Εργασία στα «Δίκτυα Υπολογιστών ΙΙ»

(2018-2019)

Χριστόφορος Σπάρταλης

56785

Υπ. Εργαστηρίου:

Χρήστος-Αλέξανδρος Σάρρος

Ιωάννα-Αγγελική Καπετανίδου

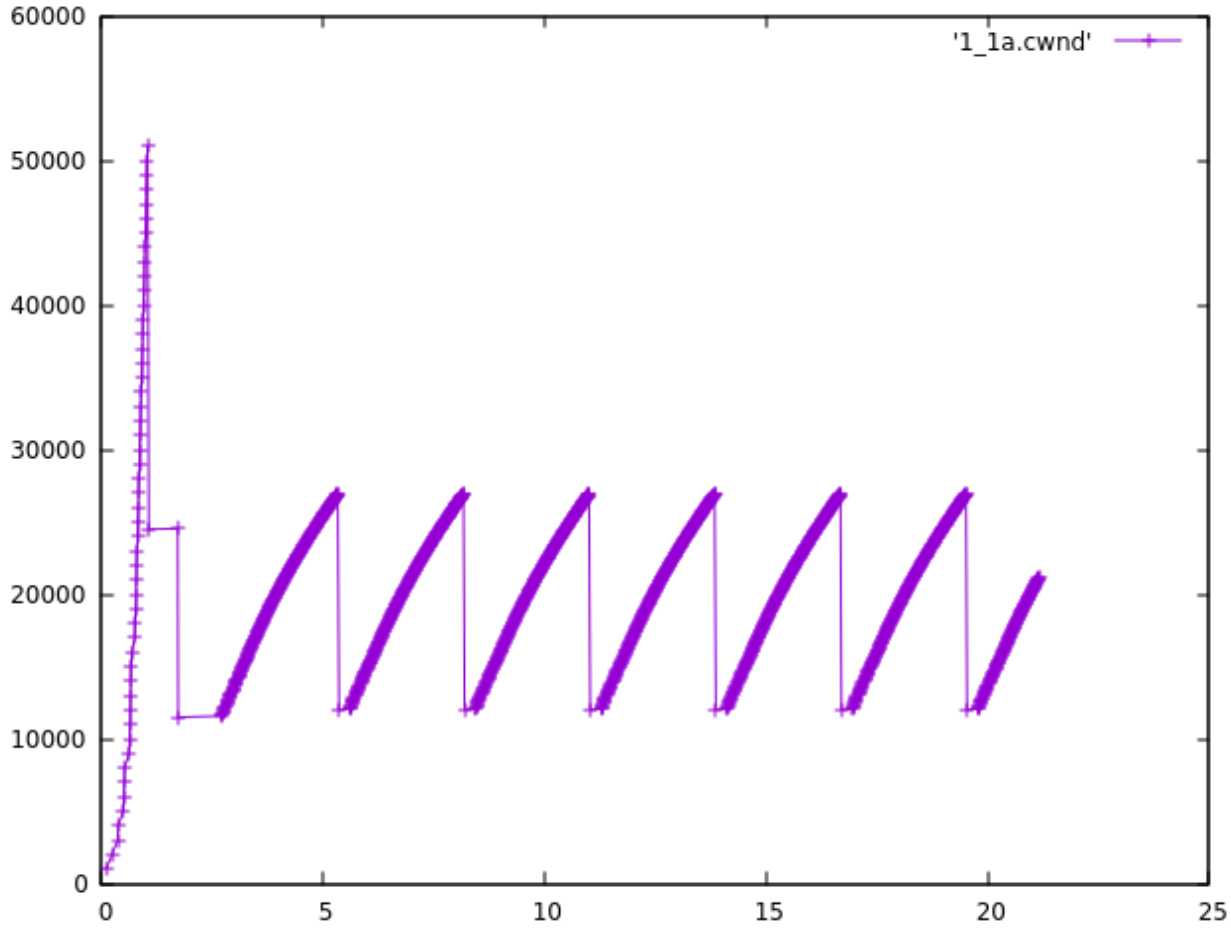
Υπ. Μαθήματος:

Βασίλειος Τσαουσίδης

Περιεχόμενα

Άσκηση 1

Ερώτημα 1α



[0 – 1,08048 s] Φάση Slow Start: Ο αποστολέας αρχίζει στέλνοντας ένα πακέτο (το πακέτο στο TCP ονομάζεται segment και όπως μπορούμε να δούμε από τον κώδικα το Maximum Segment Size είναι MSS=1000 Bytes). Ξεκινάει, λοιπόν, στέλνοντας μικρή ποσότητα δεδομένων, διότι έχει άγνοια για το capacity του δικτύου και δε θέλει να προκαλέσει συμφόρηση στην αρχή της επικοινωνίας. Στη συνέχεια για κάθε επιβεβαίωση που λαμβάνει (μηχανισμός self-clocking ή ack-clocking) αυξάνει κατά 1 το παράθυρο συμφόρησης (congestion window -cwnd). Έτσι σε κάθε RTT διπλασιάζει τη ροή δεδομένων που στέλνει. Η εκθετική αύξηση της ροής οφείλεται στην επιθυμία του αποστολέα να ανιχνεύσει γρήγορα το capacity του δικτύου. Η αύξηση του cwnd σταματάει, διότι τη χρονική στιγμή t=1,08048s ανιχνεύεται συμφόρηση, δηλαδή χάνεται κάποιο πακέτο.

Μηχανισμος Fast Retransmit: Η συμφόρηση που ανιχνεύτηκε δεν είναι καταστροφική. Αυτό το αντιλαμβάνεται ο αποστολέας, διότι παραλαμβάνει κάποιες επιβεβαιώσεις για τα πακέτα που παραλήφθηκαν επιτυχώς από τον παραλήπτη υπό τη μορφή διπλοεπιβεβαιώσεων (DACKs). Δηλαδή λαμβάνει ACKs τα οποία δείχνουν στο ίδιο Next Byte Expected. Μετά την παραλαβή τριών DACKs, Εφαρμόζεται ο μηχανισμός γρήγορης επαναμεταφοράς, όπου ο αποστολέας ξαναστέλνει τα πακέτα που ζητάει ο παραλήπτης (δηλαδή προσπαθεί να γεμίσει τις «τρύπες») πριν τελειώσει το timeout.

Μηχανισμός Fast Recovery: Αφού, λοιπόν, ανιχνεύτηκε περιορισμένη συμφόρηση, εφαρμόζεται ο μηχανισμός γρήγορης ανάκαμψης. Δηλαδή το cwnd μειώνεται στο μισό των Data_in_flight (όπου Data_in_flight = Last Byte Sent – Last Byte Acked). Ομοίως υποχωρεί και το ssthresh (slow start threshold). Επίσης στο TCP New Reno ο συγκεκριμένος μηχανισμός προβλέπει επιπρόσθετα πως αν ληφθεί ένα μερικό ACK, τότε το χαμένο πακέτο θα επαναμεταδοθεί αμέσως.

[1,08048 - 1,74122s] Φάση Αποφυγής Συμφόρησης: Όπως προαναφέρθηκε, το παράθυρο συμφόρησης γίνεται cwnd=(Data_in_flight)/2=24500B. Ίδια τιμή λαμβάνει και το ssthresh. Οπότε, όπως παρατηρούμε και από τις μετρήσεις η ροή των δεδομένων αυξάνεται γραμμικά και όχι εκθετικά. Όμως ανιχνεύεται ξανά περιορισμένη συμφόρηση.

