

原创 spfa(链式前向星)+dijkstra(链式前向星)

2019-07-24 18:54:37 _Y-_Y- 阅读数 60 更多

[编辑](#)版权声明：本文为博主原创文章，遵循 [CC 4.0 BY-SA](#) 版权协议，转载请附上原文出处链接和本声明。本文链接：https://blog.csdn.net/weixin_44410512/article/details/97145550

链式前向星

链式前向星可以存图，

它存图的方式是：

将任意一个节点的所有临边按输入顺序依次连接起来

然后头节点(数组)存的是最后一个临边的地址

```
1 | int head[maxn]; // head[i] 中 i 是 u->v 中的 u, head[i] 存的是这个头节点对应的最后临边的地址
2 | int cnt // cnt 是 edge[cnt] 中 edge 的地址
3 | struct node{
4 |     int w; // u->v 中的边权
5 |     int e; // u->v 中的 v
6 |     int next; // 就是用 next 让这个头节点下面的全部临边相连
7 | } edge[maxn];
```

```
1 | void add(int u, int v, int w){
2 |     edge[cnt].w=w;
3 |     edge[cnt].e=v;
4 |     edge[cnt].next=head[u]; // 就是这一步让这个头节点下面的全部临边相连
5 |     head[u]=cnt++;
6 | }
```

```
1 | #include<bits/stdc++.h>
2 | using namespace std;
3 | #define MAXN 100501
4 | struct NODE{
5 |     int w;
6 |     int e;
7 |     int next;
8 | } edge[MAXN];
9 | int cnt;
10 | int head[MAXN];
11 | void add(int u, int v, int w){
12 |     edge[cnt].w=w;
13 |     edge[cnt].e=v;
14 |     edge[cnt].next=head[u];
15 |     head[u]=cnt++;
16 | }
17 | int main(){
18 |     memset(head, 0, sizeof(head));
19 |     cnt=1;
20 |     int n;
21 |     cin>>n;
22 |     int a, b, c;
23 |     while(n--){
24 |         cin>>a>>b>>c;
25 |         add(a, b, c);
26 |     }
27 |     int start;
28 |     cin>>start;
29 |     for(int i=head[start]; i!=0; i=edge[i].next)
30 |         cout<<start<<"->"<<edge[i].e<<" " <<edge[i].w<<endl;
31 |     return 0;
32 | }
```

深度理解链式前向星 <https://blog.csdn.net/acdreamers/article/details/16902023>

spfa

编程语言大PK，你选谁？

关闭

我理解spfa是在图上跑的可回头的bfs

```

1  #include<stdio.h>
2  #include<string.h>
3  #include<algorithm>
4  #include<map>
5  #include<queue>
6  #include<math.h>
7  #include<vector>
8  #include<iostream>
9  #define INF 0x3f3f3f3f
10 #define ll long long
11 #define N 100000+10
12 using namespace std;
13 int n,m;
14 int x,y,z;
15 struct node
16 {
17     int y,z;
18 };
19 vector<node> mp[1000];
20 int spfa(int b,int e)
21 {
22     bool color[1000];
23     int d[1000];
24     memset(color,0,sizeof(color));
25     memset(d,INF,sizeof(d));
26     d[b]=0;
27     queue<int>q;
28     q.push(b);
29     color[b]=1;
30     while(!q.empty())
31     {
32         int st=q.front();
33         q.pop();
34         color[st]=0;//这里就是和bfs的唯一区别，bfs没有这里，所以color表示的就是这个点有没有进过，进过就不用进了
35         //spfa里color表示的是队列里有没有st，要是有的话就不用进了
36         for(int i=0;i<mp[st].size();i++)
37         {
38             if(d[st]+mp[st][i].z<d[mp[st][i].y])
39             {
40                 d[mp[st][i].y]=d[st]+mp[st][i].z;
41                 if(!color[mp[st][i].y])
42                 {
43                     q.push(mp[st][i].y);
44                     color[mp[st][i].y]=1;
45                 }
46             }
47         }
48     }
49     return d[e];
50 }
51 int main()
52 {
53     scanf("%d%d",&m,&n);
54     for(int i=1;i<=m;i++)
55     {
56         scanf("%d%d%d",&x,&y,&z);
57         mp[x].push_back((node){y,z});
58         mp[y].push_back((node){x,z});
59     }
60     cout<<spfa(1,n)<<endl;
61 }
62

```

SPFA详解 <https://blog.csdn.net/hlg1995/article/details/70242296>

spfa (链式前向星)

