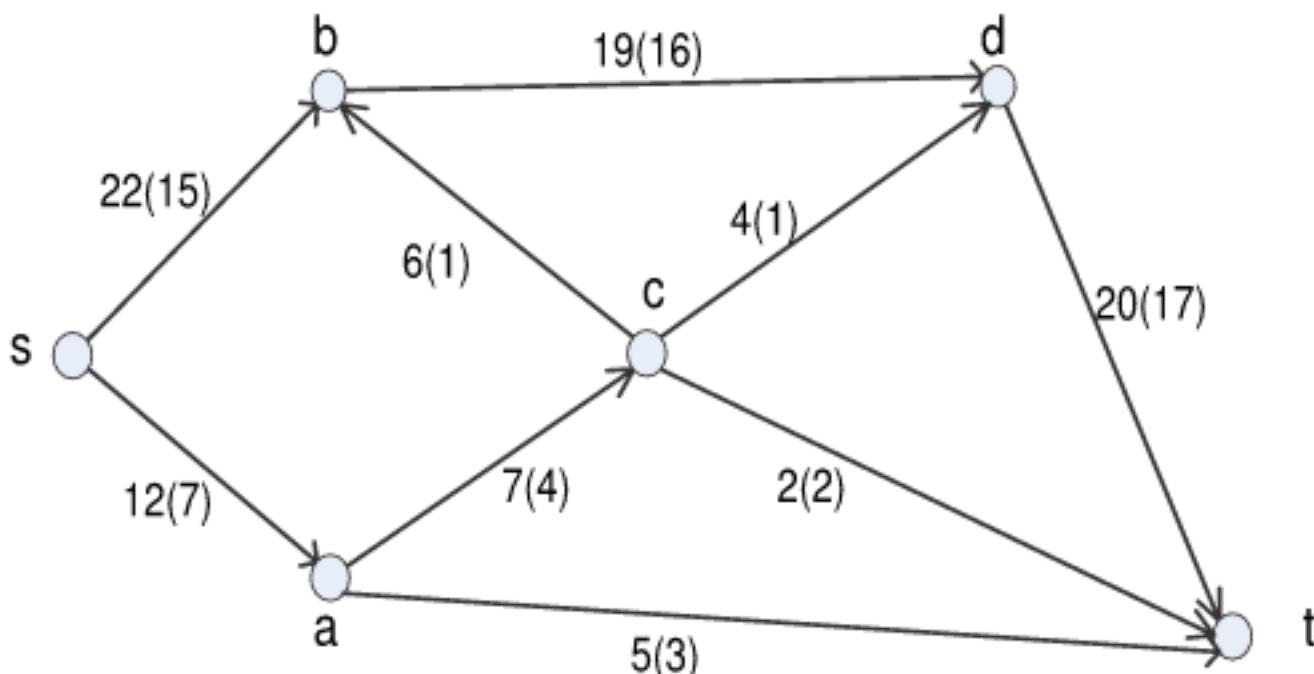


Hình 11.10

Nhận xét: Khả năng thông qua của lát cắt hẹp nhất (X , X^*) = $5+3+1=9$, bằng với giá trị của luồng cực đại.

BÀI TẬP CHƯƠNG 11

1. Cho mạng $G = (V, E)$ và một luồng f có giá trị là 22. Hãy áp dụng thuật toán Ford-Fulkerson để tìm luồng cực đại cho mạng. (Thứ tự xét các đỉnh ưu tiên theo bảng chữ cái alphabet).



Hình 11.11

2. Đám cưới vùng quê

Có m chàng trai và n cô gái, $n \geq m$. Mỗi chàng trai có thích một số cô gái. Có thể tổ chức được một đám cưới tập thể cho m chàng trai sao cho mỗi người đều lấy được cô gái mà họ thích hay không?

3. Hôn nhân bền vững

Có m chàng trai, n cô gái và k bà mối. Mỗi bà mối p ($p=1, 2, \dots, k$) có một danh sách L_p một số chàng trai và cô gái. Một số các chàng trai và cô gái này là khách hàng của bà ta. Bà mối p có thể se duyên cho bất kỳ cặp trai gái nào là khách hàng của bà ta nhưng không đủ sức tổ chức quá d_p đám cưới. Hãy xây dựng thuật toán căn cứ vào danh sách L_p, d_p ($p=1, 2, \dots, k$) để đưa ra cách tổ chức nhiều nhất các đám cưới giữa m chàng trai và n cô gái với sự giúp sức của bà mối.

Phụ lục A

HƯỚNG DẪN VÀ ĐÁP ÁN MỘT SỐ BÀI TẬP

Chương 1

Câu 1

a) Bảng chân trị

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

P	q	$p \rightarrow q$	$\neg(p \rightarrow q)$
1	1	1	0
1	0	0	1
0	1	1	0
0	0	1	0

b) Kiểm tra dạng mệnh đề $\neg(p \rightarrow q) \leftrightarrow (p \wedge \neg q)$ là hằng đúng.

Bước	Quy luật logic
$\neg(p \rightarrow q)$	
$\Leftrightarrow \neg(\neg p \vee q)$	Thay thế tương đương
$\Leftrightarrow \neg(\neg p) \wedge \neg q$	De Morgan
$\Leftrightarrow p \wedge \neg q$	Phủ định của phủ định

Câu 5

Biểu thức	Quy luật logic
$(q \wedge p) \vee \neg(q \rightarrow p)$	
$\Leftrightarrow (q \wedge p) \vee \neg(\neg q \vee p)$	Thay thế tương đương
$\Leftrightarrow (q \wedge p) \vee (\neg(\neg q) \wedge \neg p)$	De Morgan
$\Leftrightarrow (q \wedge p) \vee (q \wedge \neg p)$	Phủ định của phủ định
$\Leftrightarrow q \wedge (p \vee \neg p)$	Phân phối
$\Leftrightarrow q \wedge 1$	Phản tử bù
$\Leftrightarrow q$	Phản tử trung hòa

Câu 6

a) Ta có $p(1)=1$, $p(3)=1$, $q(1)=1$, $q(3)=1$ nên với $x=1,3$ ta có $(p(x) \rightarrow q(x))=1$.

