饮水思源　　爱国荣校

随机过程大作业

Stochastic　Processes

高逸飞

18221159

知行1801班

北京交通大学

知行班

M/G/1排队论模型的Python实现

编者:高逸飞 学号：18221159
专业班级：知行班数学与应用数学专业

摘 要： 排队论 (Queuing Theory) 起源于二十世纪初，也称随机服务系统理论，顾名思义，它是研究排队相关问题的一门学科，旨在探究排队系统的平衡与优化。本文从排队论系统的一个问题M/G/1 模型入手，利用计算机模拟实现 M/G/1 过程，并探究三种数字特征的分布情况。

关键词：排队论　M/G/1模型　Python

一、排队论背景介绍

排队论发源于二十世纪初。当时美国贝尔电话公司发明了自动电话，以适应日益繁忙的工商业电话通讯需要。这个新发明带来了一个新问题，即通话线路与电话用户呼叫的数量关系应如何妥善解决，这个问题久久未能解决。1909 年，丹麦的哥本哈根电话公司工程师 Erlang 在热力学统计平衡概念的启发下解决了这个问题，并与1917年发表文章。目前，排队论已广泛应用于解决军事、运输、维修、生产、服务、等排队系统的问题，显示了其强大的生命力。

二、M/G/1排队论模型

2.1　基本概念

基本的排队论模型如下图所示：图中虚线所包含的部分为排队系统。各个顾客从顾客源出发，随机地来到服务机构，按一定的排队规则等待服务，直到按一定的服务规则接受完服务后离开排队系统。这样构成一个完整的排队过程。

对于一个服务系统来说，如果服务机构过小，以致不能满足众多顾客的需要，那么就会产生拥挤现象而使服务质量降低。因此，对于顾客而言，服务台应当是越多越好，但是，如果服务台开设过多，相应的人力和物力方面的开支也就增加，对于企业而言会造成损失，因此研究排队模型的目的就是要在顾客需要和服务机构的规模之间进行权衡决策，使其达到合理的平衡。

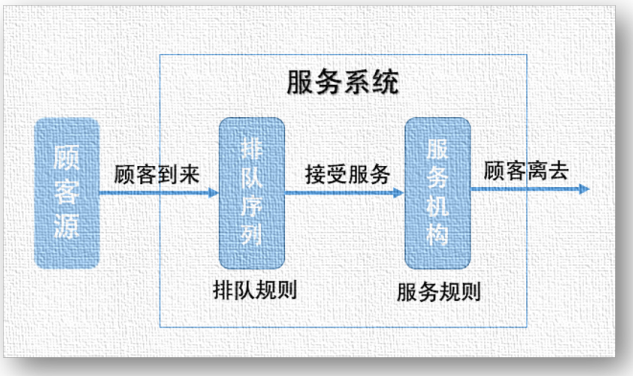

图 1: 排队系统

2.2　组成与特征

一般的排队过程都由输入过程、排队规则、服务过程三部分组成：

- 输入过程：在M/G/1排队模型中，我们考虑在有限时间内的顾客到达情况，显然，顾客的组成是有限的，且顾客到达构成一时齐的泊松序列。
- 等待规则：顾客到达排队系统时，将以等待制、损失制或者混合制三种情况进行选择，目前我们考虑等待制的情况，即顾客进入排队序列后，不会在未接受服务前离去。
- 服务过程：M/G/1系统考虑额是先到先服务的服务台服务过程。

2.3　排队系统的运行指标

为了研究排队系统运行的效率，估计其服务质量，确定系统的最优参数，评价系统的结构是否合理并研究其改进的措施，必须确定用以判断系统运行优劣的基本数量指标，这些数量指标通常是：

- 平均等待时长 S_{wait}：指顾客在系统内等待接受服务的平均时间。
- 平均逗留时间 S_{stay} 顾客在系统内逗留时间（包括排队等待的时间和接受服务的时间）的数学期望。
- 平均忙期 T_busy 指服务机构连续繁忙时间。

三、算法流程

考虑一个简单的M/G/1过程，我们认为顾客到达服务台时间为一时齐的泊松过程，

为此我们需要在规定时间T内生成一个到达时间的时间序列，为此我们通过两种算法生成该时间序列。

3.1　算法一

命题2.2.1[1]　强度为λ 的Poisson 过程$\{N_t, t \geq 0\}$ 的到达时间分布序列$\{X_n, n = 1, 2, \cdots\}$ 是独立同分布的随机变量序列，且服从 $Exp(\lambda)$，均值为 $\frac{1}{\lambda}$。

利用书中命题2.2.1，我们通过生成一列数据间隔服从指数分布的数据来模拟在$[0, T]$ 时间内的一列泊松过程，生成的泊松序列用来表示顾客到达系统的时间。下面是算法一的伪代码：

