ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ
ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

3η Ομάδα Ασκήσεων

Συστήματα Αναμονής (Queuing Systems)

ΔΗΜΗΤΡΙΟΣ ΓΕΩΡΓΟΥΣΗΣ

03119005

georg.dimi00@gmail.com

Προσομοίωση συστήματος M/M/1/10

Ο κώδικας για αυτή τη προσομοίωση είναι: (‘problem3.m’)

```
%M/M/1/10 simulation. We have 11 states. We will find the
probabilities
%of all states.
%This chain is ergodic since it has a finite amount of states.

clc;
clear all;
close all;

lambda = [1, 5, 10];
mu = 5;

convergence_lim = 0.00001;
max_trans = 1000000;

states = 11;

%do the simulation for all 3 systems.
for i = 1:1:length(lambda)

  rand("seed",1);

  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; %used for arrivals array

  arrivals = zeros(1, states); %arrivals at each state

  P = zeros(1, states); %probabilities of each state

  %the threshold used to calculate probabilities.
  threshold = lambda(i)/(lambda(i) + mu);

  %holds the transitions of the simulation in transitions steps
  transitions = 0;

  %the output - graph
  to_plot(1) = 0;

  while (transitions >= 0)
    ++transitions; %one more transition step

    if mod(transitions, 1000) == 0 %check for convergence every 1000
steps
      index = index + 1;
      for j = 1:1:states
        P(j) = arrivals(j)/total_arrivals;
      endfor

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

```
      to_plot(index) = mean_clients; %this function records the
development of
                                     %the mean per 1000 transitions
      if abs(mean_clients - previous_mean_clients) <
convergence_lim...
         || transitions > max_trans %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
      total_arrivals += 1;
      arrivals(current_state + 1) += 1;
      if current_state < (states - 1)
        current_state += 1;
      endif
    else %departure, current_state can't be 0 here.
      current_state -= 1;
    endif
  endwhile

  %we have calculated probabilities of states.
  P_blocking = P(states); %we have calculated P_blocking.
  %previous_mean_clients has the mean clients in the system stored.
  %we need to calculate mean delay time of a client.
  gamma = lambda(i)*(1 - P_blocking); %this is also called
throughput.
  mean_delay = previous_mean_clients / gamma; %we calculated delay.

  %we need to display all of these...
  disp(["Lambda is ", num2str(lambda(i))]);
  disp("");
  disp(["Transitions to convergence: ", num2str(transitions)]);
  disp("Ergodic probabilities of the system: ");
  disp(P');
  disp(["Blocking probability: ", num2str(P_blocking)]);
  disp(["Average clients: ", num2str(previous_mean_clients)]);
  disp(["Average delay time: ", num2str(mean_delay)]);
  disp("");

  %plot the ergodic probabilities
  figure(i);
  bar([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10], P, "m", 0.5);
  title(["Lambda = ", num2str(lambda(i))]);
  xlabel("States");
  ylabel("Ergodic Probabilities");
  set(gca, "fontsize", 20);
```

```
%plot the mean clients over time
  figure(length(lambda) + i);
  plot(to_plot, "b", "LineWidth", 2);
  title(["Lambda = ", num2str(lambda(i))]);
  xlabel("Thousand transitions");
  ylabel("Average number of clients");
  set(gca, "fontsize", 20);

    clear to_plot;%so that we get to the next plot
endfor
```

Δείχνουμε και τον κώδικα για το debugging: ('problem3_debug.m')

```
%M/M/1/10 simulation. We have 11 states. We will find the
probabilities
%of all states.
%This chain is ergodic since it has a finite amount of states.

clc;
clear all;
close all;

lambda = [1, 5, 10];
mu = 5;

convergence_lim = 0.00001;
max_trans = 1000000;

states = 11;

%do the simulation for all 3 systems.
for i = 2:1:length(lambda)

  rand("seed",1);

