Συστήματα Αναμονής (Queuing Systems)

3ο Εργαστήριο
Θοδωρής Παπαρρηγόπουλος (el18040)

Ερώτηση 1)

Για λ = 1

trans	state	num_ar	arr	dep
1	0	0	1	0
2	1	0	0	1
3	0	1	1	0
4	1	0	1	0
5	2	0	0	1
6	1	1	0	1
7	0	2	1	0
8	1	1	1	0
9	2	0	0	1
10	1	2	1	0
11	2	0	0	1
12	1	3	0	1
13	0	3	1	0
14	1	3	0	1
15	0	4	1	0
16	1	3	0	1
17	0	5	1	0
18	1	3	0	1
19	0	6	1	0
20	1	3	0	1
21	0	7	1	0
22	1	3	1	0
23	2	0	0	1
24	1	4	0	1
25	0	8	1	0
26	1	4	0	1
27	0	9	1	0
28	1	4	0	1
29	0	10	1	0
30	1	4	1	0

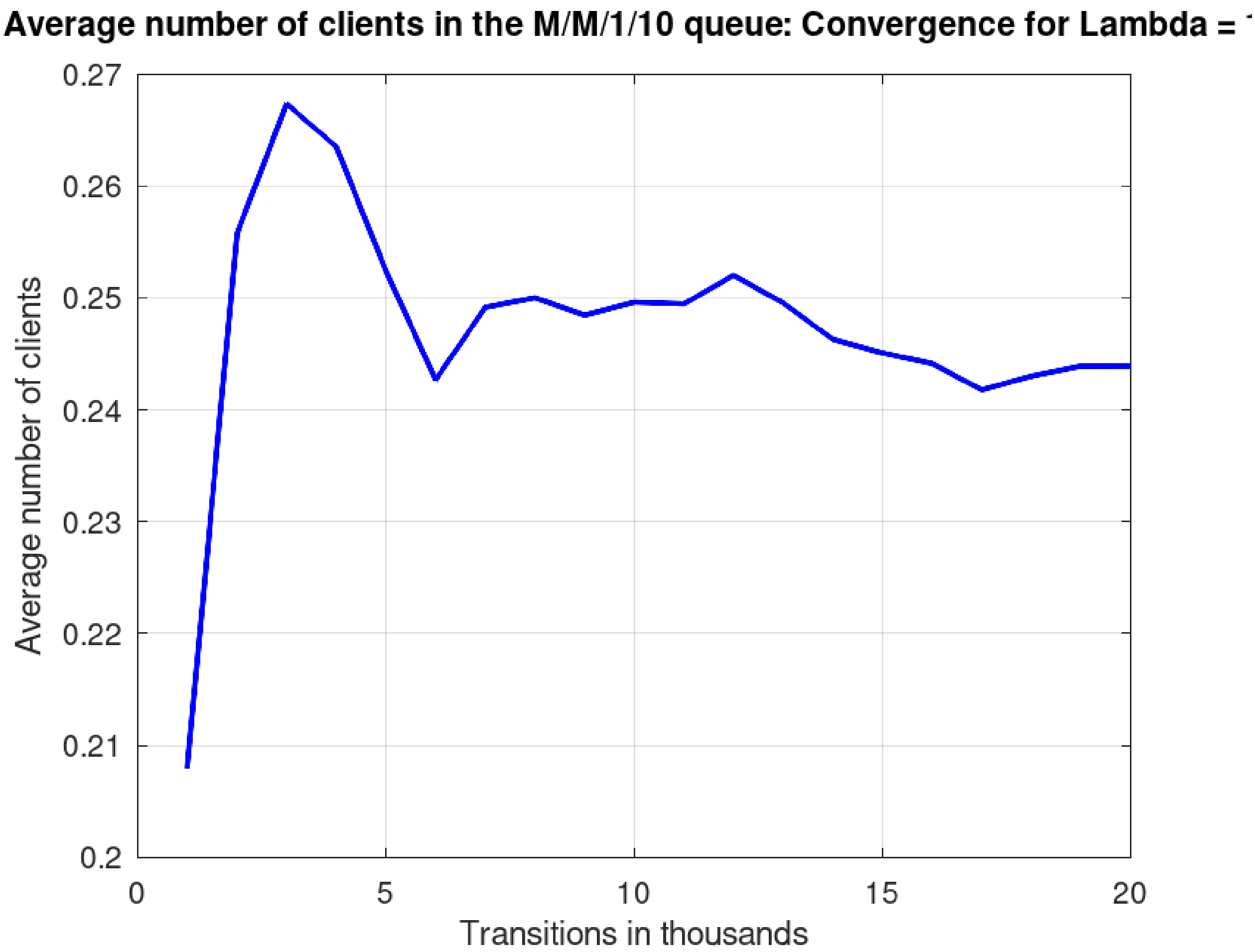
Για λ = 5

trans	state	num_ar	arr	dep
1	0	0	1	0
2	1	0	1	0
3	2	0	0	1
4	1	1	1	0
5	2	0	0	1
6	1	2	0	1
7	0	1	1	0
8	1	2	1	0
9	2	0	0	1
10	1	3	1	0
11	2	0	0	1
12	1	4	1	0
13	2	0	0	1
14	1	5	0	1
15	0	2	1	0
16	1	5	1	0
17	2	0	0	1
18	1	6	1	0
19	2	0	0	1
20	1	7	1	0
21	2	0	0	1
22	1	8	1	0
23	2	0	0	1
24	1	9	1	0
25	2	0	0	1
26	1	10	0	1
27	0	3	1	0
28	1	10	1	0
29	2	0	1	0
30	3	0	1	0