[1,74122 – 5,34461s] Φάση Αποφυγής Συμφόρησης: Όπως και προηγουμένως, το νέο παράθυρο συμφόρησης γίνεται cwnd=(Data_in_flight)/2=11500B και το ssthresh υποχωρεί εξίσου. Από εκεί και πέρα η αύξηση το cwnd γίνεται γραμμικά, αφού cwnd>ssthresh. Αυτό σημαίνει ότι το cwnd αυξάνεται κατά 1 κάθε φορά που επιστρέφει πλήθος ACK ίσο με τα πακέτα του cwnd. Η γραμμική αύξηση συνεχίζεται μέχρις ότου cwnd=26965B, όπου εντοπίζεται περιορισμένη συμφόρηση.

Η ίδια συμπεριφορά επαναλαμβάνεται μέχρι το τέλος της μετάδοσης.

Ερώτημα 1β

Throughput=(Σταλθέντα Δεδομένα [B]) / (Χρόνος [s]) = 2254000B / 21,1609s = 106,517 KBps

Ερώτημα 2α

1η περίπτωση: bandwidthRB=z+2=5+2= 7 Mbps
Το cwnd παίρνει μεγαλύτερες τιμές από την αρχική προσομοίωση. Μάλιστα το παράθυρο συμφόρησης παίρνει μέγιστη τιμή cwnd=112892B ενώ στην αρχική προσομοίωση ήταν cwnd=51000B.

2η περίπτωση: bandwidthRB=10-z=10-5= 5 Mbps
Το cwnd παίρνει μεγαλύτερες τιμές από την αρχική προσομοίωση, αλλά από αυτή της πρώτης περίπτωσης. Το παράθυρο συμφόρησης παίρνει μέγιστη τιμή cwnd=83936B.

Από τα αρχικά δεδομένα γνωρίζουμε πως:
bandwidthAR= 15Mbps και delayAR= 20ms και
bandwidthRB= 1Mbps και delayRB= 40ms

Οπότε εύκολα μπορούμε να εντοπίσουμε το αδιαχώρητο (bottleneck) στο κανάλι RB. Αυτό σημαίνει ότι στο δρομολογητή (router) ο ρυθμός άφιξης (λ) είναι μεγαλύτερος από το ρυθμό εξυπηρέτησης (μ). Με άλλα λόγια, όσο αυξάνεται η ροή των δεδομένων από τον αποστολέα, θα υπάρξει κάποια στιγμή όπου η ουρά αναμονής του router θα γεμίσει και τότε θα πέσει κάποιο πακέτο (drop). Δηλαδή θα εμφανιστεί συμφόρηση.

Όσο εμείς αυξάνουμε το bandwidthRB, αυξάνουμε το ρυθμό εξυπηρέτησης του δρομολογητή. Επομένως, όσο λ>μ, αν αυξήσουμε το bandwidthRB, τότε καθυστερούμε την εμφάνιση της συμφόρησης. Έτσι προκύπτουν και μεγαλύτερες τιμές του cwnd.

Ερώτημα 2β

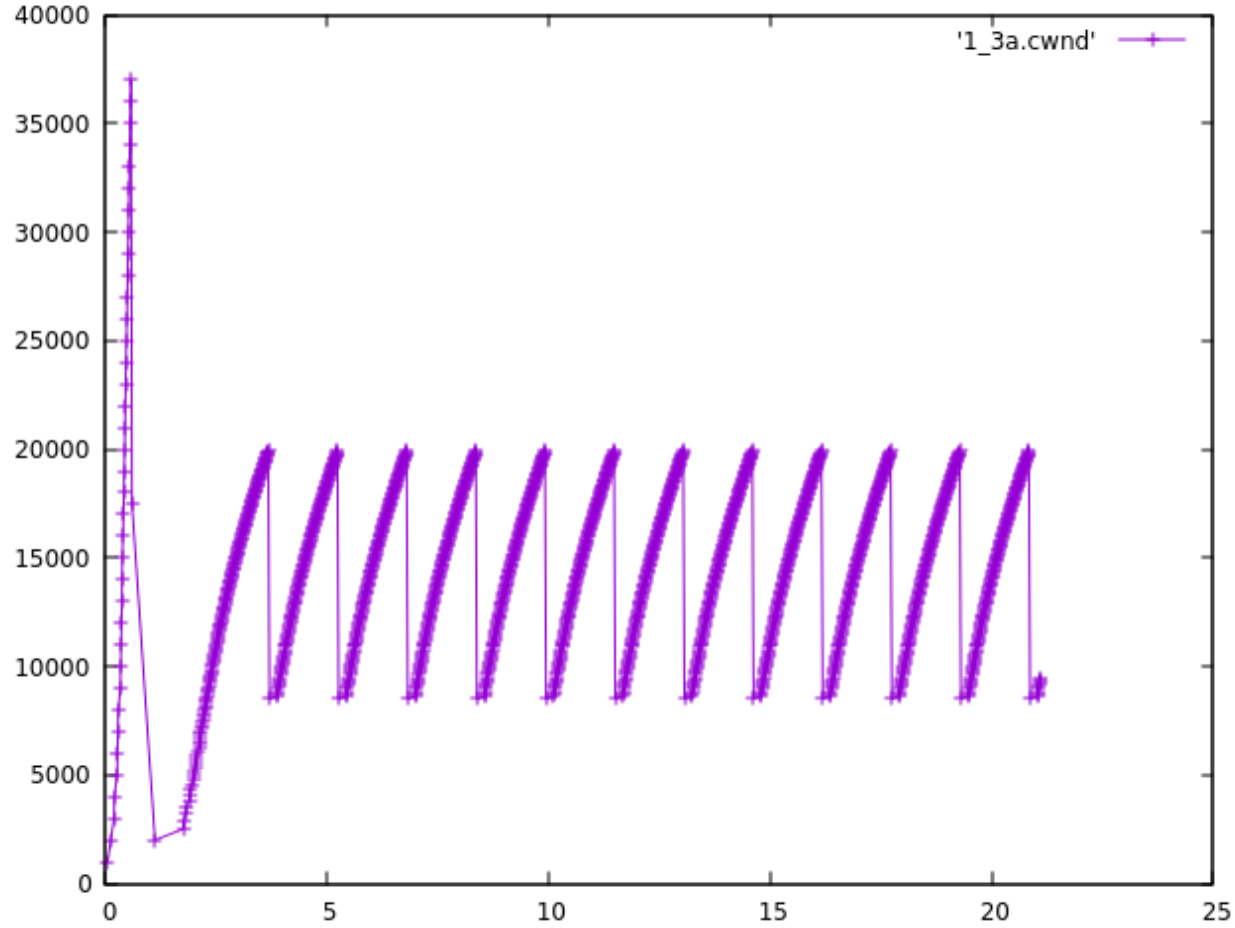
1η περίπτωση:
Throughput=(Σταλθέντα Δεδομένα [B]) / (Χρόνος [s])= 8870000B / 20,2436s = 438,163 KBps

2η περίπτωση:
Throughput=(Σταλθέντα Δεδομένα [B]) / (Χρόνος [s])= 8084000B / 20,2549s = 399,133 KBps

Παρατηρούμε πως αυξάνεται η απόδοση του δικτύου, αφού «θεραπεύουμε» ως ένα βαθμό το bottleneck.

Ερώτημα 3α

bandwidthRB= 1Mbps και delayRB= 11ms



[0 – 0,614434s] Φάση Slow Start: Όπως περιγράφηκε και στο πρώτο υποερώτημα, το παράθυρο συμφόρησης αυξάνεται εκθετικά, μέχρις ότου cwnd=17500B.