Với $x \neq 1,3$, $p(x)=0$ nên $(p(x) \rightarrow q(x))=1$. Vậy mệnh đề đã cho là đúng.

b) Ta có $p(0)=0$ nên $(p(0) \rightarrow \neg q(0))=1$. Vậy mệnh đề đã cho là đúng.

c) Xét $\neg(\forall x \in \mathbf{R}, p(x) \rightarrow \neg q(x))$

$$= \exists x \in \mathbf{R}, \neg(p(x) \rightarrow \neg q(x))$$

$$= \exists x \in \mathbf{R}, \neg(\neg p(x) \vee \neg q(x))$$

$$= \exists x \in \mathbf{R}, p(x) \wedge q(x)$$

Ta có $p(1)=1$, $q(1)=1$ nên $(p(1) \wedge q(1))=1$.

Vậy mệnh đề $(\exists x \in \mathbf{R}, p(x) \wedge q(x))=1$.

Do đó mệnh đề $\forall x \in \mathbf{R}, p(x) \rightarrow \neg q(x)$ là sai.

Câu 7

$$p \vee s$$

$$\Leftrightarrow p \vee \neg \neg s$$

$$\Leftrightarrow \neg s \rightarrow p$$

$$\neg s$$

$\therefore p$	[phương pháp khẳng định]
mà $p \rightarrow (q \rightarrow r)$	
$\therefore q \rightarrow r$	[phương pháp khẳng định]
$t \rightarrow q$	
$\therefore t \rightarrow r$	[tam đoạn luận]
$\Leftrightarrow \neg t \vee r$	
$\Leftrightarrow \neg t \vee \neg \neg r$	
$\Leftrightarrow \neg r \rightarrow \neg t$	

Câu 9

Bước	Lý do
1. $\neg p \rightarrow q$	Tiền đề
2. $q \rightarrow s$	Tiền đề
3. $\neg p \rightarrow s$	B_1, B_2 và tam đoạn luận
4. $p \rightarrow r$	Tiền đề
5. $\neg r \rightarrow \neg p$	B_4 và tương đương logic
6. $\therefore \neg r \rightarrow s$	B_5, B_3 và tam đoạn luận

Vậy suy luận trên là suy luận đúng.

Câu 10

Bước	Lý do
1. $\forall x \in A, p(x) \rightarrow q(x)$	Tiền đề
2. $\forall x \in A, q(x) \rightarrow r(x)$	Đặc biệt hoá phô dụng
3. $p(a) \rightarrow q(a)$	B_1 và đặc biệt hoá phô dụng
4. $q(a) \rightarrow r(a)$	B_2 và đặc biệt hoá phô dụng
5. $p(a) \rightarrow r(a)$	$B_2, 4$ và tam đoạn luận
6. $\therefore \forall x \in A, p(x) \rightarrow r(x)$	B_5 và tổng quát hoá phô dụng

Chương 2

Câu 1

Phản xạ: c) vì với mọi x trong tập đã cho , $x \geq x$.

Đối xứng : Không có quan hệ nào.

Phản xứng : a), b), c)

Bắc cầu : b), c)

Thứ tự : c)

Câu 2

a) Sup {e, b}=e

b) Inf {e, t }= b

c) Sup {d, v}= x

Câu 3

a) Dùng định nghĩa.

b) $[(1,3)] = \{ (1,3), (3,1), (2,2) \}$

$[(2,4)] = \{ (2,4), (4,2), (1,5), (5,1), (3,3) \}$

$[(1,1)] = \{ (1,1) \}$

c) Các lớp tương đương:

$[(1,1)], [(1,2)], [(1,3)], [(2,3)], [(2,4)], [(2,5)], [(3,5)], [(4,5)], [(5,5)]$.

Câu 4

a) sup {b, c}= e, inf {b, c}= a

b) sup {a, d, e}= e, inf {a, d, e}= a

Câu 5

a) sup {b, c, d} = d, inf {b, c, d}= a

b) sup {b, c} không có, inf {b, c}= a

c) sup {d, e} = f, inf {d, e} không có

Chương 3

Câu 1

a)

Các tế bào lớn : yzt , xyt , $y\bar{z}t$, $\bar{x}zt$, $xy\bar{z}$.

Chỉ có một công thức tối thiểu : $f = yzt \vee \bar{x}zt \vee xy\bar{z}$

b)

Các tế bào lớn : $\bar{x}z$, \bar{xy} , \bar{yzt} , $x\bar{yt}$, $x\bar{zt}$, $y\bar{zt}$.

Các công thức đa thức tối thiểu:

$$\bar{x}z \vee \bar{xy} \vee x\bar{yt} \vee x\bar{zt},$$

$$\bar{x}z \vee \bar{xy} \vee x\bar{yt} \vee y\bar{zt}$$

$$\bar{x}z \vee \bar{xy} \vee x\bar{zt} \vee \bar{yzt}$$

c)

Các tế bào lớn : \bar{yt} , \bar{zt} , \bar{zy} , yt .

