

北京大学暑期课 《ACM/ICPC竞赛训练》

北京大学信息学院 郭炜

guo_wei@PKU.EDU.CN

<http://weibo.com/guoweiofpku>

课程网页: http://acm.pku.edu.cn/summerschool/pku_acm_train.htm



北京大学
PEKING UNIVERSITY

深度优先搜索

寻路问题

ROADS (POJ1724)

N个城市，编号1到N。城市间有R条单向道路。

每条道路连接两个城市，有长度和过路费两个属性。

Bob只有K块钱，他想从城市1走到城市N。问最短共需要走多长的路。如果到不了N，输出-1

$2 \leq N \leq 100$

$0 \leq K \leq 10000$

$1 \leq R \leq 10000$

每条路的长度 L, $1 \leq L \leq 100$

每条路的过路费T, $0 \leq T \leq 100$

输入:

K

N

R

$s_1 e_1 L_1 T_1$

$s_1 e_2 L_2 T_2$

...

$s_R e_R L_R T_R$

s e是路起点和终点

解题思路

从城市 1 开始深度优先遍历整个图，找到所有能过到达 N 的走法，选一个最优的。

解题思路

从城市 1 开始深度优先遍历整个图，找到所有能过到达 N 的走法，选一个最优的。

优化：

1) 如果当前已经找到的最优路径长度为 L ，那么在继续搜索的过程中，总长度已经大于 L 的走法，就可以直接放弃，不用走到底了

解题思路

从城市 1 开始深度优先遍历整个图，找到所有能到达 N 的走法，选一个最优的。

优化：

1) 如果当前已经找到的最优路径长度为 L ，那么在继续搜索的过程中，总长度已经大于 L 的走法，就可以直接放弃，不用走到底了

2) 用 $midL[k][m]$ 表示：走到城市 k 时总过路费为 m 的条件下，最优路径的长度。若在后续的搜索中，再次走到 k 时，如果总路费恰好为 m ，且此时的路径长度已经超过 $midL[k][m]$ ，则不必再走下去了。

```
#include <iostream>
#include <vector>
#include <cstring>
using namespace std;
int K,N,R;
struct Road {
    int d,L,t;
};
vector<vector<Road> > cityMap(110); //邻接表。cityMap[i]是从点i有路连到的城市集合
int minLen = 1 << 30; //当前找到的最优路径的长度
int totalLen; //正在走的路径的长度
int totalCost ; //正在走的路径的花销
int visited[110]; //城市是否已经走过的标记
int minL[110][10100]; //minL[i][j]表示从1到i点的，花销为j的最短路的长度
```

```
void Dfs(int s) //从 s开始向N行走
```

```
{
```

```
    if( s == N ) {
```

```
        minLen = min(minLen,totalLen);
```

```
        return ;
```

```
    }
```

```
    for( int i = 0 ;i < cityMap[s].size(); ++i ) {
```

```
        int d = cityMap[s][i].d; //s 有路连到d
```

```
        if(! visited[d] ) {
```

```
            int cost = totalCost + cityMap[s][i].t;
```

```
            if( cost > K)
```

```
                continue;
```

```
            if( totalLen + cityMap[s][i].L >= minLen
```

```
                || totalLen + cityMap[s][i].L >= minL[d][cost])
```

```
                continue;
```



```
totalLen += cityMap[s][i].L;  
totalCost += cityMap[s][i].t;  
minL[d][cost] = totalLen;  
visited[d] = 1;  
Dfs(d);  
visited[d] = 0;  
totalCost -= cityMap[s][i].t;  
totalLen -= cityMap[s][i].L;
```

```
}
```

```
}
```

```
}
```

```
int main()
{
    cin >>K >> N >> R;
    for( int i = 0;i < R; ++ i) {
        int s;
        Road r;
        cin >> s >> r.d >> r.L >> r.t;
        if( s != r.d )
            cityMap[s].push_back(r);
    }
    for( int i = 0;i < 110; ++i )
        for( int j = 0; j < 10100; ++ j )
            minL[i][j] = 1 << 30;
    memset(visited,0,sizeof(visited));
    totalLen = 0;
    totalCost = 0;
    visited[1] = 1;
```

```
minLen = 1 << 30;  
Dfs(1);  
if( minLen < (1 << 30))  
    cout << minLen << endl;  
else  
    cout << "-1" << endl;
```

```
}
```