Συστήματα Αναμονής

# 6η Εργαστηριακή Άσκηση

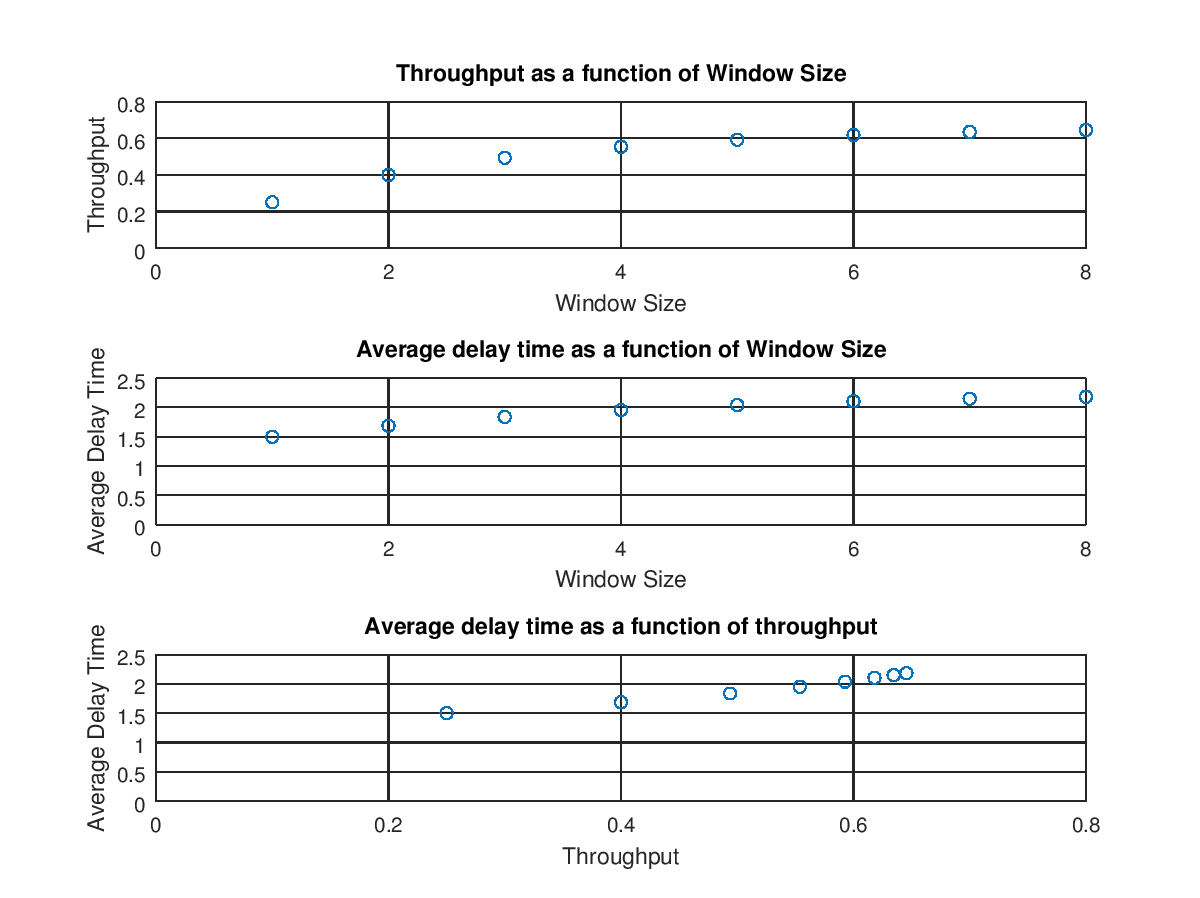
**Όνομα:** Σταύρος Σταύρου

**ΑΜ:** 03115701

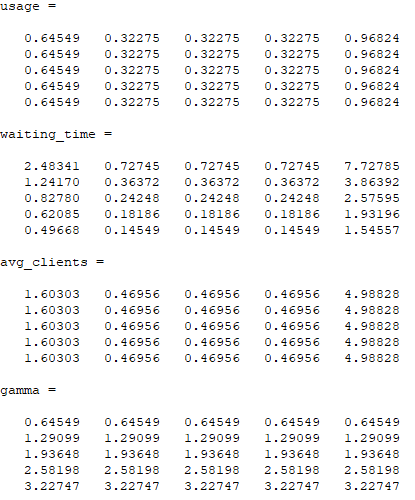
**Εξάμηνο:** 6ο-ΣΗΜΜΥ

**Μηχανισμός ελέγχου ροής παραθύρου:**

1. Η Octave δίνει τα εξής διαγράμματα για τα μεγέθη που ζητούνται:



1. Η Octave δίνει τα εξής (θεώρησα σταθερό μέγεθος παραθύρου W = 8):



Παρατηρούμε πως ο μέσος αριθμός πελατών και χρησιμοποίησης του κάθε υποσυστήματος παραμένει σταθερός συναρτήσει των διαφόρων k. Αυτό φαίνεται λογικό καθώς οι εξισώσεις μετάβασης κατάστασης που παίρνουμε για τα διάφορα k είναι γραμμικά εξαρτημένες με την περίπτωση k=1 και ως εκ τούτου παίρνουμε τα ίδια αποτελέσματα.

Επίσης, παρατηρούμε πως η ρυθμαπόδοση είναι ίση μεταξύ των υποσυστημάτων (λογικό αφού το σύστημα έχει κοινή ρυθμαπόδοση) και όπως είναι λογικό αυξάνεται με τη μείωση των χρόνων εξυπηρέτησης, αφού πλέον στον ίδιο χρόνο εξυπηρετούμε περισσότερα πακέτα.

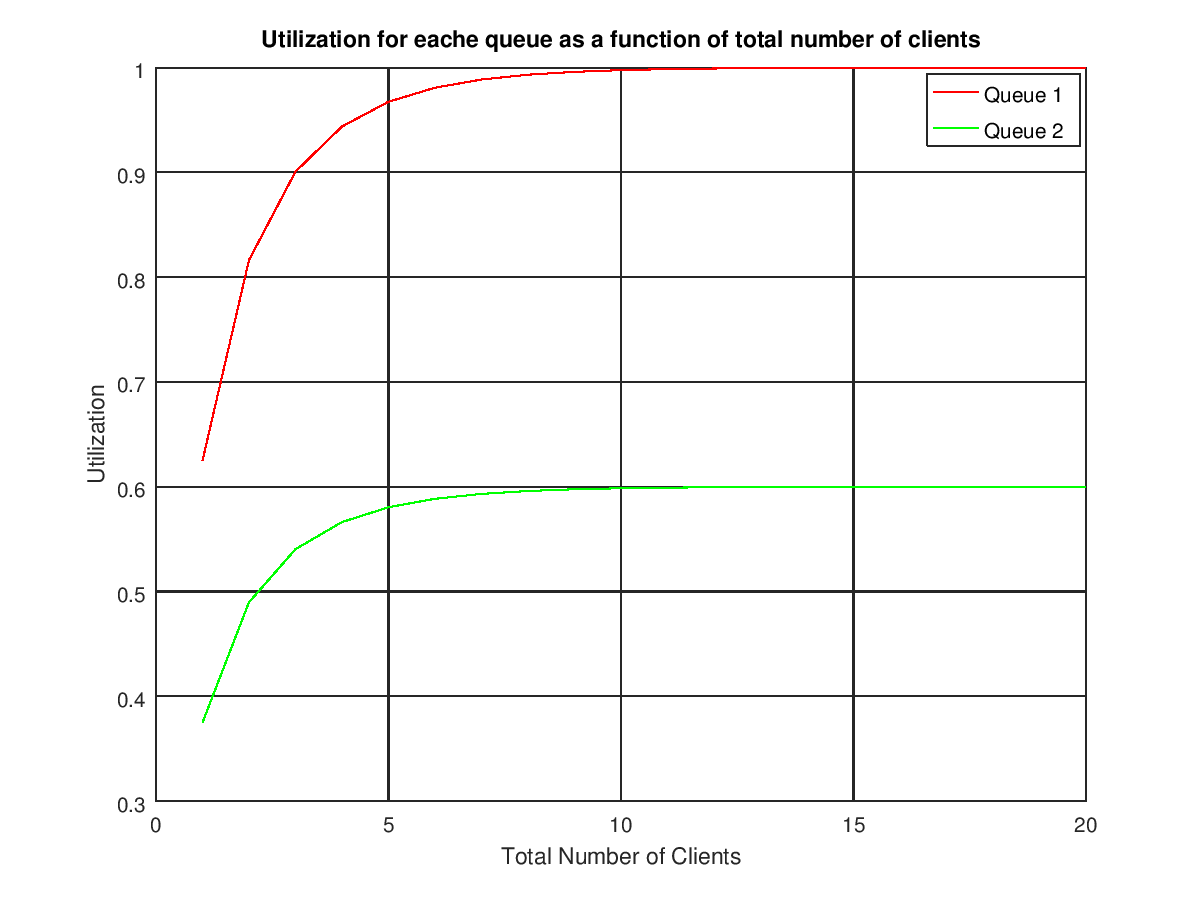
Τέλος, επίσης λόγω της μείωσης των χρόνων εξυπηρέτησης κάθε υποσυστήματος, βλέπουμε και μείωση στο χρόνο καθυστέρησης κάθε υποσυστήματος, αποτέλεσμα αναμενόμενο.

