100

VG441 Problem Set 2

Pan, Chongdan ID:516370910121

June 14, 2020

1 Problem 1

Let one day be the unit time, then we get $\lambda = 500, K = 2250, h = \frac{1}{1460}$

$$c = \begin{cases} 1490 & Q < 1200 \\ 1220 & 1200 \leq Q < 2400 \\ 1100 & Q \geq 2400 \end{cases}$$

When we apply the all-units discount structure:
$Q_0^* = \sqrt{\frac{2K\lambda}{hc_0}} = \sqrt{\frac{2\times2250\times500\times1460}{1490}} \approx 1484.8$
$Q_1^* = \sqrt{\frac{2K\lambda}{hc_1}} = \sqrt{\frac{2\times2250\times500\times1460}{1220}} \approx 1640.9$
$Q_2^* = \sqrt{\frac{2K\lambda}{hc_2}} = \sqrt{\frac{2\times2250\times500\times1460}{1100}} \approx 1728.1$

Only Q_0^* and Q_1^* are realizable
$g_1(Q_1) = 500 \times 1200 + \sqrt{2 \times 2250 \times 500 \times \frac{1}{1460} \times 1200} \approx 611371.2$

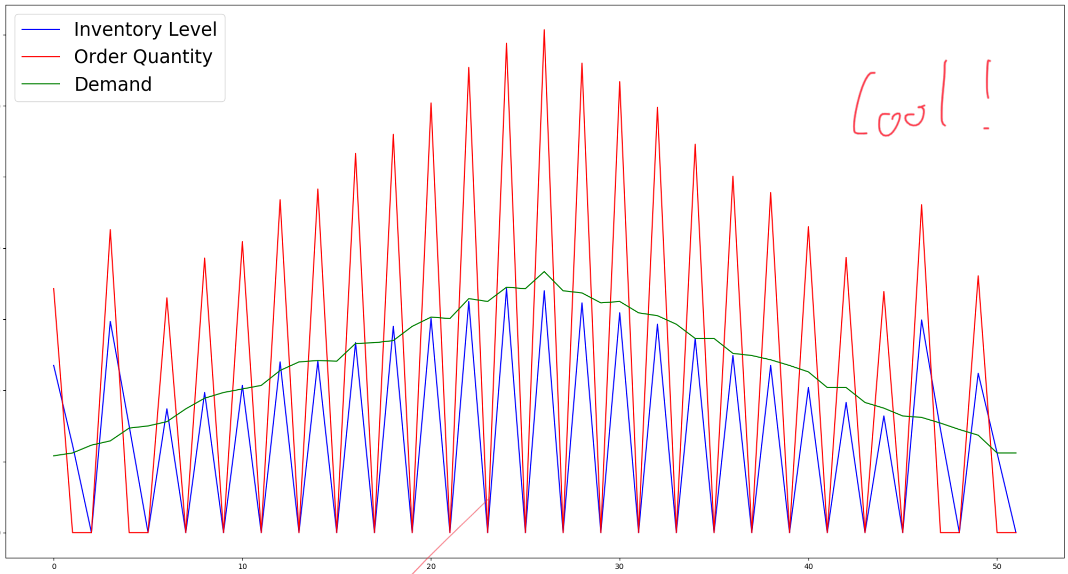
For the breakpoints
$g_1(1200) = 500 \times 1220 + \frac{2250\times500}{1200} + \frac{1220\times1200}{2\times1460} \approx 611438.9$
$g_2(2400) = 500 \times 1100 + \frac{2250\times500}{2400} + \frac{1100\times2400}{2\times1460} \approx 551372.9$
Therefore, it's the optimal order quantity is $Q = 2400$ ton, which incurs a purchase cost of 1100$ and the daily cost is 551372.9$

When we apply the incremental discount structure:
$\bar{c_1} = 1490 \times 1200 - 1220 \times 1200 = 324000$
$\bar{c_2} = 1490 \times 1200 + 1220 \times 1200 - 1100 \times 2400 = 612000$
$Q_0^* = \sqrt{\frac{2\times2250\times500\times1460}{1490}} = 1484.8$
$Q_1^* = \sqrt{\frac{2\times(2250+324000)\times500\times1460}{1220}} = 19759.3$
$Q_2^* = \sqrt{\frac{2\times(2250+612000)\times500\times1460}{1100}} = 28553.1$

