图论作业(第二周)

黄瑞轩 PB20111686

Chap 1 Prob. 24

取 G 中最长的轨道 $P(u,v)$ 。对于 u ，其下一个顶点记为 w ，除了边 uw 外，至少还有 $\delta(G)-1$ 个待分配度数，说明 u 与 $P(u,v)$ 上除了 w 外的另外 $\delta(G)-1$ 个点相邻，不妨根据离 u 的距离分别设这 $\delta(G)-1$ 个点为 $v_1,\dots,v_{\delta(G)-1}$ ，则在 P 轨道上

$$\operatorname{dist}(u,v_{\delta(G)-1})\geq\delta(G)$$

加之 u 与 $v_{\delta(G)-1}$ 相邻，则圈 $uwv_1\dots v_{\delta(G)-1}u$ 的长度

$$L(uwv_1\dots v_{\delta(G)-1}u)\geq\delta(G)+1$$

至此我们构造出了这样一个长度不小于 $\delta(G)+1$ 的圈，是否一定能取到等号?

考虑图 G' : $(\{v_1,v_2,v_3,v_4\},\{v_1v_2,v_2v_3,v_3v_4,v_4v_1\})$ ， $\delta(G')=2$ 而图中不存在长度为 3 的圈。故等号不能取得，题中描述有误，应该是一定有一个长度不小于 $\delta(G)+1$ 的圈。

Chap 1 Prob. 26

直接用Dijkstra算法即可，程序如下:

```
#include<iostream>
#include<vector>
#define G 500

using namespace std;

int map[7][7] = {
    {0,0 ,0 ,0 ,0 ,0 ,0 },
    {0,0 ,50,G ,40,25,10},
    {0,50,0 ,15,20,G ,25},
    {0,G ,15,0 ,10,20,G },
    {0,40,20,10,0 ,10,25},
    {0,25,G ,20,10,0 ,55},
    {0,10,25,G ,25,55,0 } };

int d[7] = { G,G,G,G,G,G,G };
int l[7] = { -1,-1,-1,-1,-1,-1,-1 };
bool s[7] = { false,false,false,false,false,false,false };
vector<int> vertex;

int main() {
    vertex.push_back(1);
    d[vertex[0]] = 0;
    l[vertex[0]] = vertex[0];
    s[vertex[0]] = true;
    int i = 0;
loop:;
    for (int j = 1; j <= 6; j++) {
        if (s[j] == false) {
            if (d[vertex[i]] + map[vertex[i]][j] < d[j]) {
                d[j] = d[vertex[i]] + map[vertex[i]][j];
                l[j] = vertex[i];
            }
        }
    }
    int least = G;
    int least_node = -1;
    for (int j = 1; j <= 6; j++) {
        if (s[j] == false) {
            if (least > d[j]) {
                least = d[j];
                least_node = j;
            }
        }
    }
    vertex.push_back(least_node);
    d[vertex[i + 1]] = least;
    s[vertex[i + 1]] = true;
    if (i == 5) {
        goto start;
```

```
    }
    else {
        i++;
        goto loop;
    }
start:;
    for (int i = 6; i >= 2; i--) {
        int f = i;
        cout << f << " ";
        for (; l[f] != 1;) {
            cout << l[f] << " ";
            f = l[f];
        }
        cout << "1" << endl;
    }
}
```

程序输出：

```
6 1
5 1
4 6 1
3 5 1
2 6 1
```

即制定的路线图为：

$$
\begin{aligned}
&c_1 \to c_6 \to c_2 \\
&c_1 \to c_5 \to c_3 \\
&c_1 \to c_6 \to c_4 \\
&c_1 \to c_5 \\
&c_1 \to c_6
\end{aligned}
$$

Chap 1 Prob. 27

我们考察河一边的元素情况，按人、狼、羊、菜的顺序排列构造一个二进制数，如状态为人、狼、菜时状态为 1101 ，由题可列出总状态数为 10 。

将这 10 种状态作为图 G 的顶点，可以互相转化的两个顶点之间连一条边，用邻接表的形式存图如下：

状态	下一个状态1	下一个状态2	下一个状态3
1010	转0000	转0010	
1011	转0001	转0010	
1101	转0100	转0001	转0101
1110	转0100	转0010	
1111	转0101		
0101	转1111	转1101	
0100	转1101	转1110	
0010	转1011	转1110	转1010
0001	转1011	转1101	
0000	转1010		

设此图每边边权均为 1 。对此图，我们要找 1111 到 0000 的最短路径，运用Dijkstra算法，得到可取的方案为：

$$
\begin{aligned}
1111 - 0101 - 1101 - 0100 - 1110 - 0010 - 1010 - 0000 \\
1111 - 0101 - 1101 - 0001 - 1011 - 0010 - 1010 - 0000
\end{aligned}
$$

