Εθνικό Μετσόβιο Πολυτεχνείο

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Συστήματα Αναμονής

3η ομάδα ασκήσεων

Ελευθερία Αρκαδοπούλου

el19442

Προσομοίωση συστήματος M/M/1/10

1) Για το debugging της προσομοίωσης χρησιμοποιήθηκε ο παρακάτω κώδικας:

```
%debug
if transitions <=30
  if current_state == 0 || random_number < threshold % arrival
    str = strcat("current state is_", int2str(current_state), "_arrival with total arrivals_", int2str(arrivals(current_state+1)));
 else
    str = strcat("current state is_", int2str(current_state), "_departure with total arrivals_", int2str(arrivals(current_state+1)));
 endif
disp(str)
endif
```

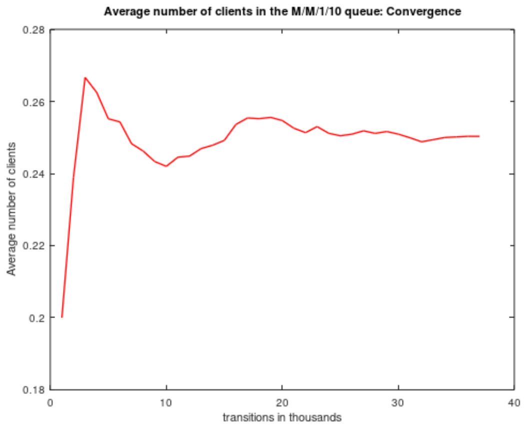
Ο κώδικας παράγει λεπτομερές trace των πρώτων 30 μεταβάσεων (transitions <= 30), και τυπώνει κάθε φορά την κατάσταση του συστήματος (current_state), το αν η επόμενη μετάβαση είναι άφιξη ή αναχώρηση (ελέγχεται στα φωλιασμένα if) και το συνολικό αριθμό αφίξεων στην παρούσα κατάσταση (arrivals(current_state+1)). Η εκτύπωση (για λ=1):

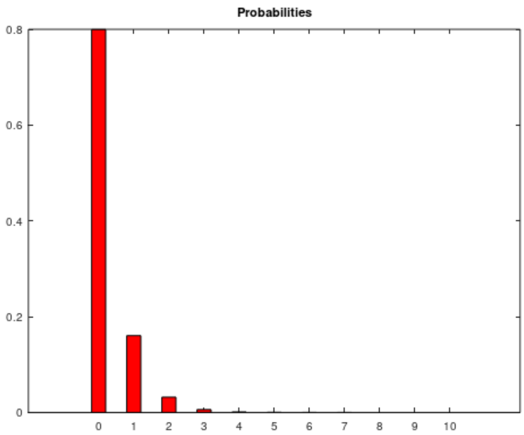
```
current state is_0_arrival with total arrivals_0
current state is_1_departure with total arrivals_0
current state is_0_arrival with total arrivals_1
current state is_1_arrival with total arrivals_0
current state is_2_departure with total arrivals_0
current state is_1_departure with total arrivals_1
current state is_0_arrival with total arrivals_2
current state is_1_arrival with total arrivals_1
current state is_2_departure with total arrivals_0
current state is_1_arrival with total arrivals_2
current state is_2_departure with total arrivals_0
current state is_1_departure with total arrivals_3
current state is_0_arrival with total arrivals_3
current state is_1_departure with total arrivals_3
current state is_0_arrival with total arrivals_4
current state is_1_departure with total arrivals_3
current state is_0_arrival with total arrivals_5
current state is_1_departure with total arrivals_3
current state is_0_arrival with total arrivals_6
current state is_1_departure with total arrivals_3
current state is_0_arrival with total arrivals_7
current state is_1_arrival with total arrivals_3
current state is_2_departure with total arrivals_0
current state is_1_departure with total arrivals_4
current state is_0_arrival with total arrivals_8
current state is_1_departure with total arrivals_4
current state is_0_arrival with total arrivals_9
current state is_1_departure with total arrivals_4
current state is_0_arrival with total arrivals_10
current state is_1_arrival with total arrivals_4
```

Εφόσον ελέγξαμε ότι η προσομοίωση λειτουργεί, απενεργοποιούμε τον κώδικα debugging.

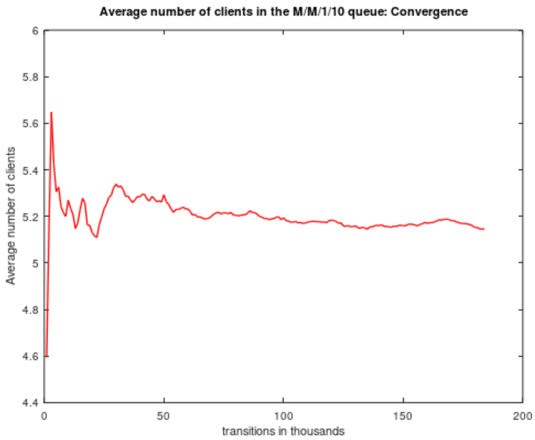
2) Εκτελούμε την προσομοίωση για λ=1,5,10 αντίστοιχα. Παρακάτω παρίστανται γραφικά οι εργοδικές πιθανότητες και η εξέλιξη του μέσου αριθμού πελατών στο σύστημα.