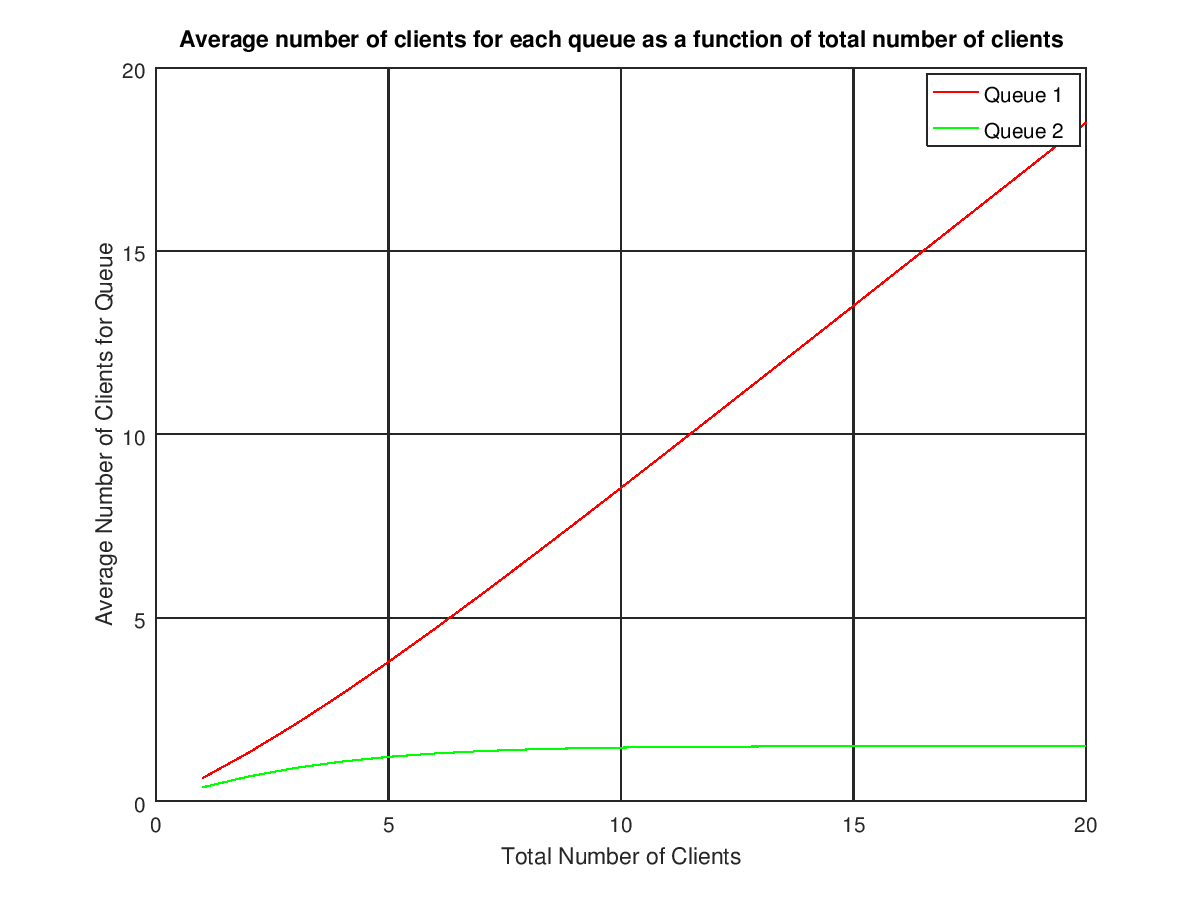
**Ο Αλγόριθμος του Buzen:**

1. Έχουμε . Επίσης :
   * 🡪 🡪
   * 🡪

Βλέπουμε πως οι 2 εξισώσεις δίνουν την ίδια σχέση. Συνεπώς θέτουμε «αναγκαστικά» , από το οποίο λαμβάνουμε .

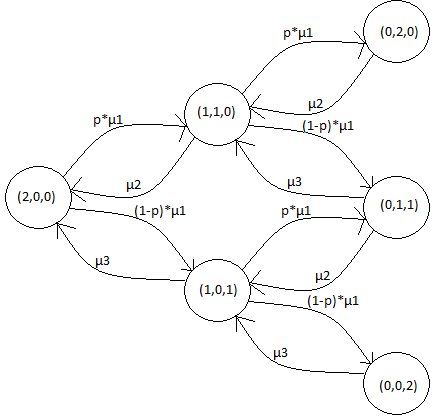
1. Ο κώδικας στο παράρτημα. Η συνάρτηση επιστρέφει όλη την τελευταία στήλη του πίνακα ώστε οι τιμές αυτές να χρησιμοποιηθούν για το ερώτημα 3.
2. Η Octave δίνει τις εξής γραφικές παραστάσεις με τη συνάρτηση που γράψαμε:



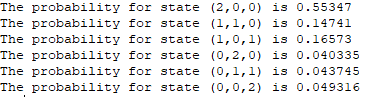


**Προσομοίωση σε κλειστό δίκτυο εκθετικών ουρών αναμονής:**

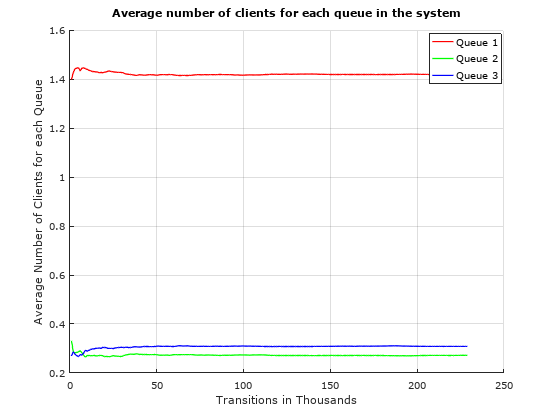
Έχουμε το εξής διάγραμμα μεταβάσεων:



1. Παίρνουμε τα εξής από την Octave:



1. Παίρνουμε την εξής γραφική παράσταση από την Octave:



Παρατηρούμε, πως το άθροισμα των μέσων όρων είναι ίσο με 2, γιατί τόσους πελάτες έχουμε στο σύστημα. Το άθροισμα πρέπει πάντα να είναι ακριβώς 2.

Όλοι οι κώδικες που χρησιμοποιήθηκαν για την άσκηση ακολουθούν στο παράρτημα.