事实上这也是此图从 1111 到 0000 的唯二的两条轨道。

Chap 2 Prob. 3

因为这是树（记为 T ），所以连通，只需证明所有顶点的度数不超过 2 即可。

考虑 T 中一条最长的轨道 P ，假设 P 上的一个节点（记为 u ）在 T 中作为非叶子节点的度数超过 2 ，由于这是树，树中没有圈，因此 u 一定还与 $V[T-P]$ 中的节点相邻，不妨记为 v 。以此类推，以 u 为起点，往 $u \to v$ 方向延伸出去的一条轨道一定终结于某个 $V[T-P]$ 中的顶点，不妨记为 w ，不妨设 w 的度数为 1 ，否则仿前继续递推，这导致 T 中出现了至少三片树叶，不可。故所有顶点的度数不超过 2 ，结合连通，故这棵树是一条轨道。

Chap 2 Prob. 4

设 T 中度数为 $\Delta(T)$ 的节点为 u ，设与 u 相邻的顶点分别是 $v_1, v_2, \ldots, v_j (j \geq n)$ ，则 $T-u$ 一定是由 j 棵树（记为 $t_1, t_2, \ldots, t_j$ ）构成的森林，这由树中没有圈很容易证得。而在森林 $\{t_i + v_i u | 1 \leq i \leq j\}$ 中，每棵树都至少有一片树叶与 T 中的树叶一一对应，故 T 中一定有 $j \geq n$ 片树叶。

Chap 2 Prob. 6

首先，一棵树（记为 T ）一定有中心，这是显然的（从离心率和半径的定义就可以看出）。这使得我们可以取出一个中心点来，不妨记为 c ，可以选出 $t_k \in V(T)$ ，使得 $\mathrm{dist}(c, t_k) = l(c)$ ，这样的 t_k 一定是叶子，否则与 $l(c) = \max\{\mathrm{dist}(c, v) | v \in V(T)\}$ 相悖。

记 T 中所有叶子为 $t_1, t_2, \ldots, t_s (s \geq k)$ ，叶子 t_i 的前驱记为 $p(t_i)$ ，则 $\mathrm{dist}(c, t_i) = \mathrm{dist}(c, p(t_i)) + 1$ ，由 $l(c)$ 的定义知道，去除 T 的所有叶子后，有

$$l(c) = \mathrm{dist}(c, p(t_k)) = \max\{\mathrm{dist}(c, v) | v \in V - \{t_i | 1 \leq i \leq s\}\}$$

即原来是 T 的中心点的节点，删除 T 的所有叶子后仍然是 T 的中心点。不断地递归此过程，现在断言边界条件为两种情况：

（1）此时 T 仍然有叶子，但再进行一次操作会导致 T 为空；

（2）此时 T 没有叶子，但不为空，是孤立的一个点。

下面来证明上面的断言是正确的。假设递归到某过程中， T ：

（a）仍然有叶子，再进行一次操作 T 也不为空；

（b）仍然有叶子，但再进行一次操作会导致 T 为空；

（c）是空树；

（d）是孤立的一个点；

上面四种情况就是全集。对（a）还可以继续递归为（d）或者此时可以转化为（b），所以（a）不是边界条件；（c）也不是，此时已无中心点可言。而容易验证（b）（d）即上面的（1）（2）是递归的边界条件，因为递归到这两种情况时不会删除原有的中心点。

这里的（1）（2）即对应两种情况：

（1）原来的树有两个相邻的中心点；

（2）原来的树只有一个中心点。

综上，一棵树或有两个相邻的中心点，或只有一个中心点。

Chap 2 Prob. 8

必要性：

若此序列是树的度数序列，由树的性质知道

$$\varepsilon = \nu - 1$$

由Euler定理知道

$$\sum_{i=1}^{\nu} d_i = 2\varepsilon = 2(\nu - 1)$$

充分性：

（1）当 $\nu = 2$ 时， $d_1 + d_2 = 2 \Leftrightarrow d_1 = d_2 = 1$ ，该图是 K_2 ，显然是树；

（2）假设当 $\nu = k - 1$ 时， $d_1 + d_2 + \ldots + d_{k-1} = 2(k-2)$ 可以推出这是某棵树的度数序列；

（3）当 $\nu = k$ 时，假设这 ν 个顶点构成的图为 G ， $d_1 + d_2 + \ldots + d_{k-1} + d_k = 2(k-1) = 2(k-2) + 2$ ，我们能够构造一个图，使得

$$d_1 + d_2 + \ldots + d_{k-1} = 2(k-2)$$

这由归纳假设保证，并且归纳假设保证这可以是一棵树（不妨记为 T ）的度数序列，这棵树是 G 的顶点 $(v_1, v_2, \ldots, v_{k-1})$ 导出子图。现在只需要保证 $d_k = 2$ ，这很容易做到，选 T 的一片树叶 v_j ，其前驱记为 $p(v_j)$ ，将路径 $p(v_j)v_j$ 改造为 $p(v_j)v_kv_j$ 即可，显然改造后的图仍然是一棵树。

综上，原命题成立。

Chap 2 Prob. 11