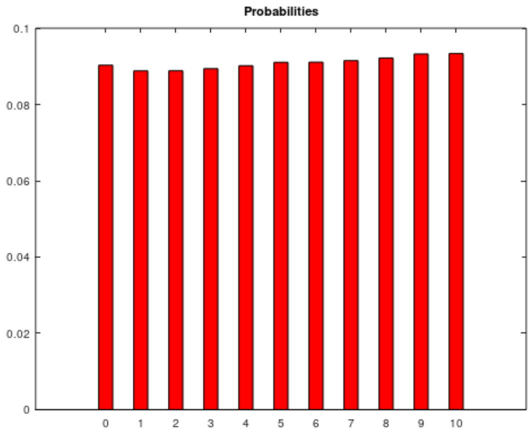
λ=1:



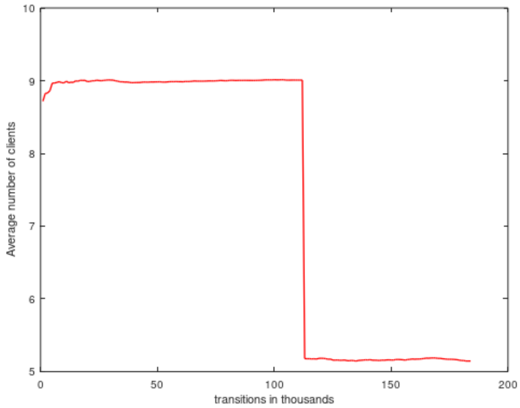


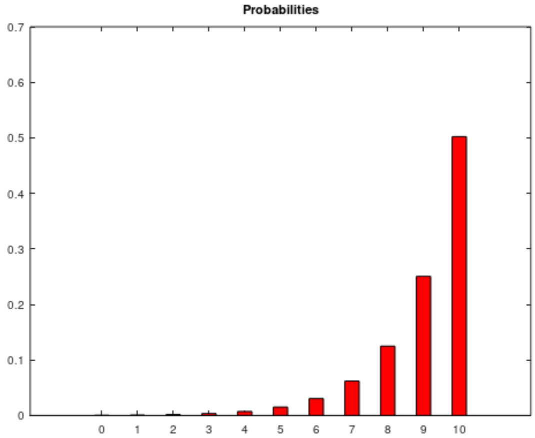
λ=5:





λ=10:





3) Ο απαιτούμενος αριθμός μεταβάσεων μέχρι να ικανοποιηθεί το κριτήριο σύγκλισης για λ=1,5,10 αντίστοιχα είναι ο εξής:

```
Total transitions are
 37000
Total transitions are
 184000
Total transitions are
 112000
```

Για την ταχύτητα σύγκλισης καθώς μεγαλώνει η τιμή του λ παρατηρούμε τα εξής:

- Για λ=1, δηλαδή για ρ=λ/μ=0.2, απαιτείται ο μικρότερος αριθμός μεταβάσεων. Από το διάγραμμα, θα μπορούσαμε να αγνοήσουμε με ασφάλεια περίπου τις 20 χιλιάδες πρώτες μεταβάσεις.
- Για λ=5, δηλαδή για ρ=1, απαιτείται ο μεγαλύτερος αριθμός μεταβάσεων. Από το διάγραμμα, θα μπορούσαμε να αγνοήσουμε με ασφάλεια περίπου τις 70 χιλιάδες πρώτες μεταβάσεις.
- Για λ=10, δηλαδή για ρ=2, απαιτείται αριθμός μεταβάσεων μεγαλύτερος του ρ=0.2 αλλά μικρότερος του ρ=1. Από το διάγραμμα, θα μπορούσαμε να αγνοήσουμε με ασφάλεια περίπου τις 30 χιλιάδες πρώτες μεταβάσεις.

Συμπεραίνουμε ότι, όσο το ρ συγκλίνει στο 1, ο αριθμός απαιτούμενων μεταβάσεων αυξάνεται.

4) Αν το σύστημα είχε εκθετικές εξυπηρετήσεις με μεταβλητό μέσο ρυθμό εξυπηρέτησης $\mu_i = \mu * (i+1)$, θα έπρεπε σε κάθε κατάσταση να αλλάζει το κατώφλι (threshold), το οποίο στην προσομοίωση υπολογίζεται από τον τύπο $threshold_i = \frac{\lambda}{\lambda+\mu_i}$.

