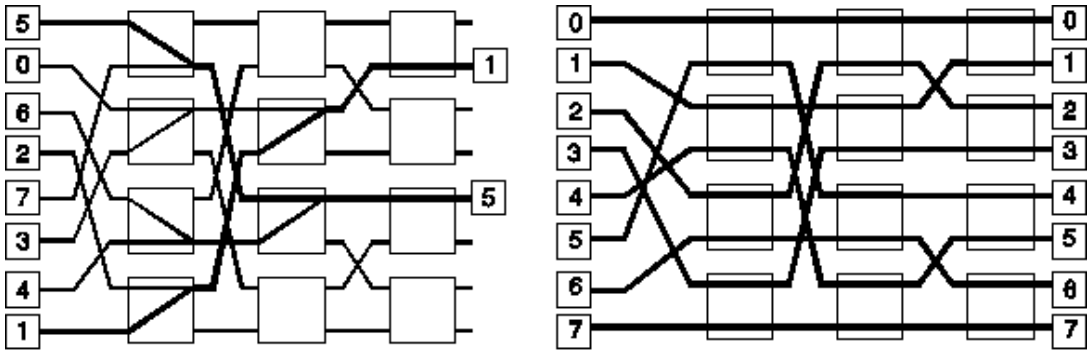
Επίδοση σχημάτων ενταμίευσης κελιών σε μεταγωγείς ATM



Παπαδόπουλος Κωνσταντίνος (8677) konserpap@auth.gr

Πίνακας Περιεχομένων

Ζητούμενα

Οι τεχνικές ενταμίευσης κελιών σε μεταγωγείς ATM χρησιμοποιούνται για την ρύθμιση της πρόσβασης κελιών στις θύρες εξόδου. Οι ενταμιευτές είναι δυνατό να βρίσκονται στις μονάδες εισόδου, στις μονάδες εξόδου ή εσωτερικά στον μεταγωγέα.

Στην παρούσα εργασία ζητείται η μελέτη της περίπτωσης όπου η ενταμίευση συμβαίνει στην έξοδο και η επίδοση του συγκεκριμένου σχήματος. Πιο συγκεκριμένα ζητείται να υπολογισθεί και να παρουσιασθεί σε διαγράμματα:

1) η πιθανότητα απώλειας κελιού σε συνάρτηση με τη χωρητικότητα του ενταμιευτή, για διάφορες τιμές του πλήθους των εξόδων του μεταγωγέα $N=\{2,4,8,16,32,\infty\}$ και για διάφορες τιμές προσφερόμενου φορτίου $p=\{0.8, 0.9\}$. Η χωρητικότητα του ενταμιευτή να μεταβάλλεται από 0 έως 80.

2) η πιθανότητα απώλειας κελιού σε συνάρτηση με την χωρητικότητα του ενταμιευτή, όταν το πλήθος των εξόδων του μεταγωγέα , N, είναι άπειρο και για διάφορες τιμές προσφερόμενου φορτίου $p=\{0.7,0.75, 0.8, 0.85, 0.9, 0.95\}$. Η χωρητικότητα του ενταμιευτή να μεταβάλλεται από 0 έως 50.

3) η μέση καθυστέρηση στην ουρά εξόδου σε σχέση με το προσφερόμενο φορτίο, για άπειρες γραμμές εξόδου και για μεταβαλλόμενη χωρητικότητα $b=\{1,2,4,8,\infty\}$. Το προσφερόμενο φορτίο, p, να μεταβάλλεται από 0 έως 1.

Παραδοχή:

Σε όλες τις περιπτώσεις θα θεωρήσουμε ότι το πλήθος των γραμμών εισόδου είναι ίσο με αυτό των γραμμών εξόδου.

Εισαγωγή

Με την ουρά στην έξοδο, τα κελιά ενταμιεύονται μόνο στις εξόδους, στις οποίες διατηρείται μία ξεχωριστή ουρά FIFO. Κάθε έξοδος μπορεί να δεχτεί μέχρι b κελιά σε κάθε χρονοθυρίδα. Αν το b < N, κάποια κελιά μπορεί να απορριφθούν. Δεν υπάρχει HOL (Head Of Line) blocking, άρα ο μεταγωγέας μπορεί να πετύχει 100% ρυθμαπόδοση.

