ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ

ΣΤΗΝ ΕΠΙΣΤΗΜΗ ΤΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Διπλωματική Εργασία
Μεταπτυχιακού Διπλώματος Ειδίκευσης

«A gossip dissemination protocol for a Publish/Subscribe P2P system»

Παναγιώτης Χασάπης

Επιβλέπων: κ. Γεώργιος Ξυλωμένος

ΑΘΗΝΑ, ΜΑΙΟΣ 2010

Περίληψη

Ο τρόπος με τον οποίο λειτουργεί το Internet έχει περάσει από πολλές φάσεις τα τελευταία δέκα χρόνια. Τα μέχρι τώρα μοντέλα προώθησης μηνυμάτων, όπως το HTTP, στηρίζονται στο TCP και είναι καθαρά client-server. Παλαιότερα, η πληροφορία που θέλαμε υπήρχε σε συγκεριμμένο κόμβο του οποίου την διεύθυνση αναφοράς γνωρίζαμε και μας ενδιέφερε να παραλάβουμε δεδομένα από αυτή και μόνο.

Πλέον, το σκηνικό έχει αλλάξει δραματικά. Πέρα από αιτίες που σχετίζονται με την επεκτασιμότητα των δικτύων, η ίδια η διαδικασία παραγωγής της πληροφορίας έχει επηρεαστεί πολύ από διάφορους παράγοντες. Οι παραγωγοί αλλά και οι καταναλωτές πληροφορίας είναι πολλοί, και σχηματίζουν μεταξύ τους σχέσεις εξάρτησης. Αυτές οι σχέσεις εξάρτησης πολλές φορές αποτελούν κοινωνικά δίκτυα. Κάθε ένας από αυτούς παράγει πληροφορία που πολλές φορές εντάσσεται σε κατηγορίες. Οι καταναλωτές θα προτιμούσαν να δηλώνουν το ενδιαφέρον τους για κατηγορίες και όχι για συγκεκριμένους χρήστες-παραγωγούς. Αυτό έχει ως αποτέλεσμα την ανάγκη δημιουργίας αρχιτεκτονικών που είναι πολλές προς πολλούς και όχι 1-1, όπως είναι η παραδοσιακή αρχιτεκτονική πελάτη-εξυπηρετητή.

Στην παρούσα εργασία θα χρησιμοποιηθεί η δυναμική των κοινωνικών δικτύων για την γρηγορότερη διάδοση πληροφορίας πολλών προς πολλούς. Η διάδοση των γεγονότων που θα αφορούν την δήλωση προσφοράς πληροφορίας μιας κατηγορίας (topic) αλλά και της εκδήλωσης ενδιαφέροντος για κάποια, θα γίνετε με μια τεχνολογία τύπου topic-based Publish/Subscribe. Το δίκτυο μας θα αποτελεί ένα αδόμητο ομότιμο (P2P σύστημα, όπου η διάδοση των publish και subscribe μηνυμάτων θα γίνετε μέσω ενός πρωτοκόλλου gossip.

Τέλος, θα παρουσιαστούν διάφορα αποτελέσματα από πειράματα που θα γίνουν με διάφορες παραμέτρους και θα περιγραφούν σενάρια χρήσης του παραπάνω μοντέλου.

Λέξεις κλειδιά:
Unstructured peer-to-peer, OMNET++, Oversim, Publish/Subscribe, Gossip/epidemic protocol, dissemination protocol, social networks

Abstract

The way the Internet works has changed a lot over the last decade. Current message exchange protocols, such as HTTP, are based upon TCP and are purely client-server. In the first days of the networks we wanted specific data objects, located at computer nodes, to which specific reference addresses we knew.

The scene has changed since then. The whole process of creating information and knowledge has been influenced by several factors. Today, there are many producers and consumers of data objects and dependence relationships are formed between them, usually resulting in social graphs. The information objects that are produced or consumed are usually categorized in topics. Consumers would prefer to declare their preference for a specific category rather than for specific users-producers. This creates the need to create communication protocols and frameworks that are many-many and not one-one, as the traditional client-server architecture.

In this thesis, the dynamics of social networks will be used to simplify approach to many-many communication. A topic based Publish/Subscribe architecture will be used to exchange the advertisements for the publication of data objects and the subscription to topics. The nodes of this network will form an unstructured P2P system, and the dissemination process will be described by an epidemic/gossip algorithm.

Finally, several experimental results will be presented, as well as usage scenarios of the proposed method.

Keywords:
Unstructured peer-to-peer, OMNET++, Oversim, Publish/Subscribe, Gossip/epidemic protocol, dissemination protocol, social networks

Ευχαριστίες

Οι πρώτες ευχαριστίες μου πηγαίνουν στους καθηγητές κ. Γεώργιο Ξυλωμένο και κ. Γεώργιο Πολύζο. Ήταν μεγάλη μου τιμή να δεχτούν να είναι επιβλέποντες στην εργασία μου και η καθοδήγηση του όλα τα χρόνια των σπουδών μου με βοήθησαν να φέρω εις πέρας το μεταπτυχιακό.

Μεγάλες ευχαριστίες στους κύριους Κωνσταντίνο Κατσαρό και Χαρίλαο Στάη, για την βοήθεια τους σε θέματα διαδικαστικά, στα πρόγραμματα προσομοίωσης Omnet++ και Oversim, αλλά και σε θέματα πιθανοτήτων και στατιστικής.

Να ευχαριστήσω επίσης τον συμφοιτητή και φίλο κ. Ηλία Φουνταλή για την παρέα του και που με ανέχτηκε τόσο καιρό, καθώς και για τις συμβουλές του σε θέματα θεωρίας γράφων. Ακόμα, τους φίλους αλλά και τους συμφοιτητές μου για την σημαντική τους παρέα κατά την διάρκεια των σπουδών μου.