算法 1 利用到达时间间隔服从指数分布生成泊松过程

输入: 顾客到达速率 λ，时间范围 T
输出: 泊松序列 $\{X_n, n = 1, 2, \cdots\}$

```
function PoissonProcesses1(T, λ):
    result ← [t]   t ← 0   N ← 0   n ← []
    while t ≤ T do
        U ← uniform(0, 1)
        E ← − log U/λ
        t ← [t] + E
        N ← N + 1
        result ← result + [t]
    end while
    return result
end function
```

3.2　算法二

定理2.3.1[1]　强度为λ 的Poisson 过程$\{N_t, t \geq 0\}$ 在已知$\{N_t = n\}$ 的条件下，$(S_1, \cdots S_n)$ 的联合分布函数与n 个在 $[0, t]$ 上独立同均匀分布的顺序统计量的分布函数相同。

利用书中定理2.3.1，在 $[0, T]$ 时间间隔内，利用到达时刻的联合分布函数与 $[0, t]$ 上独立同均匀分布的顺序统计量相同，生成一列泊松序列，生成的泊松序列用来表示顾客到达系统的时间。下面是算法二的伪代码：

3.3　M/G/1算法

我们考虑顾客在服务台接受服务的时间服从指数分布，且每名顾客的服务时长相互独立。当顾客i 进入系统，进行第一次判断：如果排队序列不为空，则顾客进入

算法 2 在 $N_t = n$的条件下，到达时刻的联合分布函数同 $[0,t]$ 上独立同均匀分布的顺序统计量

输入： 顾客到达速率 λ，时间范围 T
输出： 泊松序列 $\{X_n, n = 1,2,\cdots\}$

```
function PoissonProcesses2(T, λ):
    result ← 0   t ← []   N ← 0   n ← 0
    N ← Poisson(λT)
    while n ≤ N do
        t ← rd.uniform(0, T)
        n ← n + 1
    end while
    results ← sort t
    return result
end function
```

排队序列中排队等待，否则进行第二次判断：如果当前服务台正忙，顾客需进入排队序列中排队等待，算法中通过变量 $Objective-Time$ 标志客观时间，即每名顾客理论上的接受服务的时间。下图表示$M/G/1$ 过程的流程图。

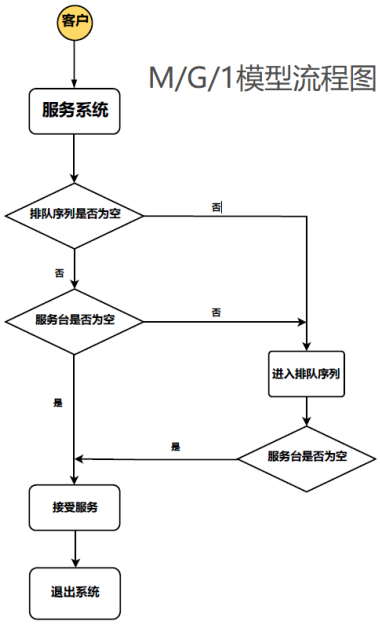

图 2: M/G/1模型流程图

四、程序运行结果

上图表示一个$M/G/1$ 排队过程，左上第一幅图表示通过算法一、算法二生成的两个泊松过程序列，在后续模拟中，我们选择 Method 1 的泊松序列，用来表示顾客的到达时间。右上的图像表示每名顾客的排队等待时间和系统逗留时间，下方的图像表示了整个排队过程，两条曲线分别表示顾客的到达时刻与开始接受服务的

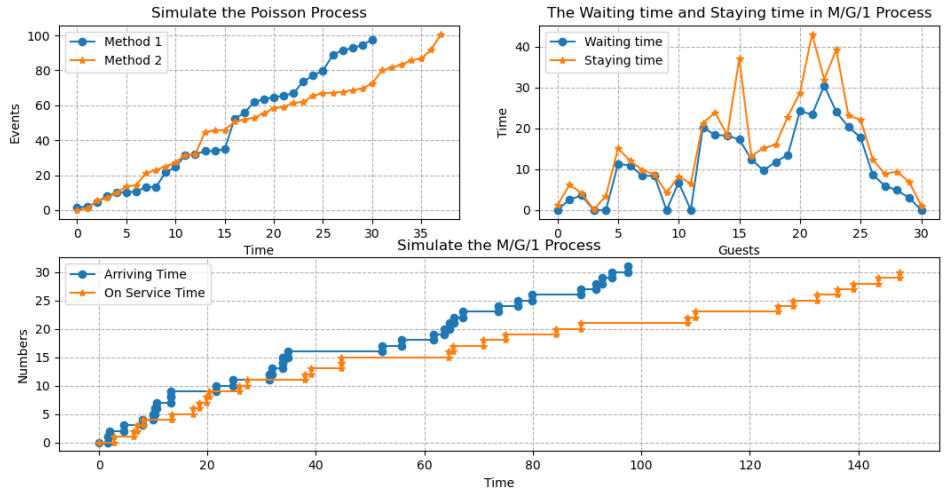

图 3: 一次排队过程的计算机模拟

时刻。可以明显观察出，系统在是定时间间隔内，产生了4个忙期[1]

