SPFA算法

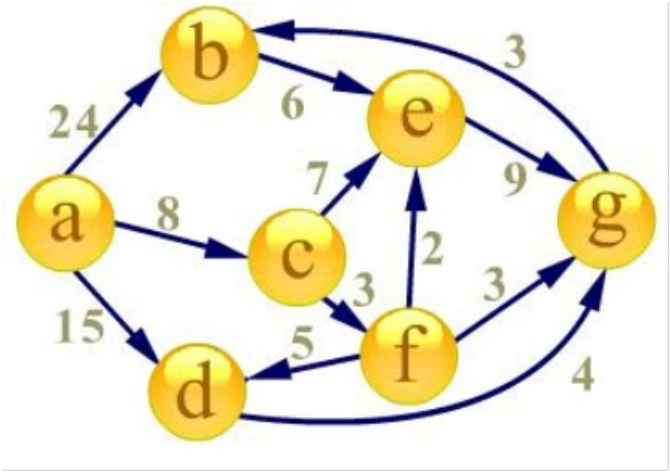
闲聊

今天在网上查关于SPFA的算法的资料发现有好多都是一样的图片什么的都是一样的真尴尬。

我找了一个自认为比较好的按照上面学习，上面有些说法很模糊比如松弛什么的我开始什么都不知道没有一点概念真尴尬

题目描述

给定一个有向图a-g求最短路径



- 这个图片是我抄别人的,表示不会做图

为了更好的写代码我把上面的图片转化成为如下形式

```
1 2 24
1 3 8
1 4 15
2 5 6
3 5 7
3 6 3
4 7 4
5 7 9
6 7 3
6 4 5
7 2 3
6 5 2
```

> 比如第一行 表示的就是A->B他们之间的长度为24

算法简介

SPFA算法是求单源最短路径的一种算法,它是一种十分高效的最短路算法。

很多时候，给定的图存在负权边，这时类似Dijkstra等算法便没有了用武之地，而Bellman-Ford算法的复杂度又过高，SPFA算法便派上用场了。SPFA的复杂度大约是O(kE),k是每个点的平均进队次数(一般的，k是一个常数，在稀疏图中小于2)。

但是，SPFA算法稳定性较差，在稠密图中SPFA算法时间复杂度会退化。

实现方法

1. 建立一个队列，初始时队列里只有起始点
2. 建立一个表格记录起始点到所有点的最短路径（该表格的初始值要赋为极大值，该点到他本身的路径赋为0）

3. 执行松弛操作，用队列里有的点去刷新起始点到所有点的最短路
4. 如果刷新成功且被刷新点不在队列中则把该点加入到队列最后。重复执行直到队列为空。
5. 此外，SPFA算法还可以判断图中是否有负权环，即一个点入队次数超过N。

- 第五点还有点不明白

具体实现

```
#include <iostream>
#include <vector>
#include <queue>
#include <string.h>

#define INF 0x3f
using namespace std;

struct node{
int y;
int quan;

}t;

vector<node>mymap[20];
queue<int>Q;
int d[20];
int xx[20];
int main()
{
    int n=7;
    int l=12;

    memset(d,INF,sizeof(d));
    memset(xx,INF,sizeof(xx));
    /*
1 2 24
1 3 8
1 4 15
2 5 6
3 5 7
3 6 3
4 7 4
5 7 9
6 7 3
6 4 5
7 2 3
6 5 2
    */
    for(int i=1;i<=l;i++)
    {
        int a,b,c;
        cin>>a>>b>>c;
        t.quan=c;
        t.y=b;
        mymap[a].push_back(t);
    }
    Q.push(1);
    d[1]=0;
    xx[1]=0;
    while(!Q.empty())
    {
```

```
        int tmp=Q.front();
        Q.pop();
        xx[tmp]=INF;
        for(unsigned int i=0;i<mymap[tmp].size();i++)
        {
        if(mymap[tmp][i].quan+d[tmp]<d[mymap[tmp][i].y]) //
        {
            d[mymap[tmp][i].y]=mymap[tmp][i].quan+d[tmp];
            if(xx[mymap[tmp][i].y]!=0)
            {
                Q.push(mymap[tmp][i].y);
                xx[mymap[tmp][i].y]=0;
            }
        }

        }
        for(int i=1;i<n+1;i++)
        cout<<d[i]<<"  ";
        cout<<endl;
    }
//    for(int i=1;i<n+1;i++)
//        cout<<d[i]<<"  ";
//    cout<<endl;
    return 0;
}
```

在代码里面我把每一次更新以后的最短路都打了出来如下：

```
0  24  8  15  1061109567  1061109567  1061109567
0  24  8  15  30  1061109567  1061109567
0  24  8  15  15  11  1061109567
0  24  8  15  15  11  19
0  24  8  15  15  11  19
0  24  8  15  13  11  14
0  17  8  15  13  11  14
0  17  8  15  13  11  14
0  17  8  15  13  11  14
```

最后一行就是我们要得a到其他各点的最短路了

作者语

开始的时候我对“”如果刷新成功且被刷新点不在队列中则把该点加入到队列最后“” 这一句话理解有一点问题 这一句话的意思是先刷新如果刷新成功就看这一点是否在队列中如果在队列里面就不入队否则就入队

- 不知道自己说的明白不，我就是这么理解的