Τέλος, αν και αυτονόητο, θα ήθελα να ευχαριστήσω τους γονείς μου για όλη την υποστήριξη τους κατά την διάρκεια των σπουδών μου.

Περιεχόμενα

Λίστα σχημάτων και πινάκων

Περιγραφή του προβλήματος

Ο τρόπος με τον οποίο λειτουργεί το Internet (και ειδικά το Web) έχει αλλάξει πολύ τα τελευταία δέκα χρόνια. Πέρα από νέες τεχνολογίες που εμφανίστηκαν, έχουν συμβεί και πολλές αλλαγές στην νοοτροπία των χρηστών. Υπηρεσίες όπως το StumbleUpon[20], το YouTube, το MySpace κτλ, έχουν δείξει ότι ο χρήστης:

α) ενδιαφέρεται για πληροφορία που μπορεί με αυτόματο ή μη τρόπο, να ταξινομηθεί σε θεματικές κατηγορίες,

β) ενδιαφέρεται για πληροφορία από συγκεκριμένα groups ή φίλους

γ) δεν ενδιαφέρεται (πάντοτε) από που μπορεί να προέρχεται η πληροφορία

δ) υπάρχουν παραγωγοί και καταναλωτές πληροφορίας που εντάσσεται σε θεματολογίες

Τα μέχρι τώρα μοντέλα προώθησης μηνυμάτων, όπως το HTTP, στηρίζονται στο TCP και είναι καθαρά client-server. Τα μοντέλα αυτά εξυπηρετούσαν τους σκοπούς των αρχικών υπηρεσιών της δεκαετίας του '90:

α) η πληροφορία που θέλουμε υπάρχει σε συγκεριμμένο κόμβο του οποίου την διεύθυνση αναφοράς γνωρίζουμε

β) μας ενδιαφέρει να παραλάβουμε αυτή και μόνο αυτή

γ) ο αποστολέας οφείλει να στέλνει τα δεδομένα μόνο σε αυτούς που τα ζητήσανε ρητά εκ των προτέρων (άρα γνωρίζει τις διευθύνσεις τους)

Το κύριο πρόβλημα που καλούμαστε να λύσουμε στην σύγχρονη διαδικτύωση αφορά την σύνδεση όχι 1-1 χρήστη, αλλά πολλών προς πολλούς. (Οι λύσεις που αφορούν την 1-1 θα μπορούσαν να χρησιμοποιηθούν και εδώ, με προβλήματα όμως στην κλιμάκωση). Οι παραγωγοί αλλά και οι καταναλωτές πληροφορίας είναι πολλοί και σχηματίζουν μεταξύ τους σχέσεις εξάρτησης. Αυτές οι σχέσεις εξάρτησης αποτελούν κοινωνικά δίκτυα. Κάθε ένας από αυτούς παράγει πληροφορία που πολλές φορές εντάσσεται σε κατηγορίες. Οι καταναλωτές ενδιαφέρονται να δηλώνουν το ενδιαφέρον τους σε κατηγορίες και όχι μόνο σε συγκεκριμένους χρήστες-παραγωγούς.

Επιπλέον, τα ομότιμα (peer-to-peer, p2p για συντομία) συστήματα ανταλλαγής αρχείων καθορίζουν πως γίνετε η ανταλλαγή πληροφορίας πλέον. Πολλοί διαφορετικοί χρήστες έχουν το ίδιο υλικό σε διάφορα μέρη του κόσμου. Ο καταναλωτής αυτών των στοιχείων ενδιαφέρετε μόνο να τα βρει γρήγορα και απλά, χωρίς να εξαρτάται από κεντρικούς κόμβους (και από συγκεκριμένους χρήστες) στο δίκτυο. Η υπερπληθώρα της πληροφορίας σήμερα (σε αριθμό αντιγράφων ενός πρωτότυπου αρχείου, αλλά και πρωτότυπων) μας δείχνουν ότι για λόγους οικονομίας αλλά και επεκτασιμότητας (scalability), πρέπει να αναζητηθούν άλλοι τρόποι ενημέρωσης για την παραγόμενη πληροφορία.

Σε αυτή την εργασία, λαμβάνουμε υπόψιν όλα τα παραπάνω και προσανατολιζόμαστε σε τεχνολογίες p2p με ανταλλαγή μηνυμάτων που στηρίζεται σε Publish/Subscribe μοντέλο. Στην επόμενη ενότητα ακολουθούν θα περιγραφούν εν συντομία τα δύο αυτά είδη αρχιτεκτονικών.

Συστήματα p2p

Εισαγωγή

Τα ομότιμα συστήματα είναι μια ειδική περίπτωση κατανεμημένων συστημάτων. Υλοποιούν ένα δίκτυο υπολογιστών στο επίπεδο εφαρμογής όπου κάθε οντότητα (peer) είναι ένα πρόγραμμα που λειτουργεί και ως εξυπηρετητής και ως πελάτης. Κάθε οντότητα-κόμβος συντηρεί πληροφορία που σε κάποια χρονική στιγμή χρησιμοποιούνται από άλλους (ως πελάτες) του συστήματος. Σε άλλες χρονικές στιγμές, αυτή η οντότητα που συγκέντρωνε πληροφορίες, στην συνέχεια μπορεί να ζητήσει πληροφορίες (ως πελάτης) από άλλους που τώρα θα λειτουργούν ως εξυπηρετητές. Τα συστήματα αυτά λειτουργούν πάνω από το επίπεδο δικτύου των ISP's δίνοντας την δυνατότητα σε χρήστες από όλο το κόσμο να εισέλθουν σε αυτά. Το δίκτυο που σχηματίζετε από αυτή την διαδικασία και βρίσκεται (ιεραρχικά) «πάνω» απο τα επίπεδα δικτύου και μεταφοράς, έχει επικρατήσει να λέγεται δίκτυο overlay.

