18124045 潘勉之

Problem Set 10

10

Problem 1. Algorithm: Sort array P and S. Assign the smallest shoes to the person with the smallest feet and do this recursively on the remaining shoes and people.

Proof: 设P、S排序后如下（升序）：

$P: \{P_1, P_2, \cdots, P_n\}$ $S: \{S_1, S_2, \cdots, S_n\}$ $\Rightarrow$ The average difference by above algorithm is $(1/n)\sum_{i=1}^{n}|P_i - S_i|$ ------ ①

假设above algorithm错误，不妨设(P_i, S_j)，(P_j, S_i)配对，$i<j$。其余不变。

有 $P_i \le P_j$，$S_i \le S_j$ $\Rightarrow$ average difference为 $\frac{1}{n}[\sum_{k=1, k\neq i, k\neq j}^{n}|P_k - S_k| + |P_i - S_j| + |P_j - S_i|]$ ---②

② − ① $= \frac{1}{n}[|P_i - S_j| + |P_j - S_i| - (|P_i - S_i| + |P_j - S_j|)]$ ----- ③

有如下6种情况：C_4^2

1° $P_i \le P_j \le S_i \le S_j$ $\Rightarrow$ ③ = 0

2° $S_i \le S_j \le P_i \le P_j$ $\Rightarrow$ ③ = 0

3° $P_i \le S_i \le P_j \le S_j$ $\Rightarrow$ ③ = $(2P_j - 2S_i)/n \ge 0$

4° $P_i \le S_i \le S_j \le P_j$ $\Rightarrow$ ③ = $(2S_j - 2S_i)/n \ge 0$

5° $S_i \le P_i \le P_j \le S_j$ $\Rightarrow$ ③ = $(2P_j - 2P_i)/n \ge 0$

6° $S_i \le P_i \le S_j \le P_j$ $\Rightarrow$ ③ = $(2S_j - 2P_i)/n \ge 0$

$\Rightarrow$ ③ ≥ 0 恒成立

$\Rightarrow$ 其他算法只会使average difference变得更大

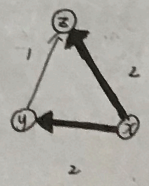
$\Rightarrow$ above algorithm is correct

B.0

Problem 2: (a) MST:

10

shortest path (from node x):



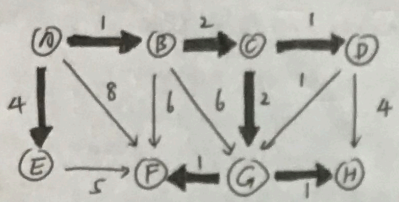
(b) 将"遍历所有edges并update"记作一个整体，循这个for loop我们不break，按照后来的Bellman-Ford alg，这个for loop要执行 $|V|-1$ 次。

只要记录每次执行这个for loop时更新的dist个数，如果发现某次for loop更新的dist个数为0，则terminate.

(c)

A	B	C	D	E	F	G	H
0	∞	∞	∞	∞	∞	∞	∞
0	1	∞	∞	4	8	∞	∞
0	1	3	∞	4	7	7	∞
0	1	3	4	4	7	5	∞
0	1	3	4	4	7	5	∞
0	1	3	4	4	7	5	8
0	1	3	4	4	6	5	6

(II)



Problem 3: (a) Go through every edge $e \in E$ and remove it if $l_e > L$ (Given L), then just do BFS from s to see if we can reach t.

Correctness: 1° ~~若为，如果 route from s to t 存在。假设全体 length no more than L 的 edge 集合为 E'，则上述路径的边集 $R \subseteq E'$ ⇒ 在 E' 上做 BFS 是可行的~~

1° 设去掉 length 大于 L 的边后，剩下的边集为 E'，显然 E' 中所有边的 length 均不大于 L ⇒ 在新图上做 BFS 是 safe 的.

2° 如图果新图中存在由 $S \to t$ 的 route，由 BFS 的性质，从 S 开始的 BFS 必能 find t ⇒ 正确. ✓

(b) Algorithm: ~~① Use the algorithm in part (a) to determine if there is a route from $s \to t$.~~

~~If not, return.~~

~~otherwise, go to step ②.~~

~~② Suppose the new graph is G', then we call Dijkstra SSSP(G', s) to find the shortest path from s to t.~~

~~③ Suppose the length of the longest edge in that path is L', set the minimum fuel tank capacity is L' as a result.~~

对 Dijkstra alg. 做修改，s.t. 最终找到的从 $S \to t$ 路径上边的最大值是所有路径中最小的。

✗ 太简洁了，如何改

```
Dijkstra(G, s):
for each u in V:
    u.maxedge = ∞ , u.parent = NIL, u.visited = false
s.maxedge = 0
Build priority queue based on maxedge
while !Q.empty():
    u = Q.remove()
    if !u.visited:  u.visited = true.
      for each edge (u,v) in E:
          if !v.visited and v.maxedge > max(u.maxedge, len(u,v)):
              v.maxedge = max(u.maxedge, len(u,v))
              v.parent = u
return t.maxedge
```