**ΠΑΡΑΡΤΗΜΑ**

**windows.m:**

clc;

clear all;

close all;

Prob = [0 1 0 0 0; 0 0 1 0 0; 0 0 0 1 0; 0 0 0 0 1; 1 0 0 0 0];

V = qncsvisits(Prob);

S = [1 0.5 0.5 0.5 3/2];

m = [1 1 1 1 1];

gamma = zeros(1,8);

tDS = zeros(1,8);

for n = 1:1:8

[U R Q X] = qnclosed(n, S, V, m);

gamma(n) = mean(X);

tDS(n) = (Q(2) + Q(3) + Q(4)) / gamma(n);

endfor

w = 1:1:8;

figure(1);

subplot(3, 1, 1);

scatter(w, gamma);

grid on;

title("Throughput as a function of Window Size");

xlabel("Window Size");

ylabel("Throughput");

xlim([0 8]);

ylim([0 0.8]);

subplot(3, 1, 2);

scatter(w, tDS);

grid on;

title("Average delay time as a function of Window Size");

xlabel("Window Size");

ylabel("Average Delay Time");

xlim([0 8]);

ylim([0 2.5]);

subplot(3, 1, 3);

scatter(gamma, tDS);

grid on;

title("Average delay time as a function of throughput");

xlabel("Throughput");

ylabel("Average Delay Time");

xlim([0 0.8]);

ylim([0 2.5]);

**windows\_diff\_times.m:**

clc;

clear all;

close all;

Prob = [0 1 0 0 0; 0 0 1 0 0; 0 0 0 1 0; 0 0 0 0 1; 1 0 0 0 0];

V = qncsvisits(Prob);

S = [1 0.5 0.5 0.5 3/2];

m = [1 1 1 1 1];

usage = zeros(5,5);

waiting\_time = zeros(5,5);

avg\_clients = zeros(5,5);

gamma = zeros(5,5);

for k = 1:1:5

myS = (1 / k) \* S;

[usage(k,:) waiting\_time(k,:) avg\_clients(k,:) gamma(k,:)] = qnclosed(8, myS, V, m);

endfor

display(usage);

display(waiting\_time);

display(avg\_clients);

display(gamma);

**buzen.m:**

function [G Gconst] = buzen(N, M, X )

if ( nargin != 3)

print\_usage();

endif

gamma = zeros(N+1, M+1);

gamma(1, :) = ones(1, M+1);

gamma(:, 1) = 0:1:N;

for i = 2:1:N+1

gamma(i, 2) = X(1) \*\* (i-1);

endfor

for i=2:1:N+1

for j=3:1:M+1

gamma(i,j) = gamma(i,j-1) + X(j-1) \* gamma(i-1, j);

endfor

endfor

Gconst = gamma(N+1, M+1);

G = gamma(:, M+1);

endfunction

**buzen\_example.m:**

clc;

clear all;

close all;

X = [1 0.6];

M = 2;

util1 = zeros(1,20);

util2 = zeros(1,20);

clients1 = zeros(1,20);

clients2 = zeros(1,20);

for N = 1:1:20

[A B] = buzen(N, M, X);

util1(N) = X(1) \* A(N) / A(N+1);

util2(N) = X(2) \* A(N) / A(N+1);

for k = 1:1:N

clients1(N) = clients1(N) + X(1) \*\* k \* A(N+1-k) / A(N+1);

clients2(N) = clients2(N) + X(2) \*\* k \* A(N+1-k) / A(N+1);

endfor

endfor

N = 1:1:20;

figure(1);

hold on;

plot(N, util1, 'r');

plot(N, util2, 'g');

hold off;

grid on;

title("Utilization for eache queue as a function of total number of clients");

xlabel("Total Number of Clients");

ylabel("Utilization");

legend("Queue 1" , "Queue 2");

legend("show");

figure(2);

hold on;

plot(N, clients1, 'r');

plot(N, clients2, 'g');

hold off;

grid on;

title("Average number of clients for each queue as a function of total number of clients");

xlabel("Total Number of Clients");

ylabel("Average Number of Clients for Queue");

legend("Queue 1", "Queue 2");

legend("show");