Các công thức đa thức tối thiểu:

$$\bar{yt} \vee yt \vee \bar{zy}$$

$$\bar{yt} \vee yt \vee \bar{zt}$$

d)

Các tế bào lớn : $\bar{x}\bar{zt}$, \bar{xyz} , \bar{xyt} , $y\bar{zt}$, $xy\bar{z}$, \bar{xyt} .

Các công thức đa thức tối thiểu :

$$xy\bar{z} \vee \bar{xyt} \vee \bar{xyz} \vee y\bar{zt}$$

$$xy\bar{z} \vee \bar{xyt} \vee \bar{x}\bar{zt} \vee \bar{xyt}$$

$$xy\bar{z} \vee \bar{xyt} \vee \bar{xyz} \vee \bar{xyt}$$

Chương 4

Câu 2: Không tồn tại đồ thị này, vì số đỉnh bậc lẻ trong đồ thị phải là số chẵn (hệ quả của định lý bắt tay).

Câu 4

- a) 28
- b) 12
- c) 30
- d) 12

Câu 5

Đáp án: 17 đồ thị con

Câu 6

Đáp án: Tổng số đồ thị $C_{15}^5 + C_{15}^7$

Hướng dẫn: Đồ thị đầy đủ 6 đỉnh có $6*(6-1)/2=15$ cạnh. Đồ thị có 5 cạnh, 6 đỉnh là một cách chọn không tính thứ tự 5 trong 15 cạnh. Tương tự với đồ thị 7 cạnh, 6 đỉnh.

Câu 8

Hướng dẫn: Giả sử không tồn tại đường đi nối hai đỉnh bậc lẻ của đồ thị. Như vậy đồ thị có ít nhất hai thành phần liên thông. Hai đỉnh lẻ duy nhất của đồ thị thuộc hai thành phần liên thông khác nhau. Điều này vô lý (xét theo hệ quả của định lý bắt tay, mỗi thành phần liên thông cũng là một đồ thị).

Câu 10

Hướng dẫn: Trong một đồ thị đơn vô hướng, bậc lớn nhất của một đỉnh là $n-1$. Giả sử không tồn tại hai đỉnh có bậc bằng nhau, như vậy danh sách bậc các đỉnh là: $0, 1, \dots, n-1$. Điều này vô lý, vì nếu tồn tại đỉnh có bậc là $n-1$ thì không thể tồn tại đỉnh có bậc là 0.

Câu 11

Hướng dẫn: Chuyển bài toán thành dạng đồ thị, mỗi đội là một đỉnh, mỗi trận đấu là một cạnh của đồ thị. Áp dụng tính chất cơ bản của đồ thị để chứng minh.

Chương 5

Câu 1

a)

1	2	3	4	5	6
1	0	0	0	0	0
2	0	0	1	0	1
3	0	1	0	1	0
4	0	0	1	0	0
5	0	1	0	0	1
6	0	1	1	0	1

b)

1	2	3	4	5
1	0	1	0	0
2	1	0	1	0
3	0	1	0	1
4	0	0	1	2
5	2	1	0	1

Danh sách cạnh (cung):

- a) (1, 2), (1, 3), (1, 5), (2, 3), (2, 5), (3, 4), (4, 5).
- b) (2, 1), (3, 1), (5, 1), (3, 2), (2, 4), (4, 5), (6, 4), (6, 5).

Danh sách kề:

a) Đỉnh 1: 2, 3, 5

Đỉnh 2: 1, 3, 5

Đỉnh 3: 1, 2, 4

Đỉnh 4: 3, 5

Đỉnh 5: 1, 2, 4

b) Đỉnh 1: \emptyset

Đỉnh 2: 1, 4

Đỉnh 3: 1, 2

Đỉnh 4: 5

Đỉnh 5: 1

Đỉnh 6: 4, 5

Câu 3

	a	b	c	d	e	f	g	h	k	l
1	0	1	0	1	1	0	0	0	0	0
2	2	1	1	0	0	1	0	0	0	0
3	0	0	0	0	0	1	1	1	0	1
4	0	0	1	1	1	0	1	1	1	0
5	0	0	0	0	0	0	0	0	1	1

Câu 4

	1	2	3	4	5	6
1	11	∞	-8	∞	∞	∞
2	∞	∞	0	3	-9	∞
3	-8	0	∞	13	1	∞
4	∞	3	13	∞	6	2
5	∞	-9	1	6	∞	0
6	∞	∞	∞	2	0	∞

Câu 5

- a) Đẳng cấu
- b) Không đẳng cấu
- c) Đẳng cấu

Chương 6

Câu 1

- a) Thứ tự duyệt các đỉnh: 0, 1, 7, 2, 6, 9, 3, 4, 5, 8
- b) Thứ tự duyệt các đỉnh: 0, 1, 2, 7, 6, 9, 3, 4, 5, 8

Câu 2

- a) Thứ tự duyệt các đỉnh: a, b, e, g, c, d, f
- b) Thứ tự duyệt các đỉnh: a, b, g, e, c, d, f

Câu 5

Hướng dẫn: Xét tất cả các cạnh của đồ thị, thực hiện bỏ 1 cạnh, kiểm tra xem số thành phần liên thông của đồ thị có tăng không. Nếu tăng, cạnh là cầu.

Câu 8

Hướng dẫn: Điều kiện để một đồ thị là đồ thị vòng: liên thông và tất cả các đỉnh có bậc là 2.

Câu 9

Hướng dẫn: Chuyển bài toán về dạng đồ thị, mỗi ô là một đỉnh. Khi hai ô có thể đi qua lại với nhau tương ứng là một cạnh. Bài toán trở thành: Đếm số thành phần liên thông của đồ thị, cho biết thành phần liên thông lớn nhất có bao nhiêu đỉnh.