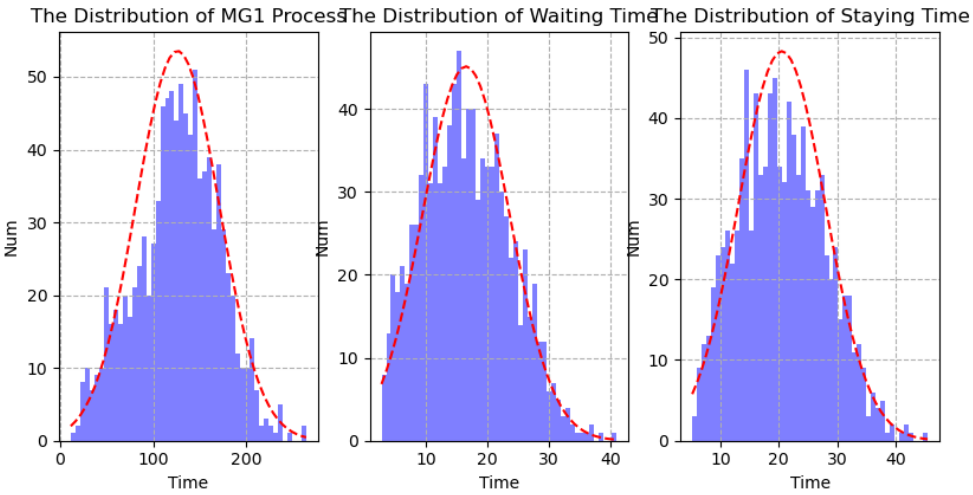

图 4: 三种数字特征的分布

上图分别表示出平均忙期，平均逗留时间，平均等待时间的分布情况，我们发现，他们都近似服从正态分布。

我们考虑服务台的多个服务高峰期，图中清楚的可以发现，系统在时刻为130 的时候，结束了第一的高峰期，在时刻为160 的时候，开始第二次服务期，并于时刻 210达到第二次高峰，程序较好的反映了系统在两次高峰期时，顾客到达以及离开的情况。

[1]该过程的输出日志见附录二

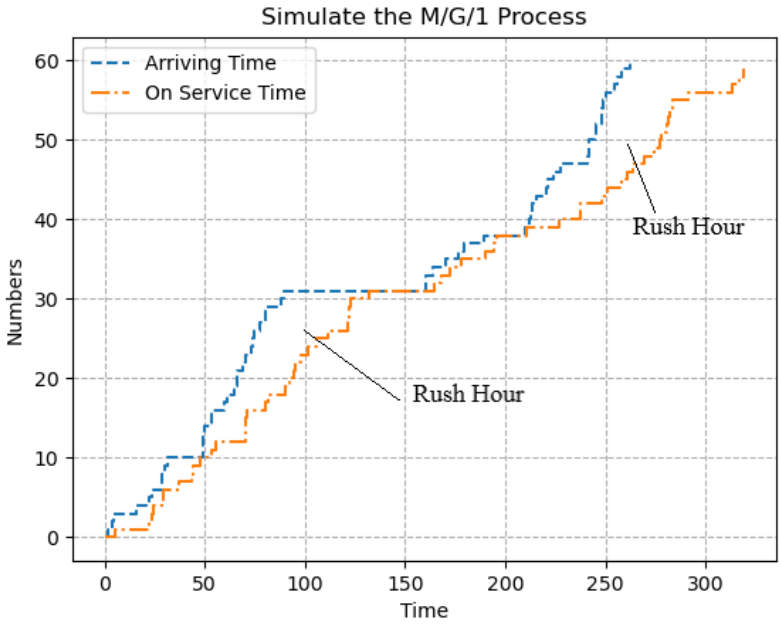

图 5: 考虑服务台的多个高峰期

五、结语

本次实验旨在模拟出排队论系统中M/G/1 过程的顾客到达情况，给出两种生成泊松序列的方式，设计出在固定时间间隔内，M/G/1过程的算法，并利用Python程序很好的实现模拟仿真，此外，实验还探究了M/G/1过程三种数字特征的分布情况，为后续的继续探索奠定了一定的基础。然而，实验之中的数据均为计算机生成的数据，在今后的改进工作当中，应该实际的考量真正的排队系统的运行情况。

参考文献

[1] SheldonM Ross. 《随机过程》[J]. 数理统计与管理, 1999.