Αν και έχουν γίνει αρκετά δημοφιλή τα τελευταία δέκα χρόνια εξαιτίας μιας ειδικής χρήσης αυτών για ανταλλαγή αρχείων, η πρώτη αναφορά σε p2p σύστημα γίνεται τα πρώτα χρόνια λειτουργίας του Internet (συγκεκριμένα στο RFC 1). Πρωτόκολλα όπως το SMTP αλλά και ο τρόπος λειτουργίας του Usenet, αφορούν την επικοινωνία μεταξύ ομότιμων (peer) προγραμμάτων.

Οι δύο κυριότερες κατηγορίες ομότιμων συστημάτων αφορούν το τρόπο οργάνωσης του δικτύου τους. Σε κάθε κατηγορία θα αναλυθεί ο τρόπος σκέψης και θα περιγραφούν σύντομα, υλοποιημένα παραδείγματα τέτοιων συστημάτων.

Δομημένα (Structured) p2p

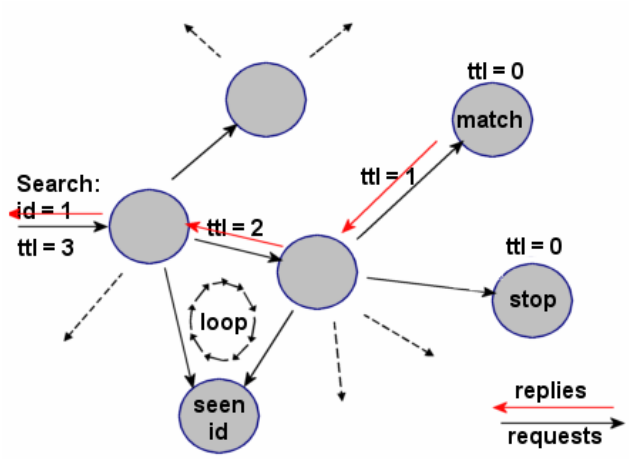
Τα δομημένα p2p συστήματα είναι η βασικότερη κατηγορία ομότιμων συστημάτων. Εδώ, υπάρχει ένα προκαθορισμένο πρωτόκολλο το οποίο ακολουθούν όλοι και αναλαμβάνει να αντιστοιχεί κάποια αντικείμενα με συγκεκριμένους peers. Όταν κάποιος θέλει κάτι (με το ίδιο τρόπο που αντιστοιχεί) ζητάει να μάθει ποιος έχει αναλάβει να ασχοληθεί με αυτό. Η υλοποίηση ενός τέτοιου συστήματος γίνετε συνήθως με κατανεμημένους πίνακες κατακερματισμού (distributed hash tables, DHTs).

Τα DHTs ανήκουν στην κατηγορία των αποκεντρωμένων κατανεμημένων συστημάτων που παρέχουν μια υπηρεσία αναζήτησης παρόμοια με έναν πίνακα hash: ζεύγη (κλειδί, αξία) αποθηκεύονται στο DHT, και κάθε συμμετέχων κόμβος μπορεί να ανακτήσει αποτελεσματικά την αξία που συνδέεται με ένα συγκεκριμένο κλειδί. Η ευθύνη για τη διατήρηση της χαρτογράφησης από τα κλειδιά για τις τιμές κατανέμεται μεταξύ των κόμβων, κατά τρόπον ώστε μια μεταβολή στο σύνολο των συμμετεχόντων να προκαλεί ελάχιστη αναστάτωση. Αυτό επιτρέπει στα DHT's να κλιμακώνονται σε εξαιρετικά μεγάλο αριθμό κόμβων και να χειρίζεται συνεχείς αφίξεις κόμβο. Συνήθως οι κόμβοι είναι συνδεδεμένοι με στατικό τρόπο σε κυκλικό σχήμα και φροντίζουν να διατηρούν αυτή την δομή με μηνύματα ενημέρωσης κατάστασης. Στα μειονεκτήματα τους είναι ότι ανταλλάσσουν πολύ πληροφορία για να λειτουργήσουν και αν και είναι επεκτάσιμα, όταν ένας κόμβος αποχωρήσει με «άτσαλο» (ungraceful) τρόπο, το δίκτυο υπερφορτώνεται με μηνύματα για να αποκατασταθεί η λειτουργία του.

Εφαρμογές που χρησιμοποιούν DHTs υπάρχουν αρκετές. Ο κύριος χώρος ανάπτυξης και χρήσης αυτών είναι ο ακαδημαϊκός. Υπάρχουν παραδείγματα όπου χρησιμοποιούνται ως πρόσθετη λύση: η βιβλιοθήκη Kademlia στηρίζει τον τρόπο λειτουργίας του BitTorrent, ενώ η Kad Net χρησιμοποιείτε στο eMule.