  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; %used for arrivals array

  arrivals = zeros(1, states); %arrivals at each state

  P = zeros(1, states); %probabilities of each state

  %the threshold used to calculate probabilities.
  threshold = lambda(i)/(lambda(i) + mu);

  %holds the transitions of the simulation in transitions steps
  transitions = 0;

  %the output - graph
  to_plot(1) = 0;

  while (transitions >= 0 && transitions <= 30)
    ++transitions; %one more transition step
    %debug message begin
    disp(["Total arrivals: ", num2str(total_arrivals)]);
    disp(["State: ", num2str(current_state)]);
    %debug message end
    if mod(transitions, 1000) == 0 %check for convergence every 1000
steps
```

```
      index = index + 1;
      for j = 1:1:states
        P(j) = arrivals(j)/total_arrivals;
      endfor

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

      to_plot(index) = mean_clients; %this function records the
development of
                                     %the mean per 1000 transitions
      if abs(mean_clients - previous_mean_clients) <
convergence_lim...
         || transitions > max_trans %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
      %debug message begin
      disp("Arrival");
      %debug message end
      ++total_arrivals;
      arrivals(current_state + 1) += 1;
      %debug message begin
      disp(["Total arrivals at state ", num2str(current_state)," are
",...
      num2str(arrivals(current_state + 1))]);
      %debug message end
      if current_state < (states - 1)
        ++current_state;
      endif
    else %departure
      %debug message begin
      disp("Departure");
      disp(["Total arrivals at state ", num2str(current_state)," are
",...
      num2str(arrivals(current_state + 1))]);
      %debug message end
      --current_state; %current_state can't be 0 here.
    endif
    %newline for debug messages
    disp("");
  endwhile
  %{
  %we have calculated probabilities of states.
  P_blocking = P(states); %we have calculated P_blocking.
  %previous_mean_clients has the mean clients in the system stored.
  %we need to calculate mean delay time of a client.
  gamma = lambda(i)*(1 - P_blocking); %this is also called
throughput.
  mean_delay = previous_mean_clients / gamma; %we calculated delay.
```

```
  %we need to display all of these...
  disp(["Lambda is ", num2str(lambda(i))]);
  disp("");
  disp(["Transitions to convergence: ", num2str(transitions)]);
  disp("Ergodic probabilities of the system: ");
  disp(P);
  disp(["Blocking probability: ", num2str(P_blocking)]);
  disp(["Average clients: ", num2str(previous_mean_clients)]);
  disp(["Average delay time: ", num2str(mean_delay)]);
  disp("");

  %plot the ergodic probabilities
  figure(i);
  bar([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10], P, "m", 0.5);
  title(["Lambda = ", num2str(lambda(i))]);
  xlabel("States");
  ylabel("Ergodic Probabilities");
  set(gca, "fontsize", 20);

  %plot the mean clients over time
  figure(length(lambda) + i);
  plot(to_plot, "b", "LineWidth", 2);
  title(["Lambda = ", num2str(lambda(i))]);
  xlabel("Thousand transitions");
  ylabel("Average number of clients");
  set(gca, "fontsize", 20);

    clear to_plot;
    %}
  break;
endfor
```

Ο παραπάνω κώδικας θα τρέξει μόνο για τις πρώτες 30 επαναλήψεις για το σύστημα με λ = 5 πελάτες/min.

(1)

Η έξοδος που μας δίνει είναι το σύνολο των αφίξεων γενικά στο σύστημα μέχρι στιγμής (αυτό δεν ζητείται). Έπειτα μας λέει την τρέχουσα κατάσταση, αν είχαμε άφιξη ή αναχώρηση πελάτη στο σύστημα και τον συνολικό αριθμό αφίξεων στη τρέχουσα κατάσταση (αφού ενημερωθεί σε περίπτωση άφιξης). Βλέπουμε τις εξόδους παρακάτω:

```
Total arrivals: 0
State: 0
Arrival
Total arrivals at state 0 are 1

Total arrivals: 1
State: 1
Departure
Total arrivals at state 1 are 0

Total arrivals: 1
State: 0
Arrival
Total arrivals at state 0 are 2

Total arrivals: 2
State: 1
Arrival
Total arrivals at state 1 are 1

Total arrivals: 3
State: 2
Departure
Total arrivals at state 2 are 0

Total arrivals: 3
State: 1
Departure
Total arrivals at state 1 are 1

Total arrivals: 3
State: 0
Arrival
Total arrivals at state 0 are 3

Total arrivals: 4
State: 1
Arrival
Total arrivals at state 1 are 2

Total arrivals: 5
State: 2
Departure
Total arrivals at state 2 are 0

Total arrivals: 5
State: 1
Arrival
Total arrivals at state 1 are 3
```

```
Total arrivals: 6
State: 2
Departure
Total arrivals at state 2 are 0

Total arrivals: 6
State: 1
Arrival
Total arrivals at state 1 are 4

Total arrivals: 7
State: 2
Arrival
Total arrivals at state 2 are 1

Total arrivals: 8
State: 3
Arrival
Total arrivals at state 3 are 1

Total arrivals: 9
State: 4
Arrival
Total arrivals at state 4 are 1

Total arrivals: 10
State: 5
Arrival
Total arrivals at state 5 are 1

Total arrivals: 11
State: 6
Arrival
Total arrivals at state 6 are 1

Total arrivals: 12
State: 7
Arrival
Total arrivals at state 7 are 1

Total arrivals: 13
State: 8
Arrival
