ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

Μάθημα: Συστήματα Αναμονής

Ονοματεπώνυμο: Ειρήνη Δόντη

Α.Μ: 03119839

3η Ομάδα Ασκήσεων

Αθήνα 2022

Προσομοίωση συστήματος M/M/1/10

(1)

Επιλέγουμε τις τιμές λ=5 και μ=5 πελάτες/min, για να προσομοιώσουμε το σύστημα M/M/1/10. Παράγουμε επαναλαμβανόμενα, με χρήση while, έναν τυχαίο αριθμό (με τη βοήθεια της εντολής rand()) ο οποίος δηλώνει την άφιξη ή την αναχώρηση ενός πελάτη από το σύστημα. Η παραπάνω επανάληψη τελειώνει στην περίπτωση που γίνουν 1.000.000 μεταβάσεις ή διαφορά μεταξύ δύο διαδοχικών μέσων αριθμών πελατών είναι μικρότερη από 0.001%. Εντός του προγράμματος, δηλώνουμε το διάστημα $\frac{\lambda}{\lambda+\mu}$ στο οποίο έχουμε άφιξη λ προς το διάστημα πιθανών μεταβάσεων λ+μ (άφιξη ή αναχώρηση). Στην περίπτωση που ο αριθμός είναι μεγαλύτερος από την τιμή $\frac{\lambda}{\lambda+\mu}$ (threshold), τότε δηλώνεται η αναχώρηση από το σύστημα, ενώ στην περίπτωση που ο αριθμός είναι μικρότερος από το την τιμή $\frac{\lambda}{\lambda+\mu}$ (threshold) τότε δηλώνεται η άφιξη πελάτη στο σύστημα. Επίσης, πρέπει να ελέγξουμε, με έναν μετρητή που ελέγχει τις συνολικές αφίξεις, την κατάσταση στην οποία αναχωρεί ο πελάτης από την κατάσταση 0 ή γίνεται άφιξη πελάτη στην κατάσταση 10. Υπολογίζουμε τον μέσο όρο των πελατών και τις πιθανότητες κατάστασης του συστήματος για κάθε τέτοια επανάληψη.

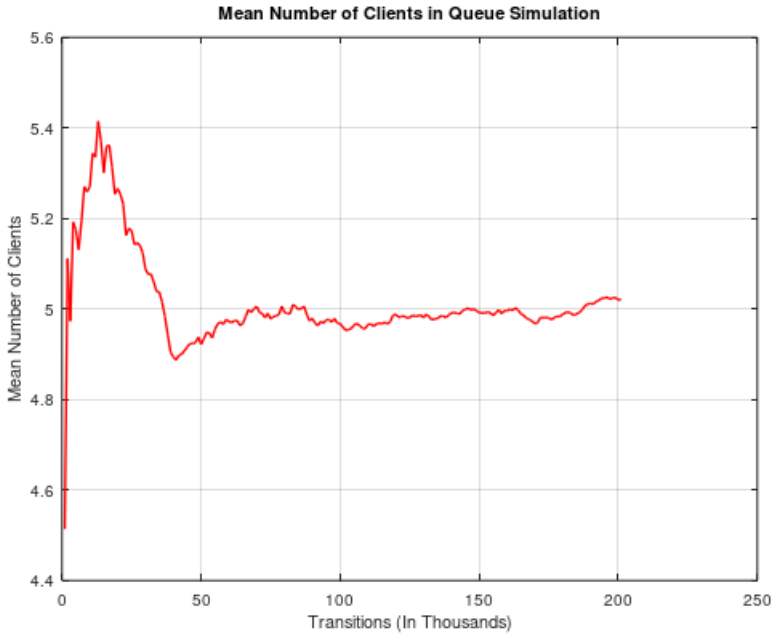
- Οι πιθανότητες των καταστάσεων του συστήματος που παρήχθησαν αφότου εκτελέσαμε το πρόγραμμα , φαίνονται παρακάτω:

```
Propabilities of Every State:
ans =  0.090801
ans =  0.089434
ans =  0.089633
ans =  0.090592
ans =  0.090706
ans =  0.091039
ans =  0.092852
ans =  0.089405
ans =  0.091561
ans =  0.092482
ans =  0.091494
```

