ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ

ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

5η Ομάδα Ασκήσεων

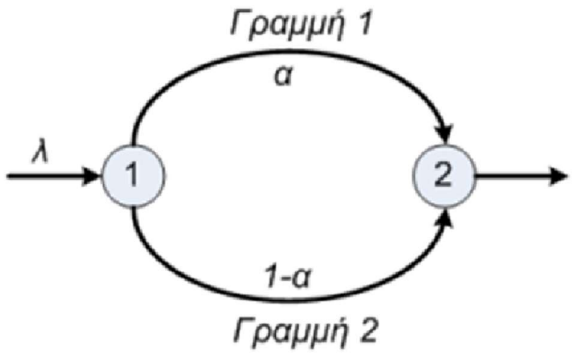
Συστήματα Αναμονής (Queuing Systems)

ΔΗΜΗΤΡΙΟΣ ΓΕΩΡΓΟΥΣΗΣ

03119005

georg.dimi00@gmail.com

Δίκτυο με εναλλακτική δρομολόγηση



(1)

Παραδοχές:

- Η ροή πακέτων ακολουθεί κατανομή Poisson με λ = 10 Kpps.
- Η διάσπαση της ροής αυτής στον κόμβο 1 στις 2 γραμμές γίνεται με τυχαίο τρόπο (κατανομή Bernoulli) ώστε στις Γραμμές 1 και 2 να έχουμε άφιξη πακέτων σύμφωνα με κατανομή Poisson με παράμετρο λ1 = αλ και λ2 = (1 – α)λ αντίστοιχα.
- Οι χρόνοι εξυπηρέτησης των πακέτων να είναι εκθετικά κατανεμημένοι με μέσες τιμές 1/μ1 και 1/μ2 αντίστοιχα (εδώ ως χρόνους εξυπηρέτησης θεωρούμε τον χρόνο που θέλει το κάθε πακέτο για να τελειώσει η μετάδοσή του). Επειδή οι χωρητικότητες των γραμμών είναι σταθερές αυτή η παραδοχή μπορεί να εκφραστεί καλύτερα ως εξής: Το μέγεθος (μήκος) των πακέτων να ακολουθεί εκθετική κατανομή.
- Για τις εντάσεις φορτίου – ρ1 = λ1/μ1 και ρ2 = λ2/μ2 – στις 2 γραμμές να ισχύει ρ1 < λ/μ1 < 1 και ρ2 < λ/μ2 < 1.

Έστω L το μέγεθος του πακέτου τότε E(L) = 128 bytes

Για τη γραμμή 1:

λ1 = 10α Kpps

μ1 = C1/E(L) = (15*10^6 bits/sec) / (128*8 bits) = 14.648 Kpps

ρ1 = λ1/μ1 = 0.683α < 0.683 < 1 (δεκτό)

Για τη γραμμή 2:

λ2 = 10(1 – α) Kpps

μ2 = C2/E(L) = (12*10^6 bits/sec) / (128*8 bits) = 11.719 Kpps

ρ2 = λ2/μ2 = 0.853(1 – α) < 0.853 < 1 (δεκτό)

Παρατηρούμε ότι αν ισχύουν οι 3 πρώτες παραδοχές θα ισχύει και η $4^{η}$.

(2)

Throughput: γ = λ(1 – P_blocking) = λ (δεν υπάρχει πιθανότητα απόρριψης πακέτου αφού και οι 2 γραμμές είναι M/M/1)

Τύπος του Little: E(T) = E(n)/γ = E(n)/λ

Ας υπολογίσουμε το E(n):

Για ένα σύστημα M/M/1 έχουμε E(n) = ρ/(1 – ρ), χρησιμοποιούμε αυτόν τον τύπο για να υπολογίσουμε το E(n) στο δικό μας σύστημα:

$$E(n) = E(n_1) + E(n_2) = \lambda_1 \cdot E(T_1) + \lambda_2 \cdot E(T_2)$$

$$E(T) = \frac{\lambda_1 \cdot E(T_1) + \lambda_2 \cdot E(T_2)}{\lambda} = a \cdot E(T_1) + (1 - a) \cdot E(T_2)$$