Câu 10

Hướng dẫn: Chuyển bài toán về dạng đồ thị, mỗi lập trình viên là một đỉnh. Khi hai người có địa chỉ email của nhau tương ứng là một cạnh. Bài toán trở thành: Đếm số thành phần liên thông của đồ thị, cho biết tên 1 đỉnh thuộc mỗi thành phần liên thông.

Chương 7

Câu 1

- a) Có chu trình Euler.
- b) Không có chu trình Euler.

Câu 2

- a) Có chu trình Hamilton.
- b) Không có chu trình Hamilton.

Câu 6

- a) m, n chẵn.
- b) m=1 và n=1.

Nếu m và n không đồng thời bằng 1: Đồ thị có đường đi Euler khi m chẵn và n lẻ, hoặc m lẻ và n chẵn.

Câu 9

Hướng dẫn: Chuyển bài toán về dạng đồ thị. Mỗi ô trên bàn cờ là 1 đỉnh. Các ô mà quân mã có thể di chuyển qua lại tương ứng là một cạnh. Chuyển toàn bộ thông tin trên bàn cờ thành một đồ thị. Việc xác định lộ trình của quân mã để đi qua tất cả các ô tương ứng với việc xác định chu trình Hamilton của đồ thị.

Chương 8

Câu 1

Số lá: 75 đỉnh

Câu 2

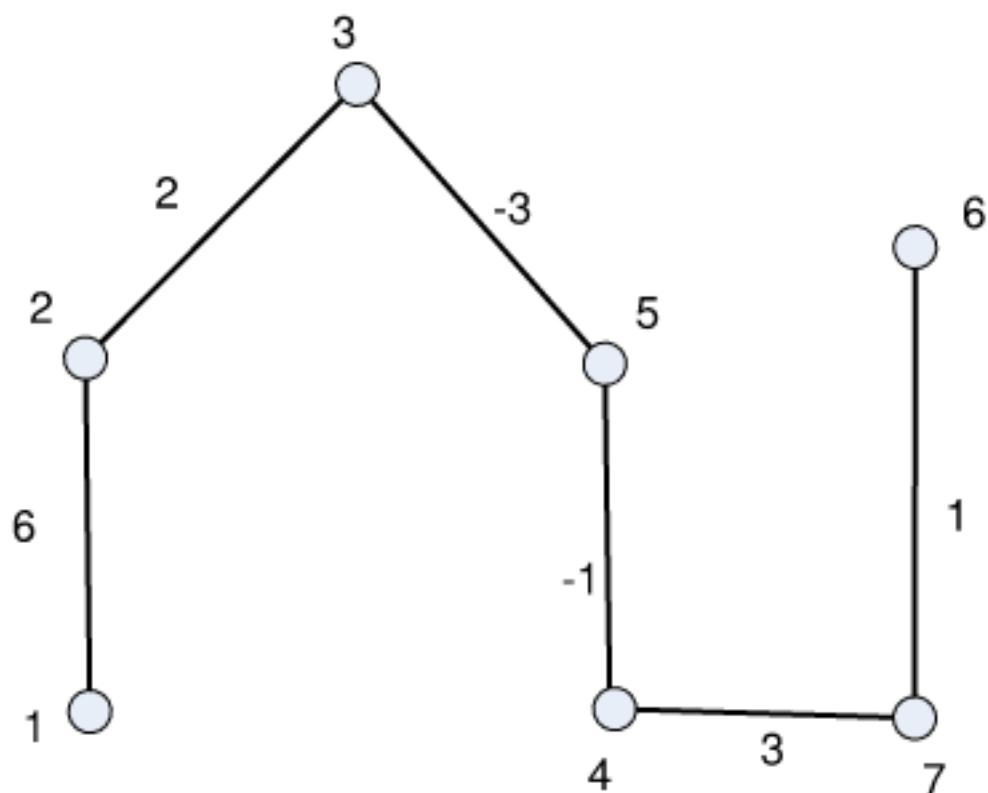
Số cạnh cần xóa: $m-n+1$

Câu 3

Khi cạnh đó là cầu

Câu 5

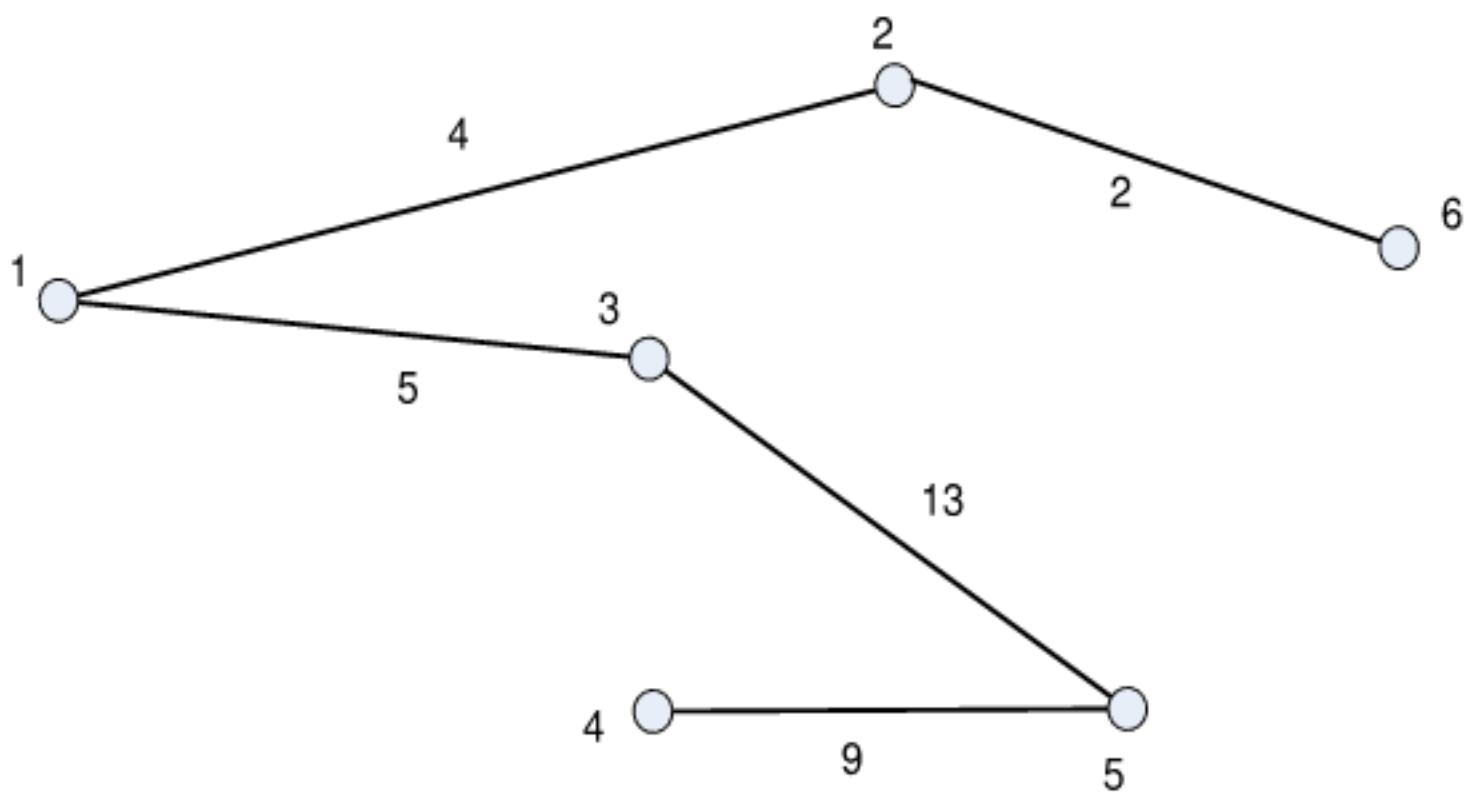
Cây khung nhỏ nhất là:



Tổng trọng số: 8

Câu 6

Cây khung nhỏ nhất là:



Tổng trọng số: 33.

Câu 7

Hướng dẫn: Kiểm tra tính liên thông và có hay không có chu trình của đồ thị.

