ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
ΣΤΗΝ ΕΠΙΣΤΗΜΗ ΤΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Διπλωματική Εργασία
Μεταπτυχιακού Διπλώματος Ειδίκευσης

«*Improving Download Time & Traffic via Locality in BitTorrent Protocol*»

Αντώνιος Πρελορέντζος

Επιβλέπων: κ. Γεώργιος Ξυλωμένος

ΑΘΗΝΑ, ΙΟΥΝΙΟΣ 2010

ΠΕΡΙΛΗΨΗ

Το BitTorrent είναι ένα πρωτόκολλο διαμοιρασμού αρχείων peer-to-peer που χρησιμοποιείται κυρίως για τη διανομή αρχείων μεγάλου μεγέθους. Η απλότητα με την οποία είναι υλοποιημένο το BitTorrent, μας επιτρέπει να κάνουμε διάφορες παραμετροποιήσεις και να βελτιώσουμε τη συμπεριφορά του πρωτοκόλλου. Στην παρούσα διπλωματική εργασία προσπαθούμε να εκμεταλλευτούμε τους γειτονικούς peers έτσι ώστε με κριτήρια τοπικότητας να πετύχουμε καλύτερους χρόνους κατεβάσματος αρχείων καθώς και καλύτερη διαχείριση κίνησης μεταξύ των Αυτόνομων Συστημάτων (AS).

Τα κριτήρια τοπικότητας έχουν αρχίσει ολοένα και περισσότερο να απασχολούν τους ερευνητές σε αυτή την περιοχή και πολλές προσεγγίσεις είναι αρκετά καλές ώστε να τείνουν να επιβληθούν σε επόμενη αλλαγή του πρωτοκόλλου. Επίσης σε μια εποχή στην οποία η κίνηση περιεχομένου BitTorrent στο Internet αποτελεί κάτι παραπάνω από το 50% της διαδικτυακής κίνησης, προσπαθώντας να "εγκλωβίσουμε" την κίνηση μέσα στα δίκτυα των παρόχων (ISPs), ίσως ωφελήσει τους παρόχους να μειώσουν την εξερχόμενη κίνηση και της δαπάνες που αυτή δημιουργεί.

Στη δική μας περίπτωση χρησιμοποιήθηκε ένα δικό μας module για το BitTorrent το οποίο δημιουργήθηκε για το περιβάλλον εξομοίωσης OMNeT++ από φοιτητές του τμήματος. Ο σκοπός μας είναι να τροποποιήσουμε τα κριτήρια επιλογής των peers που δέχεται κάθε peer από τον tracker, έτσι ώστε να μην είναι τελείως τυχαία όπως είναι στη γενική περίπτωση του πρωτοκόλλου αλλά να ευνοούν την επιλογή γειτονικών χρηστών, και να δούμε πως αυτό επηρεάζει την απόδοση του module μας.

Θα παρουσιαστούν αποτελέσματα από πειράματα που έγιναν με διάφορες παραμέτρους και θα περιγραφούν σενάρια χρήσης του παραπάνω μοντέλου που έγιναν πάνω στη γνωστή για περιβάλλοντα εξομοίωσης γεννήτρια τοπολογιών GT-ITM.

Λέξεις κλειδιά

BitTorrent, block, end-game mode, INET, OMNeT++, OverSim, peer, peer-to-peer, piece, seeder, tracker, traffic, ανέβασμα αρχείου, αρχείο, εξομοίωση δικτύου, λήψη αρχείου, πρωτόκολλο.

ABSTRACT

BitTorrent is a peer-to-peer file transfer protocol mainly used for large file transfers. The simplicity of BitTorrent's implementation, permits us make different type of changes in its structure to improve protocol behavior. In this master thesis we try to utilize neighboring peers using locality criteria in order to achieve better average file download times and also better traffic management among the Autonomous Systems (AS).

The locality criteria become more and more attractive in this area of research and many approaches are good enough to become a part of a next update of the BitTorrent protocol. In addition, nowadays, BitTorrent traffic has something more than a 50% of all the Internet traffic, so trying to enclose traffic into Internet Service Providers'(ISPs) networks may help them decrease inter-domain traffic and costs that this kind of traffic has.

In our case, it is used a module for BitTorrent that was created for OMNeT++ network simulation environment by students of our department. Our goal is to modify it, so that the criteria of peer selection will not be random as in the default case but favorable to the selection of neighboring peers, and then observe the performance of our module.

The results and the measurements of the experiments are presented, with all kind of parameters used on the above model, using the well known topology generator GT-ITM for simulation models.

Key words

BitTorrent, block, download, end-game mode, file, INET, network simulation, OMNeT++, OverSim, peer, peer-to-peer, piece, protocol, seeder, tracker, traffic, upload.

Ευχαριστίες

Η παρούσα διπλωματική εργασία πραγματοποιήθηκε στα πλαίσια απόκτησης μεταπτυχιακού διπλώματος στην "Επιστήμη των Υπολογιστών" του τμήματος Πληροφορικής του Οικονομικού Πανεπιστημίου Αθηνών.

Θα ήθελα να εκφράσω τις ευχαριστίες στον επόπτη καθηγητή μου κ. Γεώργιο Ξυλωμένο, που πίστεψε σε μένα και μου έδωσε την ευκαιρία να ασχοληθώ σε ερευνητικό επίπεδο με το θέμα της εργασίας αυτής. Θα ήθελα ακόμη να ευχαριστήσω τον καθηγητή κ. Γεώργιο Πολύζο για την αποδοχή του να αποτελέσει το δεύτερο αξιολογητή.

Ένα πάρα πολύ μεγάλο ευχαριστώ επίσης θα ήθελα να δώσω στους Χαρίλαο Στάη και Ντίνο Κατσαρό, υποψήφιους διδάκτορες του τμήματος, για τις συμβουλές τους, τη βοήθειά τους στις δυσκολίες που συνάντησα σε θέματα υλοποίησης καθώς και για τη συνεχή συμπαράστασή τους καθ' όλη τη διάρκεια της εργασίας.

Τέλος θα ήθελα να ευχαριστήσω τους γονείς μου καθώς και τους συμφοιτητές και φίλους μου για την πλήρη στήριξή τους σε κάθε βήμα των σπουδών μου.

Πίνακας περιεχομένων

Ακρωνύμια

AS	Autonomous System
ASID	Autonomous System Identification Number
CDF	Cumulative Distribution Function
DHT	Distributed Hash Tables
GT-ITM	Georgia Tech Internetwork Topology Models
ISP	Internet Service Provider
IP	Internet Protocol
NED	NEtwork Description language
P2P	Peer-to-peer
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

ΕΥΡΕΤΗΡΙΟ ΕΙΚΟΝΩΝ – ΠΙΝΑΚΩΝ – ΓΡΑΦΗΜΑΤΩΝ

ΕΙΚΟΝΕΣ **Σελίδα**

ΠΙΝΑΚΕΣ **Σελίδα**

ΓΡΑΦΗΜΑΤΑ **Σελίδα**

Κεφάλαιο 1: Εισαγωγικά

1.1 Διαμοιρασμός Αρχείων (File Sharing)

Ο διαμοιρασμός αρχείων (file sharing), είναι η διανομή ή η παροχή πρόσβασης σε ψηφιακά αποθηκευμένη πληροφορία, όπως προγράμματα Η/Υ, πολυμέσα (εικόνα, ήχος), έγγραφα, ή ηλεκτρονικά βιβλία (e-books). Υπάρχει ποικιλία μοντέλων διανομής, αποθήκευσης και μετάδοσης. Οι κοινές μέθοδοι είναι ο διαμοιρασμός με τη χρήση κινητών μέσων (όπως CD, DVD, flopy disk, μαγνητικές ταινίες, μνήμες flash), η αποθήκευση σε κεντρικούς servers, 'hyperlinked' έγγραφα στον παγκόσμιο ιστό, και η χρήση κατανεμημένων peer-to-peer (p2p) δικτύων.

Το πρόβλημα του διαμοιρασμού αρχείων απασχόλησε και παλιές, κατανεμημένες client-server εφαρμογές, ωστόσο αυτές παρουσίασαν δυσεπίλυτα προβλήματα (Sun RPC, NFS). Το κενό αυτό το κάλυψαν οι peer-to-peer εφαρμογές, οι οποίες στηρίζονται στην εξής, απλή, ιδέα: κάθε χρήστης δημοσιοποιεί τα αρχεία που διαθέτει και πάνω σε αυτήν την πληροφορία γίνονται αναζητήσεις από τους υπόλοιπους χρήστες. Όταν κάποιος χρήστης εντοπίσει κάποιο αρχείο που επιθυμεί, ανοίγει μια απ' ευθείας σύνδεση με τον κάτοχο και το αντιγράφει στο σύστημα του.

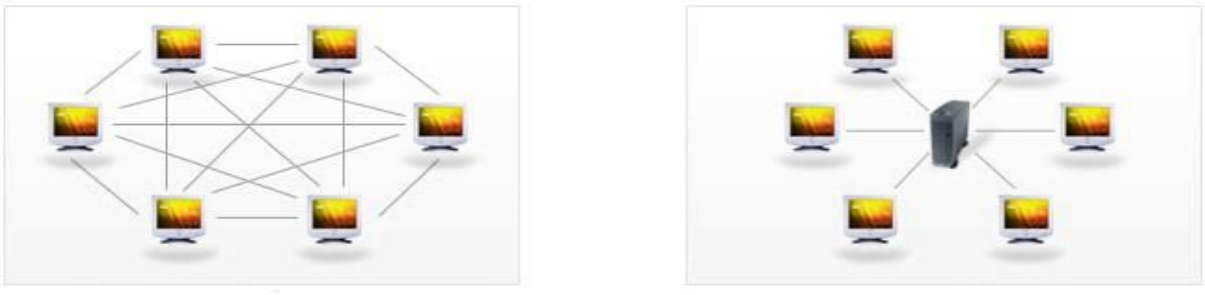
Υπάρχουν ποικίλα συστήματα αυτού του είδους, από απλές εφαρμογές με την ελάχιστη λειτουργικότητα μέχρι πιο εξειδικευμένα που παρέχουν επιπλέον υπηρεσίες, όπως κατανεμημένη αποθήκευση και άλλα. Σε γενικές γραμμές προσφέρουν τα παρακάτω χαρακτηριστικά:

- Ανταλλαγή αρχείων.
- Ασφαλής αποθήκευση.
- Υψηλή διαθεσιμότητα.
- Ανωνυμία.
- Ευχρηστία.

Τα πιο σημαντικά ζητήματα σε αυτό το πεδίο εφαρμογής είναι η κατανάλωση εύρους ζώνης, η ασφάλεια και η αναζήτηση.

1.2 Peer-to-Peer

Ο όρος συστήματα ομότιμων οντοτήτων (Peer-to-Peer, p2p) δημιουργήθηκε για να περιγράψει αυτό-οργανώσιμα, κατανεμημένα, ιδεατά δίκτυα για την από κοινού χρήση πόρων και την εκμετάλλευση της τεράστιας ποσότητας που μένει αχρησιμοποίητη στα άκρα του δικτύου. Ο διαμοιρασμός αρχείων είναι η πιο διαδεδομένη και ευρέως ανεπτυγμένη διομότιμη εφαρμογή. Ωστόσο πολλές άλλες προτείνονται και βρίσκονται υπό σχεδιασμό με σκοπό την εκμετάλλευση διαφόρων πόρων όπως η υπολογιστική ισχύς, η μνήμη, το εύρος ζώνης, και άλλοι.



Εικόνα 1: Αρχιτεκτονική P2P-based δικτύου (αριστερά) και Server-based δικτύου (δεξιά)

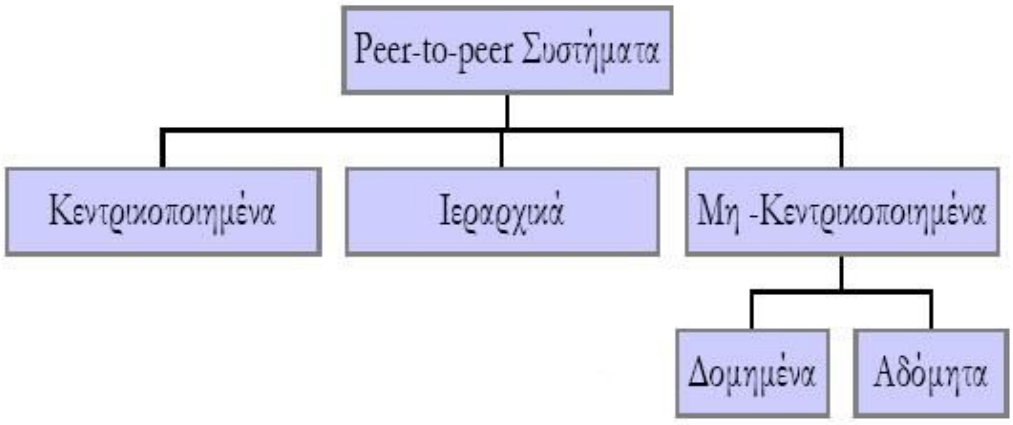
Η αρχιτεκτονική βάσει της οποίας λειτουργούν τα δίκτυα p2p είναι εντελώς διαφορετική από την αρχιτεκτονική πελάτη-εξυπηρετητή (client-server), όπως φαίνεται και στην εικόνα 1, βάσει της οποίας λειτουργεί σήμερα το μεγαλύτερο μέρος του παγκόσμιου διαδικτύου. Κύριο χαρακτηριστικό αυτής, είναι πως ο πελάτης μόνο μπορεί να χρησιμοποιήσει τους πόρους ενός εξυπηρετητή. Αντίθετα σε ένα p2p δίκτυο, ένας κόμβος του λειτουργεί ταυτόχρονα τόσο ως πελάτης όσο και ως εξυπηρετητής, αφού 'ανταλλάσσει' υπολογιστικούς πόρους με άλλους υπολογιστές. Οι σημαντικές και συχνά υπερβολικές δυνατότητες των συμβατικών προσωπικών υπολογιστών ως προς την υπολογιστική ισχύ, και η εξάπλωση των ευρυζωνικών συνδέσεων τα τελευταία χρόνια έχουν δημιουργήσει τις κατάλληλες προϋποθέσεις για την ανάπτυξη συστημάτων ομότιμων οντοτήτων. Δηλαδή συστήματα που αποτελούνται από υπολογιστές με συγκεκριμένες δυνατότητες και πανόμοιους ρόλους, για αυτό και οι χρήστες τους (peers) ονομάζονται ομότιμοι. Οι peers σχηματίζουν ένα υπερκείμενο δίκτυο (overlay network) με σκοπό την αξιοποίηση των αχρησιμοποίητων πόρων τους, όπως μνήμη υπολογιστική ισχύς, εύρος ζώνης δικτύου πρόσβασης και/ή τον διαμοιρασμό του διαθέσιμου περιεχομένου.

Ακόμα πιο ενδιαφέρουσα από τα τεχνικά θεμέλια των συστημάτων είναι οι κοινωνικές προοπτικές. Με διάφορους τρόπους διανέμουν περιεχόμενο, αποτελέσματα και έλεγχο στους χρήστες.

