Εθνικό Μετσόβιο Πολυτεχνείο

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Συστήματα Αναμονής

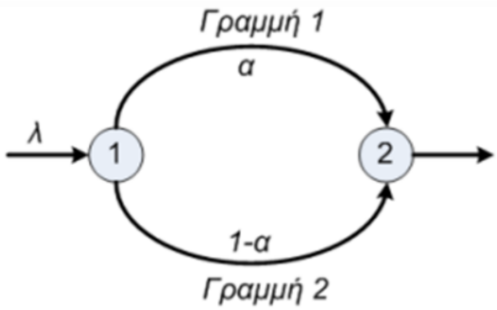
5^{η} ομάδα ασκήσεων

Ελευθερία Αρκαδοπούλου

el19442

Δίκτυο με εναλλακτική δρομολόγηση

Θεωρούμε το δίκτυο του σχήματος:



Με $\lambda=10*10^3$, μήκος πακέτου 128 bytes, c_1=15 Mbps, c_2=12 Mbps.

1. Για να μπορούν να μοντελοποιηθούν οι σύνδεσμοι σαν M/M/1 ουρές, θα πρέπει οι τυχαίες εξωτερικές αφίξεις να ακολουθούν κατανομή Poisson (2 διαδικασίες α*λ και (1-α)*λ αντίστοιχα), και οι χρόνοι μετάδοσης και αφίξεων λ, μ να είναι εκθετικά κατανεμημένοι.
   Γραμμή 1: $\mu_1 = \frac{15*10^6}{128*8} = 14.65 * 10^3 \; \kappa\alpha\iota \; \rho_1 = \frac{\lambda_1}{\mu_1} = 0.682 * \alpha$

   Γραμμή 2: $\mu_2 = \frac{12*10^6}{128*8} = 11.72 * 10^3 \; \kappa\alpha\iota \; \rho_2 = \frac{\lambda_2}{\mu_2} = 0.852 * (1-\alpha)$
   Εφόσον ρ_1, $\rho_2 < 1$ οι ουρές είναι εργοδικές.

2. Η τιμή του α που ελαχιστοποιεί τον χρόνο καθυστέρησης καθώς και ο ελάχιστος χρόνος καθυστέρησης:

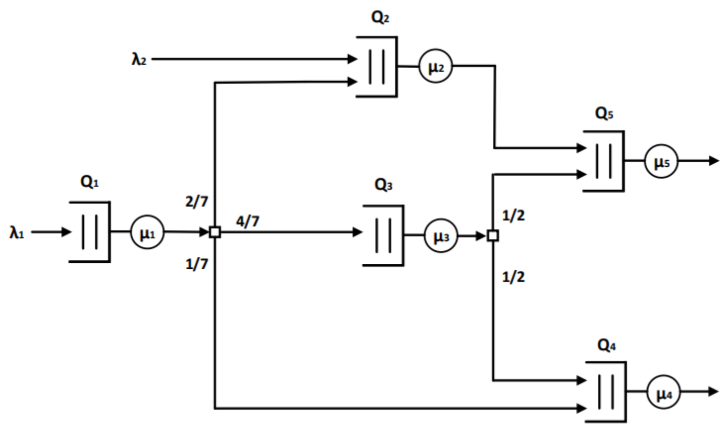
```
a for minimum waiting time =
 0.60100
Minimum waiting time =
Emin =  0.00012118
```

   Ο κώδικας:

```
pkg load queueing

clear all;
close all;
clc;

a = 0.001:0.001:0.999;
lambda = 10000;
mu1 = 14650;
mu2 = 11720;
lambda1 = a.*lambda;
lambda2 = (1-a).*lambda;

[A1 B1 C1 D1 E1] = qsmm1(lambda1,mu1); %%apo ergastiriaki 2
[A2 B2 C2 D2 E2] = qsmm1(lambda2,mu2);

R = a.*B1 + (1-a).*B2;

plot(a,R);
[Emin, position] = min(R);
display("a for minimum waiting time = ");
display(position/1000);
display("Minimum waiting time = ");
display(Emin);
```

Ανοιχτό δίκτυο ουρών αναμονής

Το ανοιχτό δίκτυο ουρών αναμονής:



1. Οι απαραίτητες παραδοχές ώστε το παραπάνω να μπορεί να μελετηθεί ως ένα ανοιχτό δίκτυο με το θεώρημα Jackson:
    - Οι ρυθμοί εξυπηρέτησης μ_i πρέπει να είναι εκθετικοί
    - Οι εξωτερικές αφίξεις σε κόμβους i πρέπει να είναι ανεξάρτητες Poisson με μέσο ρυθμό γ_i
    - Η εσωτερική δρομολόγηση πρέπει να γίνεται με τυχαίο τρόπο και πιθανότητα δρομολόγησης πελάτη από τον κόμβο Q_i στον Q_j ίση με r_{ji} (όπως φαίνεται στο σχήμα)
    - Οι χρόνοι εξυπηρέτησης πελατών στις ουρές έχουν την ιδιότητα έλλειψης μνήμης, και άρα οι χρόνοι εξυπηρέτησης πελατών υπολογίζονται ανάλογα με την κατανομή του κάθε εξυπηρετητή

2. Υπολογίζουμε την ένταση του φορτίου που δέχεται η κάθε ουρά του δικτύου:

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

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

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

$$\rho_4 = \frac{r_{34} * r_{13} * \lambda_1 + \lambda_1 * r_{14}}{\mu_5} = \frac{\frac{3}{7} * \lambda_1}{\mu_4}$$

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

```
function [r,flag_ergodic] = intensities(lambda,mu)
  r(1)=lambda(1)/mu(1);
  r(2)=(lambda(2)+lambda(1)*2/7)/mu(2);
  r(3)=(4/7)*lambda(1)/mu(3);
  r(4)=(3/7)*lambda(1)/mu(4);
  r(5)=(lambda(2)+lambda(1)*4/7)/mu(5);
  flag_ergodic = 1;
  for i=1:5
  printf("r(%d)=%d\n",i,r(i));
    if (r(i) >= 1)
      flag_ergodic=0;
    endif
  endfor
  disp(flag_ergodic);
endfunction
```

3.

```
function [R] = mean_clients(lambda,mu)
  [r,flag_ergodic]=intensities(lambda, mu);
  R = r ./ (1-r);
endfunction
```

4.

```
i =  1
and r(i) =
 0.66667
i =  2
and r(i) =
 0.42857
i =  3
and r(i) =
 0.28571
i =  4
and r(i) =
 0.24490
i =  5
and r(i) =
 0.54762
 1
Average delay time is
ET =   0.93697
```

Ο κώδικας (συμπεριλαμβάνεται στο ίδιο αρχείο .m μαζί με τις intensities, mean_clients):

```
lambda=[4,1];
mu=[6,5,8,7,6];

r = intensities(lambda,mu);
R = mean_clients(lambda,mu);
ET = sum(R)/sum(lambda);
display("Average delay time is ");
display(ET);
```

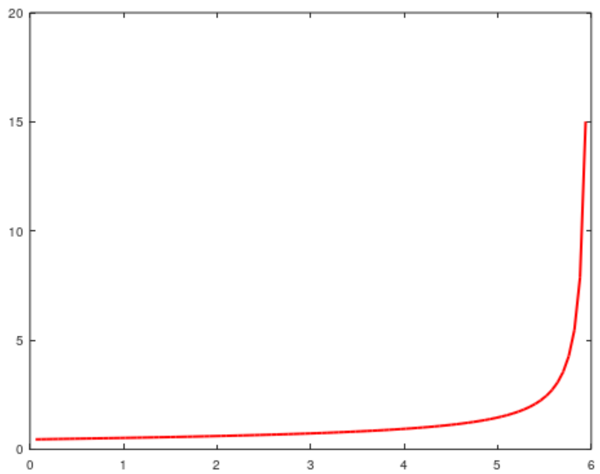
5.

```
Bottleneck queue is that of
maxposition =  1
max lambda is
max_lambda =  6
```

Ο κώδικας:

```
[bottleneck, maxposition] = max(R);
max_lambda = mu(maxposition);
display("Bottleneck queue is that of");
display(maxposition);
display("max lambda is");
display(max_lambda);
```

6.



Ο κώδικας:

```
for i=1:99
  new_lambda = max_lambda*i/100;
  saved(i) = new_lambda;
  lambda = [new_lambda,1];
  ET(i) = sum(mean_clients(lambda,mu))/sum(lambda);
endfor
figure();
plot(saved,ET,"r","linewidth",2);
```