Chương 9

Câu 1

Tham khảo chứng minh định lý đồ thị 2-màu

Câu 2

- a) 3 màu
- b) 3 màu

Câu 3

Nếu n chẵn: 3 màu.

Nếu n lẻ: 4 màu.

Chương 10

Câu 1

	d[b], Truoc[b]	d[c], Truoc[c]	d[d], Truoc[d]	d[e], Truoc[e]	d[f], Truoc[f]	d[g], Truoc[g]
1	3, a	12, a	∞ , a	∞ , a	∞ , a	∞ , a
2	3, a	12, a	-1, b	∞ , a	∞ , a,	16, c
3	3, a	12, a	-1, b	18, d	14, d	5, d
4	3, a	12, a	-1, b	16, f	12, g	5, d
5	3, a	12, a	-1, b	14, f	12, g	5, d
6	3, a	12, a	-1, b	14, f	12, g	5, d

Câu 2

	d[b], Truoc [b]	d[c], Truoc [c]	d[d], Truoc [d]	d[e], Truoc [e]	d[f], Truoc [f]	d[g], Truoc [g]	d[h], Truoc [h]
1	3, a	∞ , a	15, a	∞ , a	∞ , a	∞ , a	∞ , a
2	-	9, b	8, b	12, b	∞ , a	17, b	∞ , a
3	-	9, b	-	11, d	∞ , a	17, b	∞ , a
4	-	-	-	11, d	∞ , a	13, c	∞ , a
5	-	-	-	-	∞ , a	13, c	13, e
6	-	-	-	-	20, g	-	13, e
7	-	-	-	-	20, g	-	-
	3, a	9, b	8, b	11, d	20, g	3, c	13, e

Câu 3

Đáp án:

5	-2	-1
8	6	7
6	4	5

Ma trận D

2	0	0
1	0	1
2	0	2

Ma trận P

Câu 7

Hướng dẫn: Chuyển bài toán về dạng đồ thị, mỗi tỉnh, thành phố là một đỉnh, mỗi con đường là một cạnh, giá trị chi phí đi lại là trọng số của cạnh. Bài toán trở thành tìm đường đi ngắn nhất từ một đỉnh đến các đỉnh còn lại của đồ thị.

Chương 11

Câu 1

Luồng cực đại: $f=27$.