Αδόμητα (Unstructured) p2p

Τα αδόμητα p2p συστήματα είναι η δεύτερη κατηγορία συστημάτων, με τα οποία θα ασχοληθούμε και περισσότερο σε αυτή την εργασία. Σε ένα τέτοιο δίκτυο p2p οι όποιες συνδέσεις είναι συνήθως αυθαίρετες, χωρίς συγκερκιμμένο πρωτόκολλο, σχηματίζοντας έτσι έναν κατανεμημένο γράφο (distributed graph). Τα εν λόγω δίκτυα μπορούν εύκολα να κατασκευαστούν, με κάθε νέο peer που θέλει να ενταχθεί στο δίκτυο να αντιγράφει υπάρχουσες συνδέσεις ενός άλλου κόμβου και στη συνέχεια να διαμορφώνει τις δικές του σχέσεις με την πάροδο του χρόνου. Εάν ένας peer θέλει να βρει ένα επιθυμητό κομμάτι των δεδομένων στο δίκτυο, το ερώτημα πρέπει να πλημμυρίσει το δίκτυο για να βρει peers οι οποίοι να μοιράζονται τα αντικείμενα. Αυτό το μήνυμα αναζήτησης πηγαίνει προς όλες τις κατευθύνσεις μέχρι ένα μικρό αριθμό hops (συνήθως 6). Εάν κάποιος κατέχει ένα δεδομένο που κάνει αντιστοίχηση με αυτό ρωτήθηκε, τότε αναλαμβάνει να ενημερώσει αυτόν που έστειλε το μήνυμα ότι έχει κάτι που (πιθανότατα) τον ενδιαφέρει.



Εικόνα 1: Παράδειγμα διάδοσης ερωτήματος σε αδόμητο peer-to-peer δίκτυο

Στα βασικά πλεονεκτήματα των αδόμητων δικτύων είναι ότι μπορούν να αντεπεξέλθουν σε οποιαδήποτε αποχώρηση χρήστη, και ότι μπορούν να χρησιμοποιηθούν σε ένα πλήθος από σενάρια με ενσύρματη ή ασύρματη τεχνολογία (π.χ. ad hoc δίκτυα). Το βασικό μειονέκτημά τους όμως είναι ότι το ερώτημα που τίθεται δεν μπορεί πάντα να απαντηθεί. Δημοφιλές περιεχόμενο είναι πιθανό να είναι διαθέσιμο σε διάφορους χρήστες αλλά αν ένας peer ψάχνει για σπάνια αντικείμενα ή μοιράζεται μόνο μερικά άλλα τους ομολόγους του, τότε είναι μάλλον απίθανο ότι η αναζήτηση θα είναι επιτυχής. Δεδομένου ότι δεν υπάρχει συσχετισμός μεταξύ ομότιμων και το περιεχόμενο που αυτοί διαχειρίζονται, δεν υπάρχει εγγύηση ότι οι πλημμύρες θα βρουν έναν peer που έχει τα επιθυμητά στοιχεία. Έτσι τα αδόμητα p2p συστήματα έχουν πολύ κακή απόδοση αναζήτησης. Παρόλα αυτά, σε περιπτώσεις όπως η θεματολογία της

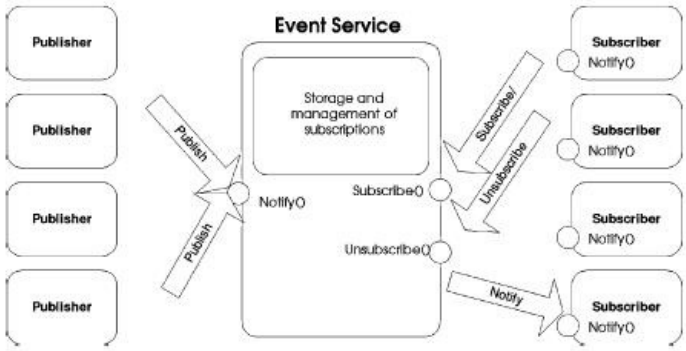
παρούσας εργασίας, η χρήση μιας κατηγορίας gossip αλγόριθμων μπορούν να δώσουν καλύτερα αποτελέσματα.

Το Gnutella αποτελεί ένα χαρακτηριστικό παράδειγμα αδόμητο δικτύου. Δημιουργήθηκε το 2000 από προγραμματιστές της εταιρίας Nullsoft, ως ένα βοηθητικό λογισμικό για την ανταλλαγή μουσικών αρχείων. Παρά τα όποια προβλήματα που αντιμετώπισε και τις πολλές αλλαγές που δέκτηκε στην πάροδο του χρόνου θεωρείτε μακράν ένα από τα καλύτερα file sharing πρωτόκολλα που κυκλοφόρησαν και από τα ποιο ανθεκτικά στο χρόνο.

Το μοντέλο Publish/Subscribe

Εισαγωγή

Το Publish/Subscribe (ή pub/sub για συντομία) αποτελεί ένα μοντέλο προώθησης μηνυμάτων χωρίς συγκεκριμένους παραλήπτες και αποστολείς. Είναι ένα παράδειγμα μετάδοσης πολλούς προς πολλούς. Πρωτοαναφέρθηκε ως συγκεντρωτική αρχιτεκτονική ενδιάμεσου λογισμικού το 1987 σε μια δημοσίευση συνεδρίου της ACM [28]. Το συγκεκριμένο άρθρο αναφερόταν σε ένα ενδιάμεσο λογισμικό στο οποίο ένα μικρό υποσύστημα αποτέλεσε την αρχή αυτό του μοντέλου προώθησης μηνυμάτων.



Εικόνα 2: Αφηρημένο σχήμα Publish/Subscribe αρχιτεκτονικής

Τα συστήματα αυτά περιέχουν δύο βασικές κατηγορίες χρηστών (agents): μια κατηγορία που ονομάζονται publishers και μια που λέγονται subscribers. Οι publishers είναι συνήθως παραγωγοί ή κάτοχοι κάποιων αντικειμένων πληροφορίας. Για την ύπαρξη αυτών των αντικειμένων θέλουν να ενημερώνονται οι subscribers. Οι publishers δηλώνουν το αντικείμενο στην υπηρεσία γεγονότων του συστήματος (event service) μέσω μιας μεθόδου που καλείται pub(). Οι subscribers δηλώνουν το ενδιαφέρον στην ίδια υπηρεσία μέσω μιας συνάρτησης που καλείται sub(). Το σύστημα της υπηρεσίας γεγονότων χρησιμοποιεί τα μηνύματα που παράγονται από αυτές τις μεθόδους και βλέπει ποιες αντιστοιχήσεις από publishers-subscribers μπορούν να γίνουν.