六、附录

6.1 代码展示

```
import random as rd
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Define Function
def plotfun(data):  # plot function
  if data[0] != 0:
  data.insert(0, 0)
  n = len(data)
  N = [x for x in range(0, n) for i in range(2)]
  data = [x for x in data for i in range(2)]
  data.pop(0); del(N[-1])
  plt.plot(data,N,'-o')
  plt.title("Simulate the Poisson Processes")
  plt.legend(['Method one', 'Method two'])
  plt.xlabel("Time"); plt.ylabel("Numbers")
  plt.grid(ls = '--')
  plt.show()

def poisson_processes1(T,lam):
  t = 0; N = 0; n=[]; tt=[t]
  while t <= T:
    U = rd.uniform(0, 1)
    E = -1/lam * np.log(U)
    t = t + E
    N = N + 1
    n.append(N); tt.append(t)
    print("The", N, "events happened in", t, "s")
    # plotfun(tt)
  return tt

def poisson_processes2(T,lam):
  t = 0; N = 0; n=[];tt=[t]
  N = np.random.poisson(T*lam)
  U = np.sort([rd.uniform(0, T) for x in range(0, N)])
  for i in range(0,len(U)):
    print("The", N, "events happened in", t, "s")
    nn = [x for x in range(1,len(U)+1)]
  return U

def dict_del_first_fun(Dict):
  keyDict = list(Dict.keys())
  ValueDict = list(Dict.values())
  index = list(Dict.keys())[0]
  indextime1 = Dict[index][0]
  indextime2 = Dict[index][1]
  delname = keyDict[0]
  Dict.pop(delname)
```

```
    keyDict.pop(0); ValueDict.pop(0)
    return Dict, indextime1, index, indextime2

def plotfun(data):  # visualization
    if data[0] != 0:
        data.insert(0, 0)
    n = len(data)
    N = [x for x in range(0, n) for i in range(2)]
    data = [x for x in data for i in range(2)]
    data.pop(0); del(N[-1])
    return data,N

def show_data(ArriveTime, ServiceTime):
    ShowData_AS = pd.DataFrame({"Arriving time":ArriveTime,"service hours":ServiceTime})
    print("-------------------------------------------")
    print("Pandas showing:")
    print(ShowData_AS)
    print("-------------------------------------------")

def writeMG1(MG1TotalTime,MG1Totaltime):
    print("\n")
    print("-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*")
    print("The Busy Time is:", MG1Totaltime,"s")
    print("-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*")
    print("\n")
    MG1TotalTime.append(MG1Totaltime)
    return MG1TotalTime

def output_Service(index,i,ObjectiveTime,StartTimeService,dict_guest,Andex):
    if Andex == 0:
    print("Guest",index, "is on service from waiting line in", ObjectiveTime, "s")
    StartTimeService.append(dict_guest[i][0])
    print("Guest",index, "left in", ObjectiveTime + dict_guest[index][1], "s")
    print("-------------------------------------------")
    print("-------------------------------------------")
    if Andex == 1:
    print("Guest",index, "on service straightly in", dict_guest[index][0], "s")
    StartTimeService.append(dict_guest[index][0])
    print("Guest",index, "left in", ObjectiveTime + dict_guest[index][1], "s")
    print("-------------------------------------------")
    print("-------------------------------------------")
    return StartTimeService

def outputWS(WaitingTime,StayTime):
    for i in range(len(WaitingTime)):
    print("Guest",i, "'s waiting time is:", WaitingTime[i],"s")
    print("Guest",i, "'s staying time is:", StayTime[i],"s")
    print("-------------------------------------------")

    print("The average waiting time is:",np.mean(WaitingTime),"s")
    print("The average staying time is:",np.mean(StayTime),"s")
```

```
def MG1(dict_guest,NumGuests):
  # Define the variable
  TotalServiceTime = 0
  Timeindex = 0
  ObjectiveTime = 0
  OnLineDict = {}
  MG1TotalTime = []; MG1Totaltime = 0
  i = 0; MG1 = 1
  ObjectiveTime = dict_guest[i][0] + dict_guest[i][1]
  Timeindex = dict_guest[i][0]
  plottime = []
  StartTimeService = []
  outputindex = 0

  StartTimeService = output_Service(i,0,0,StartTimeService,dict_guest,1)

  while MG1:
    if i < NumGuests-1:
      i = i + 1
      if dict_guest[i][0] < ObjectiveTime:
        OnLineDict.update({i:dict_guest[i]})
      if outputindex == 0:
        print("The Objective Time is:", ObjectiveTime, "s")
        outputindex = 1
        print("Guest",i, "comes into watingline in", dict_guest[i][0], "s")
      else:
        if not OnLineDict:
          MG1Totaltime = ObjectiveTime - Timeindex
          plottime.append(ObjectiveTime)
          MG1TotalTime = writeMG1(MG1TotalTime,MG1Totaltime)

          # update the statement
          StartTimeService = output_Service(i,0,0,StartTimeService,dict_guest,1)
          outputindex = 0; MG1Totaltime = 0
          ObjectiveTime = dict_guest[i][0] + dict_guest[i][1]
          Timeindex = dict_guest[i][0]
          pass
        if OnLineDict:
          OnLineDict,indextime,index,indextime2 = dict_del_first_fun(OnLineDict)
          StartTimeService = output_Service(index,i,ObjectiveTime,StartTimeService,
                                            dict_guest,0)
          outputindex = 0; plottime.append(ObjectiveTime)
          ObjectiveTime = ObjectiveTime + indextime2
          i = i - 1
          pass
        pass
    if i >= NumGuests-1:
      if OnLineDict:
        OnLineDict,indextime1,index,indextime2 = dict_del_first_fun(OnLineDict)
        StartTimeService = output_Service(index,i,ObjectiveTime,StartTimeService,dict_guest,
                                          0)
        outputindex = 0; plottime.append(ObjectiveTime)
        ObjectiveTime = ObjectiveTime + indextime2
        pass
```

```
        if not OnLineDict:
          MG1Totaltime = ObjectiveTime - Timeindex
          MG1TotalTime = writeMG1(MG1TotalTime,MG1Totaltime)
          MG1 = 0
          pass
        pass
    return MG1TotalTime, StartTimeService, plottime

def Viewplot(ArriveTime,ArriveTime1,plottime,WaitingTime,StayTime):
    # M/G/1 visualization
    data1, N1 = plotfun(list(ArriveTime))
    data2, N2 = plotfun(plottime)

    plt.figure(12)
    plt.subplot(221)
    plt.plot(ArriveTime,"-o")
    plt.plot(ArriveTime1,"-*")
    plt.title("Simulate the Poisson Process")
    plt.legend(['Method 1', "Method 2"])
    plt.xlabel("Time"); plt.ylabel("Events")
    plt.grid(ls = '--')

    plt.subplot(222)
    plt.plot(WaitingTime,"-o")
    plt.plot(StayTime,"-*")
    plt.title("The Waiting time and Staying time in M/G/1 Process")
    plt.legend(['Waiting time', "Staying time"])
    plt.xlabel("Guests"); plt.ylabel("Time")
    plt.grid(ls = '--')

    plt.subplot(212)
    plt.plot(data1,N1,'-o')
    plt.plot(data2,N2,"-*")
    plt.title("Simulate the M/G/1 Process")
    plt.legend(['Arriving Time', 'On Service Time'])
    plt.xlabel("Time"); plt.ylabel("Numbers")
    plt.grid(ls = '--')

    plt.show()

def main():
    ArriveTime = poisson_processes2(T = 100, lam = 0.35)
    ArriveTime1 = poisson_processes1(T = 100, lam = 0.35)
    NumGuests = len(ArriveTime)

    # Service Time
    # ServiceTime = [3] * NumGuests
    ServiceTime = np.random.exponential(5,NumGuests)
    # ServiceTime = np.random.normal(5,1,NumGuests)

    show_data(ArriveTime, ServiceTime)
    Zip_AS = [[k,v] for k,v in zip(ArriveTime,ServiceTime)]
    guestnum = [x for x in range(NumGuests)]
```

```
    dict_guest = dict(zip(guestnum,Zip_AS))

    # MG1 Model
    MG1TotalTime, StartTimeService, plottime = MG1(dict_guest,NumGuests)
    for i in range(len(MG1TotalTime)):
      print("The",i,"Busy Time is:",MG1TotalTime[i],"s","\n")

    WaitingTime= [StartTimeService[i] - ArriveTime[i] for i in range(len(ArriveTime))]
    StayTime = [WaitingTime[i] + ServiceTime[i] for i in range(len(ServiceTime))]
    outputWS(WaitingTime,StayTime)
    Viewplot(ArriveTime,ArriveTime1,plottime,WaitingTime,StayTime)

if __name__ == "__main__":

     main()
```