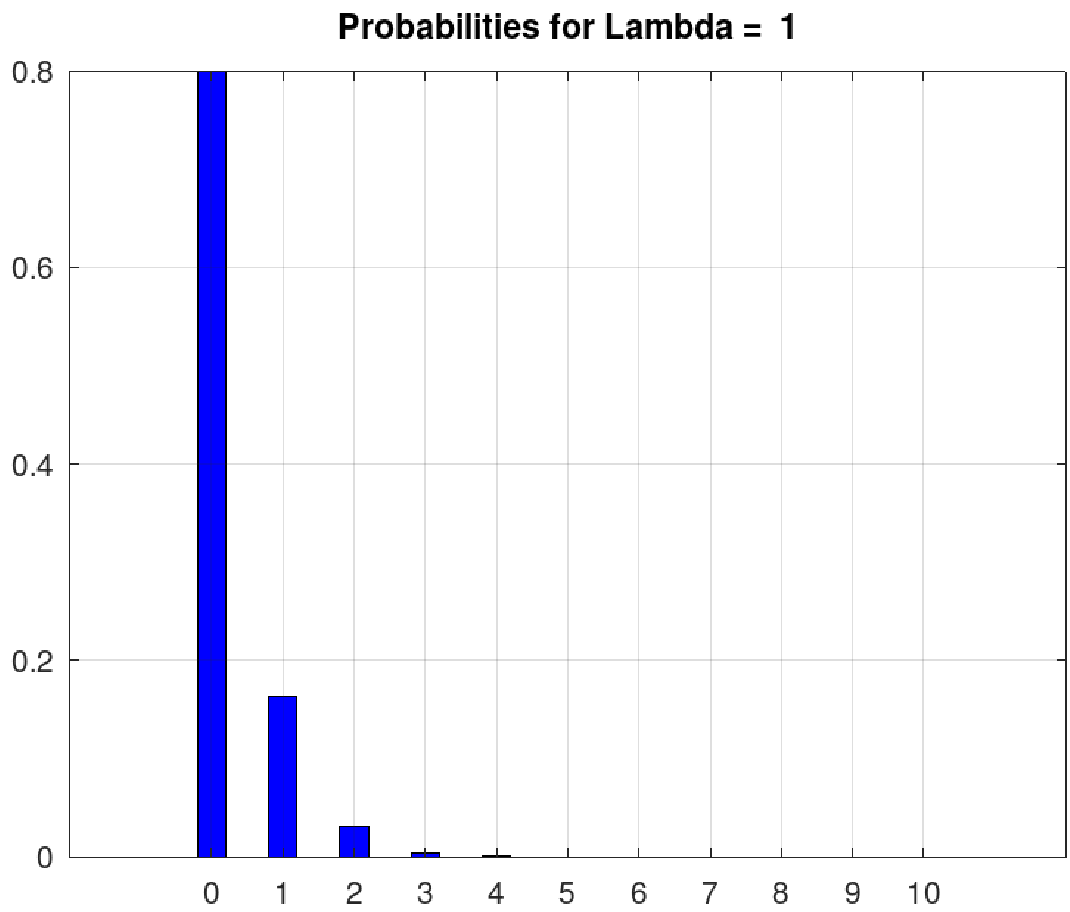
Γιο λ = 10

trans	state	num_ar	arr	dep
1	0	0	1	0
2	1	0	1	0
3	2	0	0	1
4	1	1	1	0
5	2	0	0	1
6	1	2	1	0
7	2	0	1	0
8	3	0	1	0
9	4	0	0	1
10	3	1	1	0
11	4	0	1	0
12	5	0	1	0
13	6	0	1	0
14	7	0	1	0
15	8	0	0	1