Only Q_2^* are realizable
$g_2(28553.1) = 500 \times 1100 + \frac{612000}{2\times1460} + \sqrt{2 \times 500 \times (2250 + 612000) \times 1100 \times \frac{1}{1460}} = 571722.2$
Therefore, it's the optimal order quantity is $Q = 28553.1$ ton, which incurs the daily cost 571722.2$

2 Problem 2

$T = 52, K = 1100, c = 2.4$



Cool!

Figure 1: Plot of Result

According to MILP Model computed by guroby package in python, the objective $\sum_{t=1}^{T}(Ky_t + hx_t)$ is 42583$, 1 shows demand, optimal order quantity and corresponding inventory level.

3 Problem 3

Task 1

1. First, we pick the Path:$s \to b \to t$

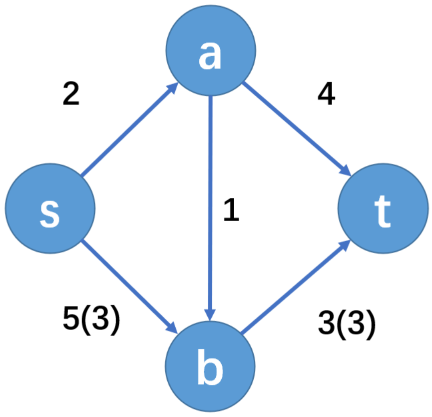


Figure 2: Flow Graph of Step 1

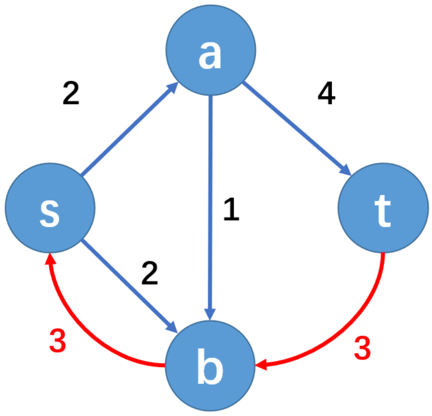


Figure 3: Residual Graph of Step 1

2. Second, we pick the Path:$s \to a \to t$

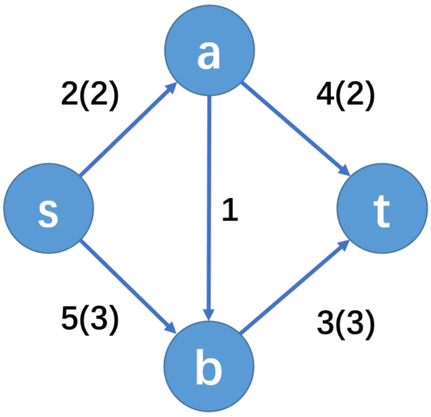


Figure 4: Flow Graph of Step 2

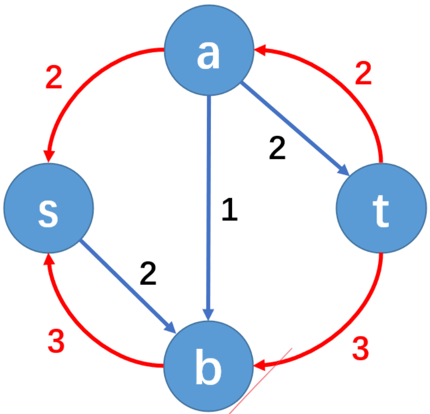


Figure 5: Residual Graph of Step 2

Since there is no more path $p \to t$, the max flow is 5

Task 2 Yes, because by looking the original graph intuitively, we can easily find the max flow is 2000, and the algorithm only needs to take two steps if it prefers $s \to a \to t$ and $s \to b \to t$

