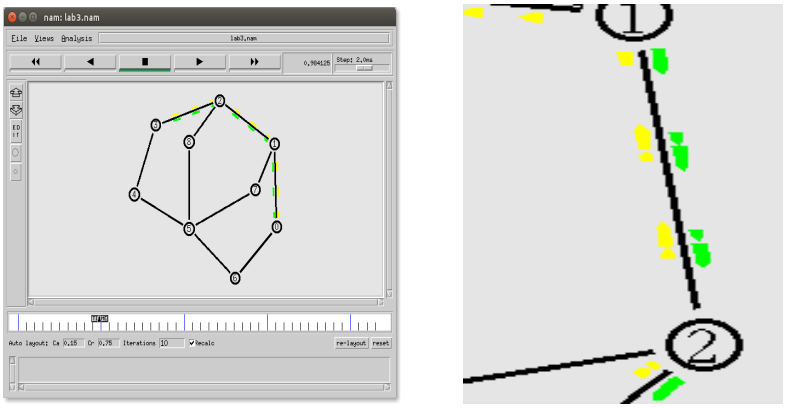
Δίκτυα επικοινωνιών
Τρίτη εργαστηριακή Άσκηση

Νικόλαος Ζαρίφης ID: 03112178

21 Απριλίου 2015

Ερωτήσεις 1

Το σχήμα που προέκυψε είναι το ακόλουθο:



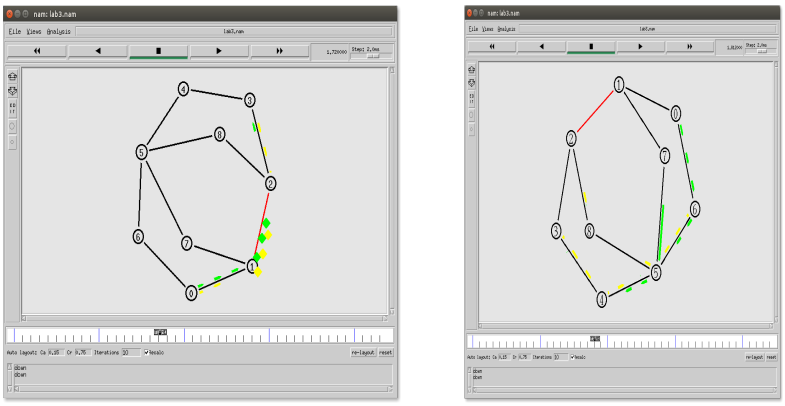
(α′) Δυκτιό (β′) without packetSize

Σχήμα 1: nam of ex 1

- **Ποια διαδρομή ακολουθούν τα πακέτα;**
  Τα πράσινα πακέτα ξεκινάνε από τον κόμβο 0 κι πάνε στον 3 ακολουθώντας την διαδρομή 0-1-2-3. Ενώ τα κίτρινα πάνε από τον 3 στον 0 από την διαδρομή 3-2-1-0.
- **Ελέγξτε αν η ροή των πακέτων και από τις δυο πλευρές ακολουθεί τη διαδρομή με τα λιγότερα βήματα.**
  Βλέπουμε ότι κι οι 2 διαδρομές έχουν 3 βήματα η οποία είναι η διαδρομή με τα λιγότερα βήματα. Εκτελώντας έναν αλγόριθμο BFS στον γράφο αφού δεν έχουμε βάρος μας επιβεβαιώνει το αποτέλεσμά μας.
- **Υπάρχει συντομότερη διαδρομή από αυτήν που ακολουθούν, όσον αφορά τη συνολική καθυστέρηση κάθε ροής;**
  Η διαδρομή που έχουμε είναι η καλύτερη και από την μεριά της καθυστέρησης έχοντας συνολική 60ms.Η μόνη εναλλακτική περνάει από το διαδρομή 1-7-5 και όποια κατεύθυνση αν πάρει θα έχει περισσότερο καθυστέρηση.Ο αλγόριθμος του Dijkstra σε τέτοιες περιπτώσεις μας επιβεβαιώνει το αποτέλεσμα.