Για να επιτραπεί αυτή η σύνδεση πολλών προς πολλούς, χρειάζονται τρία είδη αποσύνδεσης (decoupling). Το πρώτο είδος αφορά τον χρόνο: ένας subscriber μπορεί να ενημερωθεί για κάτι που έγινε published νωρίτερα από την στιγμή που δήλωσε το ενδιαφέρον του (με την sub()). Το άλλο είδος αφορά το χώρο: ούτε οι publishers χρειάζεται να γνωρίζουν ποιοι είναι οι subscribers τους, ούτε οι subscribers ποιοι είναι οι publishers – μόνο η υπηρεσία γεγονότων έχει αυτή την γνώση. Το τελευταίο είδος αποσύνδεσης είναι του συγχρονισμού.

Η αποστολή των μηνυμάτων γίνετε ασύγχρονα και χωρίς εμποδισμό (blocking) και για τις δύο κατηγορίες χρηστών.

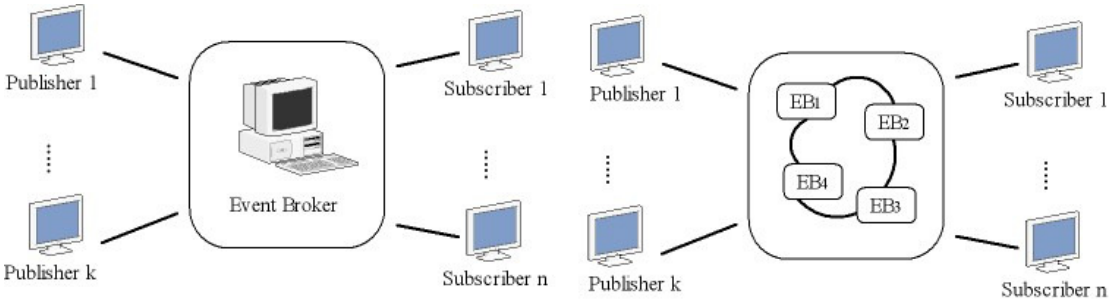
Στην ενότητα που ακολουθεί θα περιγραφούν τα Publish/Subscribe συστήματα ως προς την επεκτασιμότητα τους, τον τρόπο προώθησης μηνυμάτων και τους τύπους μηνυμάτων.

Επεκτασιμότητα

Συγκεντρωτικά (Centralized) συστήματα Publish/Subscribe

Τα συγκεντρωτικά συστήματα pub/sub ήταν και τα πρώτα που εμφανίστηκαν ιστορικά. Ένας κεντρικός κόμβος αναλαμβάνει να εκτελέσει ένα αλγόριθμο ταιριάσματος για να συνδέσει τα publications με τα subscriptions. Στο τομέα της τεχνολογίας λογισμικού έχει δημιουργηθεί το σχεδιαστικό πρότυπο (design pattern) Observer που αφορά ακριβώς αυτή την διαχείριση των γεγονότων.

Το πρόβλημα που προκαλείται με αυτήν την αρχιτεκτονική είναι το κλασικό που υπάρχει στα συγκεντρωτικά συστήματα: το όλο σύστημα στηρίζεται σε ένα κόμβο που αν καταρρεύσει, η υπηρεσία παύει να λειτουργεί. Επιπλέον, εμφανίζεται και ένα ακόμα πρόβλημα bottleneck – εμπειρικά έχει φανεί ότι publishers είναι λιγότεροι από ότι οι subscribers και οι τελευταίοι ζητάνε από πολλούς publishers δεδομένα, κάνοντας τον κεντρικό κόμβο να δυσχεραίνετε στην λειτουργία του.



Εικόνα 3: Παραδείγματα κατανεμημένων συστημάτων Publish/Subscribe

Αποκεντρωμένα (decentralized) συστήματα Publish/Subscribe

Η εμφάνιση των κατανεμημένων συστημάτων pub/sub ήταν το επόμενο λογικό βήμα. Τα συστήματα αυτά άρχισαν να εμφανίζονται εκτενώς στην βιβλιογραφία γύρω στο 2000, μετά την εμφάνιση και των πρώτων συστημάτων p2p. Εδώ μπορούν να ειπωθούν πολλά, αλλά θα γίνει κλιμακωτά στις επόμενες ενότητες, μιας και αυτά τα συστήματα δεν μπαίνουν σε μια ή δύο κατηγορίες. Εδώ απλά θα αναφέρουμε ότι η καθαυτή δομή του κατανεμημένου συστήματος διαχείρισης και μετάδοσης μηνυμάτων, εξαρτάται, εάν έχουμε δομημένα (με DHT) ή αδόμητο δίκτυο, εάν έχουμε TCP ή UDP πρωτόκολλο μετάδοσης κτλ.

Τρόπος προώθησης μηνυμάτων

Τεχνική Push

Ο τρόπος μετάδοσης push είναι μια τεχνική που επιτρέπει το ξεκίνημα της επικοινωνίας να γίνεται από τον εξυπηρετητή και όχι από τον πελάτη. Τα συμβατικά πρωτόκολλα επικοινωνίας όπως το TCP λειτουργούν ανάποδα περιμένοντας τον πελάτη να ξεκινήσει την επικοινωνία.

