算法设计与分析作业五

作者：吴润泽　　**学号：**181860109

Email：181860109@smail.nju.edu.cn

2020 年 5 月 7 日

目录

Chapter 13

problem 13.1

1)

```
FloydNext(G):
  D(0):=W /*初始情况下，两点间的最短路径长度就是它们的边权*/
  for(i:=1 to n):
    for(j:=1 to n):
      if map[i][j]!=Inf: /*如果ij间有边连接，则i的后继即为j*/
        GO[i][j]=j
  for(k:=1 to n):
    for(i:=1 to n):
      for(j:=1 to n):
        if D[i][j]>D[i][k]+D[k][j]:
          GO[i][j]=k /*如果i到j通过k的路径更短，则更新路由表*/
          D[i][j]=D[i][k]+D[k][j]
```

FloydNext.func

2) 算法更新路径 $i \to k \to j$，所选择的 j 的前驱与从选择的节点 k 到节点 j 的一条中间节点取自集合 $\{1, 2, \cdots, k-1\}$ 的最短路径上的前驱是一样的。

```
FloydPrev(G):
  D(0):=W
  for(i:=1 to n):
    for(j:=1 to n):
      if map[i][j]!=Inf: /*如果ij间有边连接，则j的前驱即为i*/
        GO[i][j]:=i
  for(k:=1 to n):
    for(i:=1 to n):
      for(j:=1 to n):
        if D[i][j]>D[i][k]+D[k][j]:
          GO[i][j]:=GO[k][j] /*如果i到j通过k的路径更短，则前缀为
    Back[k][j]*/
          D[i][j]:=D[i][k]+D[k][j]
```

FloydPrev.func

problem 13.2

1) 仿照 Dijkstra 算法，记录每条路径的最小值即可。

```
DijkstraCap(G):
  initialize the priority queue Fringe as empty
  insert some node v to Fringe
  cap:=[] /*cap 原来存放s到达其他点的吞吐量*/
  while Fringe!=empty:
    v:=Fringe.EXTRACT-MIN()
    cap[v]:=v.priority /*存放到达v的结果*/
    UPDATE-FRINGE(v,Fringe)
  return cap
UPDATE-FRINGE(v,Fringe):
  for neighbor w of v:
    w.priority:=min(vw.weight,v.priority)
    /*取路径的最小值，即当前路径已知吞吐量*/
    if w is UNSEEN: /*如果是第一次遍历到*/
      Fringe.INSERT(w,w.priority) /*将w加入队列*/
    else:
      Fringe.decrease(w,w.priority) /*将w的权值更新为w.priority*/
```

DijkstraCap.func

2) 仿照 Floyd 算法，对路径 $i \to k \to j$ 从 ij，ik，kj 中选取最小权边即可。

```
FloydCap(G):
D(0):=W /*开始的吞吐量记为相连边的边权值*/
for(k:=1 to n):
  for(i:=1 to n):
    for(j:=1 to n):
      D[i][j]:=min(D[i][j],D[i][k],D[k][j]) /*选取3者的最小值，作为最大吞吐量*/
```

FloydCap.func

problem 13.7

计算出所有点到达 v_0 的最短路径和 v_0 到达其他顶点的最短路径。则 uv 间的最短路径即为 u 到 v_0 的最短路径加上 v 到 v_0 的最短路径。

```
ShortByV(G,v):
  D:=W /*初始化所有最短路径为起始结点权值*/
  for(k:=1 to n): /*计算所有节点到达v的最短路径*/
    for(i:=1 to n):
      D[i][v]:=min(D[i][v],D[i][k]+D[k][v])
  for(k:=1 to n): /*计算v到达所有节点的最短路径*/
    for(i:=1 to n):
      D[v][i]:=min(D[v][i],D[v][k]+D[k][i])
  for(i:=1 to n):
    for(j:=1 to n):
      D[i][j]:=D[i][v]+D[v][j] /*最短路径为i到v加v到j的最短路径*/
```

ShortByV.func

Chapter 14

problem 14.2

problem 14.3

problem 14.6

problem 14.7

problem 14.11

problem 14.13

problem 14.14