Tìm được lát cắt hẹp nhất (X, X^*) với: $X=\{s, a, b, c, d\}$, $X^*=\{t\}$

Phụ lục B

CHƯƠNG TRÌNH MẪU

1. Thuật toán Prim tìm cây khung nhỏ nhất

```
//Cài đặt thuật toán PRIM, sử dụng ma trận trọng số
#include<stdio.h>
#include<conio.h>
#define maxver 50
//Khai báo cấu trúc cạnh
typedef struct
{
    int Ver1;
    int Ver2;
    int W;
}ARC;

//Khai báo cấu trúc đồ thị
typedef struct
{
    int nVer;//Số đỉnh
    unsigned int a[maxver][maxver];
    unsigned char TrongV[maxver];
    ARC T[maxver];//Danh sách cạnh
    int nT;
    int dem;
}GRAPH;
```

```

//Danh sách các hàm con
void nhap(GRAPH &g);
void xuat(GRAPH &g);
void initData(GRAPH &g);
int searchArc(GRAPH &g,ARC &e,int &x,int &y);
int searchMinArc(GRAPH&g,ARC &e,int &x,int &y);
int PrimAlg(GRAPH &g);

//Hàm main
void main()
{
    GRAPH gr;
    nhap(gr);
    xuat(gr);
}

//Hàm đọc đồ thị từ tập tin
void nhap(GRAPH &g)
{
    FILE *f;
    f=fopen("Duong dan den tap tin input","rt");
    fscanf(f,"%d",&g.nVer);
    for(int i=0;i<g.nVer;i++)
    {
        for(int j=0;j<g.nVer;j++)
            fscanf(f,"%d",&g.a[i][j]);
        fscanf(f,"\n");
    }
    fclose(f);
}

```

```
//Hàm khởi tạo  
void initData(GRAPH &g)  
{  
    g.nT=0; g.dem=0;  
    g.TrongV[0]=1;  
    for(int i=1;i<g.nVer;i++)  
        g.TrongV[i]=0;  
}
```

//Hàm tìm 1 cạnh thỏa mãn điều kiện 1 đỉnh thuộc tập V, 1 đỉnh không thuộc tập V //trong danh sách cạnh

```
int searchArc(GRAPH &g,ARC &e,int &x,int &y)  
{  
    for(int i=0;i<g.nVer;i++)  
        if(g.TrongV[i]==1)  
            for(int j=0;j<g.nVer;j++)  
                if(g.TrongV[j]==0)  
                {  
                    if(g.a[i][j]!=0)  
                    {  
                        x=i;y=j;  
                        e.Ver1=i;  
                        e.Ver2=j;  
                        e.W=g.a[i][j];  
                        return 1;  
                    }  
                    if(g.a[j][i]!=0)  
                    {  
                        x=j;y=i;
```

```

        e.Ver1=j;
        e.Ver2=i;
        e.W=g.a[j][i];
        return 1;
    }

}

return 0;
}

```

*//Tìm cạnh có trọng số nhỏ nhất trong các cạnh thỏa mãn điều kiện
1 đỉnh thuộc tập //V, 1 đỉnh không thuộc tập V.*

```

int searchMinArc(GRAPH &g,ARC &e,int &x,int &y)
{
    int min,i,j;
    if(searchArc(g,e,x,y)==0)
        return 0;//ko co canh
    min=e.W;
    for(i=0;i<g.nVer;i++)
        if(g.TrongV[i]==1)
            for(j=0;j<g.nVer;j++)
                if(g.TrongV[j]==0)
                    if((g.a[i][j]!=0)&&(g.a[i][j]<min))
                    {
                        min=g.a[i][j];
                        x=i;
                        y=j;
                        e.W=g.a[i][j];
                        e.Ver1=i;
                        e.Ver2=j;
                    }
}

```

```

        }

    return 1;
}

//Giải thuật chính
int PrimAlg(GRAPH& g)
{
    initData(g);
    while(g.nT<g.nVer-1)
    {
        ARC e;
        int x,y;
        if(searchMinArc(g,e,x,y)==0)
            return 0;
        g.dem=g.dem+e.W;
        g.T[g.nT]=e;
        g.TrongV[y] = 1;
        (g.nT)++;
    }
    return 1;
}

```

```

//Hàm xuất kết quả ra tập tin
void xuat(GRAPH &g)
{
    FILE *f;
    int i;
    f=fopen("Duong dan den tap tin output","wt");
    if(PrimAlg(g)==0)

```

```

{
    fprintf(f, "\nDo thi khong co cay khung!\n");
}
else
{
    fprintf(f, "Tong trong so:%d", g.dem);

    fprintf(f, "\n%d", g.nT);
    fprintf(f, "\n");
    for(i=0;i<g.nT;i++)
    {
        fprintf(f, "%2d--> %2d", g.T[i].Ver1+1,
g.T[i].Ver2+1);
        fprintf(f, "\n");
    }
    fclose(f);
}

```

2. Thuật toán Kruskal tìm cây khung nhỏ nhất

```

//Cài đặt thuật toán KRUSKAL
//Duyệt theo danh sách cạnh (sắp xếp cạnh theo thứ tự trọng số
tăng dần)
#include<stdio.h>
#include<conio.h>
#include<dos.h>
const MAXVER=50;
const MAXEDGE=100;

```

//Khai báo cấu trúc cạnh

```
typedef struct
```

```
{
```

```
    int    Ver1;
```

```
    int    Ver2;
```

```
    int    w;
```

```
}ARC;
```

//Khai báo cấu trúc đồ thị

```
typedef struct
```

```
{
```

```
    int    nVer;
```

```
    int    L[MAXVER][MAXVER];
```

```
    ARC   T[MAXVER-1];      //cay khung nho nhat
```

```
    int    nT;
```

```
    ARC   DS[MAXEDGE];     //danh sach canh
```

```
    int    nDS;
```

```
}GRAPH;
```

//Các hàm con

```
void    Input(GRAPH&);
```

```
void    Output(GRAPH&);
```

```
void    ReadEdgeList(GRAPH&);
```

```
void    Sort(GRAPH&);
```

```
int     Kruskal(GRAPH&);
```

//Hàm main

```
void    main()
```

```
{
```

```
    GRAPH    gr;
```

```
    Input(gr);
    Output(gr);
}
```