**closed.m:**

clc;

clear all;

close all;

mu1 = 2;

mu2 = 3;

mu3 = 4;

p = 0.4;

arrivals(200) = 0;

arrivals(110) = 0;

arrivals(101) = 0;

arrivals(20) = 0;

arrivals(11) = 0;

arrivals(2) = 0;

total\_arrivals = 0;

times = 0;

steps = 0;

sigklisi = false;

threshold0 = p\*mu1/ (mu1+ mu2 + mu3);

threshold1 = mu1 / (mu1 + mu2 + mu3);

threshold2 = (mu1 + mu2) / (mu1 + mu2 + mu3);

prev\_mean1 = 0;

prev\_mean2 = 0;

prev\_mean3 = 0;

current\_state = 200;

while !sigklisi && steps <= 300000

steps = steps + 1;

arrivals(current\_state) = arrivals(current\_state) + 1;

total\_arrivals = total\_arrivals + 1;

decision = rand(1);

if (current\_state == 200 && decision < threshold0)

current\_state = 110;

elseif (current\_state == 200 && decision > threshold0 && decision < threshold1)

current\_state = 101;

elseif (current\_state == 110 && decision < threshold0)

current\_state = 20;

elseif (current\_state == 110 && decision > threshold0 && decision < threshold1)

current\_state = 11;

elseif (current\_state == 110 && decision < threshold2 && decision > threshold1)

current\_state = 200;

elseif (current\_state == 101 && decision < threshold0)

current\_state = 11;

elseif (current\_state == 101 && decision > threshold0 && decision < threshold1)

current\_state = 2;

elseif (current\_state == 101 && decision > threshold2)

current\_state = 200;

elseif (current\_state == 20)

current\_state = 110;

elseif (current\_state == 11 && decision > threshold2)

current\_state = 110;

elseif (current\_state == 11 && decision > threshold1 && decision < threshold2)

current\_state = 101;

elseif (current\_state == 2)

current\_state = 101;

else

arrivals(current\_state) = arrivals(current\_state) - 1;

total\_arrivals = total\_arrivals - 1;

endif

if mod(steps, 1000) == 0

times = times + 1;

T200 = 1 / mu1 \* arrivals(200);

T110 = 1 / (mu1 + mu2) \* arrivals(110);

T101 = 1 / (mu1 + mu3) \* arrivals(101);

T20 = 1 / mu2 \* arrivals(20);

T11 = 1 / (mu2 + mu3) \* arrivals(11);

T2 = 1 / mu3 \* arrivals(2);

total\_time = T200 + T110 + T101 + T20 + T11 + T2;

P(2) = T2 / total\_time;

P(11) = T11 / total\_time;

P(20) = T20 / total\_time;

P(101) = T101 / total\_time;

P(110) = T110 / total\_time;

P(200) = T200 / total\_time;

current\_mean1 = P(101) + P(110) + 2 \* P(200);

current\_mean2 = P(11) + 2 \* P(20) + P(110);

current\_mean3 = 2 \* P(2) + P(11) + P(101);

clients1(times) = current\_mean1;

clients2(times) = current\_mean2;

clients3(times) = current\_mean3;

if abs(current\_mean1 - prev\_mean1) < 0.00001 && abs(current\_mean2 - prev\_mean2) < 0.00001 && abs(current\_mean3 - prev\_mean3) < 0.00001

sigklisi = true;

endif

prev\_mean1 = current\_mean1;

prev\_mean2 = current\_mean2;

prev\_mean3 = current\_mean3;

endif

endwhile

total\_steps = 1:1:times;

disp(cstrcat("The probability for state (2,0,0) is ", num2str(P(200))));

disp(cstrcat("The probability for state (1,1,0) is ", num2str(P(110))));

disp(cstrcat("The probability for state (1,0,1) is ", num2str(P(101))));

disp(cstrcat("The probability for state (0,2,0) is ", num2str(P(20))));

disp(cstrcat("The probability for state (0,1,1) is ", num2str(P(11))));

disp(cstrcat("The probability for state (0,0,2) is ", num2str(P(2))));

figure(1);

hold on;

plot(total\_steps, clients1, 'r');

plot(total\_steps, clients2, 'g');

plot(total\_steps, clients3, 'b');

hold off;

title("Average number of clients for each queue in the system");

xlabel("Transitions in Thousands");

ylabel("Average Number of Clients for each Queue");

legend("Queue 1", "Queue 2", "Queue 3");

legend("show");