- **Ποιος είναι ο ρόλος των εντολών $udp0 set packetSize_ 1500 και $udp3 set packetSize_ 1500; Τι παρατηρείτε στις ροές των πακέτων αν αφαιρεθούν οι γραμμές αυτές από τον κώδικα της προσομοίωσης;**
  Όπως βλέπουμε στην εικόνα στέλνει 2 πακέτα. Αυτό συμβαίνει γιατί συμφωνά με το documation ορίζει ως default value το size=1000B άρα ότι είναι μεγαλύτερο το σπάει σε κομμάτια.

Ερωτήσεις 2



(α΄) Static (β΄) Dynamic

Σχήμα 2: nam of ex 2

- **Εξηγήστε γιατί, με τη στατική δρομολόγηση, οι κόμβοι εξακολουθούν να στέλνουν πακέτα και μετά τη διακοπή της ζεύξης.**
  Ο λόγος είναι ότι το δίκτυο μας δεν διαθέτη κάποιον τρόπο για να ανακαλύψει ότι η ζεύξη έπεσε, όποτε συνεχίζει να στέλνει σαν να μην συνέβηκε τίποτα.

- **Τα πακέτα που χάθηκαν, θα ξαναμεταδοθούν από τους αντίστοιχους κόμβους, όταν επανέλθει η σύνδεση;**
  Όχι , γιατί όπως είπαμε οι πομποί δεν καταλαβαίνουν πως έπεσε η ζεύξη.

- **Τι παρατηρείτε όταν γίνεται διακοπή ζεύξης και έχουμε δυναμική δρομολόγηση; Περιγράψτε με απλά λόγια τη διαδικασία που λαμβάνει χώρα στο animation.Συμπίπτει η αρχική με τη μόνιμη διαδρομή δρομολόγησης για τις δύο ροές κατά τη διάρκεια της διακοπής;**
  Όταν η ζεύξη πέσει τότε το rtproto θα ειδοποιήσει το σύστημα κι θα βρεθεί μια νέα διαδρομή για να στείλει τα πακέτα όπου θα είναι η πιο σύντομη. Αλλά τα πακέτα που είχε στείλει πριν την νέα μόνιμη διαδρομή κι έχουν φτάσει στον κόμβο 1 και στον 2 αντίστοιχα, στέλνονται στον 7-5 και 8-5 . Και απο εκεί ακολουθούν την κοινή διαδρομή με τα άλλα πακέτα που είναι η 0-6-5-4-3.Τέλος όταν ξαναναίβει η ζεύξη το rtproto ενημερώνει την διαδρομή κι στέλνονται τα υπόλοιπα πακέτα από εκεί.

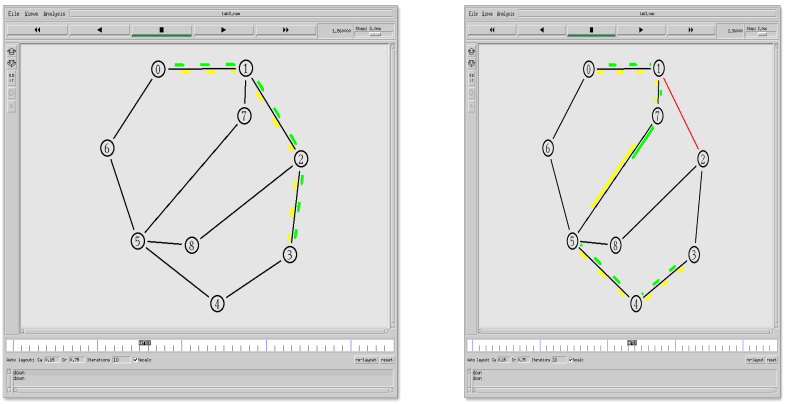
- **Με βάση το animation, προσδιορίστε για κάθε ροή τη χρονική στιγμή όπου παρατηρείται η μόνιμη διαδρομή δρομολόγησης κατά τη διάρκεια διακοπής**
  Βλέπουμε πως ταυτόχρονα κι στους δυο πομπούς ακολουθούν την μόνιμη διαδρομή πέριπου στο 1.85s.Αυτό γίνεται γιατί τότε πέρνουν την πληροφορία απο το rtproto.

- **Για ποιο λόγο τα πακέτα ακολουθούν τις συγκεκριμένες διαδρομές αφότου πέσει η σύνδεση, στην αρχική και τη μόνιμη κατάσταση;**

Αυτό συμβαίνει επειδή είναι η ελάχιστη διαδρομή από θέμα καθυστερήσεις ανάμεσα στους δύο κόμβους.Στην αρχική κατάσταση κοιτάει από τον κόμβο 1,2 και μετά απο τους κόμβους 0,3.