- Η πιθανότητα απόρριψης πελάτη από το σύστημα, είναι ουσιαστικά η πιθανότητα της τελευταίας κατάστασης του συστήματος, όπως απεικονίζεται παρακάτω:

```
Blocking Propability = p_bl =  0.091494
```

- Ο μέσος αριθμός πελατών στο σύστημα συναρτήσει των μεταβάσεων του συστήματος, απεικονίζεται στο παρακάτω διάγραμμα το οποίο δημιουργήθηκε αφού εκτελέσαμε τον κώδικα:



- Ο μέσος χρόνος καθυστέρησης ενός πελάτη στο σύστημα υπολογίζεται, με τη βοήθεια του προγράμματος ως εξής:

```
Mean Delay Time = delay =  1.1054
```

Παρακάτω, τυπώνουμε από το πρόγραμμά μας τις 30 πρώτες μεταβάσεις, οι οποίες βοηθούν στο debugging του προγράμματος μας:

```
Current State = s = 0
Next Transition - Arrival
Total Arrivals in Current State = a = 0
Current state = s =  1
Next transition - Departure
Total Arrivals In Current State = a = 0
Current State = s = 0
Next Transition - Arrival
Total Arrivals in Current State = a =  1
Current state = s =  1
Next transition - Departure
Total Arrivals In Current State = a = 0
Current State = s = 0
Next Transition - Arrival
Total Arrivals in Current State = a =  2
Current state = s =  1
Next transition - Departure
Total Arrivals In Current State = a = 0
Current State = s = 0
Next Transition - Arrival
Total Arrivals in Current State = a =  3
Current state = s =  1
Next transition - Departure
Total Arrivals In Current State = a = 0
Current State = s = 0
Next Transition - Arrival
Total Arrivals in Current State = a =  4
Current State = s =  1
Next Transition - Arrival
Total Arrivals in Current State = a = 0
Current state = s =  2
Next transition - Departure
Total Arrivals In Current State = a = 0
Current state = s =  1
Next transition - Departure
Total Arrivals In Current State = a =  1
Current State = s = 0
Next Transition - Arrival
Total Arrivals in Current State = a =  5
Current state = s =  1

Next transition - Departure
Total Arrivals In Current State = a =  1
Current State = s = 0
Next Transition - Arrival
Total Arrivals in Current State = a =  6
Current state = s =  1
Next transition - Departure
Total Arrivals In Current State = a =  1
Current State = s = 0
Next Transition - Arrival
Total Arrivals in Current State = a =  7
Current state = s =  1
Next transition - Departure
Total Arrivals In Current State = a =  1
Current State = s = 0
Next Transition - Arrival
Total Arrivals in Current State = a =  8
Current State = s =  1
Next Transition - Arrival
Total Arrivals in Current State = a =  1
Current State = s =  2
Next Transition - Arrival
Total Arrivals in Current State = a = 0
Current State = s =  3
Next Transition - Arrival
Total Arrivals in Current State = a = 0
Current State = s =  4
Next Transition - Arrival
Total Arrivals in Current State = a = 0
Current state = s =  5
Next transition - Departure
Total Arrivals In Current State = a = 0
Current state = s =  4
Next transition - Departure
Total Arrivals In Current State = a =  1
Current state = s =  3
Next transition - Departure
Total Arrivals In Current State = a =  1
Current State = s =  2

Next Transition - Arrival
Total Arrivals in Current State = a =  1
Current state = s =  3
Next transition - Departure
Total Arrivals In Current State = a =  1
Current State = s =  2
Next Transition - Arrival
Total Arrivals in Current State = a =  2
Current State = s =  3
Next Transition - Arrival
Total Arrivals in Current State = a =  1
```