Total arrivals at state 8 are 1

Total arrivals: 14
State: 9
Departure
Total arrivals at state 9 are 0

Total arrivals: 14
State: 8
Departure
Total arrivals at state 8 are 1

Total arrivals: 14
State: 7
Arrival
Total arrivals at state 7 are 2
```

```
Total arrivals: 15
State: 8
Departure
Total arrivals at state 8 are 1

Total arrivals: 15
State: 7
Arrival
Total arrivals at state 7 are 3

Total arrivals: 16
State: 8
Departure
Total arrivals at state 8 are 1

Total arrivals: 16
State: 7
Arrival
Total arrivals at state 7 are 4

Total arrivals: 17
State: 8
Arrival
Total arrivals at state 8 are 2

Total arrivals: 18
State: 9
Arrival
Total arrivals at state 9 are 1

Total arrivals: 19
State: 10
Arrival
Total arrivals at state 10 are 1

Total arrivals: 20
State: 10
Arrival
Total arrivals at state 10 are 2

Total arrivals: 21
State: 10
Arrival
Total arrivals at state 10 are 3
```

(2)

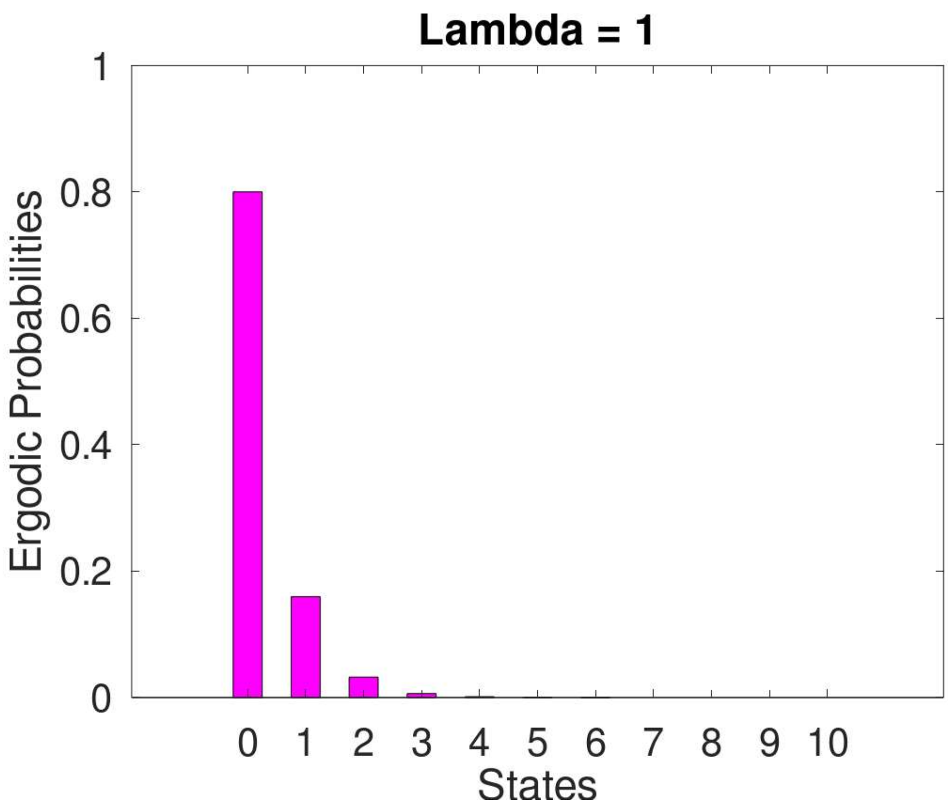
Εκτελούμε τον αρχικό κώδικα και έχουμε:

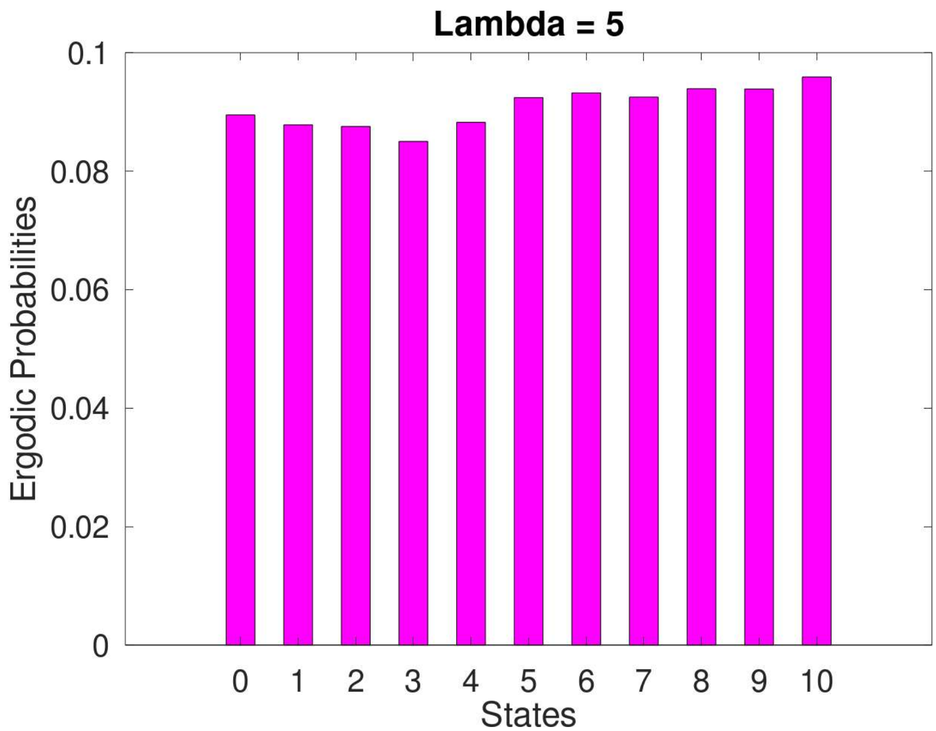
```
Lambda is 1

Transitions to convergence: 37000
Ergodic probabilities of the system:
   0.8002
   0.1596
   0.0322
   0.0062
   0.0013
   0.0004
   0.0001
        0
        0
        0
        0
Blocking probability: 0
Average clients: 0.25033
Average delay time: 0.25033