- **Θα μπορούσαν να δρομολογηθούν από άλλους κόμβους;**
  Ναι γίνεται υπάρχουν κι άλλες διαδρομές όπως βλέπουμε αλλά δεν είναι ελάχιστες.
- **Ποιος από όλους τους κόμβους καθορίζει από ποια διαδρομή θα προωθηθούν κάθε φορά τα πακέτα;**
  Πάντα αποφασίζει η πηγή αλλά πριν την μόνιμη κατάσταση αποφασίζουν οι κόμβοι που έχουν πακέτα εκείνη την στιγμή δηλαδή στην περίπτωση μας ο 1,2 .

Ερωτήσεις 3



(α΄) Before (β΄) After

Σχήμα 3: nam of ex 3

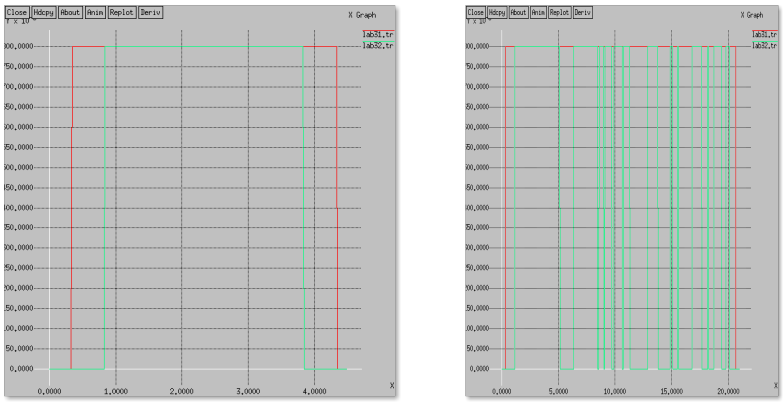
- **Ποιες διαδρομές ακολουθούν τα πακέτα πριν, κατά τη διάρκεια και μετά την πτώση της σύνδεσης για τις δύο ροές;**
  Πριν την πτώση ακολουθούν την ίδια διαδρομή με πριν. Αλλά όταν πέσει η γραμμή τα πακέτα προτειμούν την διαδρομή 0-1-7-5-4-3 γιατί έχει το μικρότερο κόστος.
- **Για ποιον λόγο τα πακέτα ακολουθούν τις συγκεκριμένες διαδρομές;**
  Έχουν κι τα 2 το ελάχιστο κόστος. Όταν πέσει η γραμμή έχει κόστος 15 κι όταν δεν έχει πέσει έχει 12.Υπολογίζοντας την διαδρομή που ακολουθούσε πριν όταν είχε πέσει η γραμμή ,τώρα έχει κόστος 16.
- **Θα μπορούσαν να δρομολογηθούν από άλλους κόμβους;**
  Ναι γίνεται αφού υπάρχουν κι άλλες διαδρομές αλλά δεν έχουν το λιγότερο κόστος.
- **Μετά την αποκατάσταση της ζεύξης μεταξύ των κόμβων "1" και "2", προσδιορίστε με βάση το animation τη χρονική στιγμή όπου παρατηρείται η μόνιμη διαδρομή δρομολόγησης για κάθε ροή.**
  Όταν ανέβει η γραμμή κι οι δύο ροές παίρνουν την μόνιμη κατεύθυνση στα 2.8s.
- **Πριν(?) την αποκατάσταση της ζεύξης μεταξύ των κόμβων "1" και "2", προσδιορίστε με βάση το animation τη χρονική στιγμή όπου παρατηρείται η μόνιμη διαδρομή δρομολόγησης για κάθε ροή.**
  Η πρώτη ροή πέρνει την μόνιμη κατασταση κατευθείαν αφού η διαδρομή αυτή περνά απο τον κόμβο 1. Η αλλή ροή όμως αργεί λίγο γιατί πρέπει να στείλει τα πακέτα από τον κόμβο 2 που έχουν μείνει κι δεν ανείκει στην ελάχιστη διαδρομή. Περίπου 1.85s.