Ο κώδικας που κατασκευάσαμε είναι ο παρακάτω:

```
clc;
clear all;
close all;

% 1st Task

P = zeros(1,11);
arrivals = zeros(1,11);
number_of_total_arrivals = 0; % Total Number of Arrivals
state = 0;                    % Current State of System
previous_mean_clients = 0;    % Previous Mean Clients
index = 0;
lambda = 5;
mu = 5;
threshold = lambda/(lambda + mu); % Threshold λ/(λ+μ)
transitions = 0;                  % Transitions of the Simulation

while transitions >= 0
  transitions = transitions + 1;  % Increase Transition Step
  if mod(transitions,1000) == 0   % Check for Convergence Every 1000 Transitions
    index = index + 1;
    for i=1:1:length(arrivals)
        P(i) = arrivals(i)/number_of_total_arrivals; % Probability for Every State
    endfor

    mean_clients = 0;                               % Calculate the Mean Number of Clients
    for i=1:1:length(arrivals)
       mean_clients = mean_clients + (i-1).*P(i);
    endfor

    help_to_plot(index) = mean_clients;

    if abs(mean_clients - previous_mean_clients) < 0.00001 || transitions > 200000 % Test Convergence
      break;
    endif

    previous_mean_clients = mean_clients;

  endif


  random_number = rand(1);                    % Generate Random Number
  if state == 0 || random_number < threshold  % In Case of Arrival
    if state < 11
      number_of_total_arrivals = number_of_total_arrivals + 1;
      if transitions < 31
        printf("Current State = "); s = state
        printf("Next Transition - Arrival\n");
        printf("Total Arrivals in Current State = "); a = arrivals(state+1)
      endif
      arrivals(state + 1) = arrivals(state + 1) + 1; % Increase Number of Arrivals In Current State
      if state < 10
        state = state + 1;
      endif
    endif
  else                                        % In Case of departure
    if state != 0 % No Departure from Empty System
      if transitions < 31
        printf("Current state = "); s = state
        printf("Next transition - Departure\n");
        printf("Total Arrivals In Current State = "); a = arrivals(state+1)
      endif
        state = state - 1;
    endif
  endif
endwhile
```

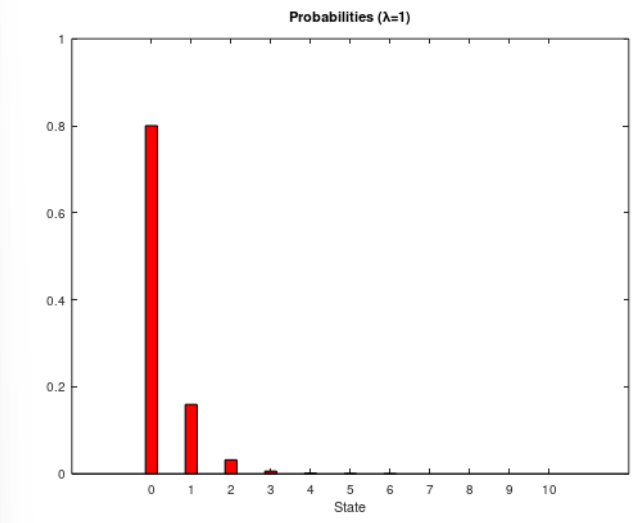
```
#(i)
printf("Propabilities of Every State:\n");
for i=1:1:length(arrivals)
  P(i)
endfor

#(ii)
printf("Blocking Propability = "); p_bl = P(11)

#(iii)
figure;
plot(help_to_plot,"r","linewidth",1.4);
title("Mean Number of Clients in Queue Simulation");
xlabel("Transitions (In Thousands)");
ylabel("Mean Number of Clients");
hold on;
grid on;

#(iv)
average_delay_time = mean_clients/(lambda*(1-P(11)));
printf("Mean Delay Time = "); delay = average_delay_time
```

(2)