6.2　输出日志

Method one

第 1 个事件在 1.6363591435423097 s发生
第 2 个事件在 1.8775840014096934 s发生
第 3 个事件在 4.496660412275666 s发生
第 4 个事件在 8.068381848971951 s发生
第 5 个事件在 9.937465493035624 s发生
第 6 个事件在 10.307470538382935 s发生
第 7 个事件在 10.66483533155428 s发生
第 8 个事件在 13.184116080414999 s发生
第 9 个事件在 13.214967913671028 s发生
第 10 个事件在 21.59389012630488 s发生
第 11 个事件在 24.70433490997551 s发生
第 12 个事件在 31.483826596494723 s发生
第 13 个事件在 31.91326106874045 s发生
第 14 个事件在 33.79612051339175 s发生
第 15 个事件在 33.91517198487759 s发生
第 16 个事件在 34.897531605720374 s发生
第 17 个事件在 52.182536329108764 s发生
第 18 个事件在 55.76510297601784 s发生
第 19 个事件在 61.73124519763963 s发生
第 20 个事件在 63.67545436662175 s发生

第 21 个事件在 64.59066547607327 s发生
第 22 个事件在 65.4890867916578 s发生
第 23 个事件在 67.06404340101047 s发生
第 24 个事件在 73.55383550386956 s发生
第 25 个事件在 77.18284744925461 s发生
第 26 个事件在 79.8174051986415 s发生
第 27 个事件在 88.87379306796501 s发生
第 28 个事件在 91.64582068550344 s发生
第 29 个事件在 92.70943447340149 s发生
第 30 个事件在 94.55213894851849 s发生
第 31 个事件在 97.5798217844719 s发生

