- 教材讨论
 - -TC第24章

问题1: Bellman-Ford算法

• 你能解释RELAX操作的意义吗?

```
RELAX(u, v, w)

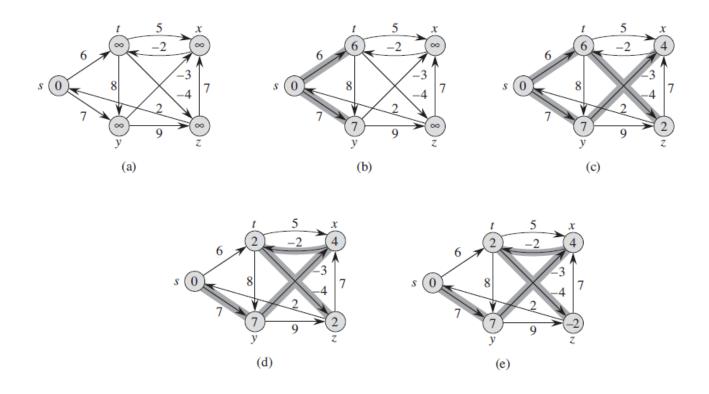
1 if v.d > u.d + w(u, v)

2 v.d = u.d + w(u, v)

3 v.\pi = u
```

问题1: Bellman-Ford算法(续)

• 你能结合这个例子说明算法执行的过程吗?



问题1: Bellman-Ford算法(续)

- 算法的循环不变量是什么?
- 或者说,第i次循环后,得到了什么重要的中间结果?

```
BELLMAN-FORD (G, w, s)

1 INITIALIZE-SINGLE-SOURCE (G, s)

2 for i = 1 to |G.V| - 1

3 for each edge (u, v) \in G.E

4 RELAX (u, v, w)

5 for each edge (u, v) \in G.E

6 if v.d > u.d + w(u, v)

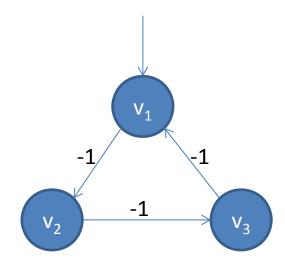
7 return FALSE

8 return TRUE
```

问题1: Bellman-Ford算法(续)

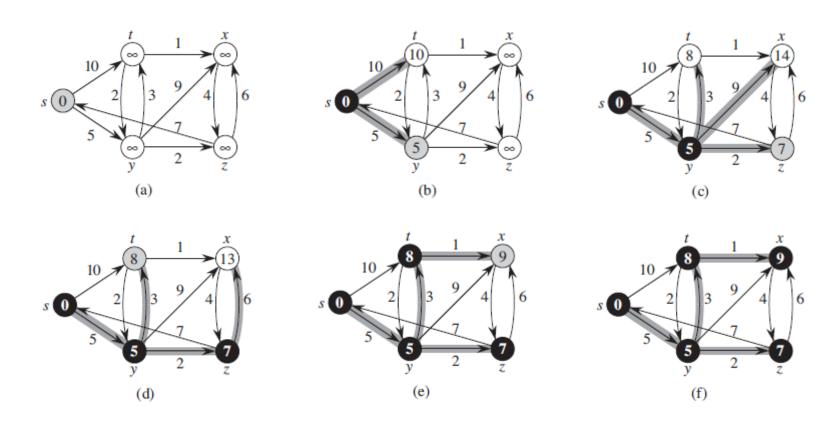
• 为什么算法一定能发现negative-weight cycles?

for each edge $(u, v) \in G.E$ if v.d > u.d + w(u, v)return FALSE



问题2: Dijkstra算法

• 你能结合这个例子说明算法执行的过程吗?