编程语言大PK，你选谁？

关闭

```
1 #include<stdio.h>
2 #include<string.h>
3 #include<algorithm>
4 #include<map>
5 #include<queue>
6 #include<math.h>
7 #include<vector>
8 #include<iostream>
9 using namespace std;
10 #define INF 0x3f3f3f
11 #define maxn 10010
12 struct node{
13     int w;
14     int e;
15     int next;
16 }edge[maxn];
17 int cnt,t,n;
18 int head[maxn];
19 void init(){
20     memset(head,0,sizeof head);
21     cnt=1;
22 }
23 void add(int u,int v,int w){
24     edge[cnt].w=w;
25     edge[cnt].e=v;
26     edge[cnt].next=head[u];
27     head[u]=cnt++;
28 }
29 int spfa(){
30     queue<int> q;
31     bool color[maxn];
32     int d[maxn];
33     memset(d,INF,sizeof d);
34     memset(color,true,sizeof color);
35     q.push(1);
36     d[1]=0;
37     color[1]=false;
38     while(!q.empty()){
39         int st=q.front();
40         q.pop();
41         color[st]=true;
42         for(int i=head[st];i!=0;i=edge[i].next){
43             if(d[st]+edge[i].w<d[edge[i].e]){
44                 d[edge[i].e]=d[st]+edge[i].w;
45                 if(color[edge[i].e]){
46                     q.push(edge[i].e);
47                     color[edge[i].e]=false;
48                 }
49             }
50         }
51     }
52     return d[n];
53 }
54 int main(){
55     while(~scanf("%d %d", &t, &n)){
56         init();
57         int u,v,w;
58         for(int i=0;i<t;i++){
59             scanf("%d %d %d", &u, &v, &w);
60             add(u,v,w);
61             add(v,u,w);
62         }
63         printf("%d\n", spfa());
64     }
65     return 0;
66 }
67
```

dijkstra

```

1  #include <iostream>
2  #include <algorithm>
3  #include <string.h>
4  #include <queue>
5  #define INF 0x3f3f3f
6  #define maxn 1005
7  using namespace std;
8  int n;
9  vector< pair<int,int> > mp[maxn];
10 int dijkstra(int b,int e){
11     priority_queue< pair<int,int> > p; //有限队列存最小节点（默认优先级较大）
12     int d[maxn]; //用于记录起点s到v的最短路
13     memset(d,INF,sizeof(d));
14     d[b]=0;
15     p.push(make_pair(0,b)); //起点
16     while(!p.empty()){
17         pair<int,int> f = p.top();
18         p.pop();
19         int u=f.second;
20         if(d[u] < f.first*(-1)) continue;
21         for(int j=0; j<mp[u].size(); j++){
22             int v=mp[u][j].first;
23             if(d[v]>d[u]+mp[u][j].second){
24                 d[v]=d[u]+mp[u][j].second;
25                 p.push(make_pair(d[v]*(-1),v)); // priority_queue（默认优先级较大）所以要*-1;
26             }
27         }
28     }
29     return d[e];
30 }
31 int main(){
32     cin>>n;
33     int k,c,u,v;
34     for(int i=0;i<n;i++){
35         cin>>u>>k;
36         for(int j=0;j<k;j++){
37             cin>>v>>c;
38             mp[u].push_back(make_pair(v,c));
39         }
40     }
41     printf("%d\n", dijkstra(1,n));
42     return 0;
43 }
44

```

最短路径问题—Dijkstra算法详解 https://blog.csdn.net/qq_35644234/article/details/60870719

dijkstra(链式前向星)

```

1  #include <iostream>
2  #include <algorithm>
3  #include <cstring>
4  #include <cstdio>
5  #include <queue>
6  #define INF 0x3f3f3f
7  #define maxn 10010
8  using namespace std;
9  struct Time{
10     int w, e;
11     bool operator < (const Time& t)const{
12         return w > t.w;
13     }
14 };
15 struct node{
16     int w;
17     int e;
18     int next;
19 }edge[maxn];

```

```
20 int cnt,t,n;
21 int head[maxn];
22 void init(){
23     memset(head,0,sizeof head);
24     cnt=1;
25 }
26 void add(int u,int v,int w){
27     edge[cnt].w=w;
28     edge[cnt].e=v;
29     edge[cnt].next=head[u];
30     head[u]=cnt++;
31 }
32 int dijkstra(){
33     priority_queue<Time> q;
34     int d[maxn];
35     memset(d,INF,sizeof d);
36     d[1]=0;
37     q.push(Time{0,1});
38     while(!q.empty()){
39         Time st=q.top();
40         q.pop();
41         if(d[st.e]<st.w) continue;
42         for(int i=head[st.e];i!=0;i=edge[i].next){
43             if(d[st.e]+edge[i].w<d[edge[i].e]){
44                 d[edge[i].e]=d[st.e]+edge[i].w;
45                 q.push(Time{d[edge[i].e],edge[i].e});
46             }
47         }
48     }
49     return d[n];
50 }
51 int main(){
52     while(~scanf("%d %d", &t, &n)){
53         init();
54         int u,v,w;
55         for(int i=0;i<t;i++){
56             scanf("%d %d %d", &u, &v, &w);
57             add(u,v,w);
58             add(v,u,w);
59         }
60         printf("%d\n", dijkstra());
61     }
62     return 0;
63 }
64
```

有 0 个人打赏

文章最后发布于: 2019-07-24 18:54:37

©2019 CSDN 皮肤主题: 大白 设计师: CSDN官方博客

编程语言大PK，你选谁？

关闭