Ο κώδικας της προσομοίωσης:

```
% M/M/1/10 simulation.

clc;
clear all;
close all;

rand("seed",1);
mu = 5;
states=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10];

%lambda=1
  disp("lambda=1")
  lambda=1;
  arrivals = zeros(1,11); %vector of arrivals in each state
  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

  while transitions >= 0
    transitions = transitions + 1; % one more transitions step

    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; % calcuate the probability of every state in the system
      endfor

      Pblock = P(11); %Pblocking calculation

      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
      g = lambda * (1 - P(11));
      mean_delay_time = mean_clients / g;

      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)


    %debug
    %if transitions <=30
    %  if current_state == 0 || random_number < threshold % arrival
    %    str = strcat("current state is_", int2str(current_state), "_arrival with total arrivals_", int2str(arrivals(current_state+1)));
    % else
    %    str = strcat("current state is_", int2str(current_state), "_departure with total arrivals_", int2str(arrivals(current_state+1)));
    % endif
    %disp(str)
    %endif


   if current_state == 0 || random_number < threshold % arrival
      total_arrivals = total_arrivals + 1;
      arrivals(current_state + 1) = arrivals(current_state + 1) + 1; % increase the number of arrivals in the current state
      if current_state<10
       current_state = current_state + 1;
      endif
   else % departure
      if current_state != 0 % no departure from an empty system
        current_state = current_state - 1;
      endif
    endif
  endwhile
```

```
disp("Now diplaying Probabilities")
for i=1:1:length(arrivals)
 display(P(i));
endfor

disp("Pblocking is");
disp(Pblock);
disp("Mean number of clients is");
disp(mean_clients);
disp("Mean delay time is");
disp(mean_delay_time);
disp("Total transitions are");
disp(transitions);

figure(1);
plot(to_plot,"r","linewidth",1.3);
title("Average number of clients in the M/M/1/10 queue: Convergence");
xlabel("transitions in thousands");
ylabel("Average number of clients");

figure(2);
bar(states,P,'r',0.4);
title("Probabilities")
```

```
%lambda=5
disp("lambda=5")
lambda=5;
arrivals = zeros(1,11); %vector of arrivals in each state
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

while transitions >= 0
  transitions = transitions + 1; % one more transitions step

  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; % calcuate the probability of every state in the system
    endfor

    Pblock = P(11); %Pblocking calculation

    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

    g = lambda * (1 - P(11));
    mean_delay_time = mean_clients / g;

    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
    total_arrivals = total_arrivals + 1;
    arrivals(current_state + 1) = arrivals(current_state + 1) + 1; % increase the number of arrivals in the current state
    if current_state<10
     current_state = current_state + 1;
    endif
 else % departure
    if current_state != 0 % no departure from an empty system
      current_state = current_state - 1;
    endif
  endif
endwhile
```

```
disp("Now diplaying Probabilities")
for i=1:1:length(arrivals)
 display(P(i));
endfor

disp("Pblocking is");
disp(Pblock);
disp("Mean number of clients is");
disp(mean_clients);
disp("Mean delay time is");
disp(mean_delay_time);
disp("Total transitions are");
disp(transitions);

figure(3);
plot(to_plot,"r","linewidth",1.3);
title("Average number of clients in the M/M/1/10 queue: Convergence");
xlabel("transitions in thousands");
ylabel("Average number of clients");

figure(4);
bar(states,P,'r',0.4);
title("Probabilities")
```

```
%lambda=10
disp("lambda=10")
lambda=10;
arrivals = zeros(1,11); %vector of arrivals in each state
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

while transitions >= 0
  transitions = transitions + 1; % one more transitions step

  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; % calcuate the probability of every state in the system
    endfor

    Pblock = P(11); %Pblocking calculation
```

```
    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
    g = lambda * (1 - P(11));
    mean_delay_time = mean_clients / g;

    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
   total_arrivals = total_arrivals + 1;
   arrivals(current_state + 1) = arrivals(current_state + 1) + 1; % increase the number of arrivals in the current state
   if current_state<10
    current_state = current_state + 1;
   endif
else % departure
   if current_state != 0 % no departure from an empty system
     current_state = current_state - 1;
   endif
  endif
endwhile
disp("Now diplaying Probabilities")
for i=1:1:length(arrivals)
 display(P(i));
endfor
```

```
disp("Pblocking is");
disp(Pblock);
disp("Mean number of clients is");
disp(mean_clients);
disp("Mean delay time is");
disp(mean_delay_time);
disp("Total transitions are");
disp(transitions);

figure(5);
plot(to_plot,"r","linewidth",1.3);
title("Average number of clients in the M/M/1/10 queue: Convergence");
xlabel("transitions in thousands");
ylabel("Average number of clients");

figure(6);
bar(states,P,'r',0.4);
title("Probabilities")
```