模擬與統計計算

HW6

N26120838 吳定洋

Homework: Write a Simulation for M/M/1 Queueing System

- Input:
  - Num. of Server (1)
  - Policy of Queue (FIFQ)
  - Customer arrival (Poisson process)
  - Customer departure
  - Service time (Exponential distribution)

- Discuss Output Can utilization go to 1? 100% busy?
  - Server utilization (lambda/mu)
  - Waiting time (expected value, distribution)

    Why plot the distribution of waiting time, but not the histogram?

1、概述

本次作業要我們模擬 M/M/1 的 queueing system，代表是 single-server 的模擬。本作業的 customer arrival time 和 service time 都是要用到 exponential distribution 的概念，因為 poisson process 也是由此而來。

2、實作

```
# All customer ------------------------------------------------------------------------------------------
    # arrive time
    this_arrival_time = -np.log(random.uniform(0,1))/lambda_possion   # possion arrival
    if not customer_arraival_time_timeline:
        customer_arraival_time_timeline.append(this_arrival_time)
    else:
        customer_arraival_time_timeline.append(customer_arraival_time_timeline[-1] + this_arrival_time)

    # service time
    this_service_time = -np.log(random.uniform(0,1))/lambda_service
    all_service_time.append(this_service_time)
    waiting_time.append(0)
```

首先我先將所有客人的到達時間點給記錄下來，每次的 arrival 都是 poisson process，我的寫法是將所以客人到達的時間放在一個時間軸上，例如在 1 分鐘

時來了一個客人，然後用 poisson process 取一個時間得到 30 秒後又來第二個，在 customer_arrival_time_timeline 上就會記錄[1 , 1.5]。

接下來 all_service_time 則紀錄該客人(index)所需要的服務時間。

```
# count wait time and leave time-----------------------------------------------------------------------------
for i in range(customers):
    if i == 0:
        customer_leave_time.append(customer_arraival_time_timeline[i] + all_service_time[i])
    else:
        if(customer_leave_time[i-1] > customer_arraival_time_timeline[i]):
            waiting_time[i] = customer_leave_time[i-1] - customer_arraival_time_timeline[i]
        else:
            waiting_time[i] = 0
        customer_leave_time.append(waiting_time[i] + customer_arraival_time_timeline[i] + all_service_time[i])
```

我需要計算前一個客人的離開時間，以便我計算後面這位客人進入排隊等待的時間等待時間有多長，只需要把前一位客人離開時間點減去現在這位客人的抵達時時間點即可，不會有負數會直接記 0 (不用等)。

這邊離開時間點也是和到達時間一樣概念用的 timeline 的方法，leave time 只需要將該客人的 wait time + service time 加上他到達的時間點即可。

3、模擬實驗

```
def mm1_queue(lambda_possion,customers,lambda_service):
```

我的主要 function 可調控參數如上。lambda_possion 調整 arrival time 的 poisson process 的 lambda、customers 代表這次模擬幾位客人、lambda_service 調整 service exponential distribution 的 lambda。