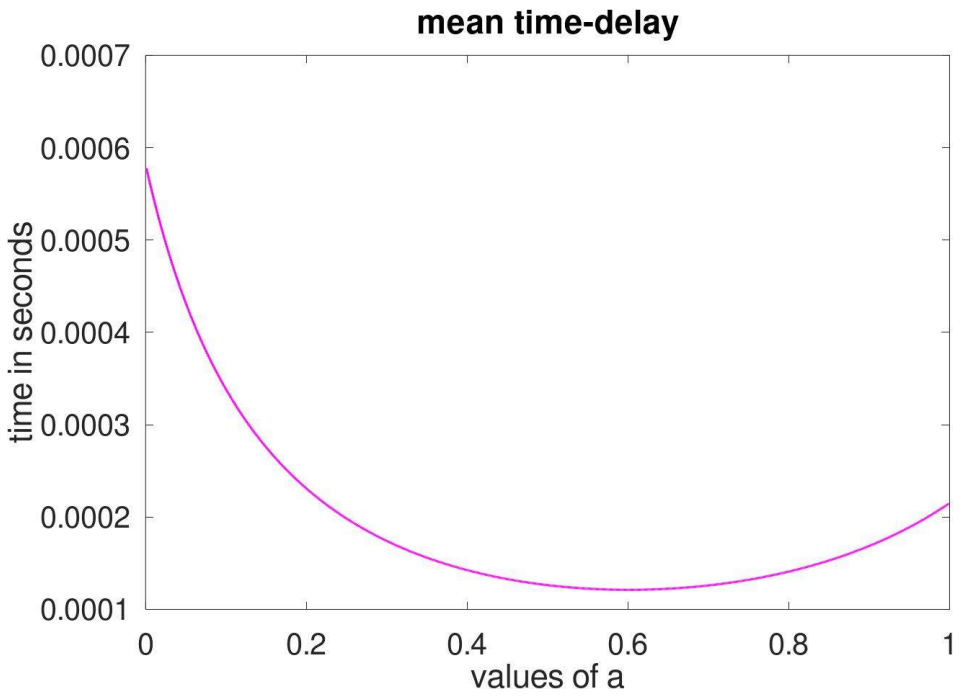
Μπορούμε, να υπολογίσουμε τον μέσο χρόνο καθυστέρησης από το Octave με κάποιον από τους εξής τρόπους χρησιμοποιώντας το queueing package:

$$E(T) = \frac{E(n_1) + E(n_2)}{\lambda} = a \cdot E(T_1) + (1 - a) \cdot E(T_2)$$

Κώδικας:

```
clc;
clear all;
close all;

pkg load queueing

a = 0.001:0.001:0.999;
lambda = 10000;

mu1 = (15 * 10^6) / (128 * 8);
mu2 = (12 * 10^6) / (128 * 8);

lambda1 = a * lambda;
lambda2 = lambda - lambda1;

%system M/M/1
%[U, R, Q, X, p0] = qsmm1(lambda, mu)
%pk = qsmm1(lambda, mu, k)
%lambda is the arrival rate
%mu is the service rate
%k is the number of requests in the system (including the one being
served)

%U is the server utilization
%R server response time
%Q average number of requests in the system
%X sever throughput. If the system is ergodic (mu > lambda) we always
have
%X = lambda
%p0 steady-state probability that there no requests in the system
%pk steady-state probability that there are k requests in the system
%(including the one being served)
```

```
%if the function is called without k, then lambda and mu can be
vectors of the
%same size

[U1, R1, Q1, X1, p01] = qsmm1(lambda1, mu1);
[U2, R2, Q2, X2, p02] = qsmm1(lambda2, mu2);


R = (Q1 + Q2)/lambda;

[minR,position] = min(R);
display(["minimum delay is\nE(T) = ",num2str(1000 * minR),...
         " milliseconds\nwhen a = ",num2str(position*0.001)]);

%used in the alternative way of solving this problem, but we won't
use it now
%R_second = a.*R1 + (1 - a).*R2;

figure(1);
plot(a,R,'m','linewidth',1.5);
title("mean time-delay");
xlabel("values of a");
ylabel("time in seconds");
set(gca, "fontsize", 20);

%alternative method
%figure(2);
%plot(a,R_second,'m');
```

