In [ ]:

```
#
#   Example 4: Finding maximum arrival rate to ensure that mean
#   waiting time is below critical
#
#   Author: Paulius Tervydis
#   Date: 2023-07-09
#
# ============================================================
#                Single Server Queueing System (ssqs)
#                 +----------------------------+
#    Arrival      |                Server      |    Output
#                 |    Queue    +--------------+|
#  arrival_rate | ------------ | | | | | | service_rate || ------------>
#                 | ---------- | +--------------+|
#                 |            +-----------------+
#
# System notation (A/B/1):
# A - distribution of inter-arrival time
# B - distribution of service time is determined
# 1 - single server
#
# Distribution types:
# M - exponential,
# D - determined (fixed),
# G - general (time variance must be provided))
# ============================================================

from qsystems import ssqs, msqs
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt


# ------------------------------------------------------------
# System initial parameters:
# ------------------------------------------------------------
# List of queueing systems to be evaluated

# arrival_rate_total = 500 # [req/h]

system_notation = ["M01"]

# Server parameters
# service_rate = 100 # [entities/t.u.]

server_no = 10
service_time_sec = 200 # single server
service_time = service_time_sec/3600
service_rate = 1/service_time
print("service_rate",service_rate)

# Arrival parameters
arrival_rate_start = 0.0 # [entities/t.u.], for example [users/hour]
arrival_rate_stop = 0.6 * service_rate * server_no
arrival_rate_step = 0.5
# Note: For stable system arrival_rate must be < service_rate
# ------------------------------------------------------------

# ------------------------------------------------------------
# Example how to estimate dependence of system parameters on arrival time,
# Results are stored in pandas dataframe.
#
arrival_rate_array = np.arange(arrival_rate_start,
                               arrival_rate_stop+arrival_rate_step,
                               arrival_rate_step)

param_df = pd.DataFrame()
for server_no in range(1,10):
    for q_system in system_notation:
        for arrival_rate in arrival_rate_array:
            # print(q_system)
            if arrival_rate/server_no < service_rate:
                sys_params = msqs(ar=arrival_rate,sn = server_no, qs=q_system,sr1=service_rate )
                df = pd.DataFrame(sys_params,index=[0])
                param_df  = pd.concat([param_df ,df], axis=0)
param_df.reset_index(drop=True, inplace=True)

print(param_df)

# ------------------------------------------------------------
# Having such data it is possible to estimate how system parameters depend on
# arrival rate. It can be used to ensure quality of service requirements.
# As it was shown in the Example 2 - system parameters highly depend
# on arrival rate or utilization.
# For example, if it is necessary to achieve that mean waiting time is
# below critical, then the maximum arrival rate can be determined.
#
# waiting_time_critical_sec = 300
# waiting_time_critical = waiting_time_critical_sec/3600
# print(waiting_time_critical)

# # Finding what is the maximum arrival rate to ensure that mean waiting time is < critical value
# max_arrival_rate_at_critical_waiting_time = {}
# for q_system in system_notation:
#     row_found = param_df[(param_df['w']<=waiting_time_critical) & (param_df['qs']==q_system)].tail(1)
#     max_arrival_rate_at_critical_waiting_time[q_system] = float(row_found['ar'].iloc[0])

# print("Max arrival rate at critical mean waiting time",
#       max_arrival_rate_at_critical_waiting_time)


# # ------------------------------------------------------------
# # Plot Waiting times vs Arrival rate
# #
plt.figure()
for q_system in system_notation:
    for server_no in range(1,10):
        plot_data_x = param_df[(param_df["qs"]==q_system)&(param_df["sn"]==server_no)]['ar']
        plot_data_y = param_df[(param_df["qs"]==q_system)&(param_df["sn"]==server_no)]['w']
        print(plot_data_x)
        plt.plot(plot_data_x,plot_data_y,label="Servers: "+str(server_no))#+
    #             ": arrival rate (at critical mean waiting time) = %.2f"%
    #             max_arrival_rate_at_critical_waiting_time[q_system])
    # plt.axvline(max_arrival_rate_at_critical_waiting_time[q_system],
    #             color='black')
# plt.axhline(waiting_time_critical,color='red')
# plt.text(0,waiting_time_critical,'critical mean waiting time')
plt.xlabel("Arrival rate")
plt.ylabel("Mean waiting time")
plt.title("Mean waiting time vs Arrival rate")
plt.legend()
plt.grid()
plt.ylim(0,1)
plt.show()
```

```
service_rate 18.0
      qs     ar   sr        a        va        s  vs        u         l  \
0    M01    0.0  18.0      inf      inf  0.055556   0  0.000000  0.000000
1    M01    0.5  18.0      inf      inf  0.055556   0  0.000000  0.000000
2    M01    1.0  18.0      inf      inf  0.055556   0  0.000000  0.000000
3    M01    1.5  18.0      inf      inf  0.055556   0  0.000000  0.000000
4    M01    2.0  18.0      inf      inf  0.055556   0  0.000000  0.000000
...  ...    ...   ...      ...      ...       ...  ..       ...       ...