- **Ποιος είναι ο ρόλος της εντολής Agent/rtProto/Direct set preference_ 200; Τι παρατηρείτε στη δρομολόγηση των πακέτων αν αφαιρεθεί η εντολή αυτή από τον κώδικα της προσομοίωσης; Αιτιολογείστε γιατί συμβαίνει αυτό.**
  Η εντολή αυτή έχει ως αποτέλεσμα να κρατάει πληροφορίες για την σύνδεση με τους γειτονικούς κόμβους,όπως επίσης από DV έχει κι όλες τις πιθανές διαδρομές.Και έτσι όταν διακοπεί η διαδρομή μπορεί να βρει μια νέα διαδρομή.Έτσι βλέπουμε πως όταν αφαιρούμε την εντολή κάνει περισσότερο χρόνο να αποφασίσει την νέα διαδρομή.Γιατί πρέπει πρώτα να ενημερωθεί ο κόμβος 0 (ή 3) κι να καθορίσει την νέα διαδρομή σε αντίθεση με όταν είχαμε την εντολή κι καθόριζε ο κόμβος πριν την γραμμή που έπεσε.

```
set ns [new Simulator]
set nf [open lab3.nam w]
set xf [open lab3.tr w]
Agent/rtProto/Direct set preference_ 200
$ns rtproto DV
$ns namtrace-all $nf
proc finish {} {
 global ns nf xf
 $ns flush-trace
 close $nf
 close $xf
 exit 0
}
for {set i 0} {$i < 9} {incr i} {
        set n($i) [$ns node]
}

for {set i 0} {$i < 7} {incr i} {
        $ns duplex-link $n($i) $n([expr ($i+1)%7]) 2Mb 40ms DropTail
        $ns cost $n($i) $n([expr ($i+1)%7]) 4
        $ns cost $n([expr ($i+1)%7]) $n($i) 4
}
$ns duplex-link $n(7) $n(1) 2Mb 20ms DropTail
$ns cost $n(7) $n(1) 2
$ns duplex-link $n(7) $n(5) 2Mb 10ms DropTail
$ns cost $n(1) $n(7) 2
$ns duplex-link $n(8) $n(5) 2Mb 10ms DropTail
$ns duplex-link $n(8) $n(2) 2Mb 40ms DropTail
set udp0 [new Agent/UDP]
$ns cost $n(8) $n(2) 4
$ns cost $n(2) $n(8) 4

$ns attach-agent $n(0) $udp0
$udp0 set fid_ 0
$ns color 0 green
set sink0 [new Agent/LossMonitor]
$ns attach-agent $n(0) $sink0
$udp0 set packetSize_ 1500
set udp3 [new Agent/UDP]

$ns attach-agent $n(3) $udp3
$udp3 set fid_ 3
$udp3 set packetSize_ 1500
$ns color 3 yellow
```

```
set sink3 [new Agent/LossMonitor]
$ns attach-agent $n(3) $sink3

$ns connect $udp0 $sink3
$ns connect $udp3 $sink0

set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 1500
$cbr0 set interval_ 0.015
$cbr0 attach-agent $udp0

set cbr3 [new Application/Traffic/CBR]
$cbr3 set packetSize_ 1500
$cbr3 set interval_ 0.015
$cbr3 attach-agent $udp3

$ns at 0.2 "$cbr0 start"
$ns at 0.7 "$cbr3 start"
$ns rtmodel-at 1.7 down $n(1) $n(2)
$ns rtmodel-at 2.7 up $n(1) $n(2)
$ns at 3.7 "$cbr3 stop"
$ns at 4.2 "$cbr0 stop"
$ns at 4.5 "finish"

$ns run
```

Ερωτήσεις 4



(α΄) CBR (β΄) Expomential

Σχήμα 4: xgraph

- **Ποιος είναι ο μέγιστος ρυθμός μετάδοσης που επιτυγχάνεται για τις δύο περιπτώσεις κίνησης, βάσει των γραφικών παραστάσεων που σχεδιάσατε;**
  Όπως βλέπουμε κι στις δύο περιπτώσεις έχουμε μέγιστο ρυθμό μετάδοσης 0.8Mbits/s
- **Αιτιολογείστε τις μέγιστες τιμές που προσδιορίσατε παραπάνω, χρησιμοποιώντας τις παραμέτρους που θέσατε για τη διαμόρφωση των δύο πηγών κίνησης (CBR και Exponential).**

Όταν έχουμε CBR έχουμε από τύπους που έχουμε χρησιμοποιείσει σε παλιότερες ασκήσεις έχουμε : $\frac{packets}{rate} = \frac{1500}{0.015} = 10^5 bytes/s = 0.8*10^6 Bits/s = 0.8 Mbits/s$. Τώρα όταν έχουμε Expomential έχουμε ρυθμίσει manualy τον ρυθμό απο την εντολή rate=800k → 0.8MBits/s.