[0,614434 – 1,13279s] Timeout: Η συμφόρηση είναι καταστροφική, οπότε δεν εφαρμόζονται οι μηχανισμοί fast retransmit και fast recovery. Το παράθυρο συμφόρησης γίνεται cwnd=2000B και το slow start threshold γίνεται το μισό του προηγούμενου cwnd, δηλαδή ssthresh=cwnd/2=17500/2=88750B.

[1,13279 – 2,3147s] Φάση Slow Start (cwnd<ssthresh): Εκθετική αύξηση του cwnd μέχρι να ξεπεραστεί το ssthresh (cwnd=88791B).

[2,3147 – 3,71405s] Φάση Αποφυγής Συμφόρησης (cwnd>ssthresh): Γραμμική αύξηση του cwnd μέχρι να εμφανιστεί συμφόρηση (cwnd=19940B).

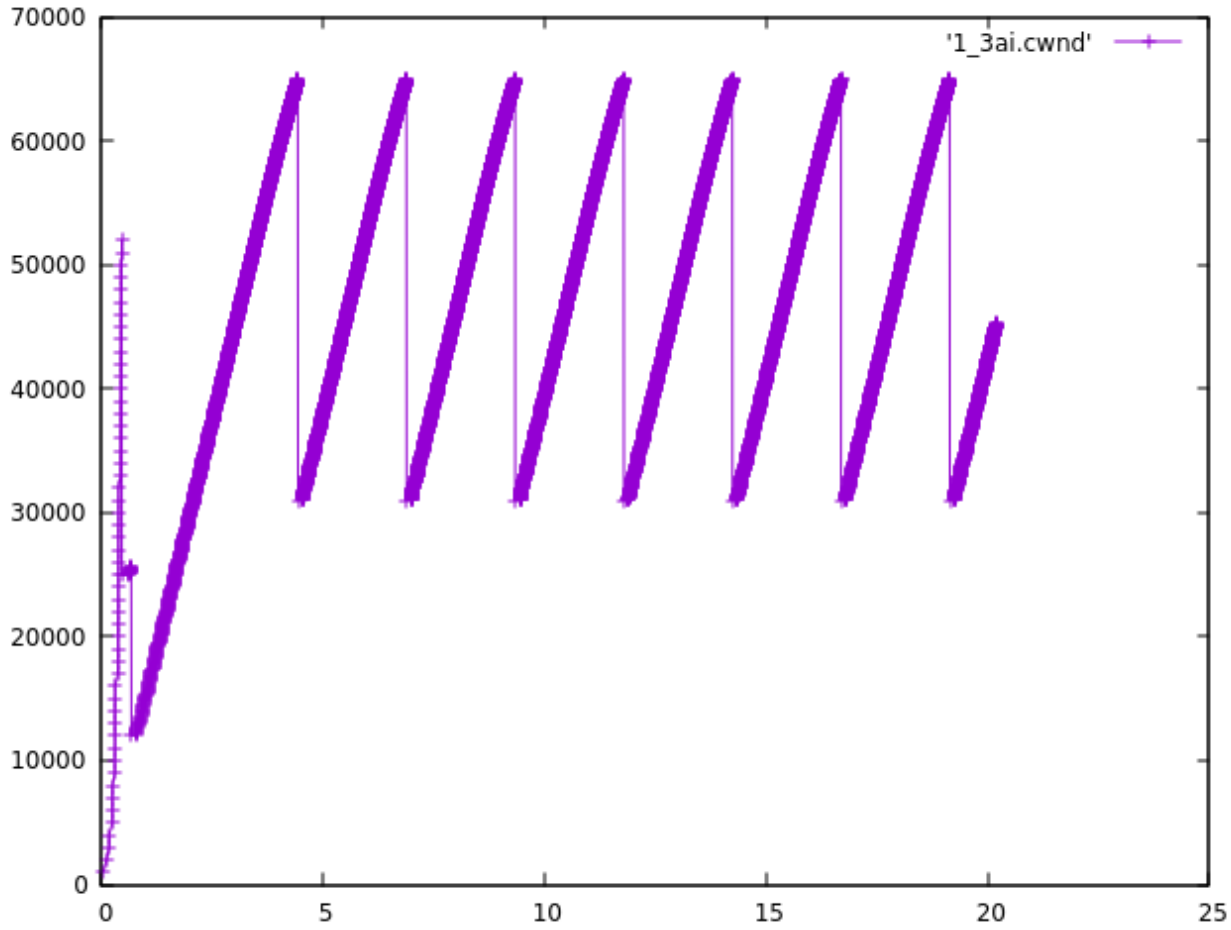
Η συμφόρηση δεν είναι καταστροφική, οπότε εφαρμόζονται οι μηχανισμοί fast retransmit και fast recovery, όπως προαναφέρθηκαν στο πρώτο υποερώτημα.

[3,71405 – 5,2736s] Φάση Αποφυγής Συμφόρησης: Η συμφόρηση που εμφανίστηκε ήταν περιορισμένη, οπότε όπως προαναφέρθηκε στο πρώτο υποερώτημα cwnd=Data_in_flight/2. Ομοίως υποχωρεί και το ssthresh. Έπειτα ακολουθεί γραμμική αύξηση του cwnd μέχρι την επόμενη συμφόρηση (που είναι επίσης περιορισμένη).

Από εδώ και πέρα επαναλαμβάνεται η ίδια συμπεριφορά, όπως και στο πρώτο υποερώτημα.

Throughput=(Σταλθέντα Δεδομένα [B]) / (Χρόνος [s])= 2345000B / 21,1048s = 111,112 KBps

bandwidthRB= 7Mbps και delayRB= 11ms



[0 – 0,471907s] Φάση Slow Start: Το παράθυρο συμφόρησης αυξάνεται εκθετικά.

Η συμφόρηση είναι περιορισμένη, άρα εφαρμόζονται οι μηχανισμοί fast retransmit και fast recovery. Το παράθυρο συμφόρησης μειώνεται στο μισό των Data_in_flight. Ομοίως υποχωρεί και το ssthresh.

[0,471907 – 0,690047s] Φάση Αποφυγής Συμφόρησης: Το παράθυρο συμφόρησης αυξάνεται γραμμικά.