Θεωρούμε μια συγκεκριμένη ουρά εξόδου. Ορίζουμε την τυχαία μεταβλητή A, που αντιπροσωπεύει τον αριθμό των αφίξεων των κελιών (cell) για την εξεταζόμενη έξοδο σε μία συγκεκριμένη χρονοθυρίδα και χρησιμοποιούμε τη σχέση

$$a_k \triangleq \Pr[A = k] = \binom{N}{k}\left(\frac{p}{N}\right)^k\left(1 - \frac{p}{N}\right)^{N-k}, \qquad k = 0, 1, 2, \ldots N$$

και για N → ∞ η σχέση γίνεται

$$a_k \triangleq \Pr[A = k] = \frac{p^k e^{-p}}{k!}, \qquad k = 0, 1, 2, \ldots .$$

Συμβολίζουμε με Q_m τον αριθμό των κελιών, στην αναφερθείσα ουρά, στο τέλος της m-οστης χρονοθυρίδας και με A_m τον αριθμό των αφίξεων κελιών κατά την διάρκεια της m-οστης χρονοθυρίδας. Έχουμε τη σχέση

$$Q_m = \min\{\max(0, Q_{m-1} + A_m - 1), b\}.$$

Αν το $Q_{m-1} = 0$ και το $A_m > 0$ δεν υπάρχει κελί που να περιμένει στο ξεκίνημα της m-οστης χρονοθυρίδας, αλλά έχουμε A_m κελιά στην άφιξη. Υποθέτουμε ότι ένα από τα κελιά που φθάνει εκπέμπεται αμέσως κατά τη διάρκεια της m-οστης

χρονοθυρίδας (θεωρούμε ότι ένα κελί περνά μέσα από τον μεταγωγέα χωρίς καθυστέρηση).

Για πεπερασμένα N και b, αυτό μπορεί να μοντελοποιηθεί ως μια πεπερασμένη αλυσίδα Markov με πιθανότητες μεταβολής κατάστασης

$$P_{ij} = \begin{cases} a_0 + a_1, & i = 0, \quad j = 0, \\ a_0, & 1 \le i \le b, \quad j = i - 1, \\ a_{j-i+1}, & 1 \le j \le b - 1, \quad 0 \le i \le j, \\ \sum_{m=j-i+1}^{N} a_m, & j = b, \quad 0 \le i \le j, \\ 0 & \text{otherwise}, \end{cases}$$

Για την αναδρομική επίλυση της εξίσωσης ισορροπίας της αλυσίδας Markov χρησιμοποιούμε τις σχέσεις

$$q_1 \triangleq \Pr[Q = 1] = \frac{1 - a_0 - a_1}{a_0} \cdot q_0$$

$$q_n \triangleq \Pr[Q = n] = \frac{1 - a_1}{a_0} \cdot q_{n-1} - \sum_{k=2}^{n} \frac{a_k}{a_0} \cdot q_{n-k}, \qquad 2 \le n \le b,$$

όπου

$$q_0 \triangleq \Pr[Q = 0] = \frac{1}{1 + \sum_{n=1}^{b} q_n / q_0} \cdot$$

Κανένα κελί δεν θα μεταδοθεί στην έξοδο που μελετάμε κατά τη διάρκεια της m-οστης χρονοθυρίδας αν και μόνον αν $Q_{m-1} = 0$ και το $A_m = 0$. Συνεπώς, η ρυθμαπόδοση του μεταγωγέα υπολογίζεται ως

$$\rho_0 = 1 - q_0 a_0 .$$

Ένα κελί θα χαθεί, αν βγαίνοντας από τη μήτρα του μεταγωγέα βρει τον ενταμιευτή εξόδου ήδη να περιέχει b κελιά. Η πιθανότητα απώλειας κελιού μπορεί να υπολογισθεί ως

$$\Pr[\text{cell loss}] = 1 - \frac{\rho_0}{p},$$

όπου p το προσφερόμενο φορτίο.