- **Υπολογίστε το πλήθος των bytes που λαμβάνονται επιτυχώς στον προορισμό για κάθε ροή, θεωρώντας ότι και οι δύο ροές ολοκληρώνονται σε χρόνο t=20+(a/10)sec, όπου a τα δύο τελευταία ψηφία του αριθμού μητρώου σας**. Για α=78 έχουμε t=27.8s.Προσθέτοντας μεταβλητές που να υπολογίζουν τα συνολικά bytes που λαμβάνονται έχουμε ως αποτέλεσμα από 0 με προορισμό 3 έχουμε 2755500Bytes κι απο 3 με 0 έχουμε 1588500Bytes.

```
set ns [new Simulator]
set nf [open lab3.nam w]
set xf [open lab31.tr w]
set xf1 [open lab32.tr w]
set calc0 0
set calc1 0
Agent/rtProto/Direct set preference_ 200
$ns rtproto DV
$ns namtrace-all $nf
proc record {} {
        global sink0 sink3 xf xf1 calc0 calc1
        set ns [Simulator instance]
        set time 0.015
        set bw0 [$sink3 set bytes_]
        set bw3 [$sink0 set bytes_]
        set now [$ns now]
        set calc0 [expr [$sink3 set bytes_]+$calc0]
        set calc1 [expr [$sink0 set bytes_]+$calc1]
        puts $xf "$now [expr (($bw0/$time *8)/1000000)]"

        puts $xf1 "$now [expr (($bw3/$time *8)/1000000)]"
        $sink0 set bytes_ 0
        $sink3 set bytes_ 0
        $ns at [expr $now+$time] "record"
}
proc finish {} {
 global ns nf xf xf1 calc0 calc1
 $ns flush-trace
 close $nf
 close $xf1
 close $xf
 puts $calc0
 puts $calc1
 exit 0
}
for {set i 0} {$i < 9} {incr i} {
        set n($i) [$ns node]
}

for {set i 0} {$i < 7} {incr i} {
        $ns duplex-link $n($i) $n([expr ($i+1)%7]) 2Mb 40ms DropTail
        $ns cost $n($i) $n([expr ($i+1)%7]) 4
        $ns cost $n([expr ($i+1)%7]) $n($i) 4
}
```

```
$ns duplex-link $n(7) $n(1) 2Mb 20ms DropTail
$ns cost $n(7) $n(1) 2
$ns duplex-link $n(7) $n(5) 2Mb 10ms DropTail
$ns cost $n(1) $n(7) 2
$ns duplex-link $n(8) $n(5) 2Mb 10ms DropTail
$ns duplex-link $n(8) $n(2) 2Mb 40ms DropTail
set udp0 [new Agent/UDP]
$ns cost $n(8) $n(2) 4
$ns cost $n(2) $n(8) 4

$ns attach-agent $n(0) $udp0
$udp0 set fid_ 0
$ns color 0 green
set sink0 [new Agent/LossMonitor]
$ns attach-agent $n(0) $sink0
$udp0 set packetSize_ 1500
set udp3 [new Agent/UDP]

$ns attach-agent $n(3) $udp3
$udp3 set fid_ 3
$udp3 set packetSize_ 1500
$ns color 3 yellow
set sink3 [new Agent/LossMonitor]
$ns attach-agent $n(3) $sink3

$ns connect $udp0 $sink3
$ns connect $udp3 $sink0

set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 1500
$cbr0 set interval_ 0.015
$cbr0 attach-agent $udp0

set cbr3 [new Application/Traffic/Exponential]
$cbr3 set packetSize_ 1500
$cbr3 set rate_ 800k
$cbr3 set interval_ 0.015
$cbr3 attach-agent $udp3
$ns at 0.0 "record"
$ns at 0.2 "$cbr0 start"
$ns at 0.7 "$cbr3 start"
$ns at 27.8 "$cbr3 stop"
$ns at 27.8 "$cbr0 stop"
$ns at 27.9 "finish"

$ns run
```