Οι σημαντικότερες σύγχρονες εφαρμογές ομότιμων οντοτήτων είναι οι:

- Napster
- BitTorrent
- SETI@Home,
- Gnutella
- KaZaA
- Skype
- Tribler

1.3 Κατηγοριοποίηση Peer-to-Peer συστημάτων



Εικόνα 2: Κατηγορίες p2p συστημάτων

1.3.1 Κεντρικοποιημένα peer-to-peer συστήματα

Στις κεντρικοποιημένες αρχιτεκτονικές υπάρχει ένας κεντρικός εξυπηρετητής (Directory Server) στον οποίο απευθύνουν οι κόμβοι τις ερωτήσεις τους για να πληροφορηθούν που βρίσκονται οι επιθυμητές πληροφορίες (π.χ Napster). Μια τέτοια αρχιτεκτονική αν και είναι αρκετά αποδοτική, δεν έχει την ιδιότητα της κλιμάκωσης ενώ έχει ενιαίο σημείο της αποτυχίας (bottleneck).

1.3.2 Ιεραρχικά

Οι κόμβοι οργανώνονται σε ιεραρχική δομή όπως γίνεται με τους DNS στο διαδίκτυο. Στα ιεραρχικά peer-to-peer συστήματα εισάγεται ή έννοια των "super-peers" (FastTrack). Η δομή τους μπορεί να είναι κεντρικοποιημένη ή μη.

1.3.3 Μη Κεντρικοποιημένα peer-to-peer συστήματα

Μια άλλη κατηγορία αρχιτεκτονικών είναι οι μη – κεντρικοποιημένες όπου οι κόμβοι συγκροτούν το overlay δίκτυο είτε δομημένα ακολουθώντας κανόνες για τον σχηματισμό του δικτύου, είτε αδόμητα όπου δεν υπάρχει ούτε κεντρικό directory ούτε ακριβείς οδηγίες για τον σχηματισμό τοπολογίας του δικτύου και την τοποθέτηση των περιεχομένων.

Α) Δομημένα: Στα δομημένα peer-to-peer συστήματα οι κόμβοι οργανώνονται σε δομημένο γράφο για το σχηματισμό του overlay δικτύου. Στα δεδομένα αντιστοιχίζεται ένα κλειδί και η τοποθέτηση τους στους κόμβους γίνεται με προκαθορισμένο τρόπο έτσι ώστε να διευκολύνεται η αναζήτησή τους και να επιτυγχάνεται η κλιμάκωση. Για να γίνει αυτό χρησιμοποιούνται πίνακες κατακερματισμού (Distributed Hash Tables, DHTs), οι οποίοι αντιστοιχούν κλειδιά σε τιμές. Η τοποθέτηση των αρχείων στα χαλαρά δομημένα συστήματα (Freenet) βασίζεται σε εκτιμήσεις για το που μπορεί να βρεθεί η αναζητούμενη πληροφορία. Στα αυστηρά δομημένα συστήματα τόσο η δόμηση του overlay δικτύου όσο και η τοποθέτηση των αρχείων είναι σαφώς καθορισμένη.

Ο εντοπισμός ενός αντικειμένου (δεδομένου) από μια εφαρμογή στα δομημένα συστήματα γίνεται σε μικρό αριθμό βημάτων (network hops), υπό την απαίτηση βέβαια να διατηρείται ένας μικρός πίνακας δρομολόγησης σε κάθε κόμβο.

Παραδείγματα τέτοιων συστημάτων αποτελούν τα: Content Addressable Network (CAN), Chord, Tapestry, Pastry, Kademlia και Viceroy.

Β) Αδόμητα: Στα συστήματα αυτά δεν υπάρχει καμιά δομή στο overlay δίκτυο και τα περιεχόμενα τοποθετούνται σε κόμβους στο δίκτυο χωρίς γνώση της τοπολογίας ή άλλης συσχέτισης με αυτό. Τα μη δομημένα συστήματα είναι κατάλληλα σε περιπτώσεις όπου μεγάλο πλήθος κόμβων μετέχει παροδικά στο δίκτυο χωρίς όμως αποδοτικούς μηχανισμούς αναζήτησης, κλιμάκωσης, διαθεσιμότητας. Υποστηρίζουν καλύτερα πολύπλοκες ερωτήσεις σε σχέση με τα δομημένα.

Αδόμητα peer-to-peer δίκτυα είναι τα: Napster, Gnutella, FastTrack, KaZaA, BitTorrent, κ.α.

Κεφάλαιο 2: Πρωτόκολλο BitTorrent

2.1 Τι είναι το πρωτόκολλο BitTorrent

Το BitTorrent είναι ένα πρωτόκολλο που σχεδιάζεται για τη μεταφορά των αρχείων. Είναι ίδιας φύσης με τα P2P, καθώς οι χρήστες συνδέουν ο ένας με τον άλλον άμεσα για να στείλουν και να λάβουν τις κομμάτια του αρχείου. Εντούτοις, υπάρχει ένας κεντρικός υπολογιστής (αποκαλούμενος tracker) που συντονίζει τη δράση όλων των χρηστών. Ο tracker διαχειρίζεται μόνο τις συνδέσεις, και δεν έχει οποιαδήποτε γνώση του περιεχομένου του διανομής των αρχείων, και επομένως ένας μεγάλος αριθμός χρηστών μπορεί να υποστηριχθεί με το σχετικά περιορισμένο εύρος ζώνης του tracker [1],[2],[3].

Ορολογία BitTorrent	
Block	Μέρος του piece το οποίο μεταφέρεται σε ένα απλό μήνυμα.
Choke	Όταν δεν επιτρέπεται στον peer να ζητήσει και να κατεβάσει blocks δεδομένων.
Client	Είναι το πρόγραμμα που χρησιμοποιούμε για να ανεβάζουμε/κατεβάζουμε torrents.
Leechers	Οι χρήστες που τώρα κατεβάζουν το αρχείο και άρα έχουν και μοιράζονται ένα μέρος του αρχείου μόνο. Όταν κάποιος leecher κατεβάσει ολόκληρο το αρχείο γίνεται seeder.
Optimistic unchoke	Περιοδικό unchoke σε έναν τυχαίο choked peer.
Piece	Κομμάτι δεδομένων το οποίο μπορεί να επαληθευτεί μέσω hash.
Peers	Το σύνολο των seeders και leechers.
Ratio	Ο λόγος του όγκου των ανεβασμένων δεδομένων προς τον όγκο των κατεβασμένων.
Seeders	Οι χρήστες που έχουν και μοιράζονται ολόκληρο το αρχείο (το 100%).
Swarm	Το σύνολο των κόμβων στο δίκτυο όπου υπάρχουν οι peers.
Torrents	Torrents αποκαλούμε γενικά τα αρχεία στον tracker, γράφοντας όμως (.torrent) θα εννοούμε το μικρό αρχειάκι τέτοιου τύπου τα οποία κατεβάζουμε από τον tracker προκειμένου να συνδεθούμε με τους άλλους χρήστες και να κατεβάσουμε το πραγματικό αρχείο.
Tracker	Ο κεντρικός υπολογιστής που συντονίζει τη δράση των χρηστών και καταγράφει τον όγκο δεδομένων που ανεβοκατεβάζουν οι χρήστες. Λέγοντας tracker επίσης μπορεί να αναφερόμαστε στη σελίδα του site.
Unchoke	Όταν επιτρέπεται στον peer να ζητήσει και να κατεβάσει blocks δεδομένων.
Uploaders	Οι χρήστες που έχουν το δικαίωμα και σηκώνουν στον tracker δικά τους torrents.

Πίνακας 1: Πίνακας με την ανάλυση όρων του BitTorrent

Πως λειτουργεί και τι χρειάζεται

Η βασική φιλοσοφία του BitTorrent είναι ότι οι χρήστες πρέπει να ανεβάσουν συγχρόνως ενώ κατεβάζουν κιόλας. Με αυτόν τον τρόπο, το εύρος ζώνης του δικτύου χρησιμοποιείται όσο το δυνατόν αποτελεσματικότερα. Το BitTorrent έχει ως σκοπό να λειτουργήσει καλύτερα όσο ο αριθμός των ενδιαφερόμενων αυξάνει σε ορισμένα αρχεία, σε αντίθεση με άλλα πρωτόκολλα μεταφοράς αρχείων. Στο πρωτόκολλο BitTorrent όλα αρχίζουν με την δημιουργία ενός αρχείου .torrent. Αυτό είναι ένα μικρό σε μέγεθος αρχείο που περιέχει μόνον βασικές πληροφορίες (μέγεθος, διεύθυνση του tracker, κλπ) για ένα άλλο αρχείο-στόχο που μπορεί να είναι οτιδήποτε (βίντεο, μουσική, εφαρμογή κλπ). Το αρχείο

.torrent δημοσιεύεται σε έναν tracker και πλέον είναι διαθέσιμο. Το αρχείο-στόχος του .torrent, διαθέσιμο αρχικά από έναν χρήστη (seeder), κόβεται αυτόματα σε κομμάτια (chunks ή pieces), συνήθως 256 KB. Κατόπιν, αρχίζει να μοιράζεται κομμάτι-κομμάτι στους διάφορους χρήστες που το ζητούν (leechers). Κάθε χρήστης παίρνει και διαφορετικό κομμάτι, και κάθε χρήστης που έχει έστω ένα κομμάτι από το αρχείο το μοιράζει με τη σειρά του στους υπόλοιπους που δεν το έχουν. Έτσι εξασφαλίζεται η ταχύτερη διανομή του αρχείου, μιας που η διανομή του δεν βασίζεται αποκλειστικά στην ταχύτητα μετάδοσης του αρχικού seeder, αλλά στην συνολική ταχύτητα μετάδοσης όλων των εμπλεκόμενων χρηστών. Εννοείται πως όποιος χρήστης ολοκληρώσει το κατέβασμα του αρχείου αυτομάτως μετατρέπεται από leecher σε seeder και αρχίζει με τη σειρά του την ίδια διαδικασία.

Τα απαραίτητα στοιχεία για το πρωτόκολλο BitTorrent είναι:

- Ένας Tracker (όπως το PirateBay.com)
- Ένας Client (πρόγραμμα που πρέπει να βρίσκεται στον υπολογιστή μας και μας επιτρέπει τις «συναλλαγές» στον κόσμο του BitTorrent, όπως το Azureus)
- Τα αρχεία .torrent

Οι trackers χωρίζονται σε δυο κατηγορίες: Δημόσιοι (Public) και Ιδιωτικοί (Private). Ενώ σε έναν δημόσιο tracker ο κάθε χρήστης μπορεί να πάρει ότι θέλει και να κάνει ότι θέλει (π.χ. να σταματήσει το torrent αμέσως μόλις πάρει το αρχείο που θέλει), σε έναν ιδιωτικό tracker απαιτείται εγγραφή και απαιτείται από τους χρήστες να μοιράσουν (seed) τα αρχεία που κατεβάζουν τουλάχιστον μέχρι να έχουν δώσει όσο έχουν πάρει (ratio 1.00). Ως εκ τούτου, σε έναν ιδιωτικό tracker οι ταχύτητες είναι υψηλότερες και η διαθεσιμότητα των αρχείων μεγαλύτερη και σε ποσότητα και μέσα στον χρόνο (μιας που το φαινόμενο «κατεβάζω-και-φεύγω» γνωστό και ως "hit 'n run" είναι περιορισμένο αν όχι απών). Επιπλέον, μιας που πρόκειται για κλειστές κοινότητες, αναπτύσσεται μια ιδιαίτερη σχέση μεταξύ των χρηστών η οποία μεταφράζεται στην διαθεσιμότητα σπάνιων αρχείων που δεν απαντά κανείς σε δημοσίους trackers η στην δημοσίευση αρχείων κατόπιν αίτησης (request).

2.2 Το πρωτόκολλο Tracker

Ένα από τα πρωτόκολλα που εφαρμόσαμε και το οποίο αποτελεί κομμάτι του BitTorrent πρωτοκόλλου είναι το πρωτόκολλο Tracker [4]. Τυπικά, για τη διανομή ενός αρχείου με το BitTorrent αρχικά δημοσιοποιούμε το .torrent metafile σε ιστοσελίδες μέσω των οποίων διανέμεται στους διάφορους peers. Όπως προείπαμε το metafile αυτό περιέχει τη διεύθυνση του tracker, το μέγεθος του αρχείου, το μέγεθος του piece, και τις τιμές hash για τα pieces του αρχείου. Οι trackers είναι υπεύθυνοι να βοηθήσουν τους peers να ανακαλύψουν ο ένας τον άλλον έτσι ώστε να δημιουργήσουν ένα swarm. Κατά τη διάρκεια της λήψης ενός αρχείου, κάθε client επικοινωνεί με τον tracker και δημοσιοποιεί τα ολικά bytes που έχει λάβει και έχει διαθέσει, καθώς επίσης και την IP διεύθυνσή του, την TCP θύρα, πληροφορίες αναγνώρισης κ.ο.κ. Ας σημειώσουμε ότι οι περισσότερες από τις πληροφορίες που ανακοινώνει ο client εξυπηρετούν στατιστικούς σκοπούς και μόνο η IP διεύθυνση και η TCP θύρα ενός client είναι σημαντικές για τον tracker. Με αυτόν τον τρόπο όλο και περισσότεροι peers ανακαλύπτουν αυξανόμενα υποσύνολα του swarm.

2.3 Το πρωτόκολλο Peer-Wire

Το πρωτόκολλο αυτό παρέχει τον πυρήνα της λειτουργικότητας του BitTorrent, δηλαδή, την αλληλεπίδραση με τους peers[4].

Αφού ο client επικοινωνήσει με τον tracker, στη συνέχεια προσπαθεί να δημιουργήσει TCP συνδέσεις με τους peers που έλαβε ως απάντηση από τον tracker. Όταν πραγματοποιηθεί η σύνδεση, οι δύο peers ανταλλάσσουν HANDSHAKE μηνύματα έτσι

ώστε να επιβεβαιώσουν ότι όντως ενδιαφέρονται για το ίδιο torrent metafile. Μετά τα HANDSHAKE μηνύματα γίνεται ανταλλαγή BITFIELD μηνυμάτων τα οποία περιέχουν το bitfield του κάθε client, δηλαδή το bitmap που δείχνει την διαθεσιμότητα κάθε piece στον client. Σύμφωνα με αυτή την πληροφορία ένας client μπορεί να αποφασίσει αν ενδιαφέρεται για ένα ή περισσότερα pieces ενός άλλου peer. Ας σημειώσουμε εδώ ότι στην εφαρμογή αυτή η ανταλλαγή μηνυμάτων αποφεύγεται στην περίπτωση που ο peer δεν έχει κανένα piece αφού θα είχε ως αποτέλεσμα την ανταλλαγή άχρηστης πληροφορίας. Γι' αυτό το λόγο αποστέλλεται ένα NOT INTERESTED μήνυμα στον peer για να επισημάνει την έλλειψη ενδιαφέροντος για τα δεδομένα του.