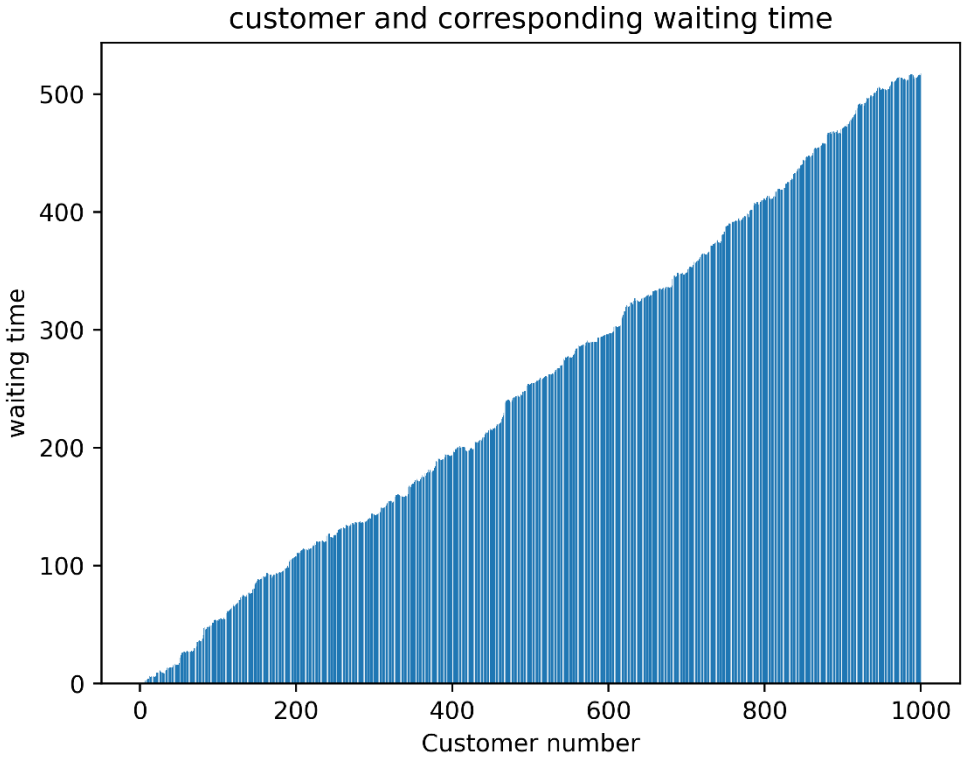
例子：

我將最小時間單位設定為 1，如果 lambda_possion = 2，代表客人平均 1/2 分鐘也就是 30 秒會來一個。lambda_service = 1，代表一個客人平均要服務 1/1 分鐘也就是一分鐘。

接下來我會分別模擬 3 個實驗，分別為

1. lambda_possion = 2 , lambda_service = 1 , customers = 1000
2. lambda_possion = 1 , lambda_service = 1 , customers = 1000
3. lambda_possion = 1 , lambda_service = 2 , customers = 1000

(1)lambda_possion = 2 , lambda_service = 1 , customers = 1000



圖表 1 客人平均 0.5 分鐘來一個 服務時間平均 1 分鐘

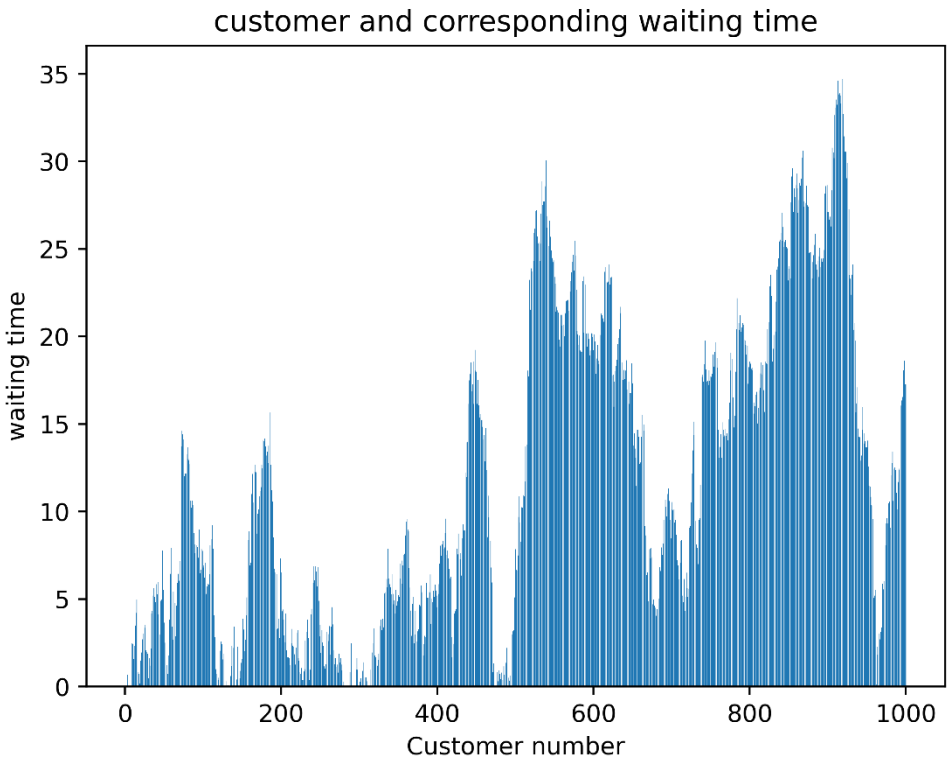
由此圖可以看出，每位客人的等待時間會越來越長，因為客人來的速度太快，而服務時間太久，導致後面來的客人前面可能還要等好多人。越後面來的人需要等越多人。由上圖可看出最後一個客人需要等將近 500 分鐘。

```
0.9964421933913044
233.55725448760253
```

