ECE 606, Fall 2019, Assignment 7

Zhijie Wang, Student ID number: 20856733

zhijie.wang@uwaterloo.ca

October 29, 2019

1. (a) $d_{ab}^{(i)} \to \begin{cases} R[a,b] & \text{if } i = 1 \ , \\ \max(d_{ab}^{(i-1)}, d_{a_k}^{(j)} \times d_{k_b}^{(i-j)}) & \text{if } i \geq 2 \ . \end{cases}$

   In the above recurrence, $d_{ab}^{(i)}$ represents the best payoff from a to b within i times exchanges, where i for each i from 1 to r, j for each j from 1 to i. The reason for this recurrence is that this problem can be divided into two optimal solution that best payoff from a to k within j times and best payoff from k to b within $i - j$ times, where k is any intermediate currency in the set $\{1, 2, \ldots, n\}$.

   (b) SOLUTION(a, b, r, R)

```
1: for each k from 1 to r − 1 do
2:     for each i from 1 to n do
3:         for each j from 1 to n do
4:             π[i][j] ← i
5:             d[k][i][j] ← 0
6: BestPayoff(a, b, r, R, d, π)
7: return Output(a, b, π, r, [])
```

   BESTPAYOFF(a, b, r, R, d, π)

```
1: if r = 1 then
2:     return R[a][b]
3: if d[r][a][b] > 0 then
4:     return d[r][a][b]
5: for each i from 1 to n do
6:     for each j from 1 to r − 1 do
7:         q ← BestPayoff(a, i, j, d, π) × BestPayoff(i, b, r − j, d, π)
8:         if d[r][a][b] < q then
9:             d[r][a][b] ← q, π[a][b] ← π[i][b]
10: return d[r][a][b]
```

   OUTPUT(a, b, π, r, s)

```
1: if r = 1 or π[a][b] = a then
2:     return
3: else
4:     Output(π, a, π[a][b], r − 1, s)
5:     s ← s + π[a][b]
6: return s
```

   In the above algorithm, each time we search the best payoff between a & i and i & b, where i is the intermediate currency and the sum of exchanges of these two parts from 2 to r. $d[k][i][j]$ represents the best payoff within k times exchanges between i and j, $\pi[i][j]$ represents the intermediate currency between i and j. Therefore, OUTPUT will return all the intermediate currencies between a and b.

2. a7p2.py