Lambda is 5

Transitions to convergence: 113000
Ergodic probabilities of the system:
   0.089492
   0.087824
   0.087554
   0.085027
   0.088245
   0.092424
   0.093216
   0.092508
   0.093924
   0.093873
   0.095912
Blocking probability: 0.095912
Average clients: 5.0953
Average delay time: 1.1272

Lambda is 10

Transitions to convergence: 400000
Ergodic probabilities of the system:
   4.7928e-04
   8.5372e-04
   1.7449e-03
   3.8567e-03
   7.7284e-03
   1.5446e-02
   3.0831e-02
   6.1939e-02
   1.2478e-01
   2.5011e-01
   5.0222e-01
Blocking probability: 0.50222
Average clients: 9.0141
Average delay time: 1.8109
```

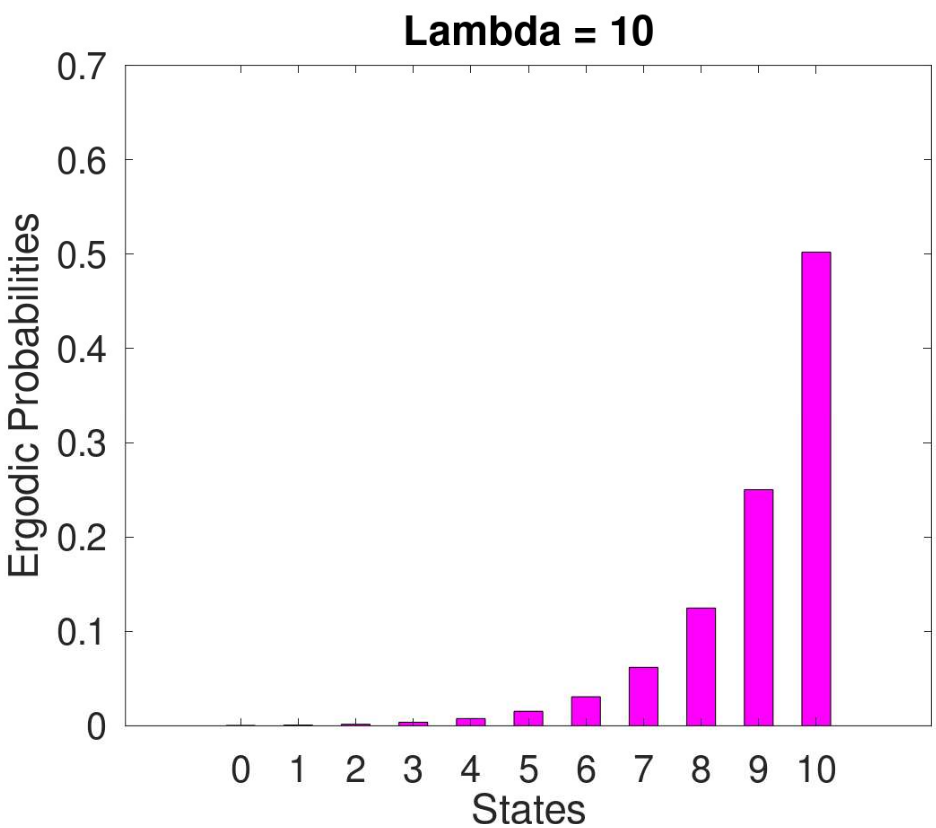
Παραπάνω βλέπουμε ότι για κάθε τιμή του λ υπολογίστηκαν:

- πόσες αλλαγές κατάστασης χρειάστηκαν για να έχουμε σύγκλιση
- οι εργοδικές πιθανότητες των καταστάσεων του συστήματος
- η πιθανότητα απόρριψης πελάτη από το σύστημα
- ο μέσος αριθμός πελατών στο σύστημα
- ο μέσος χρόνος καθυστέρησης ενός πελάτη στο σύστημα

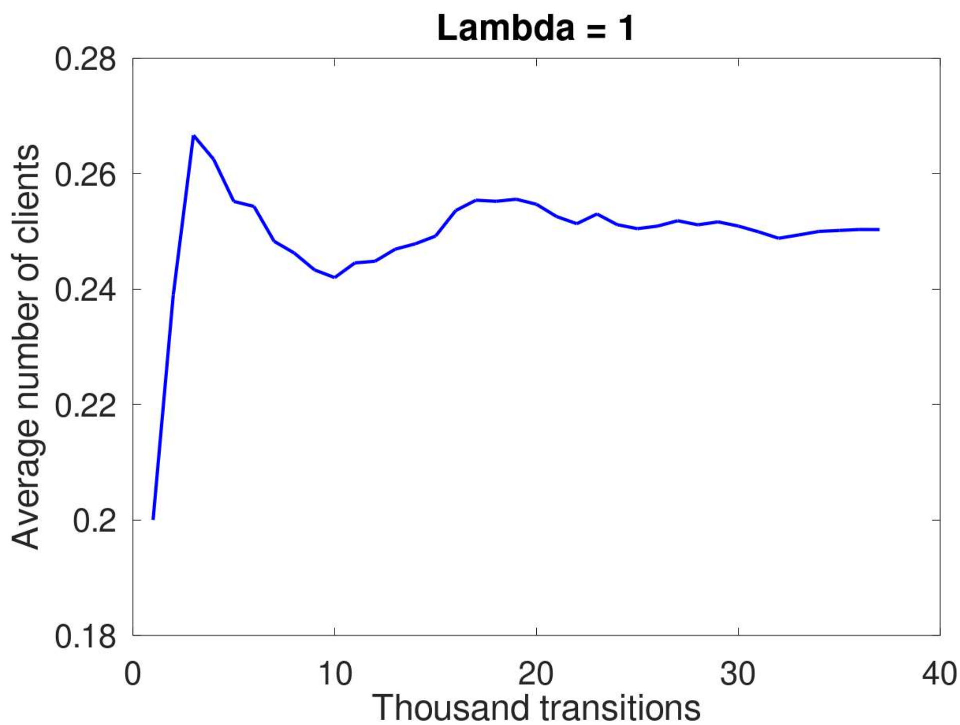
Οι γραφικές παραστάσεις που παίρνουμε είναι:

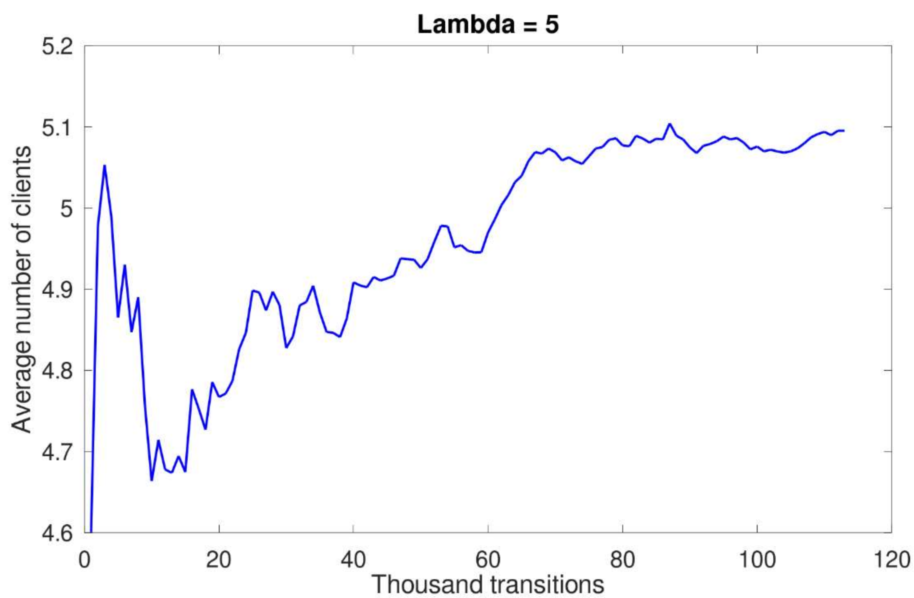