16	7	1	0	1
17	6	1	0	1
18	5	1	1	0
19	6	1	1	0
20	7	1	1	0
21	8	0	1	0
22	9	0	0	1
23	8	1	1	0
24	9	0	1	0
25	10	0	1	0
26	10	1	0	1
27	9	1	1	0
28	10	1	1	0
29	10	2	1	0
30	10	3	1	0

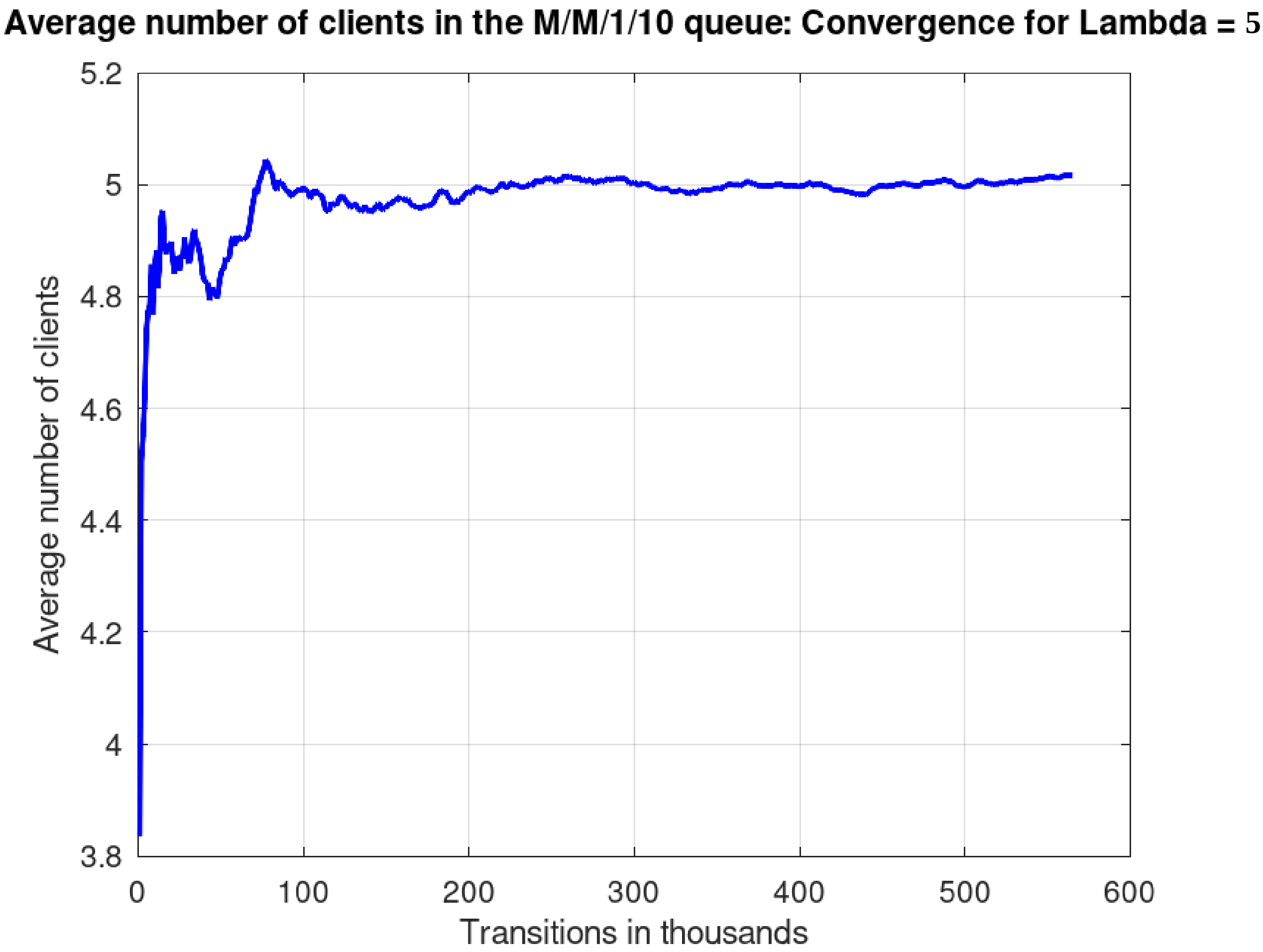
Ερώτηση 2)