Method two

第 1 个事件在 0.9518675819249702 s发生
第 2 个事件在 5.333404126534944 s发生
第 3 个事件在 6.812666185583941 s发生
第 4 个事件在 9.94878457892372 s发生
第 5 个事件在 13.644737321073638 s发生
第 6 个事件在 14.28203143131761 s发生
第 7 个事件在 21.253256777646534 s发生
第 8 个事件在 22.96997110249935 s发生
第 9 个事件在 24.963701836744548 s发生
第 10 个事件在 27.424161367447297 s发生
第 11 个事件在 31.600684962932007 s发生
第 12 个事件在 31.895428046837424 s发生
第 13 个事件在 44.73084532606872 s发生
第 14 个事件在 45.542902995172575 s发生
第 15 个事件在 45.844632057833806 s发生
第 16 个事件在 50.55955841655216 s发生
第 17 个事件在 51.864767052503105 s发生
第 18 个事件在 52.89125397456223 s发生
第 19 个事件在 55.54538644970516 s发生
第 20 个事件在 58.40499850383912 s发生
第 21 个事件在 59.09444172924151 s发生

第 22 个事件在 61.38485241372033 s发生
第 23 个事件在 61.91039188927772 s发生
第 24 个事件在 65.5640093670734 s发生
第 25 个事件在 66.95990556652056 s发生
第 26 个事件在 67.2121953698935 s发生
第 27 个事件在 67.68870174523671 s发生
第 28 个事件在 68.65600582441651 s发生
第 29 个事件在 69.69705657905187 s发生
第 30 个事件在 72.41417069651317 s发生
第 31 个事件在 80.04523641721126 s发生
第 32 个事件在 81.53941924647096 s发生
第 33 个事件在 83.26992765014818 s发生
第 34 个事件在 85.88755482023136 s发生
第 35 个事件在 86.72531941546684 s发生
第 36 个事件在 91.91233869701463 s发生
第 37 个事件在 100.74335318107288 s发生

Pandas 数据展示：

Order	Arriving time	service hours
0	1.636359	1.175870
1	1.877584	3.605634
2	4.496660	0.571811
3	8.068382	0.106285
4	9.937465	3.528266
5	10.307471	3.814009
6	10.664835	1.177299
7	13.184116	1.383918
8	13.214968	0.371294
9	21.593890	4.269265
10	24.704335	1.438934
11	31.483827	6.533115
12	31.913261	1.049604
13	33.796121	5.522497
14	33.915172	0.036120

15	34.897532	19.813144
16	52.182536	0.837758
17	55.765103	5.500327
18	61.731245	4.189109
19	63.675454	9.376792
20	64.590665	4.460000
21	65.489087	19.706802
22	67.064043	1.517094
23	73.553836	15.240765
24	77.182847	2.772156
25	79.817405	4.385532
26	88.873793	3.741254
27	91.645821	2.944924
28	92.709434	4.563601
29	94.552139	3.920167
30	97.579822	1.110182

顾客 0 在 1.6363591435423097 s 直接接受服务
顾客 0 在 2.812229329872033 s 离开服务台

服务台允许的客观时间为： 2.812229329872033 s
顾客 1 在 1.8775840014096934 s进入排队序列
顾客 1 在 2.812229329872033 s 从排队队列进入服务台
顾客 1 在 6.417863318741556 s 离开服务台

服务台允许的客观时间为： 6.417863318741556 s
顾客 2 在 4.496660412275666 s进入排队序列
顾客 2 在 6.417863318741556 s 从排队队列进入服务台
顾客 2 在 6.989673835145664 s 离开服务台

-*

本次的忙期长度为: 5.353314691603355 s
-*

顾客 3 在 8.068381848971951 s 直接接受服务
顾客 3 在 8.174667197680664 s 离开服务台
——
——

-*
本次的忙期长度为: 0.1062853487087132 s
-*

顾客 4 在 9.937465493035624 s 直接接受服务
顾客 4 在 13.465731014447005 s 离开服务台
——
——
服务台允许的客观时间为： 13.465731014447005 s
顾客 5 在 10.307470538382935 s进入排队序列
顾客 6 在 10.66483533155428 s进入排队序列
顾客 7 在 13.184116080414999 s进入排队序列
顾客 8 在 13.214967913671028 s进入排队序列
顾客 5 在 13.465731014447005 s 从排队队列进入服务台
顾客 5 在 17.279740175098333 s 离开服务台
——
——
顾客 6 在 17.279740175098333 s 从排队队列进入服务台
顾客 6 在 18.45703938930484 s 离开服务台
——
——
顾客 7 在 18.45703938930484 s 从排队队列进入服务台
顾客 7 在 19.84095729937075 s 离开服务台
——
——
顾客 8 在 19.84095729937075 s 从排队队列进入服务台