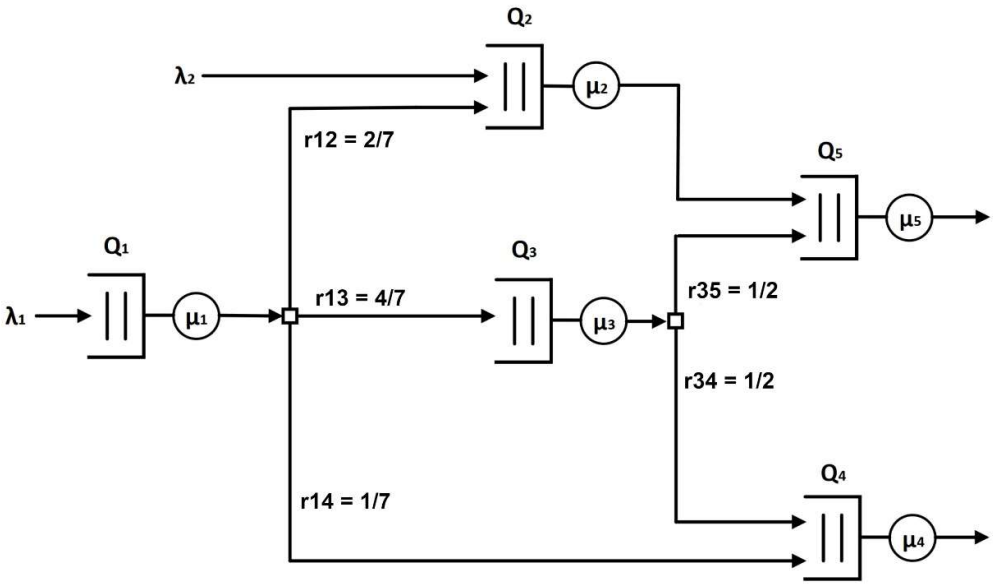


Έξοδος:

```
minimum delay is
E(T) = 0.1212 milliseconds
when a = 0.601
```

Ανοιχτό δίκτυο ουρών αναμονής

(1)



Παραδοχές:

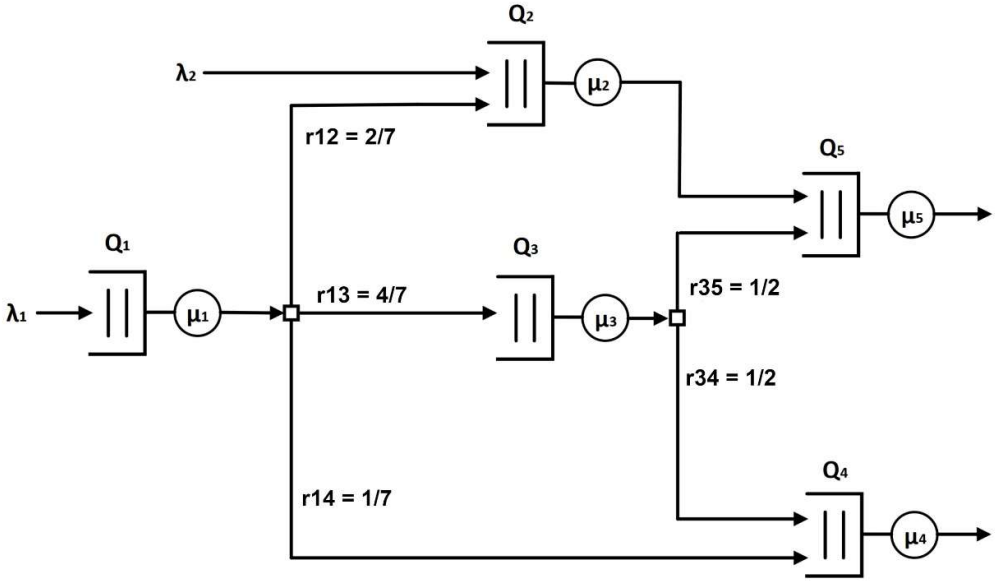
- Η κάθε ουρά αναμονής $Q_i, i = 1,2,3,4,5$ αποτελεί έναν δικτυακό κόμβο εξυπηρέτησης κορμού με εκθετικούς ρυθμούς εξυπηρέτησης $\mu_i, i = 1,2,3,4,5$. Η παραδοχή αυτή ισχύει βάση εκφώνησης.
- Οι ροές μεταξύ των δικτυακών κόμβων είναι ανεξάρτητες ροές Poisson με μέσους ρυθμούς $\gamma_{ij}, i,j = 1,2,3,4,5$ και η συνολική εξωγενής ροή Poisson στην ουρά Qi είναι ίση με $\gamma_i = \sum_{j=1,j\neq i}^{5} \gamma_{ij}$.
- Οι διασπάσεις να γίνονται με τυχαίο τρόπο (κατ' αναλογία των πειραμάτων Bernoulli που αναφέραμε στη προηγούμενη άσκηση). Εδώ έχουμε 2 διασπάσεις, μία μετά τον κόμβο Q1 και μία μετά τον κόμβο Q3. (Η εσωτερική δρομολόγηση από τον κόμβο i στον κόμβο j με πιθανότητα rij).
- Ο κόμβος εξυπηρέτησης Qj διαπερνάται από ροές με συνολικό μέσο ρυθμό

$$\lambda_j = \gamma_j + \sum_{i=1,i\neq j}^{5} r_{ij}\,\lambda_i$$

- Έλλειψη μνήμης, δηλαδή, οι χρόνοι εξυπηρέτησης των πελατών καθώς διαπερνούν το δίκτυο δεν διατηρούν την τιμή τους, αλλά αποκτούν χρόνο εξυπηρέτησης ανάλογα με την κατανομή του κάθε εξυπηρετητή.

Για τα παρακάτω ερωτήματα ο κώδικας είναι στο τέλος.

(2) Μεταφέρουμε το σχήμα και εδώ:



Προσδιορισμός εντάσεων φορτίων:

$$\rho_1 = \frac{\lambda_1}{\mu_1}$$

$$\rho_2 = \frac{r_{12}\lambda_1 + \lambda_2}{\mu_2} = \frac{\frac{2}{7}\lambda_1 + \lambda_2}{\mu_2}$$

$$\rho_3 = \frac{r_{13}\lambda_1}{\mu_3} = \frac{\frac{4}{7}\lambda_1}{\mu_3}$$

$$\rho_4 = \frac{(r_{14} + r_{13}r_{34})\lambda_1}{\mu_4} = \frac{(\frac{1}{7} + \frac{2}{7})\lambda_1}{\mu_4} = \frac{\frac{3}{7}\lambda_1}{\mu_4}$$

$$\rho_5 = \frac{(r_{12} + r_{13}r_{35})\lambda_1 + \lambda_2}{\mu_5} = \frac{(\frac{2}{7} + \frac{2}{7})\lambda_1 + \lambda_2}{\mu_5} = \frac{\frac{4}{7}\lambda_1 + \lambda_2}{\mu_5}$$

Τα παραπάνω προκύπτουν διότι σε όλους τους κόμβους έχουμε ουρές M/M/1 και χρησιμοποιούμε την ιδιότητα Markov implies Markov ή ότι η έξοδος μιας ουράς M/M/1 ακολουθεί κατανομή Poisson με ρυθμό ίδιο με το ρυθμό εισόδου σε αυτήν.