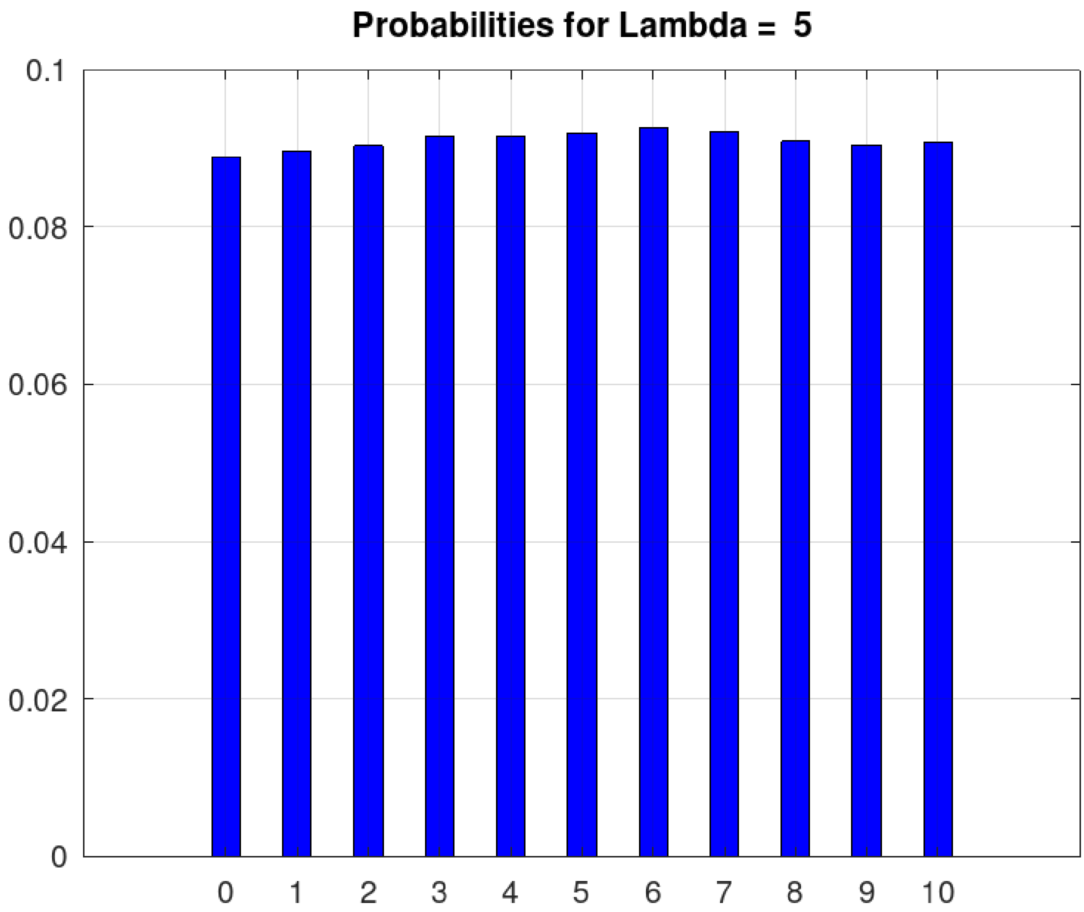
Average number of clients in the M/M/1/10 queue: Convergence for Lambda =
Average number of clients
Transitions in thousands
0.27
0.26
0.25
0.24
0.23
0.22
0.21
0.2
0
5
10
15
20