1402 M01  106.0  18.0  0.084906  0.007209  0.055556   0  0.654321  1.273589
1403 M01  106.5  18.0  0.084507  0.007141  0.055556   0  0.657407  1.288163
1404 M01  107.0  18.0  0.084112  0.007075  0.055556   0  0.660494  1.302974
1405 M01  107.5  18.0  0.083721  0.007009  0.055556   0  0.663580  1.318029
1406 M01  108.0  18.0  0.083333  0.006944  0.055556   0  0.666667  1.333333

           lq        wq         w  sn
0    0.000000  0.000000  0.055556   1
1    0.000000  0.000000  0.055556   1
2    0.000000  0.000000  0.055556   1
3    0.000000  0.000000  0.055556   1
4    0.000000  0.000000  0.055556   1
...       ...       ...       ...  ..
1402 0.619268  0.052579  0.108135   9
1403 0.630756  0.053303  0.108859   9
1404 0.642480  0.054040  0.109596   9
1405 0.654448  0.054791  0.110347   9
1406 0.666667  0.055556  0.111111   9

[1407 rows x 13 columns]
0       0.0
1       0.5
2       1.0
3       1.5
4       2.0
5       2.5
6       3.0
7       3.5
8       4.0
9       4.5
10      5.0
11      5.5
12      6.0
13      6.5
14      7.0
15      7.5
16      8.0
17      8.5
18      9.0
19      9.5
20     10.0
21     10.5
22     11.0
23     11.5
24     12.0
25     12.5
26     13.0
27     13.5
28     14.0
29     14.5
30     15.0
31     15.5
32     16.0
33     16.5
34     17.0
35     17.5
Name: ar, dtype: float64
36      0.0
37      0.5
38      1.0
39      1.5
40      2.0
       ...
103    33.5
104    34.0
105    34.5
106    35.0
107    35.5
Name: ar, Length: 72, dtype: float64
108     0.0
109     0.5
110     1.0
111     1.5
112     2.0
       ...
211    51.5
212    52.0
213    52.5
214    53.0
215    53.5
Name: ar, Length: 108, dtype: float64
216     0.0
217     0.5
218     1.0
219     1.5
220     2.0
       ...
355    69.5
356    70.0
357    70.5
358    71.0
359    71.5
Name: ar, Length: 144, dtype: float64
360     0.0
361     0.5
362     1.0
363     1.5
364     2.0
       ...
535    87.5
536    88.0
537    88.5
538    89.0
539    89.5
Name: ar, Length: 180, dtype: float64
540     0.0
541     0.5
542     1.0
543     1.5
544     2.0
       ...
751    105.5
752    106.0
753    106.5
754    107.0
755    107.5
Name: ar, Length: 216, dtype: float64
756     0.0
757     0.5
758     1.0
759     1.5
760     2.0
       ...
968    106.0
969    106.5
970    107.0
971    107.5
972    108.0
Name: ar, Length: 217, dtype: float64
973     0.0
974     0.5
975     1.0
976     1.5
977     2.0
       ...
1185   106.0
1186   106.5
1187   107.0
1188   107.5
1189   108.0
Name: ar, Length: 217, dtype: float64
1190    0.0
1191    0.5
1192    1.0
1193    1.5
1194    2.0
       ...
1402   106.0
1403   106.5
1404   107.0
1405   107.5
1406   108.0
Name: ar, Length: 217, dtype: float64
```