(3)

Για τους μέσους αριθμούς πελατών κάθε ουράς χρησιμοποιούμε τον τύπο:

$$E(n_i) = \frac{\rho_i}{1 - \rho_i}, i = 1, 2, 3, 4, 5$$

(4)

Δείχνουμε τον κώδικα (έχει και των κώδικα των ερωτημάτων (2) και (3) μέσα):

```
clc;
clear all;
close all;

pkg load queueing

%(2)
function [r, ergodicity] = intensities(lambda, mu)
  %array of ratios given by the problem
  r_ij = [0 2/7 4/7 1/7 0; 2/7 0 0 0 0; 4/7 0 0 1/2 1/2;...
          1/7 0 1/2 0 0; 0 0 1/2 0 0];
  r(1) = lambda(1)/mu(1);
  r(2) = (r_ij(1,2)*lambda(1) + lambda(2))/mu(2);
  r(3) = (r_ij(1,3)*lambda(1))/mu(3);
  r(4) = (r_ij(1,4) + r_ij(1,3)*r_ij(3,4))*lambda(1)/mu(4);
  r(5) = ((r_ij(1,2)+r_ij(1,3)*r_ij(3,5))*lambda(1) +
lambda(2))/mu(5);
  ergodicity = true;
  for i=1:1:5
    display(["rho(",num2str(i),") = ",num2str(r(i))]);
    ergodicity = ergodicity && (r(i) < 1);
  endfor
  if ergodicity
    display("Is ergodic");
  else
    display("Is not ergodic");
  endif
endfunction

%(3)
function Q = mean_clients(lambda, mu)
  [rho, erg] = intensities(lambda, mu);
  Q = rho ./ (1 - rho);
endfunction

%(4)
lambda = [4 1];
mu = [6 5 8 7 6];
Q = mean_clients(lambda,mu);

%we have to calculate mean delay time for a client in the system

display("");
display(["mean delay of a client: E(T) = ",...
         num2str(sum(Q(:))/sum(lambda(:))),' seconds']);
```

Έξοδος:

```
rho(1) = 0.66667
rho(2) = 0.42857
rho(3) = 0.28571
rho(4) = 0.2449
rho(5) = 0.54762
Is ergodic

mean delay of a client: E(T) = 0.93697 seconds
```

(5) Η μέγιστη ένταση φορτίου σημειώνεται στην ουρά Q1.

Με βάση την ουρά Q1 έχουμε max_λ1 = max_ρ1 * μ1 = 1 * 6 = 6 πελάτες/sec.

Θέλουμε όλες οι ουρές να είναι εργοδικές οπότε δεν αρκεί να ελέγξουμε μόνο για την ουρά Q1. Ο παρακάτω κώδικας αυξάνει βηματικά το λ1 μέχρι κάποια από τις ουρές να σταματήσει να είναι εργοδική ($\rho_i \geq 1$ *για κάποιο* $i = 1,2,3,4,5$).

Κώδικας:

```
%(5)
function [r, ergodicity] = intensities2(lambda, mu)
  %array of ratios given by the problem
  r_ij = [0 2/7 4/7 1/7 0; 2/7 0 0 0 0; 4/7 0 0 1/2 1/2;...
          1/7 0 1/2 0 0; 0 0 1/2 0 0];
  r(1) = lambda(1)/mu(1);
  r(2) = (r_ij(1,2)*lambda(1) + lambda(2))/mu(2);
  r(3) = (r_ij(1,3)*lambda(1))/mu(3);
  r(4) = (r_ij(1,4) + r_ij(1,3)*r_ij(3,4))*lambda(1)/mu(4);
  r(5) = ((r_ij(1,2)+r_ij(1,3)*r_ij(3,5))*lambda(1) +
lambda(2))/mu(5);
  ergodicity = true;
  for i=1:1:5
    ergodicity = ergodicity && (r(i) < 1);
  endfor
endfunction

display("");
step = 0.01;
for lambda1 = 4:step:10
  lambda = [lambda1 1];
  mu = [6 5 8 7 6];
  [r, e] = intensities2(lambda,mu);
  if e == 0
    display(["system is not ergodic for lambda1 =
",num2str(lambda1)]);
    break;
  endif
endfor
```

