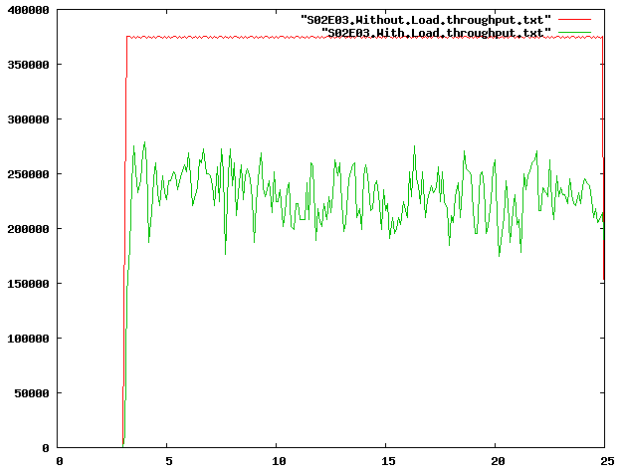
Netwerksimulatie – Verslag 2

Philippe Dellaert
3e Bachelor Computerwetenschappen-Elektrotechniek

Wouter Van Ranst
3e Bachelor Computerwetenschappen-Bedrijfsbeheer

6 mei 2007

1 Throughput

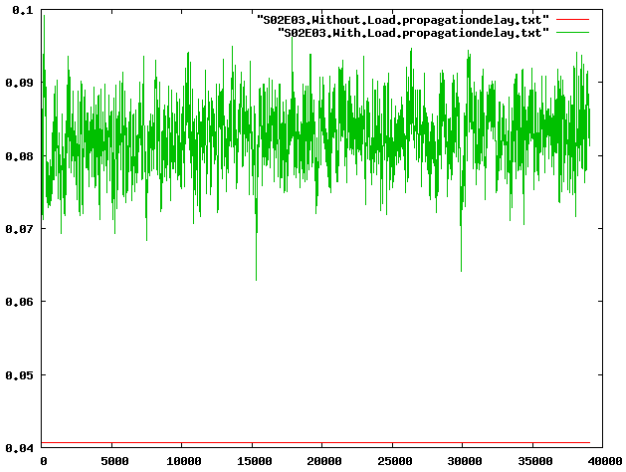


Figuur 1: Throughput in node 10 met en zonder netwerk belasting.

Uit figuur 1 blijkt duidelijk dat de throughput van de video stream zwaar lijdt onder de netwerk belasting. Zonder netwerk belasting is de throughput vrij constant rond de maximum waarde. Als er wel netwerk belasting is, is er heel wat meer schommeling in de throughput.

Dit is uiteraard te verklaren door de netwerk belasting. Wanneer we het netwerk belasten, zal de link tussen node 0 en node 1 overbelast worden. Hierdoor wordt alles vertraagd en zullen er packets gedropt worden. De TCP verbindingen zullen dit proberen opvangen door hun window aan te passen. De UDP verbindingen echter, zullen packets blijven uitzenden wat ervoor zorgt dat er veel UDP packets gedropt worden.

2 Propagation delay

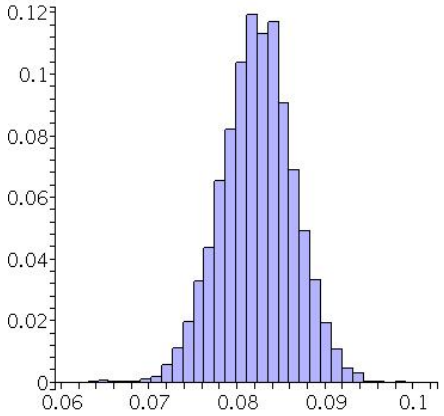


Figuur 2: Propagation delay van elk pakket van de video stream.

In figuur 2 valt het direct op dat de propagation delay voor het onbelaste netwerk constant is rond 0.04, dit is de minimum delay die zal ontstaan door de fysische limieten van de netwerkopstelling. De propagation delay bij het belaste netwerk ligt veel hoger en schommelt ook redelijk hard. Dit zorgt voor redelijk wat jitter.