Αν και σε αυτό το στάδιο ένας client γνωρίζει για ποιους peers ενδιαφέρεται, δεν μπορεί να ζητήσει δεδομένα μέχρι ο peer να του το επιτρέψει στέλνοντας ένα UNCHOKE μήνυμα. Αυτό σημαίνει ότι κάθε client είναι εξ ορισμού μπλοκαρισμένος, ή όπως αλλιώς θα λέγαμε σε ορολογία του BitTorrent, «πνιγμένος» (chocked) από τον συγκεκριμένο peer, σύμφωνα με κριτήρια που είναι ενσωματωμένα στον choking αλγόριθμο του πρωτοκόλλου.

Όταν ένας client γίνει unchoke από ένα peer, αρχίζει να στέλνει REQUEST μηνύματα, καθένα από τα οποία ζητάει ένα συγκεκριμένο block του επιλεγμένου piece. Ο peer στέλνει τα ζητούμενα δεδομένα χρησιμοποιώντας PIECE μηνύματα. Με την ολοκλήρωση της λήψης του αρχείου, ο client ενημερώνει με HAVE μηνύματα τους peers με τους οποίους έχει συνδεθεί. Αυτοί οι peers ενημερώνουν το bitfield για εκείνο τον client. Μετά μπορεί να εκφράσουν το ενδιαφέρον τους για το piece με ένα INTERESTED μήνυμα.

Στρατηγική λήψης ενός piece

Αυτή η στρατηγική αναφέρεται στη δυνατότητα των clients να λάβουν τα pieces σε τυχαία σειρά. Εμείς εφαρμόσαμε τις δύο επικρατέστερες πολιτικές λήψεως ενός piece, τις Rarest First και Random First. Η πολιτική Rarest First επιλέγει εκείνα τα pieces που εμφανίζονται λιγότερο συχνά στο σύνολο των συνδεδεμένων peers σε έναν client. Αυτή η επιλογή του piece γίνεται τυχαία ανάμεσα στα λιγότερο κοινά pieces για να αποφευχθεί πολλοί peers να συγκλίνουν στο ίδιο piece. Με αυτόν τον τρόπο οι peers λαμβάνουν pieces που οι περισσότεροι από τους άλλους peers πιθανόν να θέλουν, με αποτέλεσμα να διευκολύνεται η ανταλλαγή των δεδομένων. Ωστόσο, τα σπάνια pieces υπάρχουν σε ελάχιστους peers και είναι πιθανόν το μπλοκάρισμα ενός client να ακυρώσει τη λήψη του piece. Αυτό έχει ως αποτέλεσμα οι clients, που δεν έχουν pieces στην κατοχή τους, να περιμένουν για ένα optimistic unchoke από έναν peer που έχει το συγκεκριμένο σπάνιο piece. Η στρατηγική Random First αποφεύγει αυτό το πρόβλημα επιλέγοντας ένα τυχαίο piece το οποίο είναι περισσότερο πιθανό να είναι διαθέσιμο από πολλούς peers και έτσι η απόφαση να γίνει choke ένας client να μην έχει τόσο αρνητικά αποτελέσματα.

Queueing

Όπως αναφέρθηκε παραπάνω, τα REQUEST μηνύματα αφορούν συγκεκριμένα blocks ενός piece. Τα μεγέθη των pieces ποικίλουν από 256kb μέχρι 1MB ή και περισσότερο. Σε περίπτωση που ένας client γίνει choked θα οδηγούσε σε περιττές αναμεταδόσεις πακέτων. Γι' αυτό το λόγο χρησιμοποιούμε ένα queueing policy για αυτά τα requests, ειδάλλως θα είχαμε να αντιμετωπίσουμε καθυστερήσεις στις μεταδόσεις των πακέτων.

Στην πολιτική που εφαρμόστηκε ο χρήστης έχει τη δυνατότητα να καθορίσει το ακριβές μέγεθος του queue. Ο client μπορεί να στείλει σε έναν peer μέχρι ένα συγκεκριμένο αριθμό από REQUEST μηνύματα για blocks. Μόλις λάβει ένα PIECE μήνυμα, ο client μπορεί να στείλει το επόμενο REQUEST μήνυμα. Στην υλοποίησή μας τα blocks ζητούνται από τους clients με μια σειρά αλληλουχίας μέσα στο piece. Όταν ένα piece έχει ζητηθεί στο σύνολό του, αν η request queue δεν είναι γεμάτη, ο client επιλέγει ένα άλλο επιθυμητό piece από το bitfield ενός peer σύμφωνα και με την στρατηγική λήψης ενός piece και ξεκινάει να στέλνει REQUEST μηνύματα για τα blocks του.

Αλγόριθμος Choking

Το μπλοκάρισμα (choking) των clients γίνεται για διάφορους λόγους. Ο έλεγχος συμφόρησης TCP συμπεριφέρεται πολύ φτωχά όταν στέλνουμε πάνω από πολλαπλές συνδέσεις ταυτόχρονα. Επίσης, το choking των clients επιτρέπει στον κάθε peer να χρησιμοποιήσει έναν tit-for-tat αλγόριθμο για να εξασφαλίσει ότι έχουν ένα λογικό download rate. Ο choking αλγόριθμος που θα περιγράψουμε παρακάτω είναι αυτός που χρησιμοποιούμε στην υλοποίησή μας.

Υπάρχουν διάφορα κριτήρια τα οποία πρέπει να τηρεί ένας καλός choking αλγόριθμος. Πρέπει να ορίσει ως όριο ένα συγκεκριμένο αριθμό ταυτόχρονων uploads για καλή TCP απόδοση. Πρέπει να αποφύγει το γρήγορο choking/unchoking, το οποίο είναι γνωστό και ως 'fibrillation'. Οι peers πρέπει να κάνουν unchoke τους clients που έχουν τα καλύτερα upload rates (reciprocation). Τέλος, πρέπει να δοκιμάζει αχρησιμοποίητες συνδέσεις κάθε λίγο για να δει αν είναι καλύτερες από τις τρέχουσες, γνωστό και ως αισιόδοξο (optimistic) unchoking.

Ο τρέχων choking αλγόριθμος αποφεύγει το 'fibrillation' αλλάζοντας τους choked peers μία φορά κάθε 10 δευτερόλεπτα. Η ανταπόδοση και το ανώτατο όριο των uploads πετυχαίνεται κάνοντας unchoke τους τέσσερις peers που έχουν το καλύτερο upload rate και ενδιαφέρονται. Αυτό μεγιστοποιεί το download rate του client. Αυτοί οι τέσσερις peers ονομάζονται 'downloaders', επειδή ενδιαφέρονται να λάβουν δεδομένα από τον client. Οι peers που έχουν καλύτερο upload rate (σε σύγκριση με τους downloaders) αλλά δεν ενδιαφέρονται γίνονται choked. Αν ενδιαφερθούν, ο downloader με το χειρότερο upload rate γίνεται choked. Αν ένας client έχει ολόκληρο το αρχείο, χρησιμοποιεί το upload rate του παρά το download rate για να αποφασίσει ποιους peers θα κάνει unchoke. Όσον αφορά το αισιόδοξο unchoking, οποιαδήποτε στιγμή υπάρχει ένας μοναδικός peer ο οποίος είναι unchoked ανεξάρτητα από το upload rate του (αν ενδιαφέρεται, τότε υπολογίζεται ως ένας από τους τέσσερις downloaders). Ποιος peer γίνεται αισιόδοξα unchoked αποφασίζεται κάθε 30 δευτερόλεπτα.

Super Seeding

Το super seeding είναι ένα χαρακτηριστικό ιδιαίτερα χρήσιμο για τη διανομή περιεχομένου καθώς βοηθάει τον αρχικό seeder του αρχείου να αποφύγει την υπερβολική κατανάλωση bandwidth. Ο super seeder δεν πληροφορεί τους peers του ότι έχει διαθέσιμα όλα τα pieces, αλλά μεταμφιέζεται σε ένα συνηθισμένο client. Όταν συνδεθεί με ένα peer, ο seeder αρχικά παριστάνει πως δεν έχει κανένα piece και μόνο μετά τον ενημερώνει για τη διαθεσιμότητα ενός piece με ένα HAVE μήνυμα. Αυτό θα πείσει τον client να προσπαθήσει να κατεβάσει μόνο αυτό το piece. Όταν ο client λάβει το piece, ο seeder δε θα τον ενημερώσει για άλλα pieces μέχρι να δει ότι το piece που έστειλε προηγουμένως υπάρχει τουλάχιστον σε έναν από τους άλλους clients. Μέχρι τότε, ο client δε θα έχει πρόσβαση σε κανένα άλλο από τα pieces του seeder, με αποτέλεσμα να μη χαραμίσει το bandwidth του seeder.

Το module εφαρμόζει το χαρακτηριστικό αυτό σε όλους τους clients, αλλά το ενεργοποιεί μόνο στον αρχικό seeder γιατί ενώ βοηθάει στην διανομή σπάνιων pieces επειδή περιορίζει την επιλογή των pieces από τους clients, περιορίζει επίσης την ικανότητα αυτών των clients να κατεβάσουν δεδομένα για pieces τα οποία έχουν ήδη μερικώς λάβει.

Anti-snubbing

Περιστασιακά ένας peer θα μπλοκαριστεί από όλους τους peers από τους οποίους κατέβαζε δεδομένα. Σε τέτοιες περιπτώσεις συνήθως θα συνεχίζει να έχει χαμηλά download rates έως ότου το αισιόδοξο unchoke να βρει καλύτερους peers. Για να μετριάσουμε αυτό το πρόβλημα, όταν περνάει παραπάνω από ένα λεπτό χωρίς να πάρουμε piece από κάποιο peer,

το BitTorrent υποθέτει ότι ‘σνομπάρεται’ (snubbed) από το συγκεκριμένο peer και δεν του δίνει δεδομένα εκτός και είναι αισιόδοξο unchoke. Αυτό συχνά οδηγεί σε περισσότερα από ένα ταυτόχρονα αισιόδοξα unchoke, πράγμα που προκαλεί γρηγορότερη ανάκαμψη στα download rates.

End Game Mode

Τα downloads στο BitTorrent πραγματοποιούνται σύμφωνα με έναν αριθμό πολιτικών, ή αλλιώς βημάτων όπως είδαμε και παραπάνω. Η πολιτική που εφαρμόζεται για τη λήψη των τελευταίων εναπομείναντων pieces ονομάζεται End Game Mode.

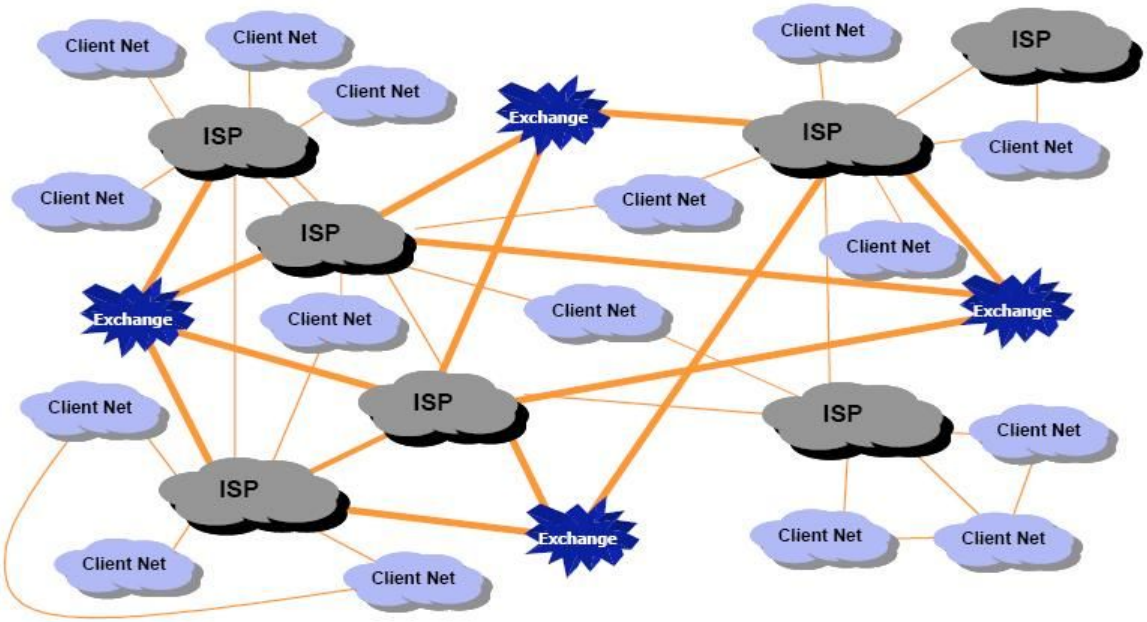
Όταν η λήψη ενός αρχείου είναι σχεδόν ολοκληρωμένη δηλαδή βρισκόμαστε στο στάδιο που τα περισσότερα pieces έχουν ληφθεί, υπάρχει μία τάση τα τελευταία blocks να λαμβάνονται πολύ αργά. Για να το επιταχύνουμε αυτό, ο client στέλνει REQUEST μηνύματα για όλα τα blocks που δεν έχει σε όλους τους peers του που δεν τον μπλοκάρουν (στη δικιά μας εφαρμογή του End Game Mode τα REQUEST μηνύματα δε στέλνονται σε όλους του peers αφού αν ένας peer μπλοκάρει τον client θα ακυρώσει απλά το request). Για να μη γίνει αναποτελεσματικό και χάσουμε σε bandwidth, ο client στέλνει επίσης ένα CANCEL μήνυμα σε όλους τους άλλους κάθε φορά που λαμβάνει ένα block. Είναι πιο «οικονομικό» να στείλουμε ένα CANCEL μήνυμα παρά να λάβουμε ολόκληρο το block και απλά να το απορρίψουμε.

Το πότε πρέπει ένας client να μπει σε End Game Mode είναι ένα θέμα συζήτησης. Στο δικό μας BitTorrent module, ο client μπαίνει σε End Game Mode όταν ο αριθμός των εναπομείναντων blocks ισούται με τον αριθμό των ζητούμενων blocks, που σημαίνει πως όλα τα εναπομείναντα blocks έχουν ζητηθεί. Στο module μας υπάρχει δυνατότητα το End Game Mode να είναι ενεργό ή όχι από τις ρυθμίσεις.

2.3 Κόστος και BitTorrent

Μπορεί το κόστος χρήσης των υπηρεσιών BitTorrent στις διάφορες υπηρεσίες να είναι μηδενικό για τους χρήστες του και μόνη προϋπόθεση να είναι η αμοιβαία συνύπαρξη όλων με την ανταλλαγή δεδομένων download/upload, αλλά σίγουρα το BitTorrent έχει δημιουργήσει άλλα δεδομένα στην κίνηση που δημιουργείται στα δίκτυα των παρόχων.