Η ενταμίευση στην έξοδο πετυχαίνει τη βέλτιστη απόδοση καθυστέρησης – ρυθμαπόδοσης. Τα κελιά καθυστερούν όταν δύο ή περισσότερα κελιά που έρχονται από διαφορετικές εισόδους προορίζονται για την ίδια έξοδο.

Για τον υπολογισμό της μέσης καθυστέρησης στην ουρά εξόδου σε σχέση με το προσφερόμενο φορτίο χρησιμοποιήθηκε η σχέση (από τον τύπο του Little)

$$\overline{W} = \frac{\overline{Q}}{\rho_0} = \frac{\Sigma_{n=1}^{b} n q_n}{1 - q_0 a_0}.$$

και όταν $N \to \infty$ και $b \to \infty$

$$\overline{W} = \frac{p}{2(1-p)}.$$

Μελέτη ζητήματος

Στη συνέχεια αναλύονται τα σχετικά ζητούμενα με τη σειρά που δόθηκαν παραπάνω (παρουσιάζεται ο κώδικας σε MATLAB, τα διαγράμματα και ο αντίστοιχος σχολιασμός).

1.

Κώδικας MATLAB

Για την ανάλυση χρησιμοποιήθηκαν ορισμένοι μαθηματικοί τύποι οι οποίοι μοντελοποιήθηκαν προγραμματιστικά. Πιο αναλυτικά:

- Για τον υπολογισμό των δυνατών συνδυασμών N ανά k χρησιμοποιούμε τη σχέση $\binom{N}{k} = \frac{N!}{(N-k)!*k!}$

Matlab:

```
function c = combinations(n,r)
% function that calculates all the possible combinations
    c = (factorial(n)) / (factorial(n-r)*factorial(r));
```

- Για την τυχαία μεταβλητή A, που αντιπροσωπεύει τον αριθμό των αφίξεων των κελιών (cell) για την εξεταζόμενη έξοδο σε μία συγκεκριμένη χρονοθυρίδα χρησιμοποιούμε τη σχέση

$$a_k \triangleq \Pr[A = k] = \binom{N}{k}\left(\frac{p}{N}\right)^k\left(1 - \frac{p}{N}\right)^{N-k}, \qquad k = 0, 1, 2, \ldots N$$

και για N → ∞ η σχέση γίνεται

$$a_k \triangleq \Pr[A = k] = \frac{p^k e^{-p}}{k!}, \qquad k = 0, 1, 2, \ldots .$$

Matlab:

```
function ak = a_var(N,k,p)
% calculates the number of cell arrivals destined for the tagged output in
% a given time slot
    if N == inf
        ak = (p^k * exp(-p)) / (factorial(k));
    elseif k > N
        ak = 0;
    else
        ak = combinations(N,k) * (p/N)^k * (1 - p/N)^(N-k);
    end
```

- Για την αναδρομική επίλυση της εξίσωσης ισορροπίας της αλυσίδας Markov χρησιμοποιούμε τις σχέσεις

$$q_1 \triangleq \Pr[Q = 1] = \frac{1 - a_0 - a_1}{a_0} \cdot q_0$$

$$q_n \triangleq \Pr[Q = n] = \frac{1 - a_1}{a_0} \cdot q_{n-1} - \sum_{k=2}^{n} \frac{a_k}{a_0} \cdot q_{n-k}, \qquad 2 \le n \le b,$$

όπου

$$q_0 \triangleq \Pr[Q = 0] = \frac{1}{1 + \sum_{n=1}^{b} q_n / q_0} \cdot$$

Matlab:

```
function q = qvar(b,p,N)
% calculates q0, which is the probability of Q = 0 (the cells in the
% examined queue at the end of the mth slot = 0)
q_n = zeros(1, b+1); % preallocating memory,  the code runs much faster be-
cause there is
                    % no need to repeatedly reallocate memory for the growing
data structure
q_n(1) = 1;
q_n(2) = (1 - a_var(N,0,p) - a_var(N,1,p)) / a_var(N,0,p); % q_n[1]: q1 di-
vided by q0
    for n = 2:b+1
        q_n(n+1) = (1 - a_var(N,1,p))/(a_var(N,0,p))*q_n(n) -
sumn(n,N,p,q_n);
    end
q = 1 / (1 + sumq(b,q_n));
```