Als men de grafiek wat beter bekijkt, valt direct op dat er ook heel wat packets gedropt worden als er netwerk belasting is. Dit is te verklaren doordat TCP probeert zijn window aan te passen zodat er geen drops voorkomen van TCP packets. UDP doet dit niet aangezien er niet geweten is dat deze packets gedropt worden. Dit valt ook direct op bij de simulatie via NAM, er worden amper TCP packets gedropt, maar wel heel veel UDP packets. Hierdoor komt er ook een deel van de videostream packets niet aan. Wat zorgt voor verlies van kwaliteit of zelfs verlies van sommige frames.

3 Jitter



Figuur 3: Histogram van de propagation delay van de video stream.

Op figuur 3 is de histogram afgebeeld van de propagation delay terwijl het netwerk belast wordt. Dit lijkt normaal verdeeld, wat wijst op een veel jitter. Doordat er zo'n verschil zit tussen de tijden dat de paketten aankomen, zal ook de kwaliteit van het beeld zwaar schommelen en het beeld schokkerig worden. Het kan eventueel helpen om een buffer op te bouwen zodat een bepaalde frame pas getoond wordt nadat alle paketten al lang zijn toegekomen.

A Code

```
#########################################
# EXERCISE SETUP                        #
#########################################

# Basename for the generated files
set filenameBase "S02E03"
set networkLoad 0

#########################################
# SIMULATOR STARTUP                     #
#########################################

# Create simulator
set ns [new Simulator]

#########################################
# COLOR SETUP                           #
#########################################

$ns color 0 Blue
$ns color 1 Red
$ns color 2 Green

#########################################
# FILES                                 #
#########################################

if {$networkLoad == 1} {
        set loadText "With.Load"
} else {
        set loadText "Without.Load"
}

# Trace file
set tf [open $filenameBase.$loadText.tr w]
$ns trace-all $tf

# Nam tracefile
set nf [open $filenameBase.$loadText.nam w]
$ns namtrace-all $nf

#########################################
# NODE SETUP                            #
#########################################

# Create nodes
set n0 [$ns node]
set n1 [$ns node]
```

```
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
set n6 [$ns node]
set n7 [$ns node]
set n8 [$ns node]
set n9 [$ns node]
set n10 [$ns node]

# Creating links
$ns duplex-link $n0 $n1 10Mb 10ms DropTail
$ns duplex-link $n0 $n7 10Mb 10ms DropTail
$ns duplex-link $n0 $n8 10Mb 10ms DropTail
$ns duplex-link $n0 $n10 10Mb 10ms DropTail
$ns duplex-link $n1 $n2 10Mb 10ms DropTail
$ns duplex-link $n1 $n5 10Mb 10ms DropTail
$ns duplex-link $n1 $n6 10Mb 10ms DropTail
$ns duplex-link $n2 $n3 10Mb 10ms DropTail
$ns duplex-link $n2 $n4 10Mb 10ms DropTail
$ns duplex-link $n2 $n9 10Mb 10ms DropTail
#$ns duplex-link $n0 $n1 10Mb 10ms RED
#$ns duplex-link $n0 $n1 10Mb 10ms SFQ

########################################
# NAM SETUP                            #
########################################

#Give node position (for NAM)
$ns duplex-link-op $n10 $n0 orient right-down
$ns duplex-link-op $n7 $n0 orient right
$ns duplex-link-op $n8 $n0 orient right-up
$ns duplex-link-op $n0 $n1 orient right
$ns duplex-link-op $n5 $n1 orient up
$ns duplex-link-op $n6 $n1 orient left-up
$ns duplex-link-op $n1 $n2 orient right
$ns duplex-link-op $n9 $n2 orient left-down
$ns duplex-link-op $n3 $n2 orient left
$ns duplex-link-op $n4 $n2 orient left-up


########################################
# VIDEO UDP CBR                        #
########################################

# Starting UDP Agent
set videoudp [new Agent/UDP]
$ns attach-agent $n9 $videoudp

# Starting Sink agent
```

```
set videonull [new Agent/Null]
$ns attach-agent $n10 $videonull