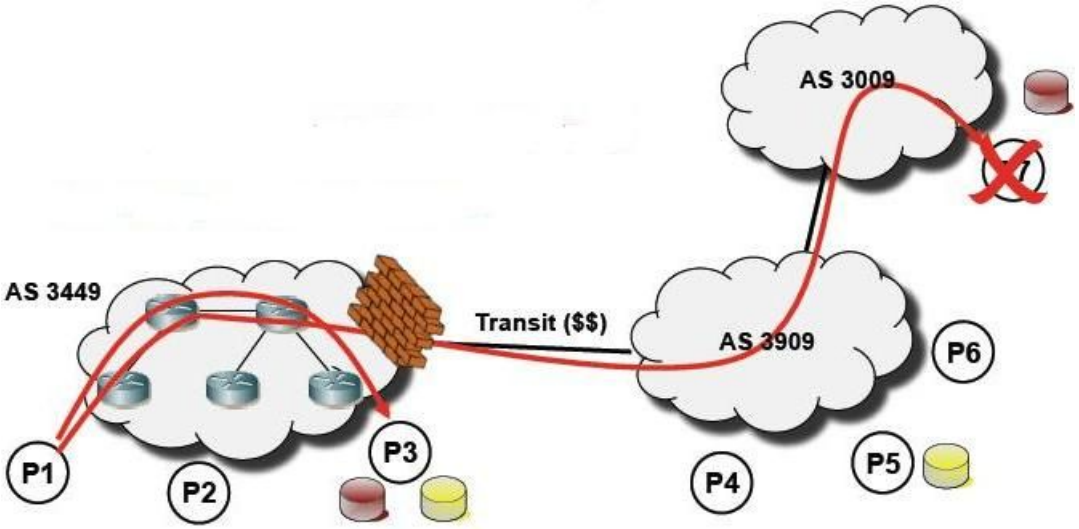
Ως μεταφορά (transit) ορίζεται η σχέση κατά την οποία ένας ISP παρέχει υπηρεσία πρόσβασης στο δίκτυο ενός άλλου ISP. Το κόστος μεταφοράς (transit cost) ορίζεται ως το κόστος της συμφωνίας μεταξύ 2 ISPs. Με βάση τον ακριβή ορισμό, έχουμε ότι ο πάροχος μεταφοράς (transit provider) είναι ένα Tier-1 δίκτυο το οποίο μπορεί να φτάσει σε όλους τους προορισμούς χωρίς να πληρώσει για transit κίνηση. Τα μικρότερου επιπέδου όμως δίκτυα τύπου Tier-2, για να καταφέρουν να έχουν πρόσβαση σε απομακρυσμένους χρήστες ή πόρους, χρειάζεται να προσπελάσουν υψηλότερου επιπέδου Tier-1 δίκτυα ώστε να φτάσει η κίνηση στον τελικό χρήστη. Το κόστος μεταφοράς είναι η χρήση σε Mbps της ποσότητας δεδομένων που στέλνεται και πληρώνεται μηνιαία. [5]



Εικόνα 3: Αναπαράσταση transit κίνησης μεταξύ ISPs η οποία απαιτεί χρηματικό αντάλλαγμα

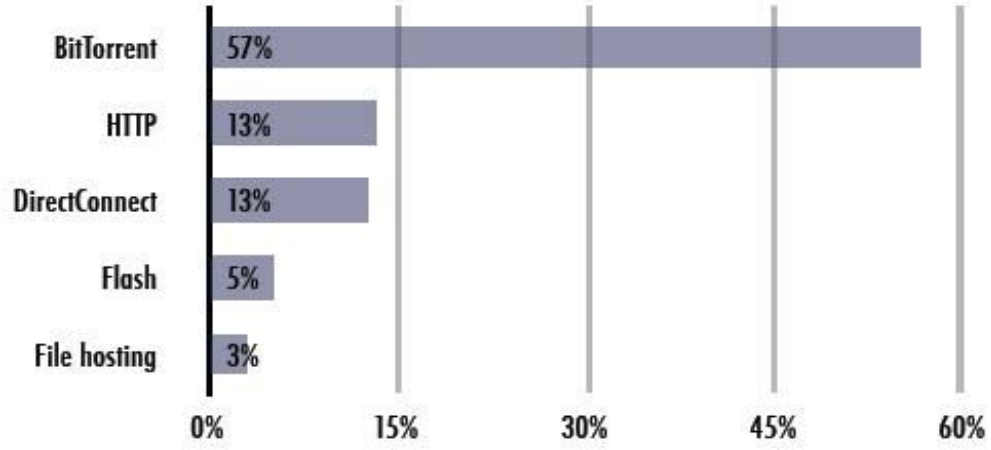
Να αναφέρουμε σε αυτό το σημείο ότι για τέτοιου είδους λόγους πολλές φορές μικρότεροι πάροχοι νοικιάζουν γραμμές διασύνδεσης ώστε να αποφύγουν τα μεγάλα κόστη σε περίπτωση συχνής διέλευσης και αυτή η διαδικασία ονομάζεται peering. Αν δύο ISPs αποφασίσουν να διασυνδεθούν μεταξύ τους είναι σίγουρα φθηνότερο αυτό να γίνει μέσω peering αλλά παρόλα αυτά ο Tier-2 ISP δεν θα μπορεί να ελέγξει τον προορισμό της κίνησης αυτής γι αυτό συνήθως προτιμάται μόνο σε περιπτώσεις όπου η τοποθεσία που θέλει να προσπελάσει βρίσκεται αρκετά κοντά για να έχει φυσική διασύνδεση.

Στην περίπτωση του BitTorrent τώρα, το κόστος για ένα πάροχο όσο αυξάνεται η διακινούμενη κίνηση έξω από τα όρια του δικτύου του, αποτελεί σημαντική πτυχή. Εφόσον η επιλογή των peers από τον tracker γίνεται με τυχαίο τρόπο αυτό σημαίνει ότι υπάρχει αρκετά μεγάλη πιθανότητα μεγάλο κομμάτι της κίνησης που θα μεταφερθεί να είναι εκτός των ορίων του δικού του δικτύου διαμέσου κάποιου Tier-1 δικτύου. Αυτός είναι και ο λόγος για τον οποίο στο παρελθόν οι ISPs αναγκάστηκαν να διακόψουν την p2p κίνηση στα δίκτυα τους για κάποιο χρονικό διάστημα μέχρι να παρουσιαστούν βέβαια αντιδράσεις από τους καταναλωτές και να αναγκαστούν να ανακαλέσουν αυτή την κίνησή τους. Όπως βλέπουμε και στο παράδειγμα της *Εικόνας 4*, ενώ το περιεχόμενο το οποίο ζητάει ο P1 θα μπορούσε να το πάρει από τον P3, αναγκάζεται λόγω της πολιτικής επιλογής peers (random selection) που ακολουθείται από τον Tracker στο BitTorrent να το πάρει από τον P7.



Εικόνα 4: Κόστος που δημιουργείται από την επιλογή peer εκτός δικτύου του ISP

Σύμφωνα με μετρήσεις που έγιναν περίπου στα μέσα του 2008 [6] , στην ανατολική Ευρώπη, η κίνηση που δημιουργεί το πρωτόκολλο BitTorrent ανέρχεται στο 57% της συνολικής κίνησης στα δίκτυα των ISPs (*Εικόνα 5*).



Εικόνα 5: Ποσοστά συνολικής δικτυακής κίνησης στην Ανατολική Ευρώπη το 2008

Από το ποσοστό τώρα που αναλογεί στη συνολική κίνηση p2p, το BitTorrent το 2008 κατείχε περίπου το 80% ενώ ένα χρόνο πριν αυτό το ποσοστό ήταν 65% (Εικόνα 6).

Eastern Europe

Protocol	2008	2007	Change
Other	1,30%	5,57%	-76,62%
BitTorrent	80,83%	65,71%	23,01%
Direct Connect	17,87%	28,72%	-37,78%

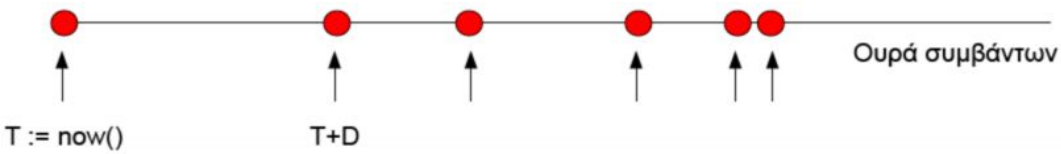
Εικόνα 6: Ποσοστά p2p κίνησης και μεταβολή τους στην Ανατολική Ευρώπη 2007-2008

Σε επόμενο κεφάλαιο θα δούμε πως μπορούμε να συνεισφέρει η υλοποίησή μας στη μείωση του Intra-domain traffic μαζί με τους υπόλοιπους παράγοντες που θα εξετάσουμε.

Κεφάλαιο 3: Περιγραφή εργαλείου εξομοίωσης και μετρήσεων

3.1 Περιγραφή του εργαλείου OMNeT++

Το OMNeT++ είναι ένα εργαλείο που αναπτύχθηκε για ακαδημαϊκούς κυρίως σκοπούς από τον A.Varga [7]. Είναι ένα εργαλείο εξομοίωσης δικτύων μέσω χρήσης διακριτών συμβάντων όπως φαίνεται και στην *Εικόνα 7*.

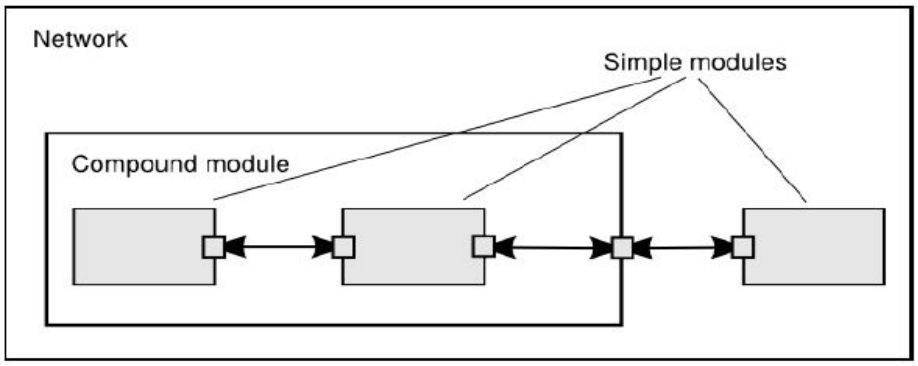


Εικόνα 7: Η ουρά συμβάντων σε μια προσομοίωση με το OMNeT++

Ο εξομοιωτής αυτός χρησιμοποιείται κυρίως για:

- Μοντελοποίηση δικτύων τηλεπικοινωνιών
- Μοντελοποίηση πρωτοκόλλων που αφορούν δίκτυα
- Μοντελοποίηση πολυεπεξεργαστικών και άλλων κατανεμημένων συστημάτων
- Μέτρηση απόδοσης πολύπλοκων συστημάτων λογισμικού μέσω της εκτέλεσης πειραμάτων και καταγραφής των αποτελεσμάτων τους

Το OMNET++ αποτελείται από ιεραρχικά φωλιασμένες ενότητες (Modules) που οι λειτουργίες τους περιγράφονται σε γλώσσα C++. Δηλαδή υπάρχουν βασικές ενότητες και υπο-ενότητες (submodules) από τις οποίες οικοδομούνται οι βασικές ενότητες. Οι ενότητες επικοινωνούν μεταξύ τους μέσω μηνυμάτων τα οποία μπορούν να φέρουν και δεδομένα. Κάθε module είναι συνδεδεμένο με κάποιο άλλο. Η κάθε ενότητα μπορεί να έχει δικές της παραμέτρους οι οποίες καθορίζουν την συμπεριφορά της. Οι ενότητες του ‘κατώτερου’ επιπέδου είναι αυτές που μοντελοποιούν την συμπεριφορά του δικτύου και ονομάζονται απλές ενότητες (simple modules) όπως φαίνεται και στην Εικόνα 8.



Εικόνα 8: Αναπαράσταση modules στο OMNeT++

Ο σχεδιασμός του δικτύου γίνεται σε γλώσσα NED ακολουθώντας αντικειμενοστραφή κριτήρια ενώ η τοπολογία του δικτύου καθορίζει πως τα διάφορα modules είναι συνδεδεμένα και πως αυτά συνδυάζονται για να σχηματίσουν μεγαλύτερα modules. Αρχικά, φτιάχνουμε την τοπολογία του δικτύου η οποία θα καθορίζει το είδος των modules που θα συμμετέχουν στο δίκτυο, τις παραμέτρους που θα έχουν αλλά και το πώς συνδέονται και αποθηκεύονται σε αρχεία της μορφής .ned. Ο ορισμός των μηνυμάτων και των παραμέτρων τους αποθηκεύεται σε αρχεία τύπου .msg ενώ ο κώδικας ο οποίος θα καθορίζει την συμπεριφορά των simple modules του δικτύου αποθηκεύεται σε αρχεία .cpp.

Για να κατασκευαστεί η εξομοίωσή μας πρέπει να καθορίσουμε αρχικά τις παραμέτρους

που θα έχει το δίκτυό μας και αυτοί οι παράμετροι αυτοί συμπεριλαμβάνονται στο αρχείο omnetpp.ini ή σε άλλα αρχεία με την κατάληξη .ini.
Η εξομοίωση φτιάχνεται ως εξής:

- Αρχικά, τα αρχεία .msg μεταγλωττίζονται σε C++ κώδικα.
- Ακολούθως, τα αρχεία .cpp μεταγλωττίζονται και ο κώδικας που παράγεται συνδέεται με τον πυρήνα εξομοίωσης του OMNET++.
- Τα αρχεία .ned μεταγλωττίζονται και αυτά και συνδέονται με τον πυρήνα εξομοίωσης για σχηματίσουν το τελικό εκτελέσιμο.

3.2 Καταγραφή στατιστικών

Η καταγραφή των αποτελεσμάτων στο OMNeT++ καθώς και η συλλογή στατιστικών για τη δημιουργία γραφημάτων και συμπερασμάτων γίνεται σε αρχεία τύπου Scalar και Vector με καταλήξεις .sca και .vec αντίστοιχα. Τα αρχεία τύπου Scalar χρησιμεύουν στην συλλογή συγκεντρωτικών στατιστικών για το μοντέλο που εξετάζουμε και είναι αρκετά χρήσιμα γιατί μπορούμε να βγάλουμε συμπεράσματα για τη συνολική συμπεριφορά του μοντέλου όπως μέσοι όροι συνολικοί χρόνοι προσομοίωσης κτλ. Τα αρχεία τύπου Vector χρησιμεύουν στη μέτρηση μεμονομένων συμβάντων όπως καθυστερήσεις χρήση δικτύου κάποια χρονική στιγμή κτλ για κάποια συμπεριφορά της προσομοίωσης.

Το OMNeT++ από μόνο του μας παρέχει μόνο την μηχανή εκτέλεσης και την γλώσσα περιγραφής NED. Δεν υποστηρίζεται κανένα πρωτόκολλο επικοινωνίας ή κάποια φυσική αναπαράσταση δικτύου. Για τη χρήση πρωτοκόλλων δικτύου χρησιμοποιούνται κάποια Frameworks τα οποία συνεργάζονται με τον εξομοιωτή για να μας δώσουν το επιθυμητό αποτέλεσμα. Δύο από αυτά τα Frameworks (INET, OverSim) είναι τα πιο βασικά από αυτά και αυτά που μας βοήθησαν στην εξομοίωση του BitTorrent.