Για τα αθροίσματα στις παραπάνω σχέσεις δίνονται οι δύο παρακάτω συναρτήσεις Matlab.

```
function s = sumn (n,N,p,q_n)
% prints the sum for qn
init = 0;
    for k = 3:n+1
        new = a_var(N,k-1,p)/a_var(N,0,p)*q_n(n+2-k);
        init = init + new;
    end
s = init;
```

```
function s = sumq (b,q_n)
% prints the sum for q0
init = 0;
    for k = 2:b+1
        new = q_n(k);
        init = init + new;
    end
s = init;
```

- Για τη ρυθμαπόδοση ρ_0 έχουμε τη σχέση

$$\rho_0 = 1 - q_0 a_0 .$$

Matlab:

```
function r = rvar(b,p,N)
% calculates the switch throughput r
    r = 1 - qvar(b,p,N)*a_var(N,0,p);
```

- Για την πιθανότητα απώλειας κελιού έχουμε τη σχέση

$$\Pr[\text{cell loss}] = 1 - \frac{\rho_0}{p},$$

όπου p το προσφερόμενο φορτίο.

Matlab:

```
function p = cell_loss(b,p,N)
% calculates the cell loss probability
    p = 1 - rvar(b,p,N)/p;
```

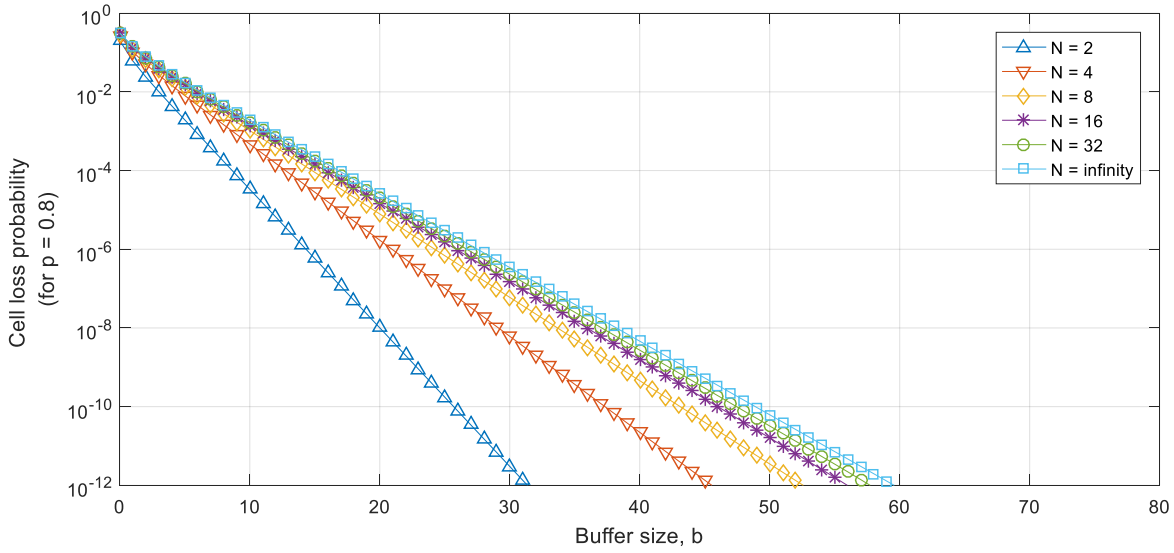
Τα διαγράμματα σχεδιάσθηκαν με τη χρήση του παρακάτω script:

Matlab:

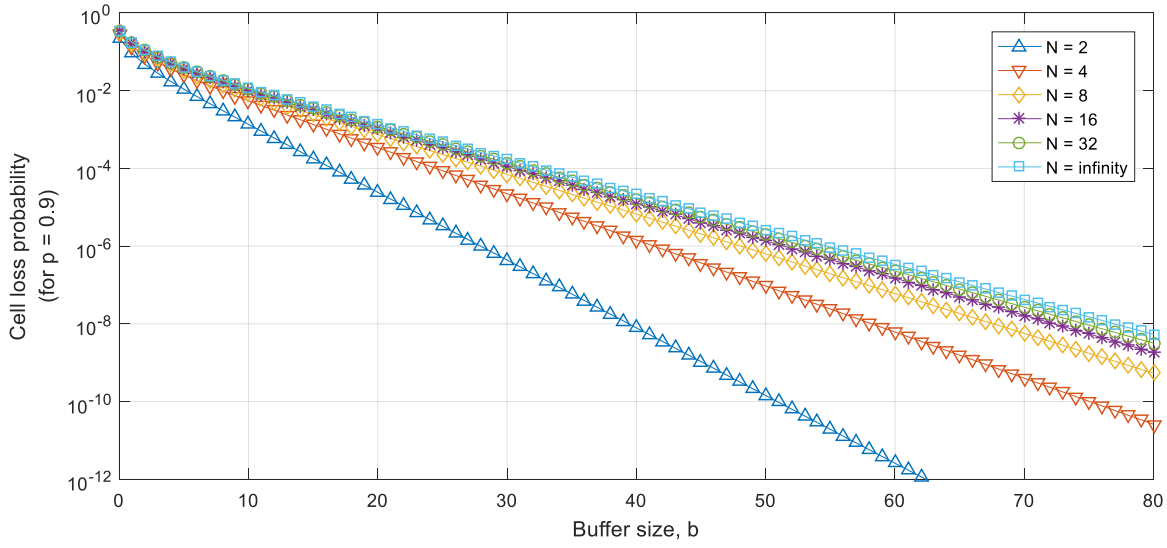
```
x = 0:80;
N = [2 4 8 16 32 inf];
p = 0.8;            % or p = 0.9
y = zeros(1,81);    % preallocating memory,  the code runs much faster be-
cause there is
                    % no need to repeatedly reallocate memory for the growing
data structure
for i = N
    for k = 0:80
        y(k+1) = cell_loss(k,p,i);
    end
    plot(x,y);
    hold on;
    ylim([10^-12,1]);   % sets max/min limit in the y axis
    xlim([0,80]);       % sets max/min limit in the x axis
    set(gca, 'YScale', 'log')   % sets logarithmic scale for axis
end
```

Διαγράμματα

i.



ii.



Σχολιασμός

Στα παραπάνω δύο διαγράμματα βλέπουμε την πιθανότητα απώλειας κελιού σε συνάρτηση με τη χωρητικότητα του ενταμιευτή, για διάφορες τιμές του πλήθους των εξόδων του μεταγωγέα και για διάφορες τιμές προσφερόμενου φορτίου.

Όπως βλέπουμε από το διάγραμμα (i), για προσφερόμενο φορτίο 80%, ένας ενταμιευτής μεγέθους b = 28 είναι αρκετά καλός για να διατηρήσει την πιθανότητα απώλειας κελιού κάτω από το 10^{-6} (για την ακρίβεια 8.1170e-07) για αυθαίρετα μεγάλο N.

Ομοίως από το διάγραμμα (ii), για προσφερόμενο φορτίο 90%, ένας ενταμιευτής μεγέθους b = 55 είναι αρκετά καλός για να διατηρήσει την πιθανότητα απώλειας κελιού κάτω από το 10^{-6} (για την ακρίβεια 9.5034e-07) για αυθαίρετα μεγάλο N.

Η καμπύλη του $N \rightarrow \infty$ είναι ένα στενό άνω όριο για τις πεπερασμένες καμπύλες των $N > 32$.

Έχοντας το ίδιο b, για μεγαλύτερα N έχουμε μεγαλύτερη πιθανότητα απώλειας κελιού (φαίνεται από την κλίση των καμπυλών).

2.