Probabilities for Lambda = 1
0.8
0.6
0.4
0.2
0
0
1
2
3
4
5
6
7
8
9
10

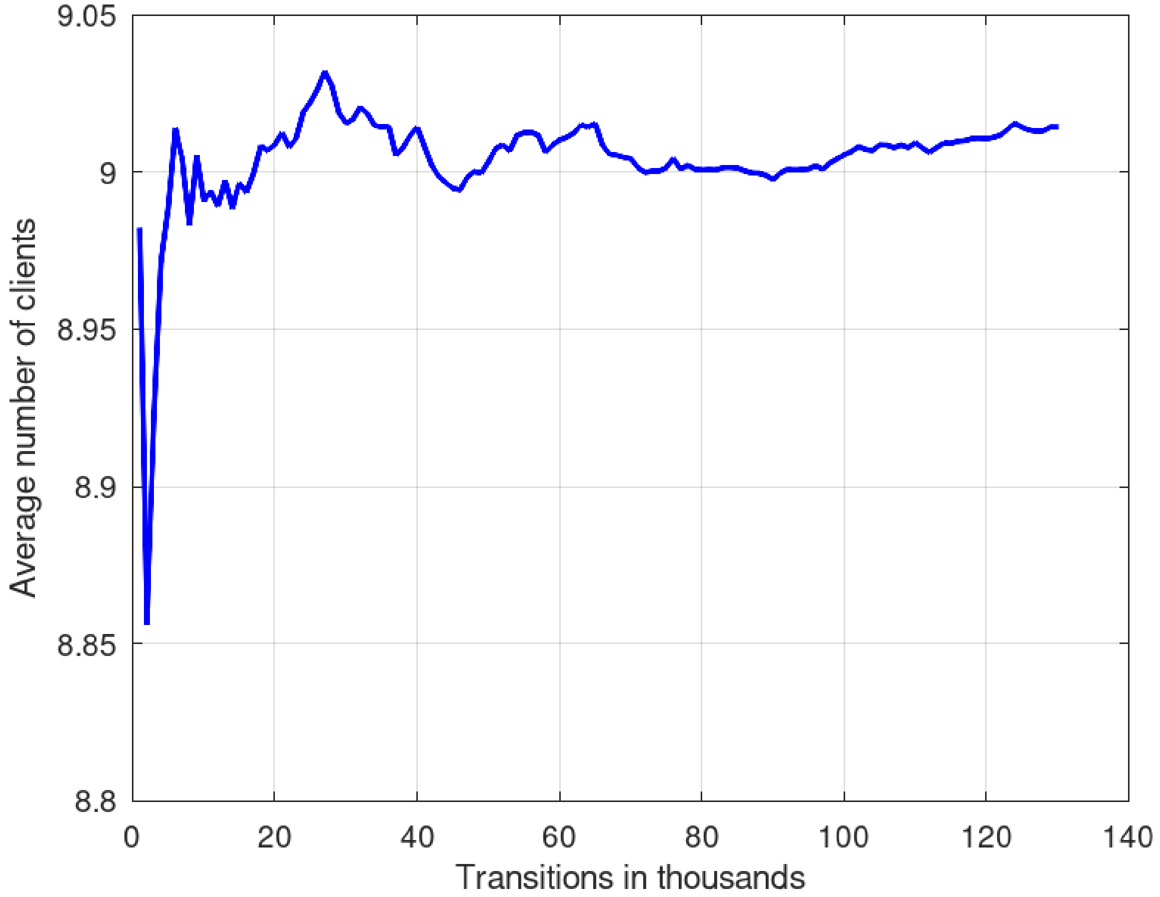


Average number of clients in the M/M/1/10 queue: Convergence for Lambda = 5
Average number of clients
Transitions in thousands
5.2
5
4.8
4.6
4.4
4.2
4
3.8
0
100
200
300
400
500
600