Παραδείγματα αυτής της τεχνολογίας υπάρχουν πολλά. Το Windows Update στο λειτουργικό σύστημα Windows, περιμένει διαρκώς μηνύματα push από τον εξυπηρετητή της Microsoft. Όταν υπάρχουν καινούρια updates, ο εξυπηρετητής αναλαμβάνει να στείλει τις ενημερώσεις στους όποιους ενδιαφερόμενους. Μια άλλη περίπτωση που χρησιμοποιεί την push τεχνική είναι στους εξυπηρετητές Web. Οι τελευταίοι, χρησιμοποιούν την δυνατότητα του HTTP πρωτοκόλλου για δημιουργία persistent HTTP συνδέσεων. Μετά την αποστολή δεδομένων που αρχικά ζήτησε ο πελάτης, ο εξυπηρετητής διατηρεί την σύνδεση ανοιχτή. Όταν ο εξυπηρετητής θέλει να στείλει κάτι, ο HTTP πελάτης (browser) το λαμβάνει και προβαίνει σε αλλαγές της σελίδας που εμφανίζετε στον χρήστη, εάν χρειάζεται κτλ. Άλλες εφαρμογές της push τεχνικής είναι η αποστολή αρχείων στο IRC πρωτόκολλο, στο MSN, το πρωτόκολλο SMTP κτλ.

Στα pub/sub συστήματα, η χρήση αυτής της τεχνικής, σημαίνει ότι ο publisher αναλαμβάνει να στέλνει τα publications του όταν έχει να πει κάτι καινούριο σε όσους ενδιαφέρονται.

Τεχνική Pull

Ο τρόπος μετάδοσης pull είναι μια τεχνική που επιτρέπει το ξεκίνημα μιας επικοινωνίας να γίνεται από τον πελάτη (π.χ. το TCP).

Ένα κλασικό παράδειγμα με αυτήν την τεχνολογία είναι το πρωτόκολλο HTTP. Η λειτουργία του Really Simple Syndication (RSS) στηρίζετε σε RSS πελάτες που περιοδικά ρωτάνε τον RSS εξυπηρετητή για καινούρια δεδομένα. Τέλος, η λειτουργία παραλαβής e-mail από τον εξυπηρετητή mail (π.χ. POP3) είναι ένα pull-based πρωτόκολλο.

Στα pub/sub συστήματα, η χρήση αυτής της τεχνικής σημαίνει ότι ο subscriber αναλαμβάνει να στέλνει τα subscriptions του όταν θέλει να μάθει εάν υπάρχουν καινούρια publications.

Παραλήπτης ενημέρωσης αντιστοίχησης

Στα παραπάνω έχουμε αφήσει ανοιχτό το θέμα της ενημέρωσης όταν μια αντιστοίχηση λάβει τόπο. Στην βιβλιογραφία παρατηρούνται δύο διαφορετικές σχολές, όπου η μία θεωρεί ότι πρέπει να ενημερώνονται οι publishers ενώ η άλλη οι subscribers.

Στα περισσότερα σενάρια χρήσης πάντως η προϋπόθεση που ισχύει είναι ότι οι subscribers είναι περισσότεροι αριθμητικά από τους publishers (μάλιστα παράγουν πολλά περισσότερα μηνύματα). Για να αποφευχθεί λοιπόν υπερφόρτωση των publishers, προτείνεται γενικά να ειδοποιούνται κυρίως οι subscribers.

Τύποι μηνυμάτων

Ανάλογα με το πως σκοπεύουμε να χρησιμοποιήσουμε τα μηνύματα, έχουμε και αντίστοιχες κατηγορίες. Μετά από το 2000 επικράτησαν να χρησιμοποιούνται τρεις κατηγορίες μηνυμάτων: τα topic-based, τα content-based και τα type-based.

Topic-based Publish/Subscribe

Σε αυτά τα συστήματα έχουμε την έννοια των θεματολογιών (topics). Οι θεματολογίες αποτελούν μια απλή και εύχρηστη αφαίρεση για την διευκόλυνση της αντιστοίχισης των γεγονότων. Κάθε subscriber και publisher οφείλει να κατατάσσει αυτό που ζητάει ή προσφέρει σε μια συγκεκριμένη κατηγορία που χαρακτηρίζεται από ένα πεπερασμένο σύνολο bytes (συνήθως αλφαριθμητικών) π.χ. «Distributed Systems». Ο κόμβος (ή κόμβοι) που αναλαμβάνει την αντιστοίχηση, ελέγχει ποια αλφαριθμητικά είναι ίδια και ενημερώνει τους subscribers.

Μια εξέλιξη αυτής της τεχνικής είναι να χρησιμοποιηθούν θεματολογίες που αποτελούν ταξινομίες (taxonomies). Κάθε topic είναι ένα σύνολο από έννοιες, η μια εξειδίκευση της άλλης π.χ. «technology / computer science / Distributed Systems». Αυτό επιτρέπει την αντιστοίχηση subscribers που ζητάνε εξειδικευμένα θέματα όπως το παραπάνω, να λαμβάνουν ενημερώσεις και για αντικείμενα με θέμα «technology / computer science». Επίσης επιτρέπει την χρησιμοποίηση wildcards για καλύτερη και ευκολότερη αντιστοίχηση θεματολογιών.

Σε κάθε περίπτωση, τα συστήματα θεματολογιών λειτουργούν πάντα με αλγόριθμους αντιστοίχησης αλφαριθμητικών. Παρόλα αυτά όμως, ακόμα και με την χρήση ταξινομιών και wildcards, πάσχουν από μικρή εκφραστικότητα και στατική κατηγοριοποίηση αντικειμένων.