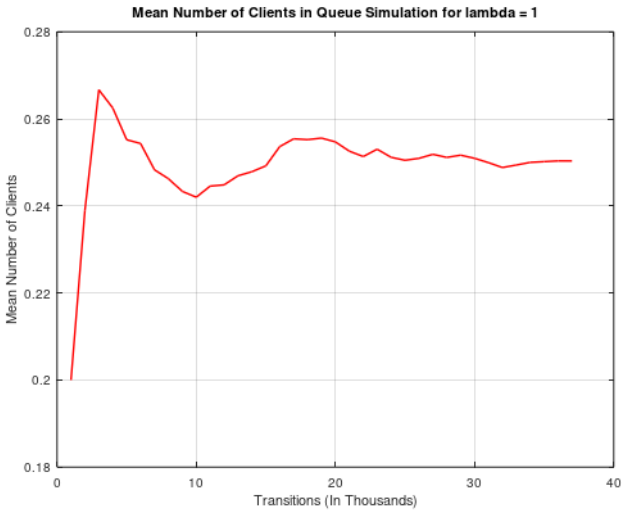
Εκτελούμε την προσομοίωσή μας για τιμές του λ = {1, 5, 10} πελάτες/min και θα αναπαραστήσουμε γραφικά τα ζητούμενα διαγράμματα:

Για παράμετρο **λ = 1** πελάτες/min:

(α) Αναπαριστούμε τις εργοδικές πιθανότητες που υπολογίζει η προσομοίωση με τη βοήθεια της εντολής bar(), όπως φαίνεται παρακάτω:



(β) Αναπαριστούμε την εξέλιξη του μέσου αριθμού πελατών στο σύστημα για τις τιμές που υπολογίσαμε, όπως φαίνεται παρακάτω:



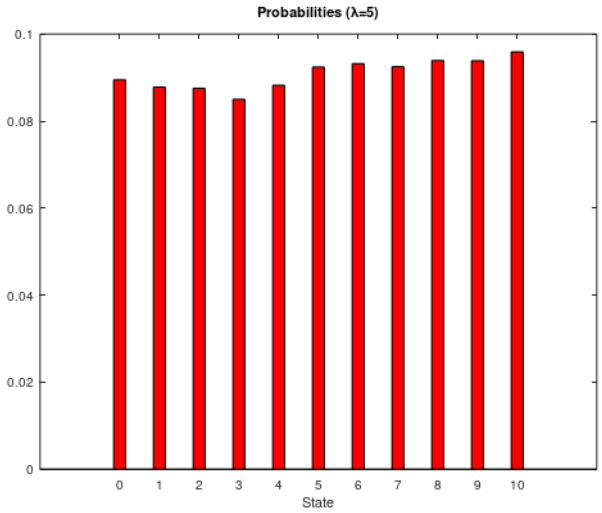
Ο κώδικας που χρησιμοποιήσαμε για τη δημιουργία των παραπάνω διαγραμμάτων (για λ = 1 πελάτες/min) είναι το παρακάτω:

```
clc;
clear all;
close all;
rand("seed",1);

% 2nd Task


printf("for lambda = 1 client/min\n"); % λ = 1 πελάτης/min
P = zeros(1,11);
arrivals = zeros(1,11);
number_of_total_arrivals = 0; % Total Number of Arrivals
state = 0;                    % Current State of System
previous_mean_clients = 0;    % Previous Mean Clients
index = 0;
lambda = 1; # λ = 1
mu = 5;
threshold = lambda/(lambda + mu); % Threshold λ/(λ+μ)
transitions = 0;                  % Transitions of the Simulation

while transitions >= 0
  transitions = transitions + 1;  % Increase Transition Step
  if mod(transitions,1000) == 0   % Check for Convergence Every 1000 Transitions
    index = index + 1;
    for i=1:1:length(arrivals)
        P(i) = arrivals(i)/number_of_total_arrivals; % Probability for Every State
    endfor

    mean_clients = 0;                              % Calculate the Mean Number of Clients
    for i=1:1:length(arrivals)
       mean_clients = mean_clients + (i-1).*P(i);
    endfor

    help_to_plot(index) = mean_clients;

    if abs(mean_clients - previous_mean_clients) < 0.00001 || transitions > 200000 % Test Convergence
      break;
    endif
```