顾客 8 在 20.212251681641753 s 离开服务台

-*
本次的忙期长度为: 10.274786188606129 s
-*

顾客 9 在 21.59389012630488 s 直接接受服务
顾客 9 在 25.863155458512473 s 离开服务台

服务台允许的客观时间为： 25.863155458512473 s
顾客 10 在 24.70433490997551 s进入排队序列
顾客 10 在 25.863155458512473 s 从排队队列进入服务台
顾客 10 在 27.3020895833912 s 离开服务台

-*
本次的忙期长度为: 5.7081994570863195 s
-*

顾客 11 在 31.483826596494723 s 直接接受服务
顾客 11 在 38.01694153045973 s 离开服务台

服务台允许的客观时间为： 38.01694153045973 s
顾客 12 在 31.91326106874045 s进入排队序列
顾客 13 在 33.79612051339175 s进入排队序列
顾客 14 在 33.91517198487759 s进入排队序列
顾客 15 在 34.897531605720374 s进入排队序列
顾客 12 在 38.01694153045973 s 从排队队列进入服务台
顾客 12 在 39.066545756154426 s 离开服务台

顾客 13 在 39.066545756154426 s 从排队队列进入服务台
顾客 13 在 44.58904246919396 s 离开服务台

顾客 14 在 44.58904246919396 s 从排队队列进入服务台
顾客 14 在 44.625162384727396 s 离开服务台

顾客 15 在 44.625162384727396 s 从排队队列进入服务台
顾客 15 在 64.43830683796256 s 离开服务台

服务台允许的客观时间为： 64.43830683796256 s
顾客 16 在 52.182536329108764 s进入排队序列
顾客 17 在 55.76510297601784 s进入排队序列
顾客 18 在 61.73124519763963 s进入排队序列
顾客 19 在 63.67545436662175 s进入排队序列
顾客 16 在 64.43830683796256 s 从排队队列进入服务台
顾客 16 在 65.27606477399965 s 离开服务台

服务台允许的客观时间为： 65.27606477399965 s
顾客 20 在 64.59066547607327 s进入排队序列
顾客 17 在 65.27606477399965 s 从排队队列进入服务台
顾客 17 在 70.77639211659292 s 离开服务台

服务台允许的客观时间为： 70.77639211659292 s
顾客 21 在 65.4890867916578 s进入排队序列
顾客 22 在 67.06404340101047 s进入排队序列
顾客 18 在 70.77639211659292 s 从排队队列进入服务台
顾客 18 在 74.96550066276748 s 离开服务台

服务台允许的客观时间为： 74.96550066276748 s
顾客 23 在 73.55383550386956 s进入排队序列
顾客 19 在 74.96550066276748 s 从排队队列进入服务台
顾客 19 在 84.34229290383615 s 离开服务台

服务台允许的客观时间为： 84.34229290383615 s
顾客 24 在 77.18284744925461 s进入排队序列
顾客 25 在 79.8174051986415 s进入排队序列
顾客 20 在 84.34229290383615 s 从排队队列进入服务台
顾客 20 在 88.80229275476356 s 离开服务台

顾客 21 在 88.80229275476356 s 从排队队列进入服务台
顾客 21 在 108.5090950346827 s 离开服务台

服务台允许的客观时间为： 108.5090950346827 s
顾客 26 在 88.87379306796501 s进入排队序列
顾客 27 在 91.64582068550344 s进入排队序列
顾客 28 在 92.70943447340149 s进入排队序列
顾客 29 在 94.55213894851849 s进入排队序列
顾客 30 在 97.5798217844719 s进入排队序列
顾客 22 在 108.5090950346827 s 从排队队列进入服务台
顾客 22 在 110.02618879597108 s 离开服务台