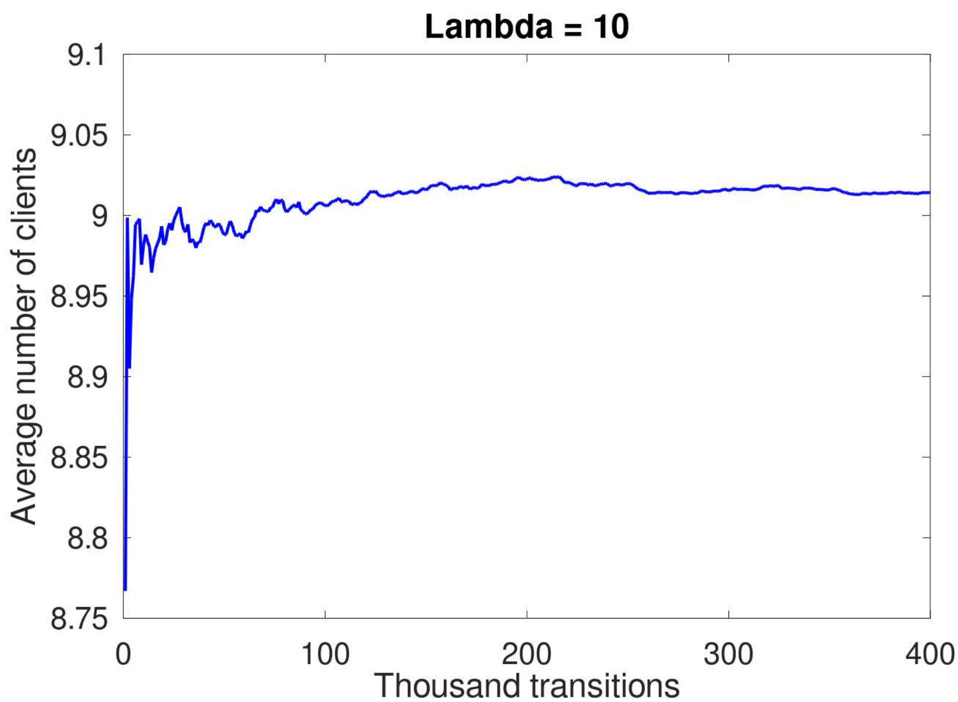

Lambda = 5
0.1
0.08
0.06
0.04
0.02
0
Ergodic Probabilities
0 1 2 3 4 5 6 7 8 9 10
States


Lambda = 10
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Ergodic Probabilities
0 1 2 3 4 5 6 7 8 9 10
States

Για την εξέλιξη των μέσων τιμών έχουμε:







Όταν ο ρυθμός άφιξης (λ) είναι μικρότερος από τον ρυθμό εξυπηρέτησης (μ = 5 πελάτες/min) βλέπουμε ότι οι εργοδικές πιθανότητες είναι μαζεμένες στις πρώτες καταστάσεις του συστήματος (περίπτωση λ = 1 πελάτης/min) αφού το σύστημα προλαβαίνει να τους εξυπηρετήσει. Όταν το λ είναι ίσο με το μ τότε οι εργοδικές πιθανότητες των καταστάσεων είναι περίπου (θεωρητικά θα ήταν όλες 1/11 = 0.0909…) ίσες μεταξύ τους. Αυτό το εξηγούμε αφού σε κάθε κατάσταση έχουμε ίδια πιθανότητα άφιξης ή αναχώρησης. Τέλος όταν το λ είναι μεγαλύτερο του, μπορούμε να θεωρήσουμε ότι το σύστημα δεν προλαβαίνει να τους εξυπηρετήσει, έτσι θα βρίσκεται συνήθως σε μη μηδενικές καταστάσεις και, πιο πιθανό είναι να βρίσκεται σε καταστάσεις κοντινότερες στον κορεσμό. Η δήλωση αυτή αποτυπώνεται στις εργοδικές πιθανότητες της περίπτωσης λ = 10 πελάτες/min, οι οποίες είναι συγκεντρωμένες στις τελευταίες καταστάσεις του συστήματος.