Content-based Publish/Subscribe

Σε αυτά τα συστήματα κάθε subscription είναι ένα σύνολο από ιδιότητες και τιμές σε αυτά. Ο κόμβος (ή κόμβοι) που αναλαμβάνει την αντιστοίχηση, διαθέτει μια γλώσσα επεξεργασίας ερωτημάτων (όπως η SQL) που επιτρέπει την χρήση αριθμητικών και λογικών τελεστών. Κάθε τι που επιστρέφεται από την επεξεργασία του ερωτήματος αποτελεί και μια αντιστοίχηση. Να σημειωθεί ότι σε αποκεντρωμένα συστήματα, αυτό προϋποθέτει μια κοινή γλώσσα επερωτημάτων και ένα προ-συμφωνημένο σύστημα επεξεργασίας της γλώσσας για να έχει νόημα η λειτουργία του συστήματος.

Type-based Publish/Subscribe

Τα type-based συστήματα είναι μια προέκταση των συστημάτων θεματολογίας. Κάθε θεματολογία συνδέετε μια έννοια, για την οποία γνωρίζουμε πως συνδέεται με άλλες έννοιες. Ο κόμβος (ή κόμβοι) που αναλαμβάνει την αντιστοίχηση, χρησιμοποιεί όχι μόνο την θεματολογία αλλά και τους τύπους εννοιών για να καταλάβει τι πρέπει να κάνει. Να σημειωθεί ότι σε αποκεντρωμένα συστήματα, αυτό προϋποθέτει ένα προ-συμφωνημένο σύστημα εννοιών για να έχει νόημα η λειτουργία του συστήματος.

Πρωτόκολλα gossip

Το πρωτόκολλα gossip (ή επιδημικά πρωτόκολλα) είναι μια ειδική κατηγορία τυχαιοκρατικών αλγόριθμων για την διάδοση και διασκόρπιση δεδομένων (data dissemination) σε κατανεμημένα συστήματα. Εφαρμόζονται κυρίως σε αδόμητα συστήματα p2p και έχουν ως κύριο σκοπό την διάδοση πολλαπλών αντιγράφων κάποιας πληροφορίας, σε όσο το δυνατόν περισσότερους peers. Προτάθηκαν για πρώτη φορά στο [23], ως ένας τρόπος επιδιόρθωσης στοιχείων βάσεων δεδομένων και από τότε έχουν πάρει διάφορες μορφές και έχουν χρησιμοποιηθεί εκτενώς κυρίως για αποκεντρωμένα κατανεμημένα συστήματα.

Οι περισσότεροι αλγόριθμοι gossip βασίζονται σε δύο βασικές συναρτήσεις. Η μια επιλέγει περιοδικά και τυχαία κάτι από μια τοπική cache και το στέλνει τυχαία σε έναν κόμβο. Η άλλη συνάρτηση λαμβάνει αυτά τα μηνύματα και τα καταχωρεί σε μια τοπική cache. Σε ένα αδόμητο σύστημα p2p κάθε peer διαθέτει πάντα και τις δύο.

Τα gossip πρωτόκολλα σε αδόμητα δίκτυα p2p χρησιμοποιούνται όταν ξέρουμε ότι έχουμε υπερπληθώρα όμοιας πληροφορίας, δεν μας ενδιαφέρει να έχουμε on demand και γρήγορα την πλήρη εικόνα ενός δικτύου καθώς και το από που θα παραλάβουμε αυτήν την πληροφορία. Είναι περισσότερο ευέλικτα από ότι οι λύσεις με DHT, επειδή ακριβώς πολλοί χρήστες στην πάροδο του χρόνου κρατάνε παρόμοια δεδομένα, και δεν χρειάζεται η εφαρμογή fault-tolerance πρωτοκόλλων.

Σχετικές εργασίες και συνεισφορά

Τα πρωτόκολλα gossip έχουν χρησιμοποιηθεί σε αδόμητα p2p συστήματα (όχι απαραίτητα Publish/Subscribe) εκτενώς πολλά χρόνια πριν. Τον τελευταίο καιρό έχουν δημιουργηθεί περιπτώσεις δικτύων που χρησιμοποιούνται σε μεγάλη κλίμακα (με αρκετούς χιλιάδες χρήστες) και χρησιμοποιούν gossip πρωτόκολλα για την διάδοση γεγονότων.

Ένα σημαντικό παράδειγμα είναι το SCAM της P2P Video Streaming εφαρμογής ονόματι Coolstreaming, που χρησιμοποιείται από την τελευταία για την διάδοση γεγονότων που αφορούν το membership χρηστών του συστήματος. Ένα άλλο παράδειγμα (βλ. [5]), το οποίο αφορά την μετάδοση λιστών προτιμήσεων μεταξύ χρηστών είναι ο αλγόριθμος Newscast σε δίκτυα που η οργάνωση μοιάζει με scale-free γράφους. Και τα δύο είναι πλήρως τυχαιοκρατικά (probabilistic): κάθε κόμβος θα στείλει κάποια πληροφορία σε ένα peer που επιλέγει τυχαία χωρίς να λαμβάνει υπόψιν πόσο τον ενδιαφέρει απαραίτητα κάτι (βλ. [24]),.

Μια άλλη εργασία (βλ. [9]) που μοιάζει πολύ με την παρούσα, αφορά την διάδοση μηνυμάτων σε ένα Publish/Subscribe p2p δίκτυο. Εκεί, ο gossip αλγόριθμος φροντίζει κάθε peer να γνωρίζει τα subscriptions των γειτόνων και των φίλων των γειτόνων του. Έτσι, όταν λαμβάνει ένα pub μήνυμα ελέγχει να δει ποιοι από όσους γνωρίζει μπορεί να ενδιαφέρονται (να είναι subscribers). Εάν δεν ενδιαφέρεται κανένας, τότε το στέλνει τυχαία σε κάποιον «γειτονικό» του peer (semi-probabilistic method).