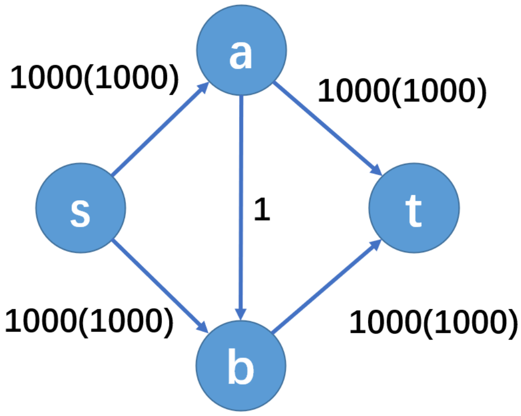


Figure 6: Intuitive Path

However, if the algorithm prefers E_{ab}, the first path will either be $s \to a \to t$ or $s \to b \to t$ with flow 1.

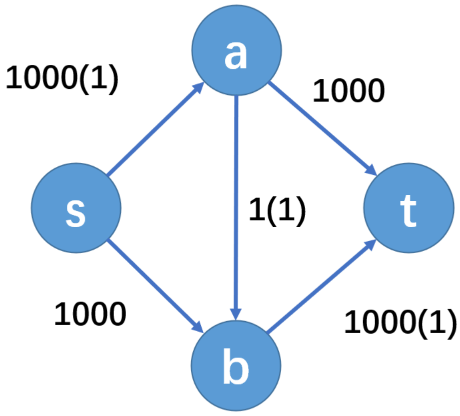


Figure 7: Flow Graph of Step 1

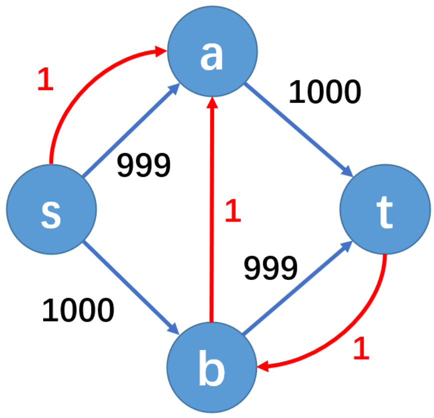


Figure 8: Residual Graph of Step 1

Then, according to the perference and residual graph, the next path will be the opposite one, but still with flow 1. Therefore, it takes 2000 steps for the algorithm to find the max flow. So the perference to *Eab* will lead to more steps for the algorithm.

Python Code

```
import numpy as np
import pandas as pd
from gurobipy import *
import matplotlib.pyplot as plt

df = pd.DataFrame(pd.read_csv(r"D:\PANDA\Study\VG441\Homework\Problem Set 2\demand.csv"))
d = df.d_t.T
T, K, h = 52, 1100, 2.4
M = 10e5
#
WW = Model()
q = WW.addVars(T, lb=np.zeros(T), vtype=GRB.CONTINUOUS, name="order_quantity")
x = WW.addVars(T, lb=np.zeros(T), vtype=GRB.CONTINUOUS, name="inventory_level")
y = WW.addVars(T, vtype=GRB.BINARY, name="if_order")

WW.setObjective(quicksum(K*y[t]+h*x[t] for t in range(T)), GRB.MINIMIZE)

c1 = WW.addConstrs(q[t] <= M*y[t] for t in range(T))
c2 = WW.addConstrs(x[t] == x[t-1] + q[t] - d[t] for t in range(1,T))
c3 = WW.addConstr(x[0] == q[0] - d[0])
WW.optimize()
# WW.printAttr('X')
print(WW.getAttr('X',q).values())

t=np.linspace(0,T-1,T)
X=[234.9999999999999, 122.99999999999835, 0.0, 297.0, 150.0, 0.0, 174.0, 0.0, 197.0, 0.0, 206.999
```

```
Q=[342.9999999999999, 0.0, 1.2505552149377763e-12, 426.0000000000004, 0.0, 0.0, 330.0, 0.0, 386.0
plt.plot(t,X,color='blue',label='Inventory Level')
plt.plot(t,Q,color='red',label='Order Quantity')
plt.plot(t,d,color='green',label='Demand')
plt.legend(loc='upper left', fontsize=10)
plt.show()
```