问题2: Dijkstra算法

- 算法的循环不变量是什么?
- 你能结合这个图解释算法的正确性吗?

```
DIJKSTRA(G, w, s)

1 INITIALIZE-SINGLE-SOURCE(G, s)

2 S = \emptyset

3 Q = G.V

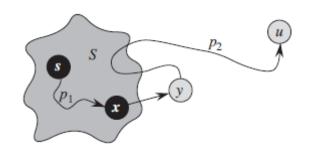
4 while Q \neq \emptyset

5 u = \text{EXTRACT-MIN}(Q)

6 S = S \cup \{u\}

7 for each vertex v \in G.Adj[u]

8 RELAX(u, v, w)
```



$$y.d = \delta(s, y)$$

$$\leq \delta(s, u)$$

$$\leq u.d$$

$$y.d = \delta(s, y) = \delta(s, u) = u.d$$

$$u.d \leq y.d$$

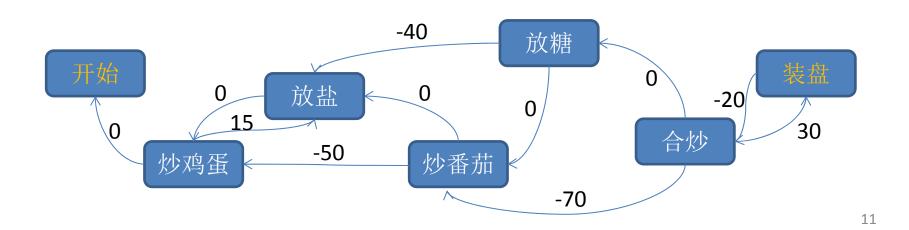
问题3: 最短路问题的应用

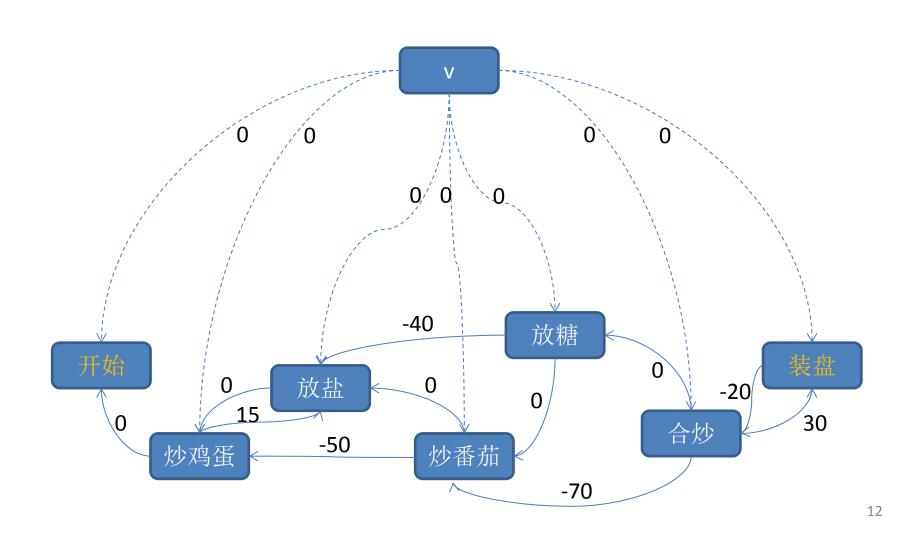
• 为什么我们很少讨论single-pair shortest-path problem?

• 你能举出一些single-source/pair shortest path(s) problem的 实际应用吗?

- 你会做番茄炒蛋吗?
 - 顺序: 先把鸡蛋炒熟,再炒番茄,再合起来炒
 - 要点:
 - 炒鸡蛋至少要50秒才能熟
 - 鸡蛋下锅的15秒内要放盐才能入味
 - 炒番茄至少要70秒才能出汁
 - 炒番茄时要放糖,但和之前放盐的时间要间隔至少40秒
 - 合起来炒至少要20秒才能混味,但不能超过30秒,否则就稀烂了
 -
- 如果一个新手要做番茄炒蛋,你能为他列出一份详细的时间表吗? (几分几秒时做什么)

- 你会做番茄炒蛋吗?
 - 顺序: 先把鸡蛋炒熟,再炒番茄,再合起来炒
 - 要点:
 - 炒鸡蛋至少要50秒才能熟
 - 鸡蛋下锅的15秒内要放盐才能入味
 - 炒番茄至少要70秒才能出汁
 - 炒番茄时要放糖,但和之前放盐的时间要间隔至少40秒
 - 合起来炒至少要20秒才能混味,但不能超过30秒,否则就稀烂了





• 怎样更新设备最合算?

	第一年	第二年	第三年	第四年	第五年
买入价	11	12	13	14	14
卖出价	4	3	2	1	0
保养费	5	6	8	11	18