//Hàm đọc đồ thị từ tập tin

```
void      Input(GRAPH&      g)
{
    FILE *f;
    f=fopen("Duong dan den tap tin input","rt");
    fscanf(f,"%d",&g.nVer);
    for(int i=0;i<g.nVer;i++)
    {
        for(int j=0;j<g.nVer;j++)
            fscanf(f,"%5d",&g.L[i][j]);
        fscanf(f,"\n");
    }
    fclose(f);
}
```

//Hàm xuất kết quả ra tập tin

```
void      Output(GRAPH&      g)
{
    FILE* f;
    f=fopen("Duong dan den tap tin output","wt");
    fprintf(f,"%d",g.nVer);
    fprintf(f,"\n");
    for(int i=0;i<g.nVer;i++)
    {
        for(int j=0;j<g.nVer;j++)

```

```

        fprintf(f,"%5d",g.L[i][j]);
        fprintf(f,"\n");
    }

    Sort(g);

    fprintf(f,"Danh sach canh sau khi da sap tang:\n");
    for(i=0;i<g.nDS;i++)
    {
        fprintf(f,"%d-->%d",g.DS[i].Ver1+1,g.DS[i].Ver2+1,g.DS[i].w);
        fprintf(f,"\n");
    }

    if(Kruskal(g)==0)
        fprintf(f,"\nDo thi khong lien thong\n");
    else
    {
        fprintf(f,"Danh sach canh trong do thi trong so nho
nhat:\n");
        for(i=0;i<g.nT;i++)
        {
            fprintf(f,"%d-->%d:
%d",g.T[i].Ver1+1,g.T[i].Ver2+1,g.T[i].w);
            fprintf(f,"\n");
        }
    }

    fclose(f);
}

```

//Hàm khởi tạo

```

void      InitData(GRAPH& g)
{

```

```

g.nT=0;
}

//Hàm xây dựng danh sách cạnh từ mà trận trọng số

void      ReadEdgeList(GRAPH& g)
{
    g.nDS=0;
    for(int i=0; i<g.nVer; i++)
        for(int j=i; j<g.nVer; j++)
        {
            if(g.L[i][j]!=0)//||(g.L[j][i]))
            {
                g.DS[g.nDS].Ver1=i;
                g.DS[g.nDS].Ver2=j;
                g.DS[g.nDS].w=g.L[j][i];
                g.nDS++;
            }
        }
}

```

```

//Hàm sắp xếp danh sách cạnh theo thứ tự trọng số tăng dần

void      Sort(GRAPH& g)
{
    ReadEdgeList(g);
    ARC tam;
    for(int i=0;i<g.nDS-1;i++)
        for(int j=i+1;j<g.nDS;j++)
            if(g.DS[i].w>g.DS[j].w)
            {
                tam=g.DS[i];

```

```
        g.DS[i]=g.DS[j];
        g.DS[j]=tam;
    }
}
```

//Giải thuật chính

```
int      Kruskal(GRAPH& g)
{
    int i,j;
    int Sohieu[MAXVER];
    InitData(g);
    //Gán nhãn
    for(i=0; i<g.nVer; i++)
        Sohieu[i]=i;
    j=0;
    do
    {
        if(Sohieu[g.DS[j].Ver1]!=Sohieu[g.DS[j].Ver2])
        {
            //Đưa cạnh j vào cây T
            g.T[g.nT].Ver1=g.DS[j].Ver1;
            g.T[g.nT].Ver2=g.DS[j].Ver2;
            g.T[g.nT].w=g.DS[j].w;

            //Cập nhật lại số hiệu
            int x=Sohieu[g.DS[j].Ver2];
            int y=Sohieu[g.DS[j].Ver1];
            for(int l=0; l<g.nVer; l++)
            {

```

```

        if(Sohieu[l]==y)
            Sohieu[l]=x;
        }
        g.nT++;
    }
    j++;
}while ((g.nT<g.nVer)&&(j<g.nDS));
if(g.nT<g.nVer-1)    return 0;
return 1;
}

```

3. Thuật toán Dijkstra tìm đường đi ngắn nhất từ một đỉnh đến các đỉnh còn lại của đồ thị

```

//Cài đặt thuật toán DIJKSTRA
#include<stdio.h>
#include"conio.h"
#define MAX 100
#define VOCUC -1

```

//Khai báo cấu trúc đồ thị

```

typedef struct
{
    int n;
    int a[MAX][MAX];
}GRAPH;
int T[MAX];
int arrnCost[MAX];
int arrnPrevious[MAX];
int nStartNode;
int nStopNode;

```

```

//Các hàm con
void Input(GRAPH &g);
void InitData(GRAPH &g);
int Dijkstra(GRAPH &g);
void PrintResult(GRAPH &g);

//Hàm main
void main()
{
    GRAPH g;
    Input(g);
    printf("Nhập vào đỉnh bắt đầu ");
    scanf("%d",&nStartNode);
    nStartNode-=1;
    printf("Nhập vào đỉnh kết thúc ");
    scanf("%d",&nStopNode);
    nStopNode-=1;
    PrintResult(g);
}

//Hàm đọc đồ thị từ tập tin
void Input(GRAPH &g)
{
    FILE *f;
    f=fopen("Đường dẫn đến tập tin input","rt");
    fscanf(f,"%d",&g.n);
    for(int i=0;i<g.n;i++)
        for(int j=0;j<g.n;j++)

```

```

        fscanf(f,"%d",&g.a[i][j]);
    fclose(f);
}

//Hàm khởi tạo dữ liệu
void InitData(GRAPH &g)
{
    for(int i=0;i<g.n;i++)
    {
        T[i]=1;
        if(i!=nStartNode)
            arrnCost[i]= VOCUC;
    }
    arrnCost[nStartNode]=0;
}

//Giải thuật chính
int Dijkstra(GRAPH &g)
{
    InitData(g);
    int v=-1;
    int c=0;
    int min;
    while(v!=nStopNode)
    {
        min=32767;
        for(int i=0;i<g.n;i++)
            if(T[i]==1 && min > arrnCost[i] &&
arrnCost[i]!=VOCUC)

