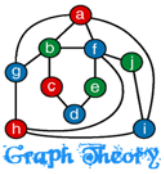
Lý Thuyết Đồ Thị
Đại Học Khoa Học Tự Nhiên Tp.HCM
227 Nguyễn Văn Cừ, Q.5

GVHD: Lê Ngọc Thành
E-mail: lnthanh@fit.hcmus.edu.vn
Updated 2013



Biểu diễn và thao tác trên đồ thị

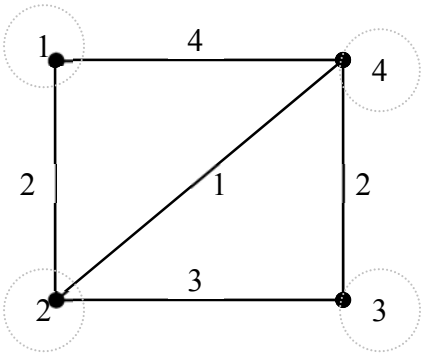
1. Tóm tắt một số khái niệm

- *Đồ thị* là một thể hiện bao gồm một tập hợp các đối tượng và giữa chúng có thể có các liên kết. Trong toán học người ta gọi các đối tượng là các đỉnh và các liên kết là các cạnh.
- *Đồ thị vô hướng* là đồ thị bao gồm các cạnh không có hướng.
- *Đồ thị có hướng* là đồ thị bao gồm các cạnh có thứ tự các đỉnh hay là các cạnh có hướng.
- *Khuyên* là cạnh nối đỉnh với chính nó.
- *Cạnh song song* là các cạnh có chung cặp điểm liên kết.
- *Đồ thị đơn* là đồ thị không có khuyên và cạnh song song.
- *Đồ thị đủ* là đồ thị vô hướng, đơn và giữa 2 đỉnh bất kì có duy nhất 1 cạnh nối chúng.
- *Đồ thị lưỡng phân* là đồ thị vô hướng bao gồm 2 tập đỉnh mà chỉ có các cạnh liên kết đỉnh từ tập này đến tập kia chứ không có liên kết nào bên trong mỗi tập.
- *Đồ thị lưỡng phân đủ* là đồ thị lưỡng phân mà có đầy đủ các cạnh nối mỗi đỉnh từ tập này đến mỗi đỉnh của tập kia. Mỗi cặp có duy nhất một cạnh.
- *Bậc của một đỉnh* là số cạnh kề với đỉnh đó đối với đồ thị vô hướng hay tổng số cạnh đi ra và đi vào đỉnh đó đối với đồ thị có hướng.
- *Đỉnh treo* là đỉnh có bậc bằng 1.
- *Đỉnh cô lập* là đỉnh có bậc bằng 0.
- *Trọng số của một cạnh* là giá trị thể hiện chi phí hay độ lớn của cạnh nối hai đỉnh trong đồ thị. Nếu đồ thị không ghi trọng số, chúng ta có thể hiểu đồ thị có các trọng số bằng nhau và bằng giá trị của 1 đơn vị.

2. Biểu diễn đồ thị

Người ta thường dùng ***ma trận kề*** để biểu diễn đồ thị. Một giá trị A[i,j] trong ma trận (dòng i cột j của ma trận A) ứng với trọng số của của cạnh (i, j) trong đồ thị. Một giá trị đặc biệt thường được sử dụng cho phần tử A[i, j] (thường là 0) để cho biết không có cạnh nối giữa cặp đỉnh (i, j).

Ví dụ:



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

Nhận xét:

- Đường chéo chính trong đồ thị có gí trị 0 (do trong đồ thị không có khuyên)
- Đồ thị vô hướng có các giá trị đối xứng qua đường chéo chính

Trong một số bài toán ta không quan tâm đến trọng số của đồ thị mà chỉ quan tâm là có tồn tại một cạnh (i, j) trong đồ thị hay không. Đối với trường hợp này, ta sử dụng giá trị 1 cho A[i, j] để chỉ ra sự tồn tại của cạnh (i, j).

3. Hướng dẫn cài đặt

3.1. Lưu trữ

Dữ liệu thể hiện ma trận kề thường được lưu trong một tập tin văn bản có cấu trúc 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.