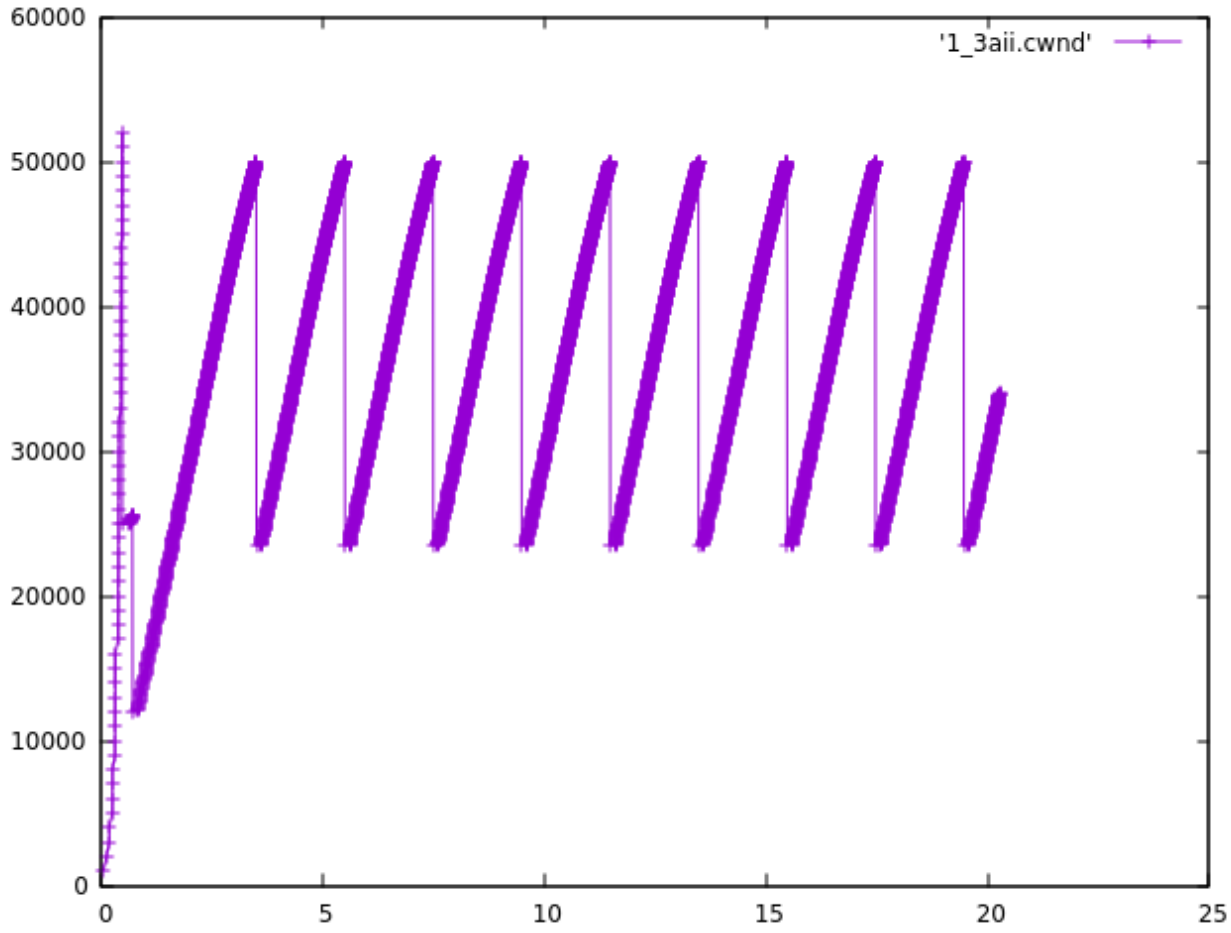
Η συμφόρηση είναι περιορισμένη, άρα εφαρμόζονται οι μηχανισμοί fast retransmit και fast recovery. Το παράθυρο συμφόρησης μειώνεται στο μισό των Data_in_flight. Ομοίως υποχωρεί και το ssthresh.

[0,690047 – 4,44546s] Φάση Αποφυγής Συμφόρησης: Το παράθυρο συμφόρησης αυξάνεται γραμμικά μέχρι την τιμή 64950B.

Ακολουθεί περιορισμένη συμφόρηση και από αυτό το σημείο κι έπειτα επαναλαμβάνεται η ίδια συμπεριφορά.

Throughput=(Σταλθέντα Δεδομένα [B]) / (Χρόνος [s])= 13281000B / 20,1967s = 657,583 KBps

bandwidthRB= 5Mbps και delayRB= 11ms



[0 – 0,485611s] Φάση Slow Start: Το παράθυρο συμφόρησης αυξάνεται εκθετικά μέχρις ότου cwnd=5200B.

Εμφανίζεται περιορισμένη συμφόρηση και εφαρμόζονται οι μηχανισμοί fast retransmit και fast recovery.

[0,485611 – 0,715888s] Φάση Αποφυγής Συμφόρησης: Το παράθυρο συμφόρησης αυξάνεται γραμμικά μέχρις ότου cwnd=25625B.

Εμφανίζεται περιορισμένη συμφόρηση και εφαρμόζονται οι μηχανισμοί fast retransmit και fast recovery.

[0,715888 – 3,50892s] Φάση Αποφυγής Συμφόρησης: Το παράθυρο συμφόρησης αυξάνεται γραμμικά μέχρις ότου cwnd=49964B.

Εμφανίζεται περιορισμένη συμφόρηση και εφαρμόζονται οι μηχανισμοί fast retransmit και fast recovery.

Στη συνέχεια παρουσιάζεται όμοια συμπεριφορά.

Throughput=(Σταλθέντα Δεδομένα [B]) / (Χρόνος [s])= 10043000B / 20,2715s = 495,425 KBps

Ερώτημα 3β

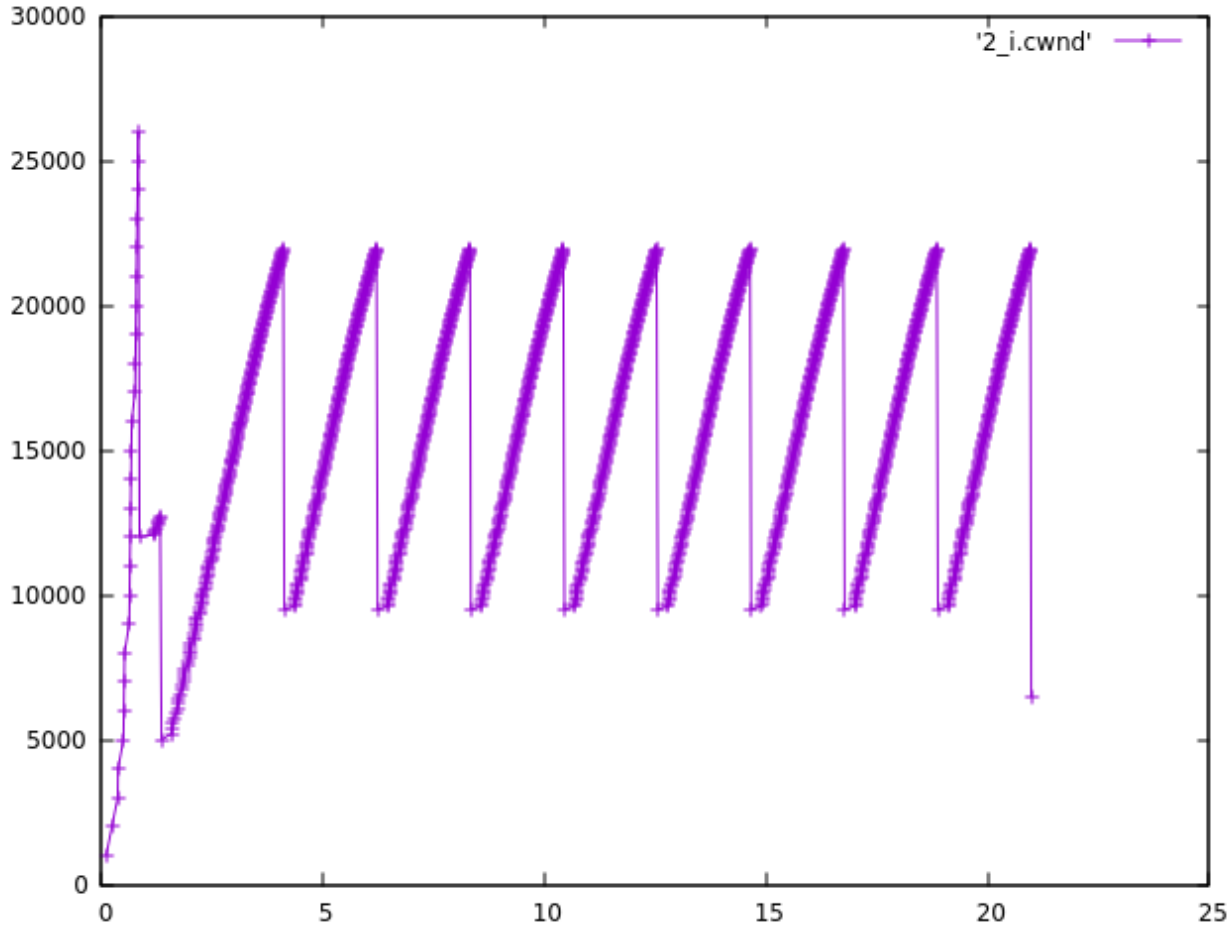
Παρατηρούμε ότι σε κάθε επιμέρους περίπτωση μειώνεται το cwnd και αυξάνεται η απόδοση.