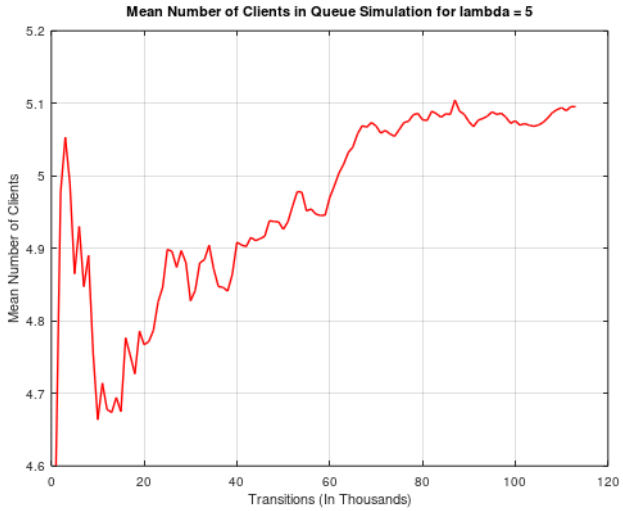
```
    previous_mean_clients = mean_clients;

  endif

  random_number = rand(1);                    % Generate Random Number
  if state == 0 || random_number < threshold  % In Case of Arrival
    if state < 11
      number_of_total_arrivals = number_of_total_arrivals + 1;
      if transitions < 31
        printf("Current State = "); s = state
        printf("Next Transition - Arrival\n");
        printf("Total Arrivals in Current State = "); a = arrivals(state+1)
      endif
      arrivals(state + 1) = arrivals(state + 1) + 1; % Increase Number of Arrivals In Current State
      if state < 10
        state = state + 1;
      endif
    endif
  else                                        % In Case of departure
    if state != 0 % No Departure from Empty System
      if transitions < 31
        printf("Current state = "); s = state
        printf("Next transition - Departure\n");
        printf("Total Arrivals In Current State = "); a = arrivals(state+1)
      endif
        state = state - 1;
    endif
  endif
endwhile



#(i)
printf("Propabilities of Every State:\n");
for i=1:1:length(arrivals)
  P(i)
endfor

#(ii)
printf("Blocking Propability = "); p_bl = P(11)

# (a)
x=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10];
figure;
bar(x,P,0.3,"r");
title("Probabilities (λ=1)");
xlabel("State");

#(iii) or (b)
figure;
plot(help_to_plot,"r","linewidth",1.4);
title("Mean Number of Clients in Queue Simulation for lambda = 1");
xlabel("Transitions (In Thousands)");
ylabel("Mean Number of Clients");
hold on;
grid on;

#(iv)
average_delay_time = mean_clients/(lambda*(1-P(11)));
printf("Mean Delay Time = "); delay = average_delay_time
```

Για παράμετρο **λ = 5** πελάτες/min:

(α) Αναπαριστούμε τις εργοδικές πιθανότητες που υπολογίζει η προσομοίωση με τη βοήθεια της εντολής bar(), όπως φαίνεται παρακάτω:



(β) Αναπαριστούμε την εξέλιξη του μέσου αριθμού πελατών στο σύστημα για τις τιμές που υπολογίσαμε, όπως φαίνεται παρακάτω:

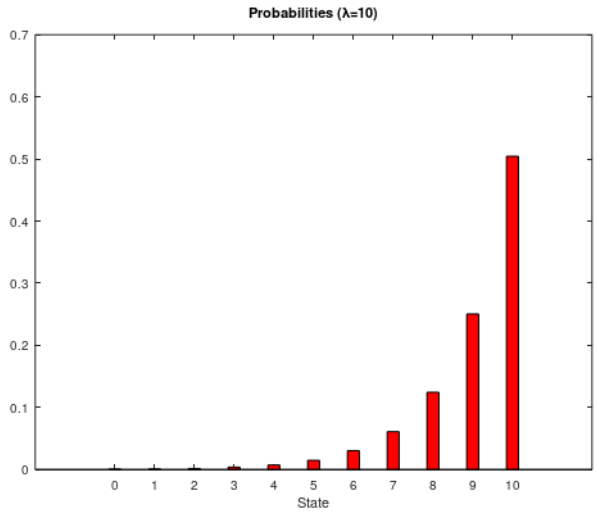


Ο κώδικας που χρησιμοποιήσαμε για τη δημιουργία των παραπάνω διαγραμμάτων (για λ = 5 πελάτες/min) είναι το παρακάτω:

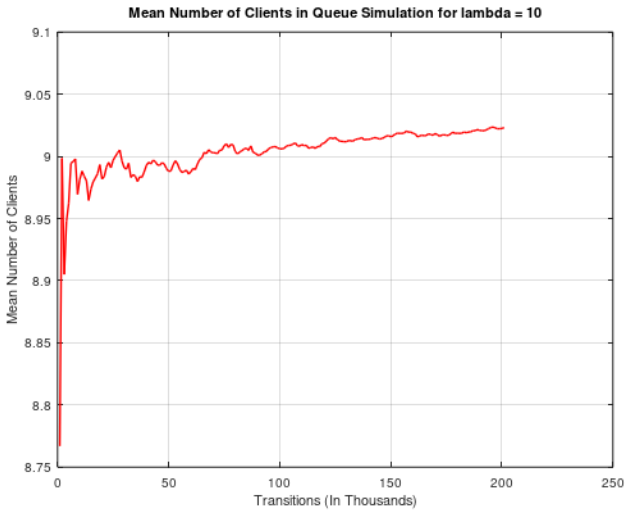
```
rand("seed",1);


% λ = 5 πελάτες/min

printf("for lambda = 5 clients/min\n"); % λ = 5 πελάτες/min
P = zeros(1,11);
arrivals = zeros(1,11);
number_of_total_arrivals = 0; % Total Number of Arrivals
state = 0;                    % Current State of System
previous_mean_clients = 0;    % Previous Mean Clients
index = 0;
lambda = 5; # λ = 5
mu = 5;
threshold = lambda/(lambda + mu); % Threshold λ/(λ+μ)
transitions = 0;                  % Transitions of the Simulation

while transitions >= 0
  transitions = transitions + 1;  % Increase Transition Step
  if mod(transitions,1000) == 0   % Check for Convergence Every 1000 Transitions
    index = index + 1;
    for i=1:1:length(arrivals)
        P(i) = arrivals(i)/number_of_total_arrivals; % Probability for Every State
    endfor

    mean_clients = 0;                                % Calculate the Mean Number of Clients
    for i=1:1:length(arrivals)
       mean_clients = mean_clients + (i-1).*P(i);
    endfor

    help_to_plot(index) = mean_clients;

    if abs(mean_clients - previous_mean_clients) < 0.00001 || transitions > 200000 % Test Convergence
      break;
    endif



    previous_mean_clients = mean_clients;

  endif

  random_number = rand(1);                    % Generate Random Number
  if state == 0 || random_number < threshold  % In Case of Arrival
    if state < 11
      number_of_total_arrivals = number_of_total_arrivals + 1;
      if transitions < 31
        printf("Current State = "); s = state
        printf("Next Transition - Arrival\n");
        printf("Total Arrivals in Current State = "); a = arrivals(state+1)
      endif
      arrivals(state + 1) = arrivals(state + 1) + 1; % Increase Number of Arrivals In Current State
      if state < 10
        state = state + 1;
      endif
    endif
  else                                        % In Case of departure
    if state != 0 % No Departure from Empty System
      if transitions < 31
        printf("Current state = "); s = state
        printf("Next transition - Departure\n");
        printf("Total Arrivals In Current State = "); a = arrivals(state+1)
      endif
        state = state - 1;
    endif
  endif
endwhile



#(i)
printf("Propabilities of Every State:\n");
for i=1:1:length(arrivals)
  P(i)
endfor

#(ii)
printf("Blocking Propability = "); p_bl = P(11)

# (a)
x=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10];
figure;
bar(x,P,0.3,"r");
title("Probabilities (λ=5)");
xlabel("State");

#(iii) or (b)
figure;
plot(help_to_plot,"r","linewidth",1.4);
title("Mean Number of Clients in Queue Simulation for lambda = 5");
xlabel("Transitions (In Thousands)");
ylabel("Mean Number of Clients");
hold on;
grid on;

#(iv)
average_delay_time = mean_clients/(lambda*(1-P(11)));
printf("Mean Delay Time = "); delay = average_delay_time
```