Vd: đồ thị có hướng ở trên sẽ được lưu trong tập tin DOTHI.TXT như sau

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

3.2. Đọc và xuất đồ thị

Trong ngôn ngữ lập trình, để lưu trữ ma trận kề người ta thường sử dụng mảng hai chiều:

```
#define  MAX  100
int  n;
int  a[MAX][MAX];
```

Tốt hơn là ta sử dụng một struct hoặc class để cài đặt, khi đó nếu có nhiều đồ thị, ta sẽ có g1, g2, … là các biến ứng với các đồ thị:

```
#define  MAX  100
struct  GRAPH
{
      int  n;
      int  a[MAX][MAX];
};
```

Sau khi đã có cấu trúc dữ liệu để lưu trữ, ta có thể tiến hành đọc đồ thị từ file lên bộ nhớ chính và thực hiện tác in ra màn hình console để kiểm thử.

```
void DocDoThi(GRAPH &g)
{
    FILE* f;
    f = fopen("dothi.txt", "rt"); // co the truyen ten file thong qua bien
cua ham
    if (f == NULL)
    {
        g.n = 0;
        return;
    }
    fscanf(f, "%d", &g.n);
    int i, j;
    for (i=0; i<g.n; i++){
        for (j=0; j<g.n; j++){
            fscanf(f, "%d", &g.a[i][j]);
        }
    }
    fclose(f);
}

void XuatDoThi(GRAPH g)
{
    int i, j;
    for (i=0; i<g.n; i++){
```

```
            for (j=0; j<g.n; j++){
                printf( "%d\t", g.a[i][j]);
            }
            printf( "\n");
        }
    }
```

3.3. Kiểm tra đồ thị không có khuyên

Nếu đồ thị của chúng ta không có khuyên, đường chéo chính của ma trận kề sẽ chỉ chứa các giá trị khác 0. Ta sẽ viết hàm *KiemTraDoThiCoKhuyen* có kết quả trả về 1 nếu đồ thị có khuyên, 0 nếu ngược lại.

```
KiemTraDoThiCoKhuyen (GRAPH &g)
{
// kiểm tra các giá trị a[0][0], a[1][1], … xem có giá trị khác 0 hay không
// nếu có, nghĩa là ma trận kề biểu diễn đồ thị có khuyên
    int i;
    for (i=0; i<g.n; i++)
        if (a[i][i] != 0)
            return 1;
    return 0;
}
// cách sử dụng trong hàm main như sau
//  if ( KiemTraDoThiCoKhuyen(g) == 1)
//  {
//    printf("Ma tran ke bieu dien do thi co khuyen ");
//    return;
//  }
```

3.4. Kiểm tra đồ thị vô hướng

Ta đã biết là đồ thị vô hướng được biểu diễn bởi ma trận kề đối xứng qua đường chéo chính, việc xây dựng hàm kiểm tra ma trận có đối xứng là cần thiết. Ta sẽ xây dựng hàm *KiemTraDoThiVoHuong* có kết quả trả về là 1 nếu ma trận đối xứng, 0 nếu ngược lại.

Ta chỉ đơn giản sử dụng hai vòng for i, j để xem a[i][j] có bằng với a[j][i] hay không, nếu có bất kỳ trường hợp nào như vậy, hàm sẽ kết thúc với giá trị 0.

```
KiemTraDoThiVoHuong(GRAPH &g)
{
  // kiểm tra xem các giá trị a[i][j] có bằng với a[j][i] hay không
    // nếu có, nghĩa là đồ thị không đối xứng
    int i, j;
    for (i=0; i<g.n; i++)    // có thể giảm bớt các bước kiểm tra thừa với
        for (j=0; j<g.n; j++)    // for (j=i+1; j<n; j++)
            if ( a[i][j] != a[j][i])
                return 0;
    return 1;
}
```

4. Bài tập áp dụng

Hãy viết thêm các hàm để làm các thao tác sau trên cả đồ thị có hướng và vô hướng:

1. Đếm số lượng đỉnh và cạnh của đồ thị.(Cần phân biệt rõ 2 trường hợp đồ thị có hướng và vô hướng)
2. Xác định đỉnh kề của 1 đỉnh bất kỳ
3. Tính bậc của từng đỉnh
4. Đếm số lượng đỉnh bậc chẵn
5. Đếm số lượng đỉnh bậc lẻ
6. Đếm số lượng đỉnh treo
7. Đếm số lượng đỉnh cô lập

8. Chuyển đổi đồ thị có hướng thành vô hướng
9. Thêm hay bớt một cạnh
10. Thêm hay bớt một đỉnh
11. Thay đổi giá trị trong số của cạnh (i,j)