Και οι δύο παρατηρήσεις είναι αναμενόμενες από τη στιγμή που το μόνο που κάναμε ήταν να μειώσουμε το delayRB. Με αυτόν τον τρόπο, μειώσαμε τη χωρητικότητα του καναλιού RB, με αποτέλεσμα η συμφόρηση να εμφανίζεται νωρίτερα. Αυτό συνεπάγεται και μικρότερο παράθυρο αποφυγής. Ταυτόχρονα η αποστολή των πακέτων από τον αποστολέα στον δρομολογητή και η προώθηση τους στον παραλήπτη καθίσταται γρηγορότερη, με αποτέλεσμα να αυξηθεί η απόδοση του δικτύου.

Άσκηση 2

1η περίπτωση

Queuesize = 10-5 = 5

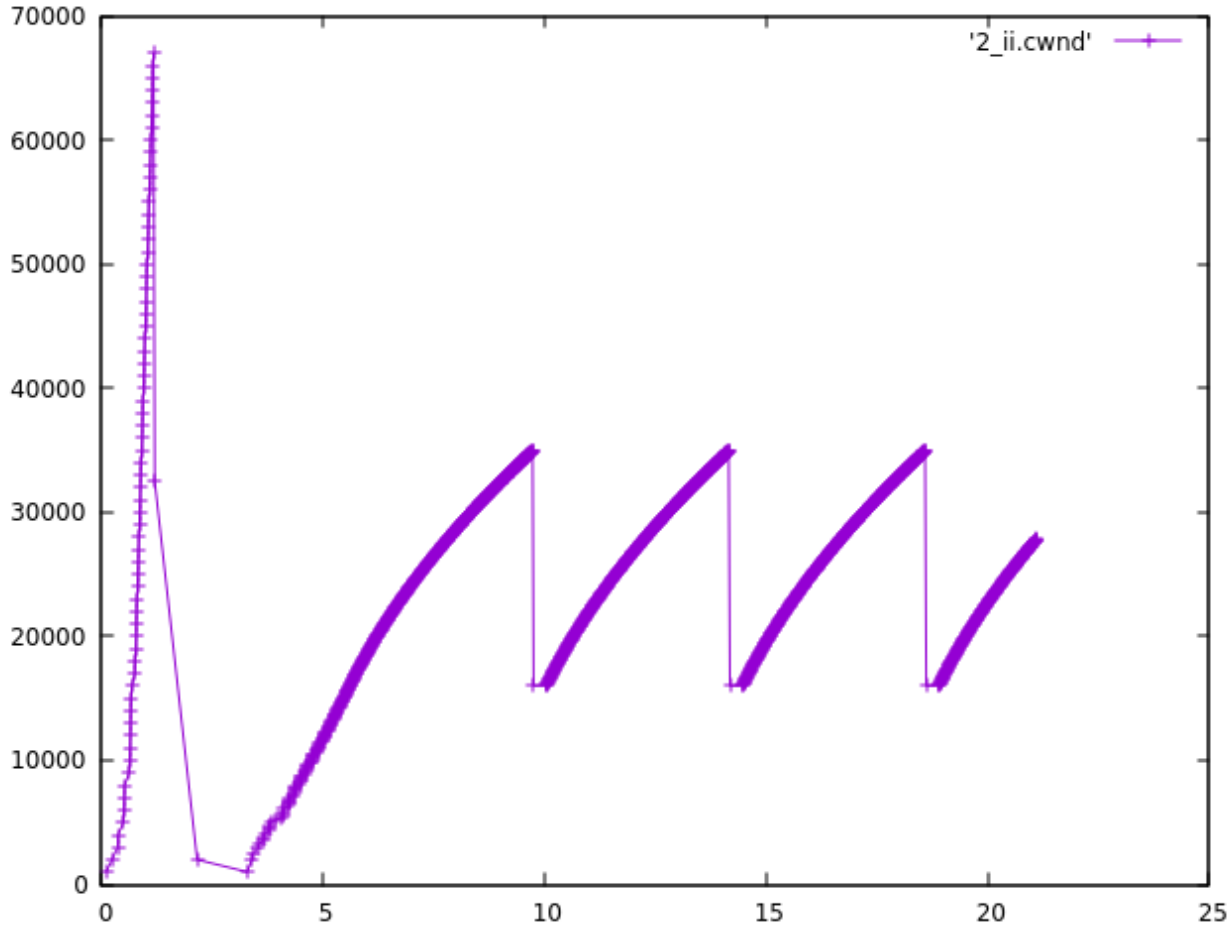


Παρατηρώ ότι το διάγραμμα έχει γίνει πιο πριονωτό. Αυτό σημαίνει βελτίωση στην ανταποκριτικότητα και επιδείνωση στην ομαλότητα του συστήματος. Επίσης παρατηρώ ότι το παράθυρο συμφόρησης παίρνει μικρότερες τιμές, το οποίο οφείλεται στο γεγονός ότι η συμφόρηση επέρχεται γρηγορότερα, αφού μειώνεται η ουρά αναμονής του δρομολογητή.

Throughput=(Σταλθέντα Δεδομένα [B]) / (Χρόνος [s])= 2084000B / 20,9888s = 99,291 KBps