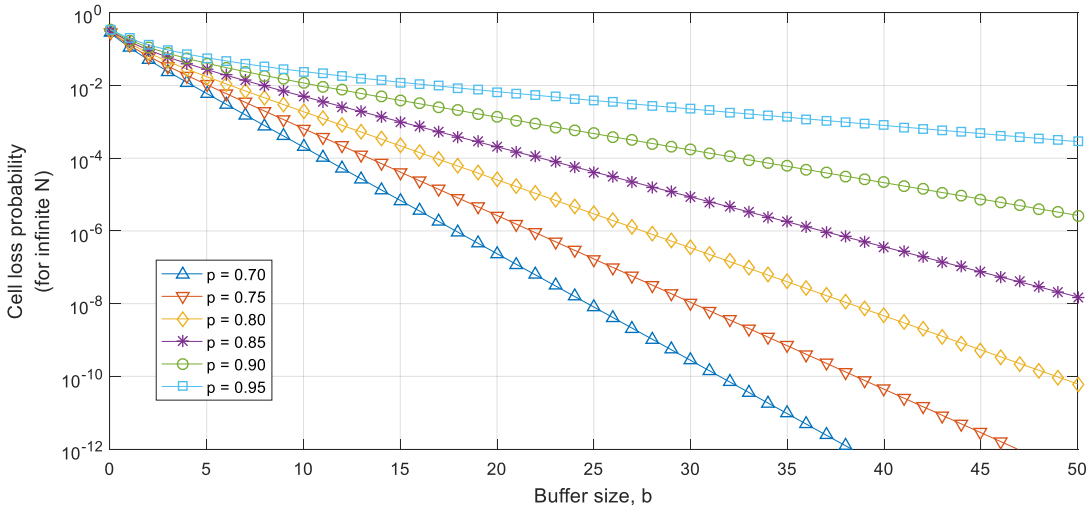
Κώδικας MATLAB

Για την ανάλυση και τη δημιουργία του διαγράμματος χρησιμοποιήθηκαν οι ίδιες συναρτήσεις με το προηγούμενο ζητούμενο και επιπλέον το παρακάτω script.

Matlab:

```
x = 0:50;
p = [0.7 0.75 0.8 0.85 0.9 0.95];
y = zeros(1,51);    % preallocating memory,  the code runs much faster be-
cause there is
                    % no need to repeatedly reallocate memory for the growing
data structure
for i = p
    for k = 0:50
        y(k+1) = cell_loss(k,i,inf);
    end
    plot(x,y);
    hold on;
    ylim([10^-12,1]);    % sets max/min limit in the y axis
    xlim([0,50]);        % sets max/min limit in the x axis
    set(gca, 'YScale', 'log')    % sets logarithmic scale for axis
end
```

Διαγράμματα



Σχολιασμός

Στο παραπάνω διάγραμμα βλέπουμε την πιθανότητα απώλειας κελιού σε συνάρτηση με τη χωρητικότητα του ενταμιευτή, όταν το πλήθος των εξόδων του μεταγωγέα , N, είναι άπειρο και για διάφορες τιμές προσφερόμενου φορτίου.

Για να διατηρηθεί η πιθανότητα απώλειας κελιού κάτω από το 10^{-6}

για τιμές προσφερόμενου φορτίου p = {0.7, 0.75, 0.8, 0.85, 0.9, 0.95}, ένας ενταμιευτής πρέπει να έχει αντίστοιχα b = {18, 22, 28, 37, 55, 105}.

Έχοντας το ίδιο b, για μεγαλύτερα p έχουμε μεγαλύτερη πιθανότητα απώλειας κελιού (φαίνεται από την κλίση των καμπυλών).

3.

Κώδικας MATLAB

Για τον υπολογισμό της μέσης καθυστέρησης στην ουρά εξόδου σε σχέση με το προσφερόμενο φορτίο χρησιμοποιήθηκε η σχέση (από τον τύπο του Little)

$$\overline{W} = \frac{\overline{Q}}{\rho_0} = \frac{\Sigma_{n=1}^{b} n q_n}{1 - q_0 a_0}.$$