圖表 2 模擬 1 utilization 與 average wait time

由此可看出 server 的利用率非常的高，但平均等待時間是 233 分鐘。

(2)lambda_possion = 1 , lambda_service = 1 , customers = 1000



圖表 3 客人平均 1 分鐘來一個 服務時間平均 1 分鐘

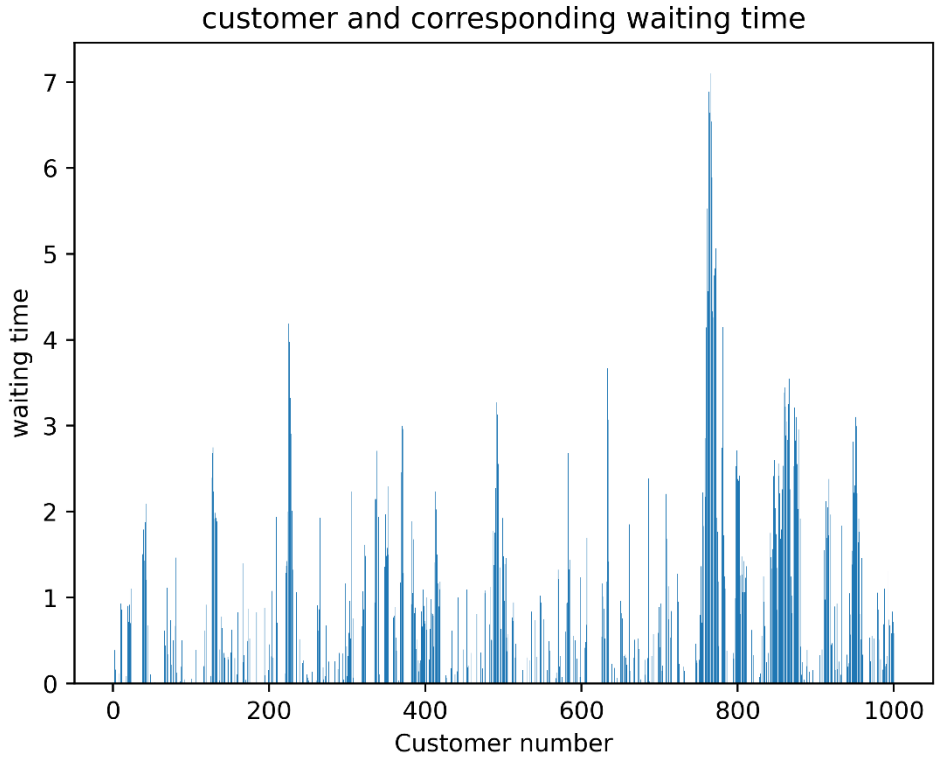
這圖就沒有前一個後面的客人會越等越久的現象了，而且最常等待時間也只有 35 分鐘左右。

```
0.9557228207392231
11.59273798934811
```

圖表 4 模擬 2 utilization 與 average wait time

Server 的 utilization 有降下來一點，只剩下 95.5%，代表有些時候客人 arrival 時 server 可能是空的。而平均的 wait time 只有 11.5 分鐘左右。

(3)lambda_possion = 1 , lambda_service = 2 , customers = 1000



圖表 5 客人平均 1 分鐘來一個 服務時間平均 0.5 分鐘

這圖的 bar 明顯更加稀疏了，代表有很多時間 server 是閒置狀態，客人來了等都不用等可以直接服務，因為員工太有效率服務太快。等最久的客人也只有等 7 分鐘左右。

```
0.5146606809340798
0.5863838730938686
```

圖表 6 模擬 2 utilization 與 average wait time

Server 的 utilization 只剩下 51%左右，而客人平均等待時間為 0.58 分鐘。

4、討論

此實驗可以看出如果 lambda_possion 比 lambda_service 大很多的話很有可能 utilization 會到無限接近 100%，反之則會導致 utilization 下降。

直方圖主要在呈現資料分布的結果，長條圖呈現的是各組資料的大小，這邊每條 bar 都獨立代表該客人的等待時間，每條 bar 之間不會呈現分布結果，所用長條圖才對。