(3)

Από τα παραπάνω είδαμε ότι:

λ (σε πελάτες/min)	μεταβάσεις
1	37.000
5	113.000
10	400.000

Βλέπουμε ότι καθώς το λ αυξάνει χρειάζονται περισσότερες μεταβάσεις μέχρι τη σύγκλιση, η ταχύτητα σύγκλισης μειώνεται. Αυτό είναι αναμενόμενο, διότι η αύξηση του λ σημαίνει περισσότερους πελάτες στη μονάδα του χρόνου, ωστόσο, ο ρυθμός εξυπηρέτησής τους μένει ίδιος, επομένως, το σύστημα υπερφορτώνεται και θα χρειαστεί περισσότερο χρόνο μέχρι να έρθει σε ισορροπία. Με άλλα λόγια η αύξηση του λ σημαίνει και μεγαλύτερη μεταβατική περίοδο μέχρι τη μόνιμη κατάσταση (ισορροπία).

Ψάχνουμε σημεία από τα οποία και μετά οι γραφικές παραστάσεις να είναι κοντά στις μέσες τιμές που υπολογίσαμε με τη προσομοίωση.

λ (σε πελάτες/min)	Μέση τιμή πελατών	Μεταβάσεις που αγνοούνται
1	0.25033	25.000
5	5.0953	95.000
10	9.0141	250.000

Για τις συγκρίσεις μεταξύ των συστημάτων στο παραπάνω ερώτημα χρησιμοποιήσαμε το rand("seed”,1) ως πρώτη εντολή του εξωτερικού for loop ώστε για κάθε τιμή του λ να πάρουμε την ίδια ακολουθία μεταβάσεων.

(4)

