THIRD ASSIGNMENT

June 7, 2020

Decision Support Methods
(CC3003)

Francisco Gonçalves
201604505

Departamento de Ciência de Computadores
Faculdade de Ciências de Universide do Porto

0.1 First Exercise

In order to answer all the given questions, a simulation model was implemented using the *Python* programming language. Given the length of the program, including it in this report in its entirety is unnecessary. The following functions are the essential parts of the simulation:

RecipientPickUp( ) in *p1.py* (PYTHON)

```
def RecipientPickUp(compensation, probability):
    global dailyCostOC, dailyCostPF
    global toHome, toLocker, dailyOC, dailyPF, dailyPickUp
    global homeRemaining, lockerRemaining

    dailyLimit = 0
    dailyCostOC = 0
    dailyCostPF = 0
    dailyOC = 0
    dailyPF = 0
    dailyPickUp = 0

    for i in range(0,lockerRemaining):
        lockerPickUp = random.randrange(0,101) # 75% chance packet gets picked up
        if lockerPickUp < 75: # Packet gets picked up
            dailyPickUp += 1
            lockerRemaining -= 1
            if homeRemaining > 0:
                courier = random.randrange(0,101)
                if courier < probability: # OC accepted
                    homeRemaining -= 1
                    dailyOC += 1
                    dailyCostOC += compensation

    while homeRemaining > 0:
        if dailyLimit < 10:
            dailyLimit += 1
            dailyCostPF += 1
        elif dailyLimit >= 10:
            dailyCostPF += 2
        dailyPF += 1
        homeRemaining -= 1
```

LockerOrHome( ) in *p1.py* (PYTHON)

```
def LockerOrHome():
    global dailyPackets
    global toHome, toLocker
    global homeRemaining, lockerRemaining

    toHome = 0
    toLocker = 0
    dailyPackets = random.randrange(10,51) # Number of packets to a Locker
    for i in range(0,dailyPackets):
        destDelivery = random.randrange(1,3) # 50/50 on packet destination
        if destDelivery == 1: # Packet is to be delivered home
            toHome += 1
        elif destDelivery == 2: # Packet stays in locker
            toLocker += 1
    homeRemaining += toHome
    lockerRemaining += toLocker
```

This implementation is set to run for a sample of 10.000 observations, meaning it will simulate the whole delivery system of 120 days and print its corresponding table ten thousand times. Therefore, the following is an example of the table output for one observation, considering a compensation of 1.8€ with a probability of acceptance of 75%:

DAY	NEW PACKAGES		DELIVERIES			ACCUMULATED DELIVERIES			COSTS			LOCKER STATUS	
t	Home	Locker	PF	OC	Locker	PF	OC	Locker	PF	OC	Acc	Home	Locker
1	27	23	0	0	0	0	0	0	0	0	0	27	23
2	15	18	9	18	26	9	18	26	9	32.4	41.4	15	25
3	10	15	2	13	19	11	31	45	2	23.4	66.8	10	21
4	10	7	1	9	14	12	40	59	1	16.2	84.0	10	14
5	11	11	3	7	12	15	47	71	3	12.6	99.6	11	13
6	19	19	6	5	9	21	52	80	6	9.0	114.6	19	23
7	22	26	5	14	18	26	66	98	5	25.2	144.8	22	31
8	27	21	4	18	23	30	84	121	4	32.4	181.2	27	29
9	22	14	9	18	22	39	102	143	9	32.4	222.6	22	21
10	15	12	9	13	17	48	115	160	9	23.4	255.0	15	16
-(...)-													
110	22	16	2	7	11	506	1143	1640	2	12.6	2603.4	22	19
111	9	11	9	13	14	515	1156	1654	9	23.4	2635.8	9	16
112	23	14	0	9	12	515	1165	1666	0	16.2	2652.0	23	18
113	7	3	15	8	12	530	1173	1678	20	14.4	2686.4	7	9
114	21	18	2	5	7	532	1178	1685	2	9.0	2697.4	21	20
115	13	8	11	10	16	543	1188	1701	12	18.0	2727.4	13	12
116	11	6	6	7	8	549	1195	1709	6	12.6	2746.0	11	10
117	9	8	8	3	6	557	1198	1715	8	5.4	2759.4	9	12
118	22	27	1	8	8	558	1206	1723	1	14.4	2774.8	22	31
119	22	16	6	16	24	564	1222	1747	6	28.8	2809.6	22	23
120	8	9	8	14	16	572	1236	1763	8	25.2	2842.8	8	16

```
> Compensation: 1.8 €
> Total Cost: 2842.8 €
> Max Items: 68 @ Day 63
```

0.1.1 Expected Total Cost

The results obtained when running the program for each of the 5 possible compensation values are as follows:

0.1.1.1 0€ Compensation

```
> Compensation: 0 €
> Expected Total Cost: (2406.909281175793 , 2413.4829188242074)
```

0.1.1.2 0.5€ Compensation

```
> Compensation: 0.5 €
> Expected Total Cost: (1919.574356005821 , 1925.1169439941789)
```

0.1.1.3 1.0€ Compensation

```
> Compensation: 1 €
> Expected Total Cost: (1940.933637947567 , 1945.735162052433)
```

0.1.1.4 1.5€ Compensation

```
> Compensation: 1.5 €
> Expected Total Cost: (2407.1384078669826 , 2412.2783921330174)
```

0.1.1.5 1.8€ Compensation

```
> Compensation: 1.8 €
> Expected Total Cost: (2835.9992239611215 , 2841.8150960388725)
```

0.1.2 Expected Maximum Items

Given that the different compensation values don't influence the number of items in the Locker at the end of a day, the following values were obtained when choosing 1.8€ as compensation:

```
> Compensation: 1.8 €
> Expected Maximum Number of Items in Locker: (67.6861797323548 , 67.74502026764519)
```

0.2 Second Exercise

In order to correctly assume which compensation would benefit UPzon the most, we would only need to compare the different values obtained in the previous exercise.

Even though these calculations are never 100% precise given that it's a simulation model, the difference in the intervals of each compensation are distinct enough to form a conclusive answer.

Therefore, using a **compensation of 0.5€** would be the most lucrative method, which is perfectly justified as the incentive is high enough with a 25% probability of acceptance and it would still only cost half the price of using a professional fleet instead.

Consecutively, using a compensation higher than 1€ would essentially give no profit, as it would have high acceptance odds and would just cost more than the professional fleet.

On the other hand, the 0€ compensation is almost a non factor given its 1% acceptance rate, guaranteeing the daily limit is almost always reached, increasing the costs of using the professional fleet even more.