3.3 INET Framework

Το INET [8], περιλαμβάνει μια βιβλιοθήκη έτοιμων modules (ned και c++) καθώς και υλοποιήσεις για πληθώρα πρωτοκόλλων που χρησιμοποιούνται στο Internet (Ipv4, Ipv6, TCP, UDP, OSPF, RTP, Ethernet, 802.11, κ.α.). Επίσης περιλαμβάνει έτοιμα modules για την αναπαράσταση κόμβων στο δίκτυο όπως routers, hosts, 802.11 Access Points, ethernet switches/hubs κ.α.

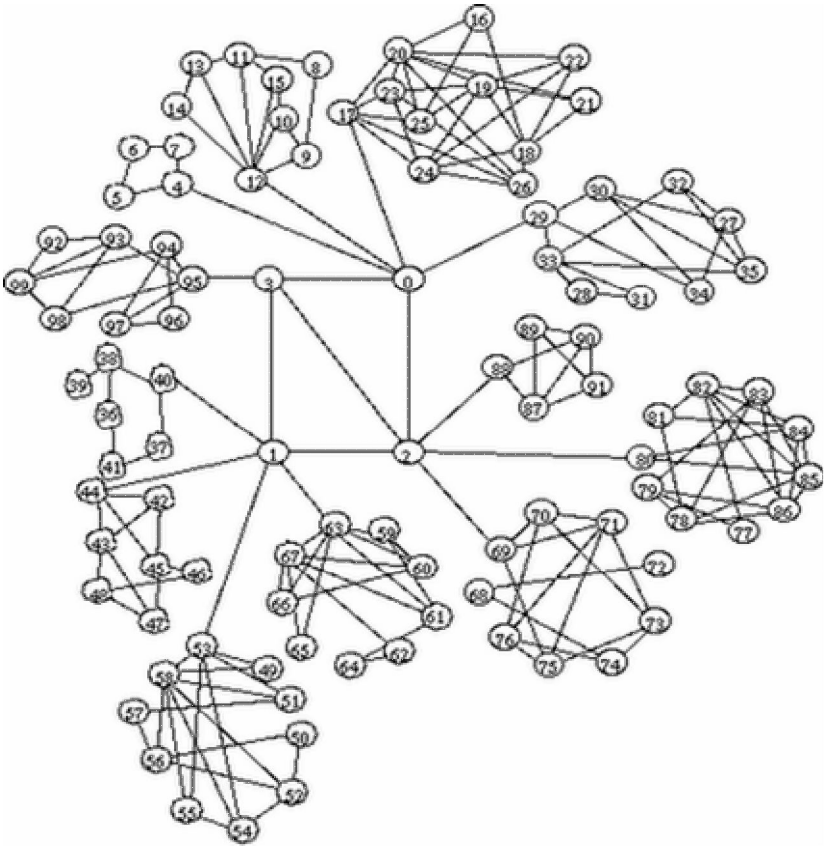
3.4 OverSim Framework

Το OverSim [9], περιλαμβάνει μερικά βασικά modules για την προσομοίωση overlay και underlay δικτύων για το OMNeT++/OMNeST περιβάλλον προσομοίωσης για την υποστήριξη peer-to-peer εφαρμογών. Περιλαμβάνει διάφορα μοντέλα για δομημένα (Chord, Kademlia, Pastry) και αδόμητα (GIA) peer-to-peer πρωτόκολλα. Επίσης παρέχεται η επιλογή υποκείμενου δικτύου(SingleHost, SimpleUnder-lay και INETUnderlay). Στην εξομοίωσή μας χρησιμοποιείται το IPv4Underlay model το οποίο μας παρέχει στην περίπτωση μας ένα αρκετά ρεαλιστικό περιβάλλον προσομοίωσης του BitTorrent πρωτοκόλλου.

3.5 Τοπολογίες (BRITE, GT-ITM)

Για την επίλυση δύο βασικών προβλημάτων στο μοντέλο IPv4Underlay που είναι η έλλειψη πολύπλοκων δομών στο backbone και access δίκτυο καθώς και η μη υποστήριξη βαρών στις πολιτικές δρομολόγησης χρησιμοποιήθηκαν οι πολύ γνωστές για περιβάλλοντα εξομοίωσης επεκτάσεις BRITE και GT-ITM [10],[11].

Το BRITE παρέχει μια γεννήτρια τοπολογιών και σε συνδυασμό με το εργαλείο GT-ITM (Georgia Tech Internet Topology Model) παρέχουν στο IPv4Underlay τη ρεαλιστικότητα που θέλουμε καθώς ένα τρόπο αναπαράστασης ιεραρχίας στο δίκτυό μας όπως φαίνεται και στην Εικόνα 9. Επίσης το BRITE μας παρέχει κάτι πολύ σημαντικό που είναι η ανάθεση Autonomous System αριθμών για τους end-host users.

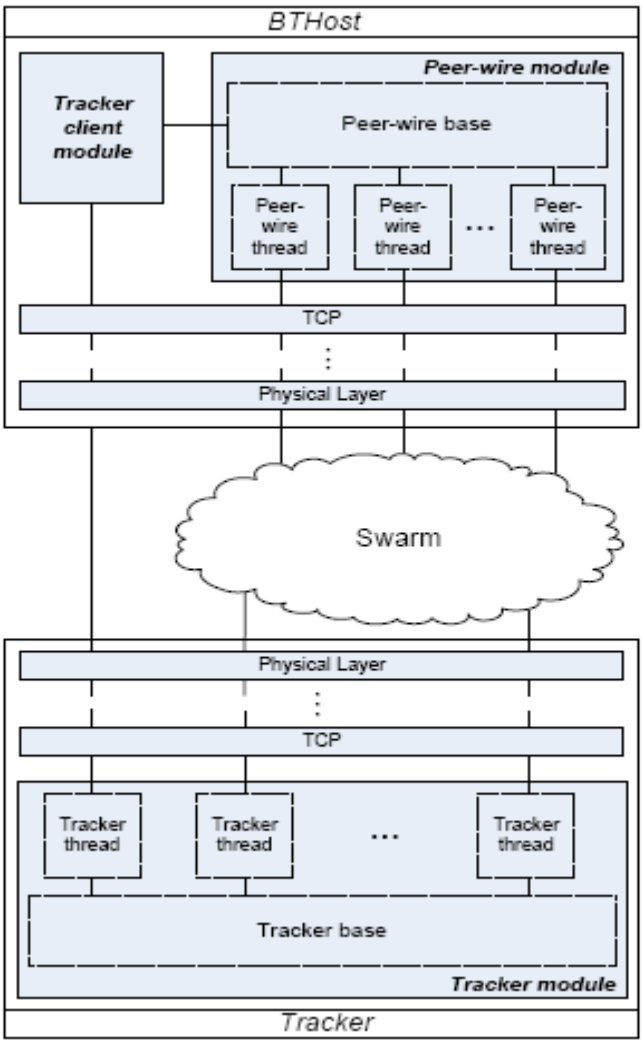


Εικόνα 9: Παράδειγμα δημιουργίας τοπολογίας με το εργαλείο GT-ITM

3.6 Δομή του Tracker πρωτοκόλλου και παράμετροι των modules

Το μοντέλο στο οποίο είναι υλοποιημένο το BitTorrent αποτελείται από 3 βασικά modules, τα TRACKER, TRACKER CLIENT, και το PEER-WIRE όπως φαίνεται και στην Εικόνα 10. Όπως προδίδουν και τα ονόματά τους, το πρώτο module παρέχει τη λειτουργικότητα του tracker, το δεύτερο module είναι υπεύθυνο για την επικοινωνία με τον tracker από τη μεριά ενός client και το τρίτο παρέχει τη λειτουργικότητα του peer-wire πρωτοκόλλου [4].

Στη συνέχεια θα αναλύσουμε τη δομή του TRACKER module, που είναι και αυτό στο οποίο θα πραγματοποιήσουμε τις αλλαγές στην υλοποίησή μας, και θα αναφερθούμε στις παραμέτρους που δίνονται στο πρωτόκολλο στα 3 παραπάνω modules.



Εικόνα 10: Η αρχιτεκτονική των modules του BitTorrent

Η δομή του πρωτοκόλλου του Tracker

Η υλοποίηση του πρωτοκόλλου του tracker αποτελείται από δύο κύρια modules: Το BTTRACKERBASE και το BTTRACKERCLIENTBASE [4]. Το πρώτο είναι το module του server, το οποίο είναι μέρος του tracker, ενώ το άλλο είναι το module του client, το οποίο είναι μέρος της εφαρμογής του BitTorrent. Η επικοινωνία μεταξύ αυτών των δύο επιτυγχάνεται μέσω του BTTRACKERMSGANNOUNCE και BTTRACKERMSGRESPONSE μηνυμάτων, τα οποία παράγονται από την κλάση CMESSAGE. Η πρώτη κλάση υλοποιεί client announce μηνύματα και περιλαμβάνει όλα τα απαραίτητα πεδία, με την ανάλογη σημασιολογία, ενώ η δεύτερη κλάση κωδικοποιεί τις απαντήσεις του tracker.

Η λειτουργικότητα του tracker υλοποιείται στο BTTRACKERBASE module όπως μια πολυνηματική (multithread) δικτυακή εφαρμογή. Έτσι σε κάθε επιτυχημένη σύνδεση στον the tracker, ένα καινούριο νήμα(thread) δημιουργείται για κάθε session μεταξύ του tracker και του peer. Το BTTRACKERCLIENTHANDLERBASE module, συγκράτησε το INET TCPSERVERTHREADBASE, και χρησιμοποιήθηκε για να ενθυλακώσει και να παράγει τις λεπτομέρειες της tracker-to-peer επικοινωνίας με τη μορφή ανταλλαγής μηνυμάτων, επικύρωσης εισόδου, κατασκευής απαντήσεων κτλ, ενώ το BTTRACKERBASE χειρίζεται τις υποκείμενες low-level λειτουργίες όπως η διαχείριση χρονομέτρου και η διαβίβαση μηνυμάτων. Παρομοίως, το client κομμάτι υλοποιείται στο BTTRACKERCLIENTBASE περιλαμβάνοντας τη λειτουργικότητα που απαιτείται για την ανάκτηση των απαντήσεων από τον tracker και την τροφοδοσία της εφαρμογής του BitTorrent με απεσταλμένη πληροφορία

(*π.χ.,* την πληροφορία επικοινωνίας με τους άλλους peers). Οι κύριες παραμετροποιήσεις των δύο modules και οι αρχικές default τιμές τους βρίσκονται στους *Πίνακες 2* (server) και *3* (client).

Parameter	**Default Value**
alwaysSendTrackerId	false
compactSupport	true
maxPeersInReply	50
announceInterval (sec)	30
cleanupInterval (sec)	60

Πίνακας 2: TRACKER SERVER PARAMETERS

Parameter	**Default Value**
connectGiveUp	3
reconnectInterval (sec)	2.0
sessionTimeout (sec)	30.0
infoHash	nil
compact	false
noPeerId	false
numWant	20
key	nil

Πίνακας 3: TRACKER CLIENT PARAMETERS

Parameter	**Default Value**
file size (MB)	700
piece size (KB)	256
block size (KB)	16
DHT port	-1
pstr	BitTorrent protocol
pstrlen	19
keep alive (sec)	120
have supression	true
choking interval (sec)	10
downloaders	4
optUnchokedPeers	1
optUnchoking interval (sec)	30
seederDownloaders	4
seederOptUnchokedPeers	1
rarest list size	5
minNumConnections	30
maxNumConnections	55
timeToSeed (sec)	0
request queue length	5
super seed mode	false
end game mode	true
maxNumEmptyTrackerResponses	5
newlyConnectedOptUnchokeProb	0.75
downloadRateSamplingDuration (sec)	20

Πίνακας 4: PEER-WIRE PROTOCOL PARAMETERS

Κεφάλαιο 4: Υλοποίηση

4.1 Περιγραφή του προβλήματος

Όπως είναι κατανοητό, η συνεχής αύξηση χρήσης των p2p τεχνολογιών και ιδιαίτερα αυτών που χρησιμοποιούν το πρωτόκολλο BitTorrent έχει αυξήσει σημαντικά τη διακίνηση δεδομένων στο Internet. Επομένως είναι άμεση ανάγκη να βρεθούν τρόποι να γίνει με κάποιο τρόπο καλύτερη διαχείριση της κίνησης είτε αυτό αφορά τους ISPs είτε αυτό αφορά το ίδιο το πρωτόκολλο.

Επίσης, μπορεί ο αλγόριθμος του tracker που επιστρέφει τη λίστα των peers που θα συνδεθεί για να πάρει το περιεχόμενο λόγω της ευρωστίας (robustness) που προσφέρουν οι random γράφοι να θεωρείται δίκαιος για τις διαδικασίες του πρωτοκόλλου, αλλά και στην κατανομή φόρτου, αλλά πολλές φορές η απόδοσή που προσδίδει στο σύστημα δεν είναι σταθερή [12]. Από τη μία, αυτό οφείλεται στο ότι οι επιλεγμένοι peers μπορεί να βρίσκονται σε μεγάλες αποστάσεις από τον peer που έχει κάνει το REQUEST, άρα δημιουργούνται πολλά hops στο routing path καθώς και πιθανή καθυστέρηση. Απ' την άλλη, δεν υπάρχει κάποια εγγύηση ποιότητας ταχύτητας τυχόν απομακρυσμένων peers και αυτό μπορεί να οδηγήσει επίσης σε μείωση της απόδοσης του συστήματος. Επομένως υπάρχουν περιθώρια βελτίωσης για το χρόνο κατεβάσματος των αρχείων αν προσπαθήσουμε να χρησιμοποιήσουμε άλλες στρατηγικές.

4.2 Στάδια-Διαδικασίες Υλοποίησης της ιδέας μας

Αυτό που εμείς προσπαθούμε να κάνουμε με την παρούσα διπλωματική εργασία είναι να επέμβουμε στο σημείο το οποίο ο tracker δημιουργεί την τυχαία λίστα με τους peers που στέλνει ως REPLY σε κάθε peer που του έκανε REQUEST, προσθέτοντας κάποια κριτήρια τοπικότητας (Locality Criteria) στη διαδικασία έτσι ώστε ένα εγγυημένο ποσοστό χρηστών να είναι από τη γειτονιά του peer. Αυτό προσπαθούμε να το πετύχουμε με τον εξής τρόπο. Κάθε φορά που κάποιος peer προσπαθεί να επικοινωνήσει με τον tracker και να μπει στο Swarm, μαζί με τις πληροφορίες που στέλνονται κανονικά (IP address, port, κτλ) στέλνουμε και την πρόσθετη πληροφορία για το Αυτόνομο Σύστημα (AS) στο οποίο ανήκει ο peer αυτός στέλνοντας το ASID του. Ο tracker επεξεργάζεται τα ASIDs που έχει στη διάθεση του από τους peers που έχουν επικοινωνήσει μαζί του και όταν στέλνει το REPLY στον peer που το χρειάζεται στέλνει ένα ποσοστό χρηστών το οποίο να ανήκει στο ίδιο AS με αυτόν.