Γνωρίζουμε για τις σχετικές πιθανότητες μεταβάσεων k -> (k + 1) και k -> (k – 1) ότι:

$$P(k \to (k+1)/\mu\varepsilon\tau\acute{\alpha}\beta\alpha\sigma\eta) = \frac{\lambda_k}{\lambda_k + \mu_k} \ \kappa\alpha\iota$$
$$P(k \to (k-1)/\mu\varepsilon\tau\acute{\alpha}\beta\alpha\sigma\eta) = 1 - P(k \to (k+1)/\mu\varepsilon\tau\acute{\alpha}\beta\alpha\sigma\eta)$$

Επομένως για $\mu_k = \mu(1+k), k = 0, 1, 2, \dots, 10$ έχουμε ότι θα χρησιμοποιούσαμε διαφορετικό threshold για την επιλογή άφιξης – αναχώρησης σε κάθε κατάστασης. Θα ήταν:

$$threshold(k) = \frac{\lambda}{\lambda + \mu(1+k)}, k = 1, 2, 3, \dots, 10 \ (\delta\varepsilon\nu\ \beta\acute{\alpha}\zeta o\upsilon\mu\varepsilon\ \tau\eta\ \mu\eta\delta\varepsilon\nu\iota\kappa\acute{\eta}\ \kappa\alpha\tau\acute{\alpha}\sigma\tau\alpha\sigma\eta)$$

Στον κώδικά μας θα ορίζαμε το threshold ως πίνακα 10 θέσεων (αρίθμηση από 1 έως 10) και σε κάθε θέση να έχει αποθηκευμένη την αντίστοιχη τιμή από τη παραπάνω συνάρτηση.

Έπειτα θα βάζαμε ξεχωριστά τις περιπτώσεις για current_state == 0 και random_number < threshold, όπου η δεύτερη περίπτωση θα γινόταν:

random_number < threshold(current_state).