2012 Chengdu Regional Online B Control Solution

write by Gestalti Lur
2012-09-18
題目鏈接：http://acm.hdu.edu.cn/showproblem.php?pid=4289

題目大意

給出一個 N(N<=200)個點和 M(M<=20000)條邊的無向無權圖計算從 s 到 t 點權最小的割點集合的權值。

算法分析

對於原圖上的每個點拆成兩個點 i,i'，從 i 到 i'連一條流量爲該點點權的邊，對於原圖的每條邊(u,v),在流網絡上加入(u+N,v)和(v+N,u)兩條邊，流量爲正無窮，原點和匯點分別爲 s 和 t',計算最大流。取最大流值和 s,t 點權值的最小者即爲答案。

參考代碼

```
/*
  2012 Chengdu Online B
  2012-09-18
  gestapolur
  ACCEPTED
*/
#include<cstdio>
#include<cstring>
#include<iostream>
#define MAXN 413
#define MAXE 450000
#define INF 100000000

int n , m;
int tn , tm , vs , vt , val_s , val_t;
int u[ MAXE << 1 ] , v[ MAXE << 1 ];
int head[ MAXN ] , next[ MAXE << 1 ];

int res[ MAXE << 1 ];
int h[ MAXN ];
int stk[ MAXN ];
short int d[ MAXN ] , vh[ MAXN ];
int di[ MAXN ];
int flow;

void dfs( int source , int sink )
{
  bool flag;//標記是否有路徑被增廣
```

```
int i , j , edg , cnt , tmp , rec = 0 , aug , mint;
vh[ 0 ] = n;
flow = 0;//初始化流量
aug = INF;
cnt = 0;
i = source;
memcpy( di , head, sizeof( di ) );
while( d[ source ] < n )
 {
  h[ i ] = aug;
  flag = false;
  for( edg = di[ i ] ; edg ; edg = next[ edg ] )
  {
   j = v[ edg ];
   if( res[ edg ] and d[ j ] + 1 == d[ i ] )
     {
      flag = true;
      di[ i ] = edg;
      aug = res[ edg ] < aug ? res[ edg ] : aug;
      stk[ ++ cnt ] = edg;
      i = j;
      if( i == sink )
      {
       flow += aug;
       while( cnt )
         {
          edg = stk[ cnt -- ];
          res[ edg ] -= aug;
          res[ ( edg - 1 ^ 1 ) + 1 ] += aug;
         }
       aug = INF;
       i = source;
      }
      break;
     }
  }
  if( flag ) continue;
  //在沒有通路的情況下，找有剩餘流量的標號最小的點，記錄到這個點的邊的標號，然後從這個
  mint = n - 1;
  for( edg = head[ i ] ; edg ; edg = next[ edg ] )
  if( res[ edg ] and d[ v[ edg ] ] < mint )
    { rec = edg ; mint = d[ v[ edg ] ]; }
  di[ i ] = rec;
  -- vh[ d[ i ] ];
  //如果調整過後某一個標號的數量爲 0，那麼就沒有增廣路了。
  if( not vh[ d[ i ] ] )
  break;
  d[ i ] = mint + 1;
```

```
    ++ vh[ d[ i ] ];
    if( i not_eq source )
     aug = h[ i = u[ stk[ cnt -- ] ] ];
   }
 return ;
}

inline void add_edge( int i , short int u1 , short int v1 , int c )
{
 u[ i ] = u1; v[ i ] = v1;
 res[ i ] = c;
 next[ i ] = head[ u1 ];
 head[ u1 ] = i;
 return ;
}

bool init()
{
 if( scanf("%d%d" , &tn , &tm ) not_eq EOF )
   {
    int i , c , u , v;
    m = 0;
    n = 2 * tn;
    scanf("%d%d" , &vs , &vt );
    for( i = 1 ; i <= tn ; ++ i )
    {
      scanf("%d" , &c );
      add_edge( ++ m , i , i + tn , c );
      add_edge( ++ m , i + tn , i , 0 );
      if( i == vs ) val_s = c;
      if( i == vt ) val_t = c;
    }
    vt = vt + tn;
    for( i = 1 ; i <= tm ; ++ i )
    {
      scanf( "%d%d" , &u , &v );
      add_edge( ++ m , u , v + tn , 0 );
      add_edge( ++ m , v + tn , u , INF );

      add_edge( ++ m , v , u + tn , 0 );
      add_edge( ++ m , u + tn , v , INF );

    }
    return true;
   }
 return false;
}
```

```
int main()
{
  while ( init() )
    {
     dfs( vs , vt );
     printf( "%d\n" , std::min( flow , std::min( val_s , val_t ) ) );

     memset( head , 0 , sizeof( int ) * ( n + 1 ) );
     memset( vh , 0 , sizeof( short int ) * ( n + 1 ) );
     memset( d , 0 , sizeof( short int ) * ( n + 1 ) );
     memset( h , 0 , sizeof( int ) * ( n + 1 ) );
    }
  return 0;
}
```