Με την παραπάνω διαδικασία, είμαστε σίγουροι ότι, μεγαλύτερο μέρος της πληροφορίας που ερχόταν σε εμάς από πριν θα έρθει από γειτονικούς peers. Έτσι εξασφαλίζουμε:

1. Λιγότερα routing hops/peer άρα και μικρότερες καθυστερήσεις καθώς και γρηγορότερα downloads.
2. Μεγαλύτερο κομμάτι της κίνησης «εγκλωβίζεται» στο intra-domain και αντίστοιχα μικρότερο κομμάτι της κίνησης δρομολογείται στο inter-domain κομμάτι του δικτύου.
3. Μικρότερο transit cost για τον ISP στον οποίο βρίσκεται ο peer και έτσι χωρίς περεταίρω κόστος (π.χ. caching, extra equipment, κ.τ.λ.) έχουμε λιγότερη κίνηση.

Με αυτήν την παραλλαγή όταν η πληροφορία που αναζητείται βρίσκεται σε κάποιο ποσοστό τοπικά, μέρος της κίνησης για την εύρεση της και την ανάκτηση της περιορίζεται σε τοπικό επίπεδο. Αυτό είναι πολύ σημαντικό για τους ISPs για παράδειγμα, διότι πολλές φορές σε συστήματα διαμοιρασμού αρχείων πολλοί peers ανακτούν δεδομένα από peers εκτός του δικτύου του ISP, ενώ η πληροφορία υπάρχει και τοπικά όπως είδαμε και σε

προηγούμενη ενότητα. Με αυτόν τον τρόπο ο ISP χρεώνεται χωρίς λόγο, ενώ όταν ισχύει η ιδιότητα του path locality οι peers θα λαμβάνουν την πληροφορία από peers έξω από το δίκτυο του ISP μόνο αν δεν υπάρχει αυτή η πληροφορία τοπικά στο βαθμό που θα έχουμε θέσει εμείς.

Η παραπάνω διαδικασία θα μπορούσε να υλοποιηθεί και στον client με τον client να επεξεργάζεται τη λίστα που τους στέλνεται από τον tracker και να επιλέγει αυτός τους peers με τους οποίους θα πάρει στοιχεία και τι ποσοστό αυτών θα είναι από γειτονικούς του peers. Ο λόγος που επιλέξαμε η διαδικασία αυτή να γίνεται στον tracker είναι πως έτσι εξασφαλίζουμε πως θα αξιοποιηθεί όλο το εύρος του maxPeersInReply που δίνει ο tracker και όχι ένα ποσοστό του, κάτι που θα οδηγούσε σε ανομοιογένεια σε ότι αφορά τους συνδεδεμένους peers.

Επομένως στον κώδικά που αλλάχτηκε στο module για το BitTorrent ακολουθήσαμε τα εξής βήματα:

1) Προστέθηκε η πληροφορία για το ASID του peer στο module του BTTRACKERCLIENTBASE.
2) Προστέθηκε επίσης η πληροφορία για το ASID του peer στο μήνυμα που θα στέλνεται από τον στον Tracker στο BTTrackerMsg.msg
3) Στο module BTTRACKERBASE, κάθε φορά που κάποιος peer μπαίνει στο Swarm, γίνεται υπολογισμός με το πόσοι peers έχουν ίδιο ASID με τον peer που εισήλθε.
4) Πριν τον αλγόριθμο για Random Selection των peers που θα αποστείλει ο Tracker στον peer γίνεται εφαρμογή του δικού μας αλγόριθμου, που θα αναλύσουμε παρακάτω, και έτσι τοποθετούμε στους peers θα αποσταλούν ένα ποσοστό peers κάθε φορά από το ίδιο AS και ο Random Selection αλγόριθμος συμπληρώνει τους υπόλοιπους peers μέχρι να συμπληρωθεί το maxPeersInReply.

info_hash	**no_peer_id**
peer_id	**event (started, stopped, completed)**
port	**ip_address**
uploaded	**numwant**
downloaded	**key**
left	**tracker_id**
compact	**+ asid +**

Πίνακας 5: Πληροφορία που στέλνει κάθε peer στον Tracker

Τέλος έχουμε και μερικούς περιορισμούς σε ότι αφορά την υλοποίηση και τον τρόπο που αυτή θα εφαρμοστεί. Αυτοί είναι οι εξής:

- Να υπάρχει ένας μεγάλος αριθμός από peers στο δίκτυό μας ώστε να μπορούμε να εξάγουμε ασφαλή συμπεράσματα για τη λειτουργία του αλγόριθμου.
- Να υπάρχουν αρκετοί peers σε κάθε AS έτσι ώστε να λειτουργήσει η παράμετρος που εισάγουμε στον αλγόριθμο και αυτό το εξασφαλίζουμε με τη συμμετρικότητα που προσδίδεται στην τοπολογία μας από τη χρήση του GT-ITM μοντέλου.
- Να μην είναι τόσο κακοί οι χρήστες (free riders) και να συμμετάσχουν και στη διαδικασία του upload.
- Να μην εκμεταλλευτούμε τα κριτήρια τοπικότητας υπερβολικά δημιουργώντας κλίκες μεταξύ γειτονικών κόμβων του δικτύου, γιατί αυτό μπορεί να οδηγήσει ακόμα και σε κατάρρευση του Tit-For-Tat. Επομένως καλό θα ήταν να βάλουμε κάποια άνω όρια στη χρήση του παράγοντα **a**.

4.3 Αλγόριθμος

Εφόσον έχουμε υπολογίσει το πόσοι peers είναι συνδεδεμένοι στον tracker με το ίδιο ASID με του peer που έκανε το REQUEST, προχωράμε στον υπολογισμό του μεγέθους των peers που θα εισάγουμε ανάλογα πάντα με τους διαθέσιμους peers αλλά και με τη βοήθεια ενός παράγοντα (factor), τον οποίο θα ονομάσουμε **a** και αναφέρεται στο ποσοστό των peers που θα εισάγουμε με ίδιο ASID από τους διαθέσιμους peers που έχουμε κάθε φορά.

Επομένως χρειάζεται να υπολογίσουμε κάθε φορά το ποσοστό των peers που θα μπαίνουν με βάση το διαθέσιμο δυναμικό των peers αλλά και το maxPeersInReply που δίνει ως απάντηση ο Tracker και ορίζουμε τη σχέση μας ως εξής:

max_same_asid_peers = a * peers_size

Το αποτέλεσμα δηλαδή είναι ο μέγιστος αριθμός των peers που θα εισαχθεί ως απάντηση του Tracker και αυτό δίνεται από το ποσοστό επί του συνόλου των peers που έχει στην κατοχή του ο Tracker τη στιγμή εκείνη. Στη συνέχεια ακολουθεί ο αλγόριθμός μας.

```
if ( same_asid_peers <= max_same_asid_peers )  //(Case 1)
{
        //Place all these peers to the list will be sent to the peer
}
else  //(Case 2)
{
        while (added_peers < max_same_asid_peers )
        {
                //Select random peer from  same_asid_peer  table
                //Place this random peer to the list will be sent to the peer
        }
}
Complete the rest of the list with random peers from all the peers table //(Default Case)
```

Στην πρώτη περίπτωση (Case 1), σε περίπτωση που οι διαθέσιμοι peers με το ίδιο ASID είναι λιγότεροι από αυτούς τους οποίους μπορεί να μας δώσει η σχέση υπολογισμού των peers που θα μπούνε στη λίστα, τότε εισάγουμε όλους τους peers που έχουν ίδιο ASID και οι υπόλοιποι συμπληρώνονται στη συνέχεια με Random Selection (Default Case) μέχρι να συμπληρωθεί το maxPeersInReply.

Στη δεύτερη περίπτωση (Case 2), όπου οι διαθέσιμοι peers που έχουν το ίδιο ASID είναι περισσότεροι από αυτούς τους οποίους μπορεί να μας δώσει η σχέση υπολογισμού των peers που θα μπούνε στη λίστα, τότε με Random Selection από αυτούς τους peers με το ίδιο ASID επιλέγουμε αυτούς τους peers μέχρι να συμπληρωθεί το max_same_asid_peers και οι υπόλοιποι συμπληρώνονται στη συνέχεια με Random Selection (Default Case) μέχρι να συμπληρωθεί το maxPeersInReply.

Στις περιπτώσεις που:

Α) το ποσοστό του factor **a** είναι ίσο με το μηδέν

Β) δεν υπάρχουν διαθέσιμοι peers με το ίδιο ASID

Γ) το ποσοστό που χρειαζόμαστε δεν ικανοποιείται από τους διαθέσιμους peers με αποτέλεσμα max_same_asid_peers = 0, τότε γίνεται απευθείας Random Selection (Default Case) και η λίστα των peers συμπληρώνεται με τον αλγόριθμο που προϋπήρχε στον Tracker.

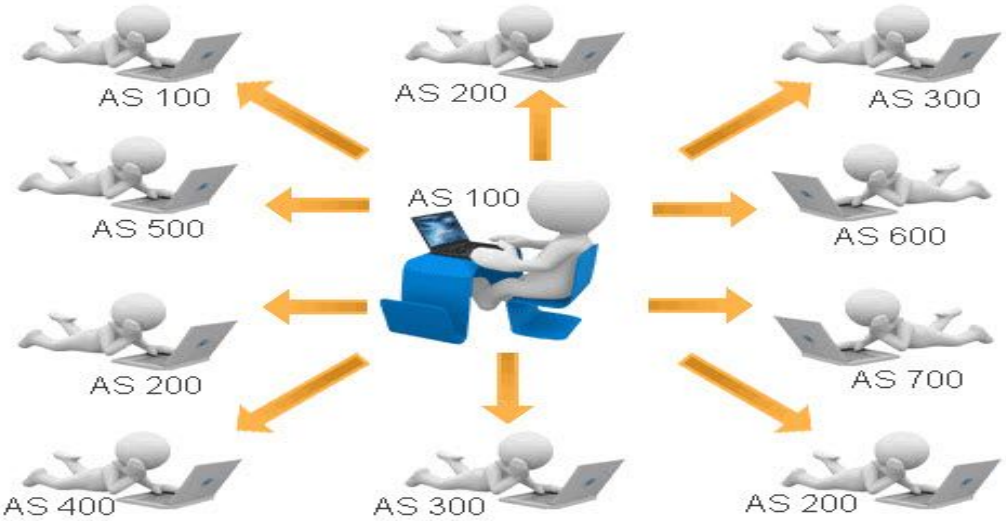
Ακολουθούν μερικά παραδείγματα για να γίνει κατανοητή η συμπεριφορά του αλγόριθμού.

Παραδείγματα

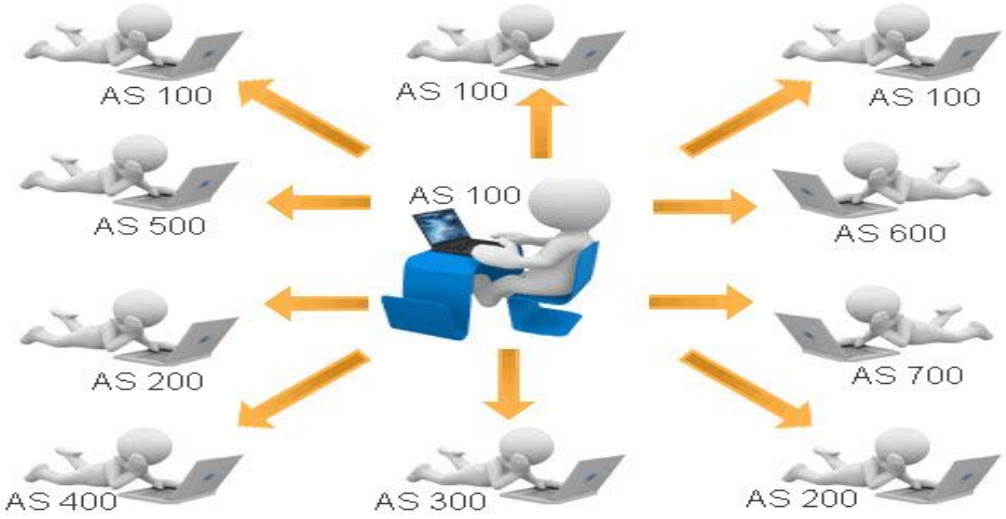
Example #	Peers in swarm	Max peers in reply	Factor a (%)	Same ASID peers	Max Same ASID peers
1	50	10	0	5	0 (default case)
2	5	10	20	5	1 (case 1)
3	50	10	10	6	1 (case 2)
4	50	10	35	5	3 (case 2)
5	50	10	40	3	3 (case 1)
6	50	10	50	7	5 (case 2)
7	50	50	10	1	0 (default case)

Πίνακας 6: Παραδείγματα με διάφορες παραμέτρους

Στο παράδειγμα 3 γίνεται επιλογή 1 peer με ίδιο ASID και οι υπόλοιποι 9 επιλέγονται με τυχαίο τρόπο*(Εικόνα 11)*. Ενώ αντίστοιχα στο παράδειγμα 4 επιλέγονται 3 peers με ίδιο ASID και οι υπόλοιποι 7 με τυχαίο τρόπο*(Εικόνα 12)*.



Εικόνα 11: Απάντηση από το παράδειγμα 3 του Πίνακα 6



Εικόνα 12: Απάντηση από το παράδειγμα 4 του Πίνακα 6

4.4 Παράμετροι-Σενάρια

Ο *Πίνακας 7* δείχνει τις σταθερές παραμέτρους στις οποίες έτρεξε η προσομοίωσή μας. Οι μεταβλητές παράμετροι μας ήταν ο παράγοντας **a** καθώς και το seed. Για το **a** τρέξαμε τις τιμές {0, 10, 20, 30, 40, 50, 60}, ενώ τρέξαμε κάθε μια από αυτές τις περιπτώσεις για 5 διαφορετικές τιμές του seed ώστε να έχουμε ένα ικανοποιητικό εύρος τυχαιότητας και πιο ασφαλή συμπεράσματα για τα αποτελέσματα μας.