Correctness: 记从 $s \to t$ 的所有路径为 $P_1, P_2, \cdots, P_m$，路径上最长边的值为 $l_1, l_2, \cdots, l_m$，则 the minimum fuel tank capacity must be $\min\{l_1, l_2, \cdots, l_m\}$。上述伪代码的第二个 for loop 做的正是这一点，其中 v.maxedge 保存的就是 $s \to v$ 所有路径最长 edge 的最小值。

Problem 4:

Algorithm: MINDIST(G):

```
for each u in V:
    v.minidist = INF
repeat |V| times:
    for each edge (u,v) in E:
        if v.minidist > min(u.minidist + w(u,v), w(u,v)):
            v.minidist = min(u.minidist + w(u,v), w(u,v))
```

10

Correctness: 对于一条边 (u,v)，值 $\min_{key}\{dist(k,v)\}$ 必为 $w(u,v)$ 和 $(\min_{key}\{dist(k,u)\} + w(u,v))$ 中的较小者。所以只需遍历所有边（第二个 for loop）并依如上更新。由于无 negative-weight cycle 的存在，一次遍历后至少有一个 node 的 minidist 已经得到了确定更新。⇒ 执行 $|V|$ 次遍历后，所有点 node 的 minidist 均为了确值。

Problem 5:

Algorithm: 首先构造一个新图 $G'=(V', E')$, where $V'=V\cup\{k\}$, $E'=E\cup\{(t,k)$

k是新增的vertex，代表一个"空"job. 同时，for each $(u,v)\in E'$，其权重为$-w(u)$

⇒ G'的唯一source记为s，唯一sink为k.

for each u in V':

u.dist = INF, u.parent = NIL, u.earliest = INF, u.latest = INF, u.IsInCriticalPath = False

s.dist = 0, s.earliest = 0, s.latest = 0

DAGSSSP(G', s)

for each $v\in V'$:

v.dist = -v.dist // 由于之前将边权取负，需要变为正数

v.earliest = v.dist // 最早开始时间即为最短距离

~~k.latest = k.earliest~~

for each $(u,v)\in E'$: // reverse all edges

w(u,v) = -w(u,v)

(u,v) = (v,u) (reversed)

for each node u in topological order: // 注意每条边的方向已经反过来了，从新的source开始

if u == k:

u.latest = u.earliest

else:

for each edge (u,v) in E':

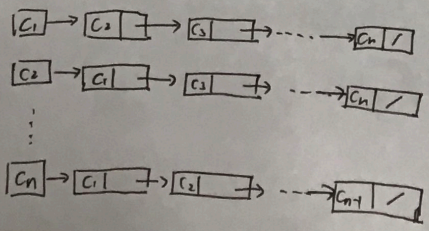
v.latest = min(v.latest, u.latest - w(u,v)) // w(u,v) is positive now

if u.earliest == u.latest:

u.IsInCriticalPath = True

Problem 6:

(a) 假设有n个currencies: $C_1, C_2, \cdots, C_n$。把每个currency视作一个vertex，于是它们之间的转换可以用如下adjacent list表示



∀一条边(C_i, C_j)表示 converting currency C_i into currency C_j，令其权重为r_{ij}.

设 $V=\{c_1, c_2, \cdots, c_n\}$

E = the set of (c_i, c_j), where $1 \le i, j \le n$.

有向 → 记图 $G=(V,E)$，则原问题转化为：在 G 上寻找一条 from vertex s to vertex t 的 path，s.t. 路径上所有边的权重积最大。(为方便起见，以下记 $c_i=s$, $c_j=t$)

设路径为 $c_i \to c_{i_1} \to c_{i_2} \to \cdots \to c_{i_m} \to c_j$

$\Leftrightarrow$ Goal: maximum $r_{ii_1} r_{i_1 i_2} \cdots r_{i_m j}$

$\Leftrightarrow$ minimum $-\log(r_{ii_1} r_{i_1 i_2} \cdots r_{i_m j}) = (-\log r_{ii_1}) + (-\log r_{i_1 i_2}) + \cdots + (-\log r_{i_m j})$

记 $w_{ij} = -\log r_{ij}$ $\Rightarrow$ for each $(c_i, c_j) \in E$, update its weight to w_{ij}.

$\Rightarrow$ 在图 G 上寻找一条 from vertex s to vertex t 的 path，s.t. 权重和最小

$\Rightarrow$ 使用 Bellman-Ford alg. 即可.

Algorithm: CONVERT:

$V=\{c_1, c_2, \cdots, c_n\}$

$E=\{(c_i, c_j)\}$, $1 \le i, j \le n$

$w(c_i, c_j) = -\log r_{ij}$

$G=(V,E)$

Bellman-Ford (G, s)

The path from s to t represents the most advantageous sequence.

(b) 仍采用 partial 的差分方法，

if there is a presence of such an anomaly, then:

$w(c_{i_1}, c_{i_2}) + w(c_{i_2}, c_{i_3}) + \cdots + w(c_{i_k}, c_{i_1})$

$= -\log r_{i_1 i_2} r_{i_2 i_3} \cdots r_{i_k i_1} < -\log 1 = 0$

$\Rightarrow$ there must be negative cycle in graph G

$\Rightarrow$ just use Bellman-Ford alg. to detect such negative cycle

Alg: DETECT:

Use Bellman-Ford alg. to detect a negative cycle.

If find it, then there is such an anomaly.