Για παράμετρο **λ = 10** πελάτες/ min:

(α) Αναπαριστούμε τις εργοδικές πιθανότητες που υπολογίζει η προσομοίωση με τη βοήθεια της εντολής bar(), όπως φαίνεται παρακάτω:



(β) Αναπαριστούμε την εξέλιξη του μέσου αριθμού πελατών στο σύστημα για τις τιμές που υπολογίσαμε, όπως φαίνεται παρακάτω:



Ο κώδικας που χρησιμοποιήσαμε για τη δημιουργία των παραπάνω διαγραμμάτων (για λ = 10 πελάτες/min) είναι το παρακάτω:

```
rand("seed",1);


# λ = 10 πελάτες/min

printf("for lambda = 10 clients/min\n"); % λ = 10 πελάτες/min
P = zeros(1,11);
arrivals = zeros(1,11);
number_of_total_arrivals = 0; % Total Number of Arrivals
state = 0;                    % Current State of System
previous_mean_clients = 0;    % Previous Mean Clients
index = 0;
lambda = 10; # λ = 10
mu = 5;
threshold = lambda/(lambda + mu); % Threshold λ/(λ+μ)
transitions = 0;                  % Transitions of the Simulation

while transitions >= 0
  transitions = transitions + 1;  % Increase Transition Step
  if mod(transitions,1000) == 0   % Check for Convergence Every 1000 Transitions
    index = index + 1;
    for i=1:1:length(arrivals)
        P(i) = arrivals(i)/number_of_total_arrivals; % Probability for Every State
    endfor

    mean_clients = 0;                               % Calculate the Mean Number of Clients
    for i=1:1:length(arrivals)
       mean_clients = mean_clients + (i-1).*P(i);
    endfor

    help_to_plot(index) = mean_clients;

    if abs(mean_clients - previous_mean_clients) < 0.00001 || transitions > 200000 % Test Convergence
      break;
    endif


    previous_mean_clients = mean_clients;

  endif

  random_number = rand(1);                    % Generate Random Number
  if state == 0 || random_number < threshold  % In Case of Arrival
    if state < 11
      number_of_total_arrivals = number_of_total_arrivals + 1;
      if transitions < 31
        printf("Current State = "); s = state
        printf("Next Transition - Arrival\n");
        printf("Total Arrivals in Current State = "); a = arrivals(state+1)
      endif
      arrivals(state + 1) = arrivals(state + 1) + 1; % Increase Number of Arrivals In Current State
      if state < 10
        state = state + 1;
      endif
    endif
  else                                        % In Case of departure
    if state != 0 % No Departure from Empty System
      if transitions < 31
        printf("Current state = "); s = state
        printf("Next transition - Departure\n");
        printf("Total Arrivals In Current State = "); a = arrivals(state+1)
      endif
        state = state - 1;
    endif
  endif
endwhile



#(i)
printf("Propabilities of Every State:\n");
for i=1:1:length(arrivals)
  P(i)
endfor

#(ii)
printf("Blocking Propability = "); p_bl = P(11)

# (a)
x=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10];
figure;
bar(x,P,0.3,"r");
title("Probabilities (λ=10)");
xlabel("State");

#(iii) or (b)
figure;
plot(help_to_plot,"r","linewidth",1.4);
title("Mean Number of Clients in Queue Simulation for lambda = 10");
xlabel("Transitions (In Thousands)");
ylabel("Mean Number of Clients");
hold on;
grid on;

#(iv)
average_delay_time = mean_clients/(lambda*(1-P(11)));
printf("Mean Delay Time = "); delay = average_delay_time
```

(3)