Έξοδος:

```
system is not ergodic for lambda1 = 6
```

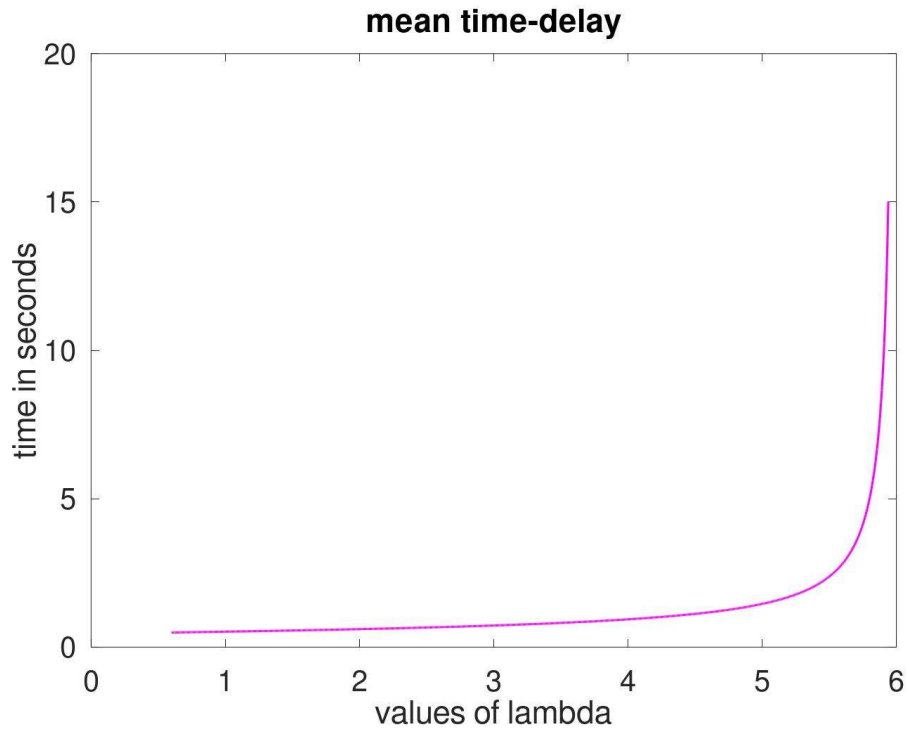
Επομένως, η τιμή 6 είναι δεκτή.

(6) Η μέγιστη τιμή του λ1 είναι 6. Θέλουμε για λ1 από 0.1 έως 0.99 της μέγιστης τιμής, δηλαδή, από 0.6 έως 5.94.

Κώδικας:

```
%(6)
%define function without the outputs
function Q = mean_clients2(lambda, mu)
  [rho, erg] = intensities2(lambda, mu);
  Q = rho ./ (1 - rho);
endfunction

ET(1) = 0;
for k = 100:1:990
  lambda1 = max_lambda*k/1000;
  lambda = [lambda1 1];
  mu = [6 5 8 7 6];
  Q = mean_clients2(lambda, mu);
  ET(k - 99) = sum(Q(:))/sum(lambda(:));
endfor

figure(1);
plot(((100:1:990).*max_lambda/1000),ET, 'm','linewidth',1.5);
title("mean time-delay");
xlabel("values of lambda");
ylabel("time in seconds");
set(gca, "fontsize", 20);
```