# Connecting UDP agent to Null agent
$ns connect $videoudp $videonull

# Specific settings
$videoudp set fid_ 0

# Starting CBR connection
set videocbr [new Application/Traffic/CBR]
$videocbr attach-agent $videoudp
$videocbr set rate_ 3Mb

########################################
# CBR CONNECTION SETUP                 #
########################################

# Number of TCP connections
set numTCPCBR 70

# Number of UDP connections
set numUDPCBR 30

# Startup times
set startTCP 0.5
set startUDP 1

########################################
# RANDOM GENERATOR FOR START TIMES     #
########################################

# Starting RNG
set rng [new RNG]
$rng seed 0

# Starting Random
set RVStart [new RandomVariable/Exponential]
$RVStart set avg_ 0.03
$RVStart use-rng $rng

########################################
# CONNECTION GENERATE                  #
########################################

if { $networkLoad == 1 } {
        # Generating TCP CBR Connections
        for {set i 0} {$i < $numTCPCBR} {incr i} {
                # Starting TCP Agent
                set tcp($i) [new Agent/TCP]
```

```
        #set tcp($i) [new Agent/TCP/Newreno]
        #set tcp($i) [new Agent/TCP/Vegas]
        if {$i <= 40} {
                $ns attach-agent $n3 $tcp($i)
        } else {
                $ns attach-agent $n5 $tcp($i)
        }

        # Starting Sink agent
        set sink($i) [new Agent/TCPSink]
        $ns attach-agent $n7 $sink($i)

        # Connecting TCP agent to Sink agent
        $ns connect $tcp($i) $sink($i)

        # Setting connection parameters
        $tcp($i) set fid_ 1
        $tcp($i) set packetSize_ 1500
        $tcp($i) set rate_ 0.043Mb

        # Starting FTP connection
        set tcpcbr($i) [new Application/Traffic/CBR]
        $tcpcbr($i) attach-agent $tcp($i)

        # Start time
        set startTCP [expr $startTCP+[$RVStart value]]

        # Timing
        $ns at $startTCP "$tcpcbr($i) start"
        $ns at 24.9 "$tcpcbr($i) stop"
}

# Generating UDP CBR Connections
for {set i 0} {$i < $numUDPCBR} {incr i} {
        # Starting UDP Agent
        set udp($i) [new Agent/UDP]

        if {$i <= 20} {
                $ns attach-agent $n4 $udp($i)
        } else {
                $ns attach-agent $n6 $udp($i)
        }

        # Starting Sink agent
        set null($i) [new Agent/Null]
        $ns attach-agent $n8 $null($i)

        # Connecting TCP agent to Sink agent
        $ns connect $udp($i) $null($i)
```

```
                # Setting connection parameters
                $udp($i) set fid_ 2

                # Starting FTP connection
                set udpcbr($i) [new Application/Traffic/CBR]
                $udpcbr($i) attach-agent $udp($i)
                $udpcbr($i) set rate_ 0.034Mb

                # Start time
                set startUDP [expr $startUDP+[$RVStart value]]

                # Timing
                $ns at $startUDP "$udpcbr($i) start"
                $ns at 24.9 "$udpcbr($i) stop"
        }
}

#########################################
# TIMING                                #
#########################################

# End of simulation
$ns at 25.0 "finish"

# Timing of video stream
$ns at 3.0 "$videocbr start"
$ns at 24.9 "$videocbr stop"

#########################################
# FUNCTIONS                             #
#########################################

# Finish function
proc finish {} {
        #finalize trace files
        global ns nf tf filenameBase loadText
        $ns flush-trace
        close $tf
        close $nf

        exec nam $filenameBase.$loadText.nam &
        exit 0
}

#########################################
# RUNNING SIMULATOR                     #
#########################################

# Running the network simulator
$ns run
```