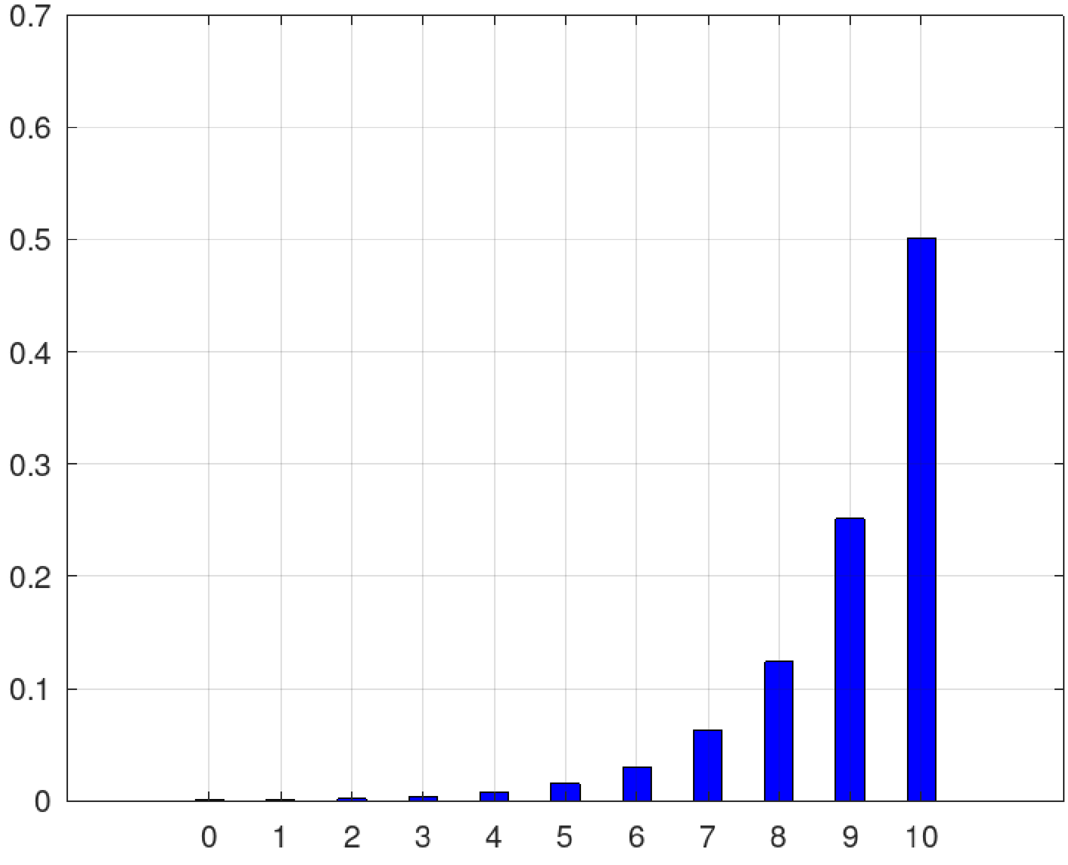


Probabilities for Lambda = 5
0.1
0.08
0.06
0.04
0.02
0
0
1
2
3
4
5
6
7
8
9
10

Average number of clients in the M/M/1/10 queue: Convergence for Lambda = 1 0



Probabilities for Lambda = 10



Ερώτηση 3)

Παρατηρούμε πως για $0 \leq |\lambda - \mu| \leq \varepsilon$ όσο το ε αυξάνει, τότε αυξάνεται και η σύγκλιση. Ο αριθμός το transitions που μπορούμε να αγνοήσουμε εξαρτάτε στο λ που εξαρτάτε στο ε. Έτσι όσο μεγαλύτερο ε τόσο μικρότερο αριθμό μπορούμε να αγνοήσουμε. Έτσι για τα 3 λ μπορούμε να αγνοήσουμε τουλάχιστον 20κ – 50κ transitions.