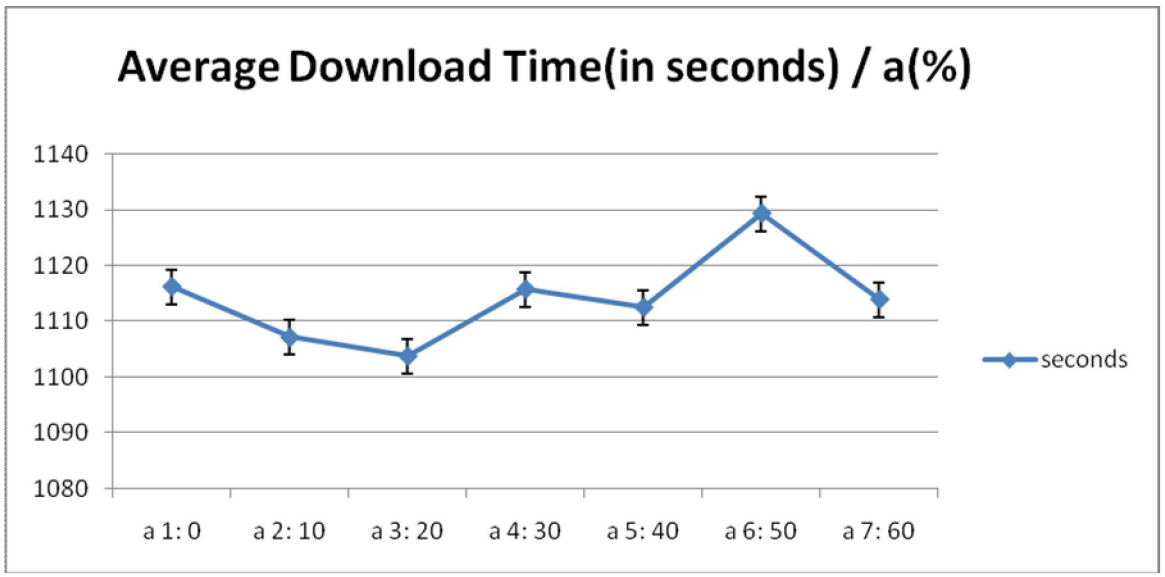
Παράμετροι υλοποίησης	
Terminals #	50
File Size	150 MB
Piece size	256 KB
maxPeersInReply	10
End Game Mode	true
Download Channels(Random)	DSL{4MB, 8MB, 16MB, 24MB}
Download Channels(Random)	DSL{1MB, 2MB}

Πίνακας 7: Παράμετροι στις οποίες έτρεξε η εξομοίωση

4.5 Αποτελέσματα και ανάλυση

Σε αυτό το σημείο θα αναλύσουμε τα αποτελέσματα που είχαμε από τα σενάρια που τρέξαμε. Θα παρουσιάσουμε έξι γραφήματα από τα οποία ένα για το μέσο χρόνο κατεβάσματος αρχείων, τρία με αθροιστικές κατανομές πιθανότητες (CDF) για το χρόνο κατεβάσματος αρχείων και δύο για inter-domain και intra-domain κίνηση στο δίκτυό μας.

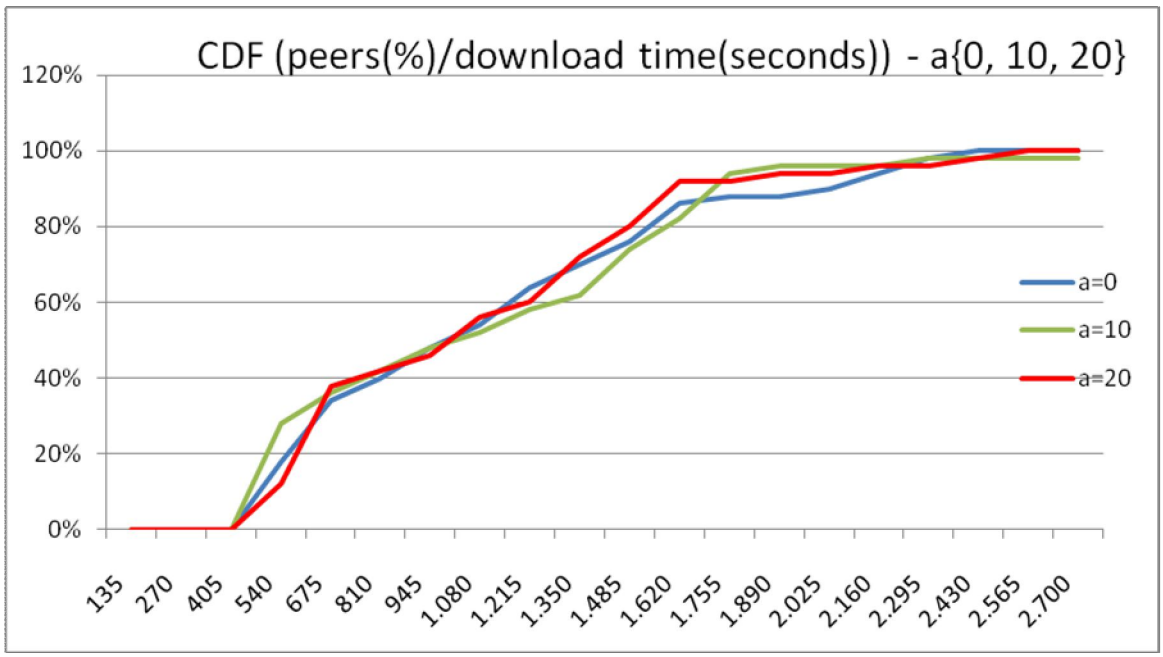
Στο *Γράφημα 1* βλέπουμε ότι για τιμές του a 10-20%, έχουμε μια αισθητή βελτίωση του μέσου χρόνου κατεβάσματος του αρχείου μας σε σχέση με τη default περίπτωση του αλγόριθμου. Στις τιμές του a 30-40% ο μέσος χρόνος είναι περίπου ίσος με αυτόν του default αλγόριθμου, ενώ στις τιμές 50-60% υπάρχει αύξηση του μέσου χρόνου. Η αύξηση αυτή οφείλεται στο γεγονός ότι αρκετοί peers επικοινωνούν μόνο μεταξύ τους, με αποτέλεσμα πολλά pieces να μην κυκλοφορούν γρήγορα αν δεν υπάρχουν στο ίδιο AS.



Γράφημα 1: Μέσος χρόνος κατεβάσματος αρχείου στις διάφορες τιμές του παράγοντα a

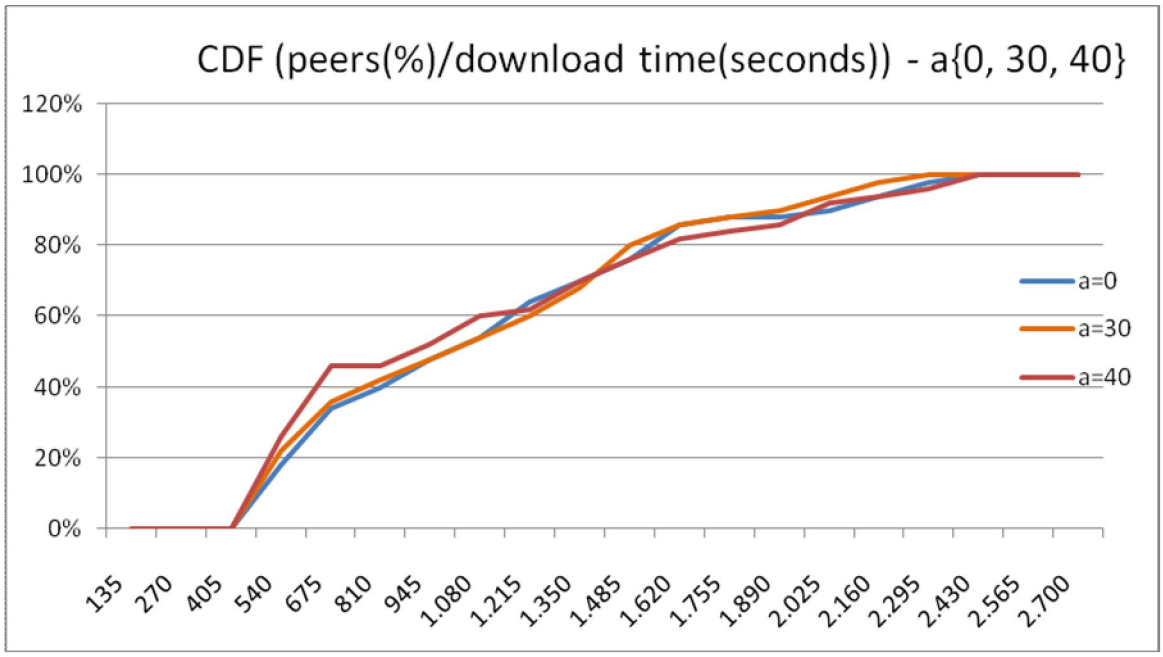
Στα τρία CDF γραφήματα μπορούμε να δούμε πόσο γρήγορα τελειώνουν οι 50 peers το κατέβασμα του αρχείου στις διάφορες τιμές του **a** σε σχέση με την default περίπτωση (a=0). Οι συγκρίσεις γίνονται ανά 2 τιμές του **a** σε σχέση με την περίπτωση που a=0, για λόγους σύγκρισης και για να μη δημιουργηθεί σύγχυση με τα γραφήματα.

Για τις περιπτώσεις όπου a=10 και a=20 όπως βλέπουμε στο *Γράφημα 2* έχουμε μια μικρή βελτίωση στην ταχύτητα όπου οι peers τελειώνουν το κατέβασμα του αρχείου και επίσης διακρίνουμε και μια αρκετά καλή σταθερότητα στο ρυθμό τους.



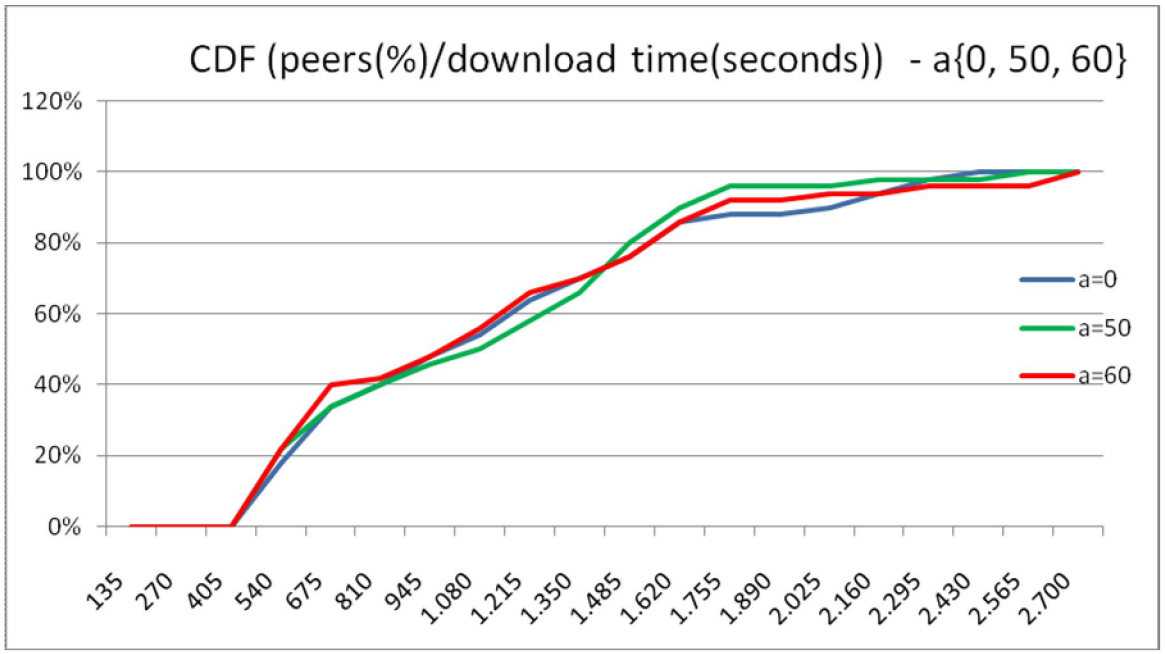
Γράφημα 2: CDF για τιμές του a{0, 10, 20}

Για τις περιπτώσεις όπου a=30 και a=40 του *Γραφήματος 3*, διακρίνουμε μια ισορροπία σε σχέση με την περίπτωση όπου a=0. Για a=40 υπάρχουν αρκετοί peers οι οποίοι τελειώνουν γρήγορα σε χρόνο μέχρι 1100 seconds αλλά στη συνέχεια υπάρχει μια μικρή καθυστέρηση, και αυτό οφείλεται στο ότι αρκετοί peers έφυγαν γρήγορα και έτσι μείνανε λίγοι στο swarm για να κάνουν τις ανταλλαγές pieces.



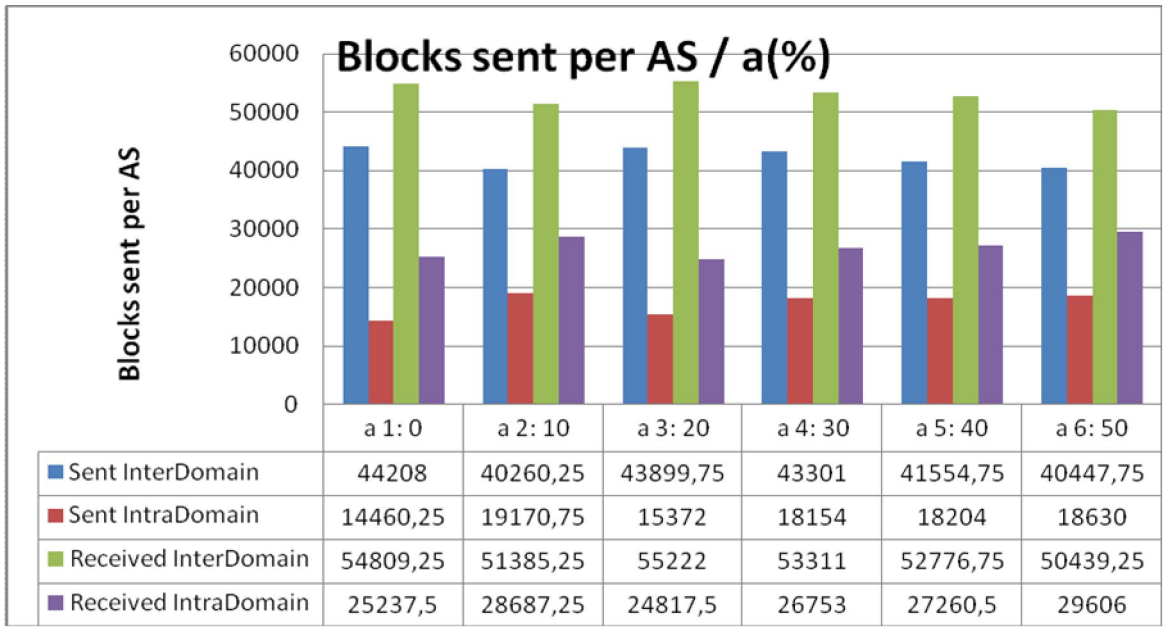
Γράφημα 3: CDF για τιμές του a{0, 30, 40}

Στις περιπτώσεις όπου a=50 και a=60 του *Γραφήματος 4*, μπορούμε να δούμε ότι πρόβλημα που υπήρχε και στην περίπτωση όπου a=40 μεγαλώνει καθώς πολλοί χρήστες παίρνουν το αρχείο γρήγορα και φεύγουν ανά διαστήματα και αναγκάζουν τους επόμενους να καθυστερούν να το πάρουν λόγω λίγων peers στο swarm.