Έξοδος:



Τέλος δείχνουμε τον συνολικό κώδικα:

```
clc;
clear all;
close all;

pkg load queueing

%(2)
function [r, ergodicity] = intensities(lambda, mu)
  %array of ratios given by the problem
  r_ij = [0 2/7 4/7 1/7 0; 2/7 0 0 0 0; 4/7 0 0 1/2 1/2;...
          1/7 0 1/2 0 0; 0 0 1/2 0 0];
  r(1) = lambda(1)/mu(1);
  r(2) = (r_ij(1,2)*lambda(1) + lambda(2))/mu(2);
  r(3) = (r_ij(1,3)*lambda(1))/mu(3);
  r(4) = (r_ij(1,4) + r_ij(1,3)*r_ij(3,4))*lambda(1)/mu(4);
  r(5) = ((r_ij(1,2)+r_ij(1,3)*r_ij(3,5))*lambda(1) +
lambda(2))/mu(5);
  ergodicity = true;
  for i=1:1:5
    display(["rho(",num2str(i),") = ",num2str(r(i))]);
    ergodicity = ergodicity && (r(i) < 1);
  endfor
  if ergodicity
    display("Is ergodic");
  else
    display("Is not ergodic");
  endif
endfunction

%(3)
function Q = mean_clients(lambda, mu)
  [rho, erg] = intensities(lambda, mu);
  Q = rho ./ (1 - rho);
endfunction

%(4)
lambda = [4 1];
mu = [6 5 8 7 6];
Q = mean_clients(lambda,mu);

%we have to calculate mean delay time for a client in the system

display("");
display(["mean delay of a client: E(T) = ",...
          num2str(sum(Q(:))/sum(lambda(:))),' seconds']);
```

```
%(5)
%define function without the outputs
function [r, ergodicity] = intensities2(lambda, mu)
  %array of ratios given by the problem
  r_ij = [0 2/7 4/7 1/7 0; 2/7 0 0 0 0; 4/7 0 0 1/2 1/2;...
          1/7 0 1/2 0 0; 0 0 1/2 0 0];
  r(1) = lambda(1)/mu(1);
  r(2) = (r_ij(1,2)*lambda(1) + lambda(2))/mu(2);
  r(3) = (r_ij(1,3)*lambda(1))/mu(3);
  r(4) = (r_ij(1,4) + r_ij(1,3)*r_ij(3,4))*lambda(1)/mu(4);
  r(5) = ((r_ij(1,2)+r_ij(1,3)*r_ij(3,5))*lambda(1) +
lambda(2))/mu(5);
  ergodicity = true;
  for i=1:1:5
    ergodicity = ergodicity && (r(i) < 1);
  endfor
endfunction

display("");
step = 0.01;
max_lambda = 0;
for lambda1 = 4:step:10
  lambda = [lambda1 1];
  mu = [6 5 8 7 6];
  [r, e] = intensities2(lambda,mu);
  if e == 0
    max_lambda = lambda1;
    display(["system is not ergodic for lambda1 =
",num2str(max_lambda)]);
    break;
  endif
endfor

%(6)
%define function without the outputs
function Q = mean_clients2(lambda, mu)
  [rho, erg] = intensities2(lambda, mu);
  Q = rho ./ (1 - rho);
endfunction

ET(1) = 0;
for k = 100:1:990
  lambda1 = max_lambda*k/1000;
  lambda = [lambda1 1];
  mu = [6 5 8 7 6];
  Q = mean_clients2(lambda, mu);
  ET(k - 99) = sum(Q(:))/sum(lambda(:));
endfor

figure(1);
plot(((100:1:990).*max_lambda/1000),ET, 'm','linewidth',1.5);
title("mean time-delay");
xlabel("values of lambda");
ylabel("time in seconds");
set(gca, "fontsize", 20);
```