Ερώτηση 4)

Εάν είχαμε $\mu_i = \mu \cdot (i+1)$ με $\mu = 1 \frac{\pi\varepsilon\lambda\acute{\alpha}\tau\eta\varsigma}{sec}$ (i=1,2,3…,10) τότε θα έπρεπε σε κάθε μετάβαση να αλλάζουμε το threshold με βάση το state στο οποίο είμαστε, δηλαδή θα έπρεπε (έστω πως είμαστε στην κατάσταση $threshold_i = \frac{\lambda}{\lambda + current\_state + 1}$.

Κώδικας

```
% M/M/1 simulation. We will find the probabilities of the first states.
% Note: Due to ergodicity, every state has a probability >0.

clc;
clear all;
close all;

figure_counter = 0;
rand('seed',12163);

for lambda = [1,5,10]
  arrivals = [0,0,0,0,0,0,0,0,0,0,0]
  mu = 5;
  total_arrivals = 0; % to measure the total number of arrivals
  current_state = 0;  % holds the current state of the system
  previous_mean_clients = 0; % will help in the convergence test
  index = 0;
  threshold = lambda/(lambda + mu); % the threshold used to calculate probabilities

  transitions = 0; % holds the transitions of the simulation in transitions steps
  tracem_i = 0;
  tracem = [];
  to_plot = [];
  while transitions >= 0
    transitions = transitions + 1; % one more transitions step

    tracem_i = tracem_i + 1;
    if tracem_i > 0 && tracem_i < 31 %&& current_state > 0
      tracem(tracem_i,1) = tracem_i;
      tracem(tracem_i,2) = current_state;
      tracem(tracem_i,3) = arrivals(current_state+1);
    endif

    if mod(transitions,1000) == 0 % check for convergence every 1000 transitions steps
      index = index + 1;
      for i=1:1:length(arrivals)
          P(i) = arrivals(i)/total_arrivals; % calculate the probability of every state in the system
      endfor

      mean_clients = 0; % calculate the mean number of clients in the system
      for i=1:1:length(arrivals)
         mean_clients = mean_clients + (i-1).*P(i);
      endfor

      to_plot(index) = mean_clients;

      if abs(mean_clients - previous_mean_clients) < 0.00001 || transitions > 1000000 % convergence
test
        break;
      endif
```

```
      previous_mean_clients = mean_clients;

    endif

    random_number = rand(1); % generate a random number (Uniform distribution)
        if current_state == 0 || random_number < threshold % arrival
            %{
            if 0 < tracem_i < 31
                tracem(tracem_i,4) = 1
                %disp("arrival"), disp(current_state);
            endif
            %}
            total_arrivals = total_arrivals + 1;
            % to catch the exception if variable arrivals(i) is undefined. Required only for systems with
finite capacity.
            x = arrivals(current_state + 1) + 1;
            arrivals(current_state + 1) = x; % increase the number of arrivals in the current state
            if (current_state != 10)
                current_state = current_state + 1;
            endif
            else % departure
            %{
            if 0 < tracem_i < 31
                tracem(tracem_i,5) = 1
            endif
            %}
            if current_state != 0 % no departure from an empty system
            current_state = current_state - 1;
            endif
        endif


  endwhile

  for i=1:1:length(arrivals)
    display(P(i));
  endfor

  figure_counter += 1;
  figure(figure_counter);
  plot(to_plot,"b","linewidth",2);
  title(strjoin({"Average number of clients in the M/M/1/10 queue: Convergence for Lambda =
",num2str((lambda))},""));
  xlabel("Transitions in thousands");
  ylabel("Average number of clients");
  grid on;
  saveas (figure_counter, strjoin({"figure_",num2str(1),"_lambda_",num2str((lambda)),".png"}))

  figure_counter += 1;
  figure(figure_counter);
  bar(0:1:(length(arrivals)-1),P,'b',0.4);
```

```
  title(strjoin({"Probabilities for Lambda = ",num2str((lambda))}));
  grid on;
  saveas (figure_counter, strjoin({"figure_",num2str(2),"_lambda_",num2str((lambda)),".png"}))


  disp("trans state num_ar arr dep");
  disp(tracem);

endfor
```