```

```

    {
        min=arrnCost[i];
        v=i;
        c=1;
    }

    if(c==0) {return 0; break;}
    T[v] = 0;
    for(int k=0;k<g.n;k++)

        if(T[k]==1 && g.a[v][k]!=0 &&
(arrnCost[k]==VOCUC ||
        (arrnCost[k] >arrnCost[v] + g.a[v][k])))

    {
        arrnCost[k] = arrnCost[v] + g.a[v][k];

        arrnPrevious[k]=v;
    }

}

return 1;
}

```

//Hàm xuất kết quả ra màn hình

```

void PrintResult(GRAPH &g)
{
    int k;
    if(Dijkstra(g)==0) printf("khong co duong di");
    else
    {
        printf("\n Chi phi toi uu la: %d",arrnCost[nStopNode]);
    }
}

```

```

        printf("\n Duong di tim duoc: \n");
        k=nStopNode;
        printf("%d ",k+1);
        do
        {
            k=arrnPrevious[k];
            printf(" <---- %d ", k+1);
        }while(k!=nStartNode);
    }
}

```

4. Thuật toán Floyd tìm đường đi ngắn nhất giữa tất cả các cặp đỉnh của đồ thị

```

#include<stdio.h>
#define MAX 100
struct GRAPH
{
    int n;
    int a[MAX][MAX];
};

int d[MAX][MAX];
int p[MAX][MAX];
//Hàm nhập đồ thị
void Input(GRAPH &g)
{
    FILE* f;
    f = fopen("Duong dan den tap tin input ", "rt");
    if (f == NULL)
    {
        printf("Khong mo duoc file\n");
    }
}

```

```

        return;
    }

    fscanf(f, "%d", &g.n);
    for (int i=0; i<g.n; i++)
        for (int j=0; j<g.n; j++)
            fscanf(f, "%d", &g.a[i][j]);
    fclose(f);
}

//Hàm xuất đồ thị ra màn hình
void Output(GRAPH &g)
{
    printf("%d\n", g.n);
    int i, j;
    for (i=0; i<g.n; i++)
    {
        for (j=0; j<g.n; j++)
            printf("%5d", g.a[i][j]);
        printf("\n");
    }
}

//Hàm xuất kết quả hai ma trận d[][] và p[][]
void XuatKQ(GRAPH g)
{
    printf("\n");
    for(int i=0;i<g.n;i++)
    {
        for(int j=0;j<g.n;j++)
            printf("%10d",d[i][j]);
        printf("\n");
    }
}

```

```

}

printf("\n");
for(int i=0;i<g.n;i++)
{
    for(int j=0;j<g.n;j++)
        printf("%10d",p[i][j]);
    printf("\n");
}
//Giải thuật chính Floyd
void Floyd(GRAPH& g)
{
    for (int i = 0; i <g.n; i++) /* Khởi tạo */
        for (int j = 0; j <g.n; j++)
    {
        d[i][j] = g.a[i][j];
        p[i][j] = i;
    }
    for (int k = 0; k < g.n; k++) /* 3 vòng lặp */
        for (int i = 0; i < g.n; i++)
            for (int j = 0; j < g.n; j++)
                if (d[i][j] > d[i][k] + d[k][j])
    {
        d[i][j] = d[i][k] + d[k][j];
        p[i][j] = p[k][j];
    }
    XuatKQ(g);
}
//Hàm main

```

```
void main()
{
    GRAPH g;
    Input(g);
    Output(g);
    Floyd(g);

}
```

TÀI LIỆU THAM KHẢO

- [1]. Nguyễn Đức Nghĩa, Nguyễn Tô Thành (1999), *Toán rời rạc*, NXB Giáo Dục.
- [2]. Kenneth H. Rosen (1997), *Toán học rời rạc ứng dụng trong tin học*, NXB Khoa học và Kỹ thuật.
- [3]. Robert Sedgewick (2004), *Cẩm nang thuật toán*, NXB Khoa học và Kỹ thuật.
- [4]. Lê Tự Hỷ (2005), *Toán rời rạc*, NXB Giáo Dục.
- [5]. Reinhard Diestel (2005), *Graph theory*, Springer-Velag.
- [6]. Nguyễn Hữu Anh (2003), *Toán rời rạc*, ĐH KHTN TP HCM.

**Giáo trình
TOÁN RỜI RẠC VÀ LÝ THUYẾT ĐỘ THỊ**

TS. NGUYỄN THÀNH SƠN
TS. ĐẶNG TRƯỜNG SƠN
ThS. LÊ VĂN VINH
ThS. TRẦN CÔNG TÚ
ThS. NGUYỄN QUANG NGỌC
TS. NGUYỄN PHƯƠNG

Bản tiếng Việt ©, TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH, NXB ĐHQG-HCM và TÁC GIẢ

Bản quyền tác phẩm đã được bảo hộ bởi Luật xuất bản và Luật Sở hữu trí tuệ Việt Nam. Nghiêm cấm mọi hình thức xuất bản, sao chụp, phát tán nội dung khi chưa có sự đồng ý của tác giả và Nhà xuất bản.

ĐÊ CÓ SÁCH HAY, CẦN CHUNG TAY BẢO VỆ TÁC QUYỀN!



ISBN: 978-604-73-4457-4

A standard linear barcode representing the ISBN number 978-604-73-4457-4. The number 9 786047 344574 is printed below the barcode.

9 786047 344574