2η περίπτωση

Queuesize = 10+8 = 18

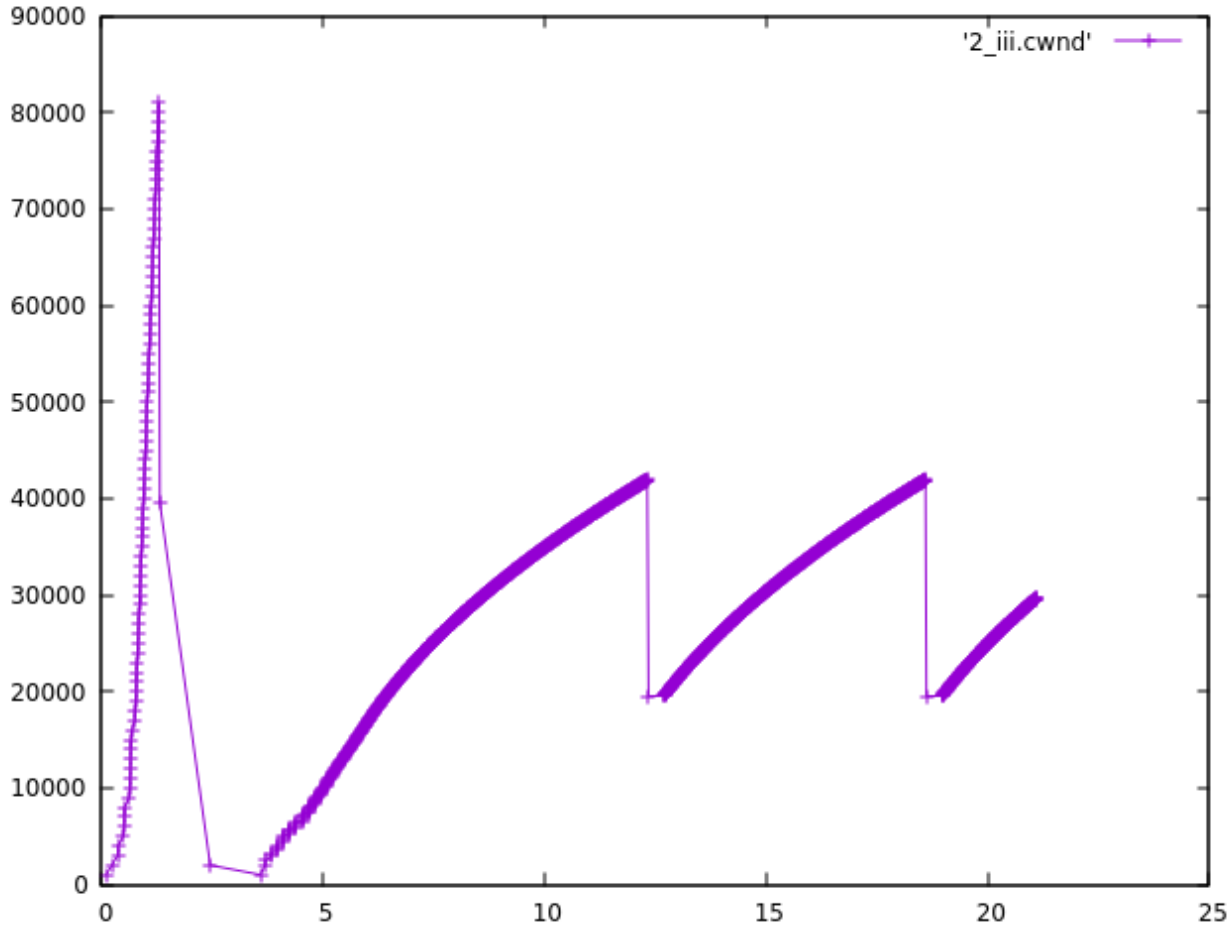


Παρατηρώ ότι το διάγραμμα έγινε λιγότερο πριονωτό (πιο ομαλό). Αυτό θα επιφέρει τα αντίθετα αποτελέσματα από την πρώτη περίπτωση. Επίσης, το γεγονός ότι παρατηρείται αύξηση στις τιμές που παίρνει το παράθυρο συμφόρησης, προκύπτει από την αύξηση της ουράς αναμονής του δρομολογητή. Η οποία συνεπάγεται την καθυστέρηση της εμφάνισης συμφόρησης. Ακόμη παρατηρώ πως η πρώτη συμφόρηση του δικτύου είναι καταστροφική (και όχι περιορισμένη). Αυτό οφείλεται πάλι στην αύξηση της ουράς αναμονής του καταχωρητή. Από τη στιγμή που θα αργήσει να πέσει κάποιο πακέτο, θα αργήσει να τελειώσει και η φάση slow start, όπου το παράθυρο συμφόρησης αυξάνεται εκθετικά. Όσο μεγαλύτερο είναι το παράθυρο συμφόρησης, τόσο αυξάνεται και η επικινδυνότητα να εμφανιστεί καταστροφική συμφόρηση.

Throughput=(Σταλθέντα Δεδομένα [B]) / (Χρόνος [s])= 2121000B / 21,1034s = 100,505 KBps

3η περίπτωση

Queuesize = 20+5 = 25



Προκύπτουν οι ίδιες παρατηρήσεις με τη δεύτερη περίπτωση, που πηγάζουν από τις ίδιες αιτίες. Μόνο που λόγω της περαιτέρω αύξησης της ουράς αναμονής του δρομολογητή, υπάρχει μεγαλύτερη αύξηση στις τιμές που παίρνει το παράθυρο συμφόρησης, όπως επίσης και καλύτερη εξομάλυνση. Βέβαια το κόστος της εξομάλυνσης είναι η καθυστέρηση αναμονής στην ουρά (Queuing Delay).

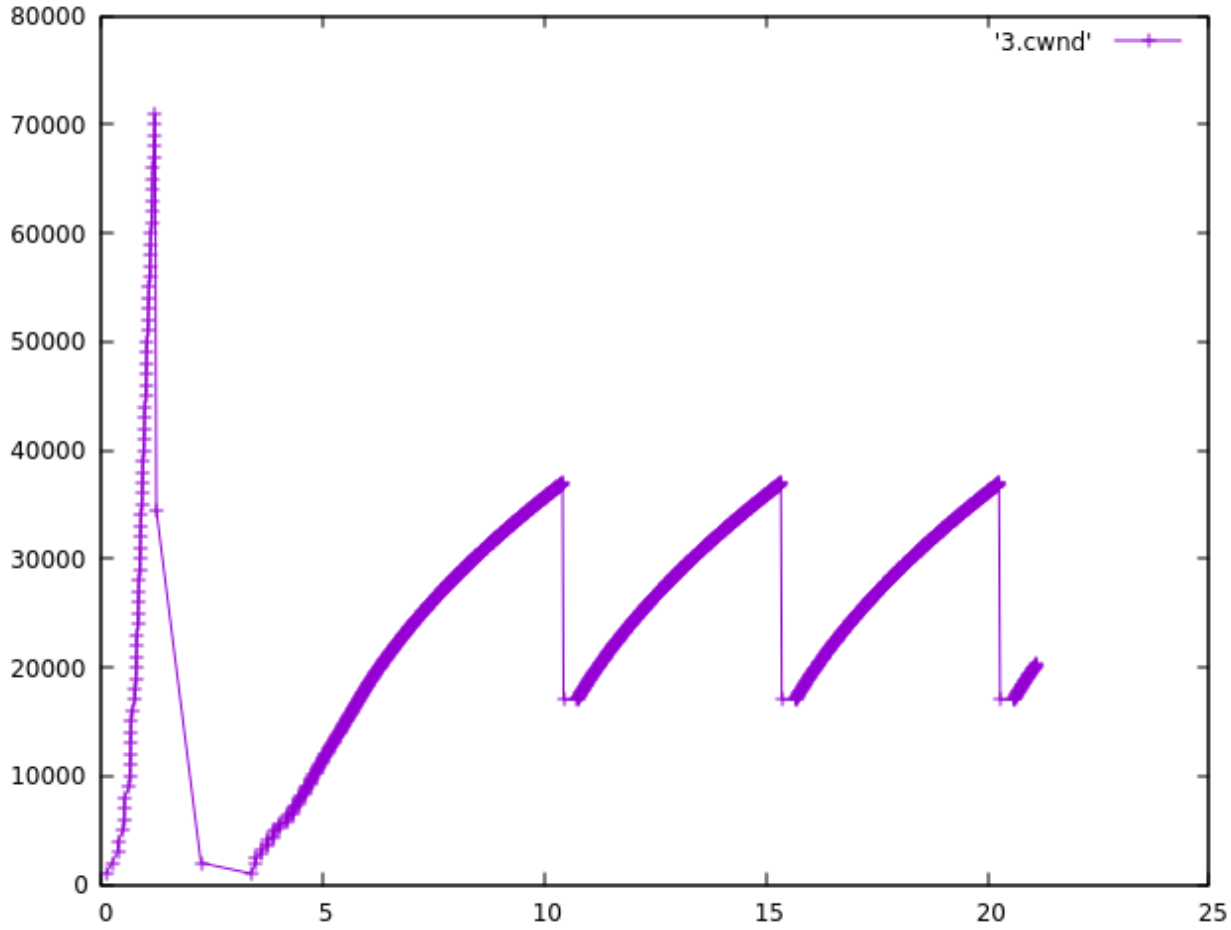
Throughput=(Σταλθέντα Δεδομένα [B]) / (Χρόνος [s])= 2116000B / 21,1034s = 100,268 KBps