Η αύξηση της παραμέτρου λ, επιφέρει τη μείωση της ταχύτητας σύγκλισης της προσομοίωσης και συνεπώς ο απαιτούμενος αριθμός μεταβάσεων αυξάνεται μέχρι να ικανοποιηθεί το κριτήριο σύγκλισης. Η παραπάνω παρατήρηση είναι λογική, καθώς ισχύει η σχέση $\rho = \frac{\lambda}{\mu}$ με μ σταθερό. Οπότε, όσο αυξάνεται η παράμετρος λ, τόσο η ένταση του φορτίου ρ αυξάνεται. Με άλλα λόγια, στο σύστημα που διατηρεί τον ίδιο ρυθμό εξυπηρέτησης, οι αφίξεις πελατών διατηρούν τον ίδιο ρυθμό εξυπηρέτησης και συνεπώς το σύστημα υπερφορτώνεται, επιμηκύνοντας την κατάσταση του συστήματος πριν επέλθει η εργοδική ισορροπία. Για τις διάφορες τιμές της παραμέτρου λ του προγράμματός μας, προσθέτουμε την εντολή transitions (χωρίς τον χαρακτήρα “;”), ώστε να εμφανιστεί η τιμή της μεταβλητής transition στο command window. Οπότε, οι μεταβάσεις που χρειάζονται το σύστημα για να επέλθει σε ισορροπία είναι τα παρακάτω:

Οι μεταβάσεις που χρειάζονται ώστε το σύστημα να επέλθει σε ισορροπία, για παράμετρο **λ = 1** πελάτες/min είναι:

```
transitions =  37000
```

Οπότε, συμπεριλαμβανομένου του παραγόμενου διαγράμματος, για λ = 1 πελάτες/min, οι αρχικές μεταβάσεις που μπορούν να αγνοηθούν ώστε να επιταχυνθεί η σύγκλιση της προσομοίωση είναι 25000 μεταβάσεις.

Οι μεταβάσεις που χρειάζονται ώστε το σύστημα να επέλθει σε ισορροπία, για παράμετρο **λ = 5** πελάτες/min είναι:

```
transitions =  113000
```

Οπότε, συμπεριλαμβανομένου του παραγόμενου διαγράμματος, για λ = 5 πελάτες/min, οι αρχικές μεταβάσεις που μπορούν να αγνοηθούν ώστε να επιταχυνθεί η σύγκλιση της προσομοίωση είναι 90000 μεταβάσεις.

Οι μεταβάσεις που χρειάζονται ώστε το σύστημα να επέλθει σε ισορροπία, για παράμετρο **λ = 10** πελάτες/min είναι:

```
transitions =  201000
```

Οπότε, συμπεριλαμβανομένου του παραγόμενου διαγράμματος, για λ = 10 πελάτες/min, οι αρχικές μεταβάσεις που μπορούν να αγνοηθούν ώστε να επιταχυνθεί η σύγκλιση της προσομοίωση είναι 100500 μεταβάσεις.

(4)

Αν ο αριθμός μ είναι μεταβλητός και εξαρτημένος από την κατάσταση στην οποία βρίσκεται το σύστημα $\mu_i = \frac{\lambda}{\lambda + \mu_i} = \frac{\lambda}{\lambda + \mu(i+1)}$ με παράμετρο i = {1, 2, …, 10} την τωρινή κατάσταση. Οι αλλαγές που πρέπει να γίνουν στον κώδικα είναι οι εξής: θα χρειαστεί να ορίζεται, μετά από κάθε αλλαγή της μεταβλητής state, η γνωστή παράμετρος μ πελάτες/min, δηλαδή mu = 5*[state + 1]; και συνεπώς θα μεταβληθεί ανάλογα και το κατώφλι. Ένας άλλος τρόπος κατάλληλης τροποποίησης είναι να δημιουργηθεί, αρχικά, ένας πίνακας 10 θέσεων και μετά από κάθε μετάβαση να υπολογίζεται το κατώφλι με την εξής εντολή: lambda/(lambda + mu(state));