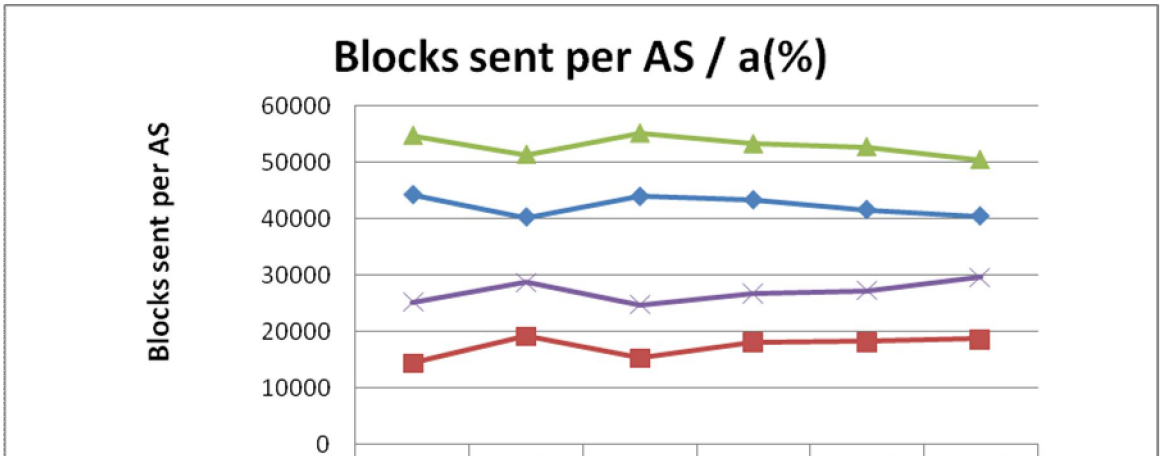
Γράφημα 4: CDF για τιμές του a{0, 50, 60}

Στο *Γράφημα 5*, μπορούμε να δούμε πως όσο αυξάνουμε την τιμή του παράγοντα a, υπάρχει μια σταθερή μείωση των inter-domain blocks που φτάνουν και στέλνονται και μια αντίστοιχη αύξηση των intra-domain blocks. Αυτό οφείλεται στο ότι όσο αυξάνεται το a, τόσο οι peers ανταλλάσουν μεταξύ τους blocks και έτσι ζητάνε από όλο και περισσότερους γείτονες κάθε φορά τα blocks των αρχείων. Αυτό το συμπέρασμα φαίνεται καθαρότερα στο *Γράφημα 6* όπου σε όλες τις περιπτώσεις η κίνηση εκτός του AS είναι μικρότερη από την αρχική και αντίστοιχα η εντός AS κίνηση μεγαλώνει με μια μικρή εξαίρεση για a=20 όπου τα ληφθέντα blocks εκτός AS γίνονται περίπου ίσα με τη default περίπτωση μάλλον λόγω του ότι δεν έχει εγκλωβιστεί ακόμα τόσο μεγάλο μέρος της κίνησης εντός AS.



	a 1: 0	a 2: 10	a 3: 20	a 4: 30	a 5: 40	a 6: 50
Sent InterDomain	44208	40260,25	43899,75	43301	41554,75	40447,75
Sent IntraDomain	14460,25	19170,75	15372	18154	18204	18630
Received InterDomain	54809,25	51385,25	55222	53311	52776,75	50439,25
Received IntraDomain	25237,5	28687,25	24817,5	26753	27260,5	29606

Γράφημα 5: Σταλμένα/ληφθέντα blocks στα Inter/Intra-domain για 6 τιμές του a ανά παράγοντα a



	a 1: 0	a 2: 10	a 3: 20	a 4: 30	a 5: 40	a 6: 50
Sent InterDomain	44208	40260,25	43899,75	43301	41554,75	40447,75
Sent IntraDomain	14460,25	19170,75	15372	18154	18204	18630
Received InterDomain	54809,25	51385,25	55222	53311	52776,75	50439,25
Received IntraDomain	25237,5	28687,25	24817,5	26753	27260,5	29606

Γράφημα 6: Αυξομείωση σε σταλμένα/ληφθέντα blocks στα Inter/Intra-domain για 6 τιμές του a

Κεφάλαιο 5: Συμπεράσματα

5.1 Συμπεράσματα εργασίας

Τα αποτελέσματα τελικά δικαιώνουν την απόπειρα που έγινε για την προσθήκη τοπικότητας στον αλγόριθμο του tracker.

Όσο αφορά το Download Time είδαμε ότι για a={10,20,30} υπάρχει αρκετά καλή και σχετικά σταθερή απόκριση του συστήματος καθώς οι peers κατεβάζουν λίγο πιο γρήγορα το αρχείο αλλά όχι τόσο ώστε να δημιουργείται πρόβλημα στους επόμενους peers που βρίσκουν το swarm πιο μικρό από ότι στις 3 πρώτες περιπτώσεις και η τοπικότητα δεν τους βοηθάει τόσο καθώς πολλά pieces από τους γείτονες πιθανό να μην καλύπτουν ολόκληρο το αρχείο όπως είδαμε και στα γραφήματα CDF.

Επίσης είδαμε ότι η transit κίνηση μειώνεται επιτυχώς σε όλες τις περιπτώσεις επομένως το κέρδος που αποκομίζουν οι ISPs από τα λιγότερα έξοδα λόγω της μετατόπισης του traffic μεταξύ άλλων ISPs και αυτού σε traffic στο εσωτερικό τους είναι μεγάλο.

Θα πρέπει όμως να ισοσκελίσουμε τις παραμέτρους ώστε να υπάρχει μεν λιγότερο Inter-AS traffic αλλά ταυτόχρονα να μην υπάρχουν και ανωμαλίες στο swarm και να διατηρείται κάποια σταθερότητα στο χρόνο κατεβάσματος των αρχείων. Μια τέτοια προσέγγιση που να συνδυάζει όλους τους παράγοντες είναι για ένα ποσοστό του factor a γύρω στο 10-20%. Επομένως σε αυτό το εύρος συνδυάζουμε ένα καλό και σταθερό χρόνο κατεβάσματος αρχείων για όλους τους peers και ταυτόχρονα μειωμένη εξερχόμενη κίνηση από το AS μας σε άλλο σε σχέση με τον pure αλγόριθμο του Random Peer Selection στον tracker.

5.2 Σχετικές εργασίες και μελλοντική δουλειά

Για την υλοποίηση της παρούσας διπλωματικής εργασίας έγινε μια αρκετά μεγάλης έκτασης ανασκόπηση στην έρευνα που έχει γίνει στον τομέα του BitTorrent και ειδικότερα σε θέματα Locality. Υπάρχει αρκετή δουλειά που έχει γίνει και πολλές έρευνες είτε έχουν γίνει σε περιβάλλοντα εξομοίωσης (OMNeT, NS-2, κ.τ.λ.), είτε σε πιο αληθοφανή σενάρια (π.χ. PlanetLab) έχουν βγάλει αρκετά καλά αποτελέσματα τα οποία αξίζουν και ίσως χρησιμοποιηθούν σε επόμενη αναβάθμιση της έκδοσης του πρωτοκόλλου.

Μερικές από τις έρευνες αυτές αποτέλεσαν κριτήριο για να σκεφτούμε την ιδέα που εφαρμόσαμε στην παρούσα υλοποίηση. Στο [13], δημιουργείται στον tracker μια λίστα με τις αποκρίσεις των peers προς αυτόν. Από αυτή τη λίστα, στο reply του tracker,στέλνεται κάθε φορά στον peer που έκανε το announce, ένα ποσοστό 25% τυχαίων peers από τους κοντινότερους της λίστας αυτής. Στο [14] και το [15] γίνεται μια κατηγοριοποίηση σε 3 (LAN, ADSL, dial-up hosts) και 2 (high, low bandwidth peers) κατηγορίες όπου από αυτές χρησιμοποιούνται ποσοστά από τις κατηγορίες χρηστών οι οποίοι θα συμπληρώνουν τη λίστα που θα είναι συνδεδεμένος κάθε peer. Στο [16] γίνεται χρήση super peers όπου αναλαμβάνουν να συλλέγουν στατιστικά όπως το τι pieces έχει κάθε peer της ομάδας, και ανάλογα παίρνουν αποφάσεις για το ποιοι peers θα συνδεθούν με ποιους αντικαθιστώντας έτσι σε αυτό το κομμάτι τη δουλειά του tracker. Στο [17] γίνεται μια κατηγοριοποίηση-χαρτογράφηση στα AS από τα οποία βρίσκεται κάθε peer που συνδέεται στον tracker και επιλέγεται κάθε φορά για κάθε peer μια λίστα με peers που έχουν το μικρότερο hop-count.

Από τη μεριά των δημοσιεύσεων οι οποίες ασχολούνται με μείωση της κίνησης έχουμε το [18] στο οποίο παρουσιάζεται η ιδέα του ISP-owned peer για τη βελτίωση της κίνησης στο δίκτυο. Επίσης, τα [19] και [20] ασχολούνται με προσπάθειες που μπορούν να γίνουν που συμβάλουν στη μείωση της Inter-domain κίνησης. Τέλος, τα [21] και [22]

αναφέρονται στην ανάγκη για συνεργασία των ISPs, σε επίπεδο τοπολογίας τους, για το διαμοιρασμό του περιεχομένου έτσι ώστε να μειωθεί η κίνηση μεταξύ τους με λιγότερα hops.

Ως μελλοντική δουλειά, με βάση την παρούσα διπλωματική εργασία, θα μπορούσε να γίνει μια επέκταση η οποία να κοιτάει αν σε κάθε AS ξεχωριστά υπάρχει συγκεντρωτικά από όλους τους peers το αρχείο, οπότε να αυξάνεται κάθε φορά το ποσοστό του παράγοντα a σε τέτοια περίπτωση ή σε περίπτωση που υπάρχει για κάποιο αρχείο ικανοποιητικός και συμμετρικός αριθμός peers ο οποίος θα ευνοούσε ένα τέτοιο σενάριο αλλαγής του παράγοντα. Επίσης θα μπορούσαν να χρησιμοποιηθούν υβριδικές τεχνικές, ίσως με κάποιο Piece Selection Strategy το οποίο θα έδινε ακόμα καλύτερα αποτελέσματα αθροιστικά.

Όπως είδαμε και πιο πάνω υπάρχει επίσης και αρκετή δουλειά σε επίπεδο ISPs. Επομένως θα ήταν καλή και κάποια υλοποίηση η οποία με κάποιο τρόπο να συνδέει ή να κάνει τους ISPs να συνεργάζονται σε περίπτωση που υπάρχει κίνηση BitTorrent στα δίκτυά τους.

Σίγουρα η συγκεκριμένη περιοχή είναι αρκετά "καυτή" και θα πρέπει να αναμένουμε εξελίξεις καθώς η κίνηση στα δίκτυα αυξάνεται ραγδαία όσο περνάει ο καιρός από BitTorrent δεδομένα. Επομένως αφού υπάρχει περιθώριο βελτίωσης του πρωτοκόλλου καλό θα ήταν να γίνει σε αυτό το επίπεδο πρώτα προσπάθεια και μετά σε δεύτερη φάση να υπάρξει κινητοποίηση στους ISPs.

Βιβλιογραφία

[1]. Thessaloniki's Wireless Metropolitan Network Wiki, http://wiki.twmn.net/index.php/Wireless_Torrent_Tracker

[2]. BitTorrent protocol, http://www.bittorrent.org/beps/bep_0003.html

[3]. BitTorrent development community. BitTorrent Protocol Specification v1.0. http://wiki.theory.org/BitTorrentSpecification

[4]. K. Katsaros, V.P.Kemerlis, Ch. Stais, G.Xylomenos, "A BitTorrent Module for the OMNeT++ Simulator," Proc. 17th Annual Meeting of the IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), London, Great Britain, September 2009 [pdf]

[5]. Marco Slot, "Latency-driven BitTorrent", Department of Computer Science-Faculty of Sciences Vrije Universiteit, [pdf]

[6]. Ipoque, "Internet Study 2008-2009", [pdf]

[7]. A.Varga, OMNET++ documentation, http://www.omnetpp.org/documentation

[8]. INET Framework, http://inet.omnetpp.org/

[9]. OverSim Framework, http://www.oversim.org/

[10]. K. Katsaros, "GT-ITM Topologies for the OMNeT++ Simulation Platform and OverSim Framework", http://mm.aueb.gr/research/GTITM-OMNeT4/#Overview

[11]. NS-2 Blog, "GT-ITM : Network Topology Generator", http://ns-2.blogspot.com/2006/05/gt-itm-network-topology-generator.html

[12]. Michael Piatek, Tomas Isdal, Thomas Anderson, Arvind Krishnamurthy, Arun Venkataramani, "Do incentives build robustness in BitTorrent?" , [pdf]

[13]. Marco Slot, Paolo Costa, Guillaume Pierre and Vivek Rai, "Zero-Day Reconciliation of BitTorrent Users with Their ISPs", (VU University Amsterdam), August 2009

[14]. Simon G. M. Koo , Karthik Kannan , C. S. George Lee, "On neighbor-Selection Strategy in hybrid Peer-to-Peer Networks", (Purdue University), February 2006 [pdf]

[15]. Bin Fan, Dah-Ming Chiu, John C.S. Lui, "The Delicate Tradeoffs in BitTorrent-like File Sharing Protocol Design", (Dept. of Computer Science & Eng. The Chinese University of Hong Kong), 2006 [pdf]

[16]. Rong, L., Burnett, I. , "BitTorrent in a dynamic resource adapting peer-to-peer network", (Wollongong Univ., Australia), December 2005[pdf]

[17]. Bo Liu, Yi Cui, Yansheng Lu, and Yuan Xue, "Locality-Awareness in BitTorrent-like P2P Applications", (Dept. of Electrical Engineering and Computer Science-University of Illinois) [pdf]

[18]. Ioanna Papafili, George D. Stamoulis, Sergios Soursos, "Insertion of ISP-owned Peer & Locality Awareness in BitTorrent", AUEB, (EuroNF workshop), Athens October 2008 [link]

[19]. Ruben Cuevas, Nikolaos Laoutaris, Xiaoyuan Yang, Georgos Siganos, Pablo Rodriguez, "Deep Diving into BitTorrent Locality", (Telefonica Research), July 2009 [pdf]

[20]. Stevens Le Blond, Arnaud Legout and Walid Dabbous, "Pushing BitTorrent Locality to the Limit", I.N.R.I.A. Sophia-Antipolis Mediterranee France [pdf]

[21]. Chen Tian, Xue Liu, Hongbo Jiang, Wenyu Liu, Yi Wang, "Improving BitTorrent Traffic Performance by Exploiting Geographic Locality", (Department of EIE, Huazhong University of Science and Technology, Wuhan, Hubei, China), 2008 [pdf]

[22]. Vinay Aggarwal, Anja Feldmann, Christian Scheideler
"Can ISPs and P2P Users Cooperate for Improved Performance?", Deutsche Telekom Laboratories/TU Berlin Berlin, Germany, July 2007 [pdf]