Άσκηση 3

Ερώτημα 1α

Χωρίς RED

queuesize = 20

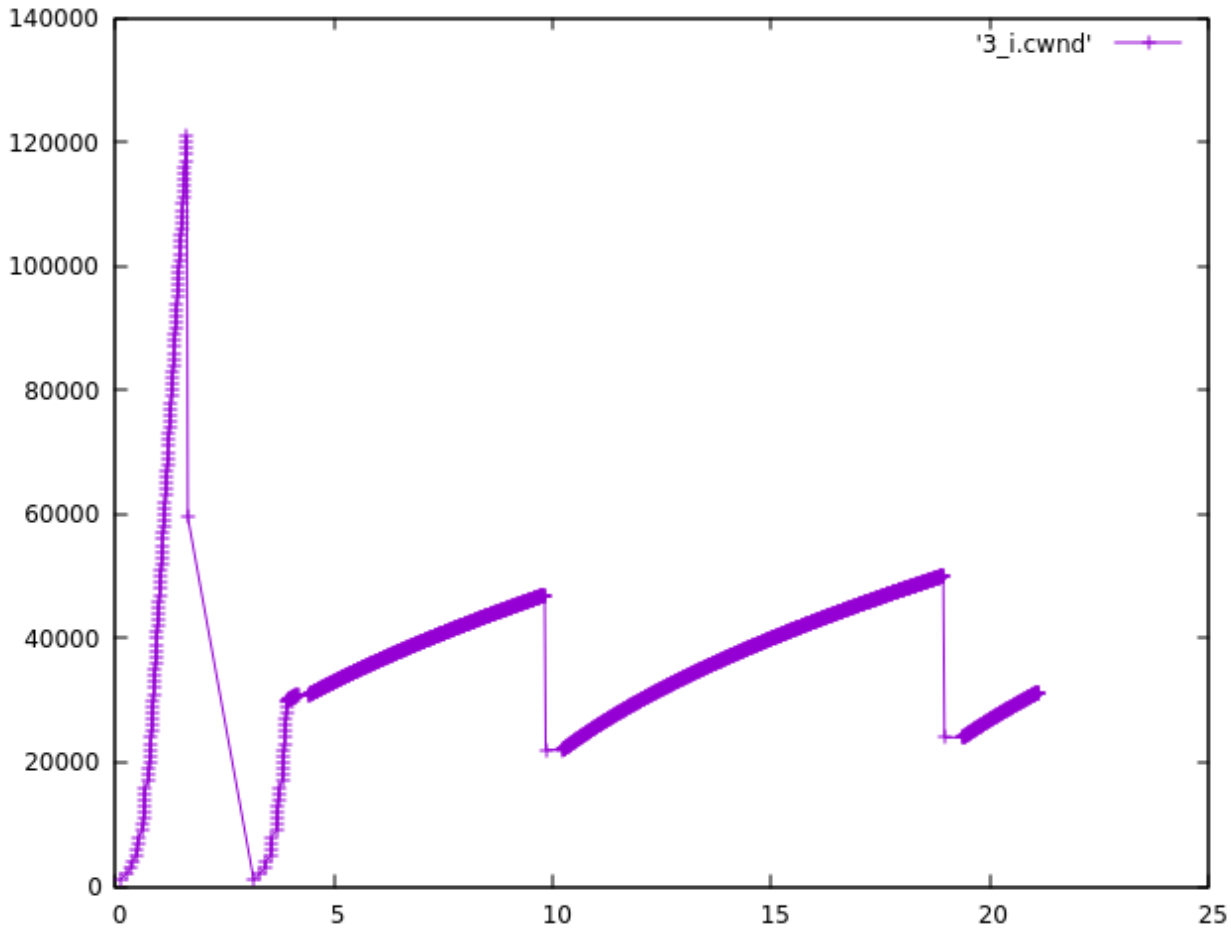


Με RED

queue_weight = 0,002

maximum_threshold = 0,5*queuesize = 0,5*20 = 10

minimum_threshold = (1/3)*maximum_threshold = 3.33 => minimum_threshold = 4



Ερώτημα 1β

Χωρίς RED:
Throughput=(Σταλθέντα Δεδομένα [B]) / (Χρόνος [s])= 2122000B / 21,1034s = 100,553 KBps

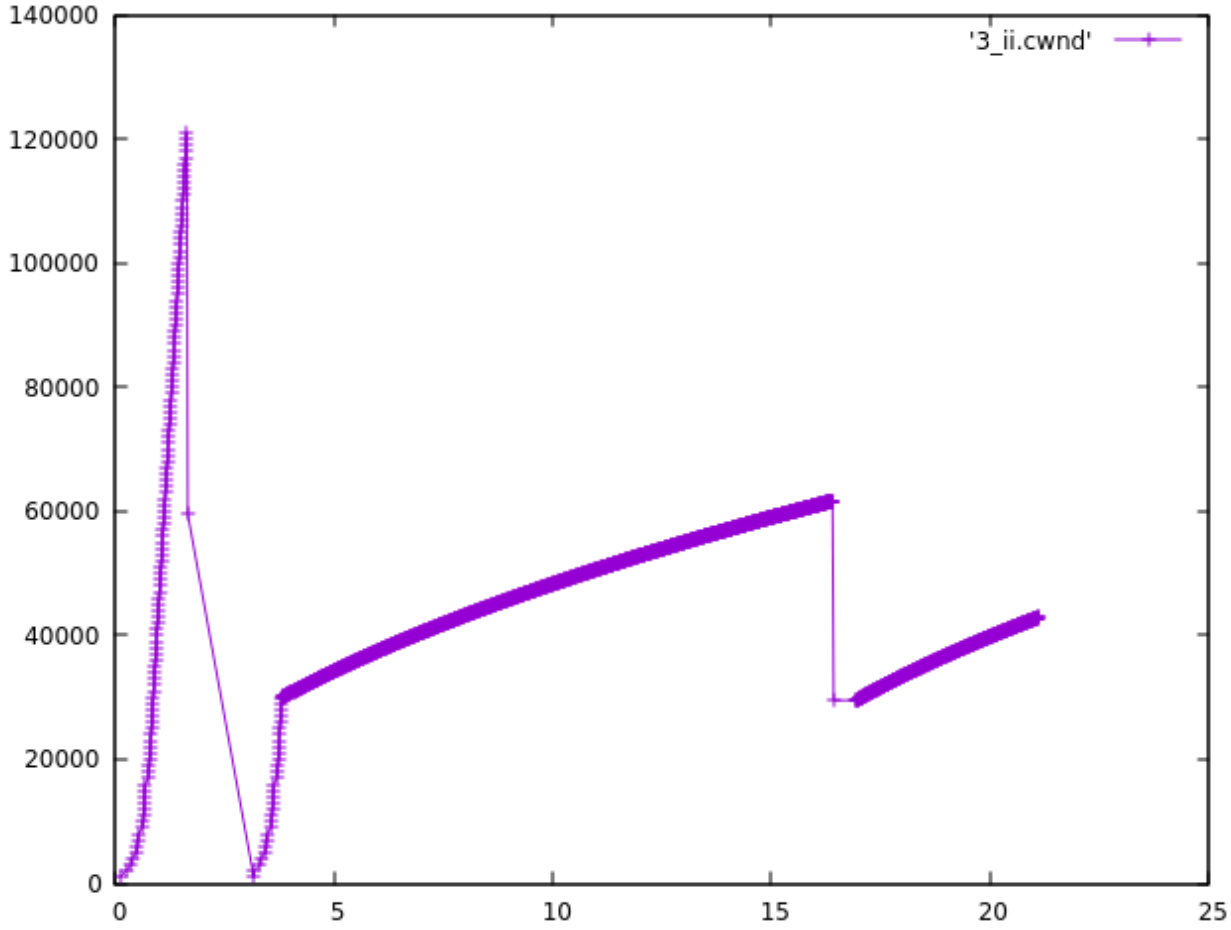
Με RED:
Throughput=(Σταλθέντα Δεδομένα [B]) / (Χρόνος [s])= 2281000B / 21,106s = 108,074 KBps