και όταν N → ∞ και b → ∞

$$\overline{W} = \frac{p}{2(1-p)}.$$

Matlab:

```
function w = waiting(b,p)
    % calculates the mean waiting time for infinite N
    if b == inf
        w = p / (2*(1-p));
    else
        w = sumnq(b,p) / rvar(b,p,inf);
    end
```

```
function s = sumnq (b,p)
% prints the sum for mean(Q)
q_n = zeros(1, b+1); % preallocating memory,  the code runs much faster be-
cause there is
                     % no need to repeatedly reallocate memory for the grow-
ing data structure
q_n(1) = 1;
q_n(2) = (1 - a_var(inf,0,p) - a_var(inf,1,p)) / a_var(inf,0,p); % q_n[1]: q1
divided by q0
    for n = 2:b+1
        q_n(n+1) = (1 - a_var(inf,1,p))/(a_var(inf,0,p))*q_n(n) -
sumn(n,inf,p,q_n);
    end
q = 1 / (1 + sumq(b,q_n));
```

```
q_n = q_n * q;
init = 0;
    for k = 2:b+1
        new = q_n(k);
        init = init + (k-1)*new;
    end
s = init;
```

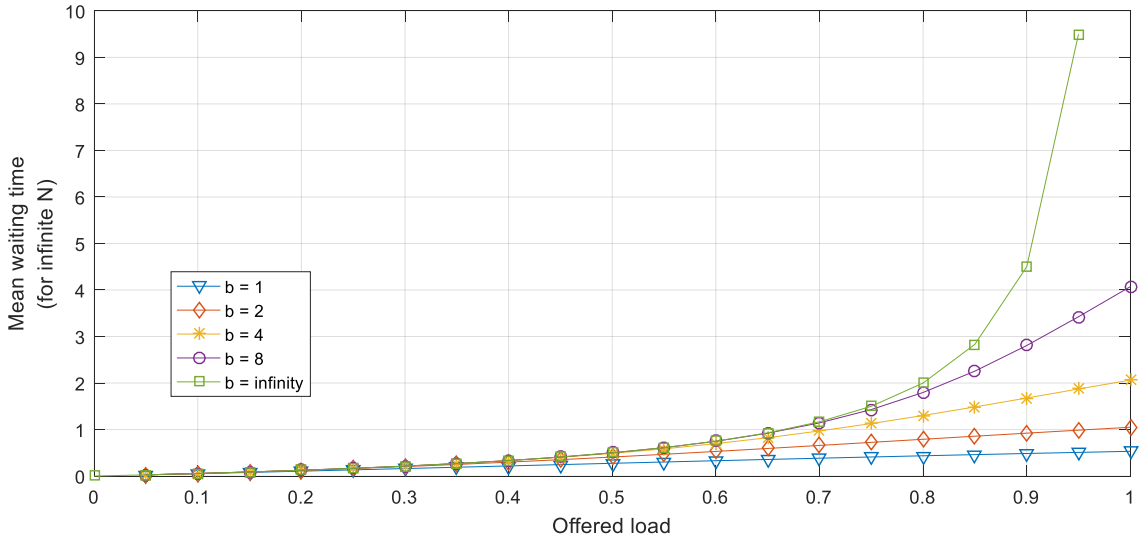
Το διαγράμματα σχεδιάσθηκε με τη χρήση του παρακάτω script:

Matlab:

```
x = 0:0.05:1;
b = [1 2 4 8 inf];
y = zeros(1,21);    % preallocating memory,  the code runs much faster be-
cause there is
                    % no need to repeatedly reallocate memory for the growing
data structure

for i = b
    k = 0;
    for m = x
        y(k+1) = waiting(i,m);
        k = k + 1;
    end
    plot(x,y);
    hold on;
    ylim([0,10]);   % sets max/min limit in the y axis
    xlim([0,1]);       % sets max/min limit in the x axis
end
```

Διαγράμματα



Σχολιασμός

Στο παραπάνω διάγραμμα βλέπουμε τη μέση καθυστέρηση στην ουρά εξόδου σε σχέση με το προσφερόμενο φορτίο, για άπειρες γραμμές εξόδου και για μεταβαλλόμενη χωρητικότητα b.

Παρατηρούμε ότι αν έχουμε αυθαίρετα μεγάλο b (ή μικρότερο), έχουμε μέση καθυστέρηση μικρότερη του 1 για φόρτο μικρότερο του 65%.

Έχοντας το ίδιο p, για μεγαλύτερα b έχουμε μεγαλύτερη μέση καθυστέρηση στην ουρά εξόδου (φαίνεται από την κλίση των καμπυλών).