顾客 23 在 110.02618879597108 s 从排队队列进入服务台
顾客 23 在 125.26695354172148 s 离开服务台

顾客 24 在 125.26695354172148 s 从排队队列进入服务台

顾客 24 在 128.03910967562607 s 离开服务台

顾客 25 在 128.03910967562607 s 从排队队列进入服务台
顾客 25 在 132.4246416491265 s 离开服务台

顾客 26 在 132.4246416491265 s 从排队队列进入服务台
顾客 26 在 136.1658959730361 s 离开服务台

顾客 27 在 136.1658959730361 s 从排队队列进入服务台
顾客 27 在 139.110819518457 s 离开服务台

顾客 28 在 139.110819518457 s 从排队队列进入服务台
顾客 28 在 143.67442085632825 s 离开服务台

顾客 29 在 143.67442085632825 s 从排队队列进入服务台
顾客 29 在 147.59458745420713 s 离开服务台

顾客 30 在 147.59458745420713 s 从排队队列进入服务台
顾客 30 在 148.7047694781835 s 离开服务台

-*
本次的忙期长度为: 117.22094288168877 s
-*

第 0 次忙期时长为: 5.353314691603355 s

第 1 次忙期时长为: 0.1062853487087132 s

第 2 次忙期时长为: 10.274786188606129 s

第 3 次忙期时长为: 5.7081994570863195 s

第 4 次忙期时长为: 117.22094288168877 s

第 0 名顾客的等待时间为: 0.0 s
第 0 名顾客的逗留时间为: 1.1758701863297232 s
————————————————-
第 1 名顾客的等待时间为: 2.619076410865973 s
第 1 名顾客的逗留时间为: 6.224710399735496 s
————————————————-
第 2 名顾客的等待时间为: 3.5717214366962846 s
第 2 名顾客的逗留时间为: 4.143531953100393 s
————————————————-
第 3 名顾客的等待时间为: 0.0 s
第 3 名顾客的逗留时间为: 0.10628534870871237 s
————————————————-
第 4 名顾客的等待时间为: 0.0 s
第 4 名顾客的逗留时间为: 3.528265521411381 s
————————————————-
第 5 名顾客的等待时间为: 11.286419587921946 s
第 5 名顾客的逗留时间为: 15.100428748573274 s
————————————————-
第 6 名顾客的等待时间为: 10.929054794750602 s
第 6 名顾客的逗留时间为: 12.106354008957108 s
————————————————-
第 7 名顾客的等待时间为: 8.409774045889883 s
第 7 名顾客的逗留时间为: 9.793691955955795 s
————————————————-
第 8 名顾客的等待时间为: 8.378922212633853 s
第 8 名顾客的逗留时间为: 8.750216594904856 s

——————————————————-
第 9 名顾客的等待时间为: 0.0 s
第 9 名顾客的逗留时间为: 4.269265332207593 s
——————————————————-
第 10 名顾客的等待时间为: 6.779491686519211 s
第 10 名顾客的逗留时间为: 8.218425811397939 s
——————————————————-
第 11 名顾客的等待时间为: 0.0 s
第 11 名顾客的逗留时间为: 6.533114933965011 s
——————————————————-
第 12 名顾客的等待时间为: 20.269275260368314 s
第 12 名顾客的逗留时间为: 21.318879486063008 s
——————————————————-
第 13 名顾客的等待时间为: 18.386415815717015 s
第 13 名顾客的逗留时间为: 23.908912528756552 s
——————————————————-
第 14 名顾客的等待时间为: 18.267364344231176 s
第 14 名顾客的逗留时间为: 18.303484259764605 s
——————————————————-
第 15 名顾客的等待时间为: 17.28500472338839 s
第 15 名顾客的逗留时间为: 37.09814917662355 s
——————————————————-
第 16 名顾客的等待时间为: 12.408129146964505 s
第 16 名顾客的逗留时间为: 13.245887083001598 s
——————————————————-
第 17 名顾客的等待时间为: 9.723983815639954 s
第 17 名顾客的逗留时间为: 15.224311158233228 s
——————————————————-
第 18 名顾客的等待时间为: 11.822590306229934 s
第 18 名顾客的逗留时间为: 16.0116988524045 s
——————————————————-
第 19 名顾客的等待时间为: 13.50739308263286 s
第 19 名顾客的逗留时间为: 22.88418532370153 s
——————————————————-

第 20 名顾客的等待时间为: 24.28312759189174 s
第 20 名顾客的逗留时间为: 28.743127442819144 s
————————————————-
第 21 名顾客的等待时间为: 23.384706276307213 s
第 21 名顾客的逗留时间为: 43.09150855622636 s
————————————————-
第 22 名顾客的等待时间为: 30.515778383461438 s
第 22 名顾客的逗留时间为: 32.03287214474981 s
————————————————-
第 23 名顾客的等待时间为: 24.02598628060234 s
第 23 名顾客的逗留时间为: 39.26675102635274 s
————————————————-
第 24 名顾客的等待时间为: 20.396974335217294 s
第 24 名顾客的逗留时间为: 23.16913046912188 s
————————————————-
第 25 名顾客的等待时间为: 17.762416585830408 s
第 25 名顾客的逗留时间为: 22.147948559330842 s
————————————————-
第 26 名顾客的等待时间为: 8.706028716506893 s
第 26 名顾客的逗留时间为: 12.447283040416503 s
————————————————-
第 27 名顾客的等待时间为: 5.934001098968466 s
第 27 名顾客的逗留时间为: 8.87892464438937 s
————————————————-
第 28 名顾客的等待时间为: 4.870387311070417 s
第 28 名顾客的逗留时间为: 9.433988648941675 s
————————————————-
第 29 名顾客的等待时间为: 3.0276828359534136 s
第 29 名顾客的逗留时间为: 6.947849433832289 s
————————————————-
第 30 名顾客的等待时间为: 0.0 s
第 30 名顾客的逗留时间为: 1.1101820239763498 s
————————————————-
平均等待时间为: 10.856506647943855 s

平均逗留时间为: 15.329523698514604 s

平均忙期为:126.18891808987726s
平均等待时间为:16.51814521081124s
平均逗留时间为:20.519577982180895s