Για να γίνει αντιληπτή η αύξηση της απόδοσης με την εφαρμογή της ενεργούς διαχείρισης της ουράς RED, αρκεί να αντιληφθούμε ποια είναι τα πλεονεκτήματα που προσφέρει στη συγκεκριμένη περίπτωση. Προφανώς, λόγω της τοπολογίας του δικτύου που έχουμε, ο RED δε συμβάλει ούτε στην αύξηση της απόδοσης δικαιοσύνης, ούτε στην αποφυγή των συγχρονισμένων ροών. Εκείνο που πετυχαίνει είναι η μείωση της παρατεταμένης συμφόρησης, είτε πρόκειται απλά για απορρίψεις ριπής, είτε για καταστροφική συμφόρηση. Δηλαδή, προκαλώντας σκόπιμα την πρόωρη απόρριψη ενός πακέτου -ώστε η έλλειψη επιβεβαίωσης του να προκαλέσει την πολλαπλασιαστική μείωση της ροής δεδομένων από τη μεριά του αποστολέα- καταφέρνει τελικά να μειώσει τις απώλειες. Αυτό έχει σαν αποτέλεσμα, στον ίδιο χρόνο που τρέχει η προσομοίωση να αυξάνεται τελικά η συνολική ροή δεδομένων και συνεπώς να αυξάνεται η απόδοση.

Ερώτημα 2α

maximum_threshold = 20 - (0,5*z)= 20-2,5 = 17,5 => maximum_threshold=18

minimum_threshold = maximum_threshold-(0,5*y) = 18-(0,5*8) = 18-4 = 14



Σύγκριση με το διάγραμμα που δεν εφαρμόζεται η πολιτική RED:

Όπως παρατηρούμε, έχουμε μία πολύ πιο ομαλή καμπύλη, η οποία φτάνει σε πολύ μεγαλύτερες τιμές παραθύρου συμφόρησης.

Σύγκριση με το διάγραμμα που εφαρμόζεται η πολιτική RED με τιμές max_{th}=10 και min_{th}=4:

Παρατηρούμε ότι μέχρι και τη χρονική στιγμή t=3,17664s τα διαγράμματα είναι ίδια. Ακόμη παρατηρούμε ότι, όπως και προηγουμένως, το διάγραμμα είναι ομαλότερο (λιγότερο πριονωτό) και αποκτά μεγαλύτερες τιμές παραθύρου συμφόρησης στο τμήμα που διαφέρουν.

Για να εξηγήσουμε αυτές τις παρατηρήσεις θα πρέπει να εμβαθύνουμε λίγο παραπάνω στον τρόπο που λειτουργεί ο αλγόριθμος RED όσον αφορά τη σκόπιμη απόρριψη ενός πακέτου.

Το μέσο μήκος της ουράς υπολογίζεται ως εξής:

AvgLen = (1-Weight)*AvgLen+Weight*SampleLen =>

AvgLen = 0.998*AvgLen+0.002*SampleLen,

όπου SampleLen είναι το μήκος της ουράς κάθε φορά που παραδίδεται ένα πακέτο στην ουρά.

- Αν AvgLen<MinThreshold, τότε το πακέτο καταχωρείται. Άρα η πιθανότητα απόρριψης του πακέτου είναι 0%
- Αν AvgLen>MaxThreshold, τότε το πακέτο πέφτει. Άρα η πιθανότητα απόρριψης του πακέτου είναι 100%
- Αν MinThreshold< AvgLen< MaxThreshold, τότε το πακέτο απορρίπτεται με μία πιθανότητα P, η οποία αυξάνεται όσο το μέσο μήκος της ουράς πλησιάζει στο MaxThreshold.

Έτσι στην περίπτωση όπου MaxThreshold=18 και MinThreshold=14, το παράθυρο συμφόρησης θα παίρνει μεγαλύτερη τιμή, αφού ο buffer έχει μεγαλύτερη ικανότητα καταχώρισης από όταν θέτουμε MaxThreshold=10 και MinThreshold=4. Η ικανότητα καταχώρισης, προφανώς, δεν έχει να κάνει με το μέγεθος της ουράς, το οποίο είναι ίδιο και στις δύο περιπτώσεις, αλλά στη δυνατότητα του router να δέχεται πακέτα χωρίς να τα απορρίπτει σκόπιμα. Όπως έχουμε αναφέρει, όταν πέφτει ένα πακέτο, το παράθυρο συμφόρησης υποχωρεί.

Το γεγονός ότι το διάγραμμα είναι πανομοιότυπο μέχρι και κάποια χρονική στιγμή, οφείλεται στο γεγονός, πως οι δρομολογητές και στις δυο περιπτώσεις έχουν όμοια συμπεριφορά, όσο το μέσο μήκος της ουράς τους δεν ξεπερνάει το ελάχιστο min_{th}, δηλαδή το 4.

Ερώτημα 2β

Throughput=(Σταλθέντα Δεδομένα [B]) / (Χρόνος [s])= 2321000B / 21,1016s = 109,992 KBps

Η απόδοση είναι μεγαλύτερη τόσο από αυτή της προσομοίωσης χωρίς την ενεργή διαχείριση της ουράς, όσο και από την προσομοίωση με RED (max_{th}=10, min_{th}=4).

Παρατηρούμε πως η απόδοση του δικτύου αυξάνεται. Αυτό μπορούμε να το αποδώσουμε σε ένα συνδυασμό δύο γεγονότων. Από τη μία, ο δρομολογητής έχει μεγαλύτερη ικανότητα καταχώρισης -με τον τρόπο που περιγράφηκε στο προηγούμενο υποερώτημα- και παράλληλα συνεχίζει να έχει την ίδια ικανότητα όσον αφορά την αποφυγή παρατεταμένης συμφόρησης.

Με αυτόν τον τρόπο, εξακολουθεί να πετυχαίνει το στόχο του -την αποφυγή παρατεταμένης συμφόρησης- ενώ παράλληλα επιτρέπει στο παράθυρο συμφόρησης να πάρει μεγαλύτερες τιμές. Άρα θα αυξηθεί η ροή δεδομένων και ως συνέπεια αυτού θα αυξηθεί και η απόδοση.

Πηγές

- Διαδικτυακά Πρωτόκολλα, Βασίλης Θ. Τσαουσίδης
- Εργαστηριακά Μαθήματα στα Δίκτυα και Διαδίκτυα Υπολογιστών, Β. Τσαουσίδης, Ε. Μαμάτας, Ι. Ψαρράς, Ε. Κοσμίδης, Σ. Δημητρίου
- Σημειώσεις εργαστηρίων του μαθήματος «Δίκτυα Υπολογιστών ΙΙ»
- The ns Manual, Kevin Fall, Kannan Varadhan