Η παρούσα εργασία διαφοροποιείτε σε διάφορα σημεία από την τελευταία και δέχεται επιρροές από τις δύο πρώτες. Η παρατήρηση που κάνουμε στην τελευταία εργασία είναι ότι αναφέρεται σε τυχαίους γράφους που ως γνωστών δεν είναι ρεαλιστικοί, ούτε collision-free (κάθε peer έχει ακριβώς την ίδια πιθανότητα να καταρρεύσει, δημιουργώντας μεγάλα κενά ασυνέχειας στον γράφο). Ακόμα, φροντίζει να γνωρίζει μόνο γείτονες, ένα γεγονός το οποίο είναι αρκετά περιοριστικό. Όλες οι εργασίες θεωρούν ότι η κατανομή των subscriptions ή των λιστών προτίμησης είναι ομοιόμορφη.

Σε αυτήν την εργασία, σκοπεύουμε να διαφοροποιηθούμε ως εξής:

- Ο γράφος που θα σχηματίζουν οι κόμβοι θα είναι ένας κοινωνικός γράφος. Έτσι θα εκμεταλλευτούμε ιδιότητες που έχουν οι κοινωνικοί γράφοι για την γρηγορότερη διάδοση πληροφορίας (βλ. [10]).
- Η κατανομή των θεμάτων των subscriptions θα είναι τύπου power-law.
- Θα έχουμε ένα push (και όχι pull) gossip πρωτόκολλο.
- Το πρωτόκολλο επικοινωνίας θα είναι το UDP.
- Φροντίζουμε να στείλουμε τα δεδομένα των subscriptions αλλά και των publications όσο μακρύτερα στο δίκτυο γίνεται, και να κρατάμε σε όλους τους ενδιάμεσους peers ποιοι είναι οι παραγωγοί αυτών. Έτσι, όλοι οι χρήστες θα μπορούν να γνωρίζουν ποιοι ζητάνε κάποια publication και να κάνουν τα matching αυτοί ακόμα και αν οι παραγωγοί των pub μηνυμάτων βρίσκονται πολλά hops μακριά.

Για να υλοποιηθούν τα παραπάνω σωστά πρέπει να λάβουμε υπόψιν μας και θέματα βελτιστοποίησης και σταθερότητας. Στην ενότητα περιγραφής της αρχιτεκτονικής και του αλγόριθμου, θα γίνει ανάλυση των δομών δεδομένων που πρόκειται να χρησιμοποιηθούν.

Γράφοι

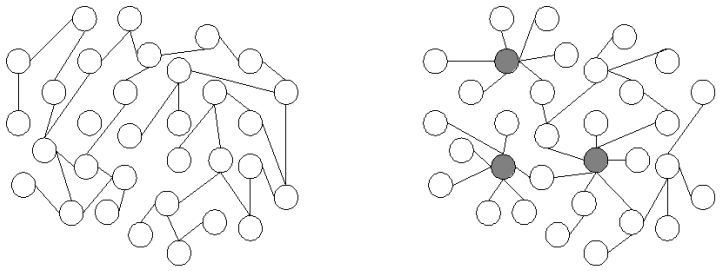
Το κύριο κομμάτι της εργασίας αφορά το είδος τοπολογίας δικτύου που θα χρησιμοποιηθούν στα πειράματα. Η κατασκευή της τοπολογίας του δικτύου είναι ένα θέμα που στις περισσότερες εργασίες δεν λαμβανόταν σοβαρά υπόψιν. Θεωρούμε ότι έχει νόημα μια τέτοια συζήτηση αφενός διότι γνωρίζουμε ότι υπάρχουν συγκεκριμένες τοπολογίες που είναι πιο ρεαλιστικές [17] και να μπορούν χρησιμοποιηθούν για να ισχυροποιήσουμε κάποια συμπεράσματα και ιδέες.

Ιδιότητες γράφου

Για το δικό μας σενάριο, θα θέλαμε ο γράφος να έχει κάποιες ιδιότητες για να έχει νόημα η χρήση του. Μας ενδιαφέρουν γράφοι:

- που είναι συνδεδεμένοι, γιατί θέλουμε να έχουμε έναν μεγάλο ενιαίο γράφο πάνω στο οποίο θα τρέξουμε τα πειράματα μας (τέτοιος γράφος είναι και ο πλήρης, που δεν αποτελεί όμως ένα ρεαλιστικό σενάριο τοπολογίας δικτύου οπότε δεν θα τον λάβουμε και υπόψιν),
- που χαρακτηρίζονται από κάποια power-law κατανομή,
- είναι μη-κατευθυνόμενοι, μιας και όπως και οι περισσότερες τοπολογίες, έχουν πάντα bi-directional connectivity, και
- δεν είναι δέντρα.

Οι οικογένειες γράφων που έχουν τις παραπάνω ιδιότητες είναι αρκετές. Μελετώντας την βιβλιογραφία γενικά για γράφους, είδαμε ότι υπάρχουν δύο συγκεκριμένες οικογένειες γράφων που κατέχουν τα χαρακτηριστικά που θέλουμε και χρησιμοποιούνται πολύ στην έρευνα, μιας και αποτελούν αρκετά ρεαλιστικά μοντέλα. Εμείς θα μελετήσουμε κυρίως τους scale-free γράφους, αφού αναφέρουμε περιληπτικά του random, για λόγους αντιπαραβολής.



(a) Random network (b) Scale-free network

Εικόνα 4: Δύο σημαντικές οικογένειες γράφων για το πρόβλημα μας

Random graphs

Οι random γράφοι ορίστηκαν για πρώτη φορά το 1959 από τους Paul Erdős και Alfréd Rényi. Αποτελούν γράφους που είναι συνδεδεμένοι αλλά δεν δίνουν καμία εγγύηση α) για τον μέσο όρο των node degrees ούτε β) για το πλήθος τον ακμών. Ένα τυχαίο γράφημα λαμβάνεται ξεκινώντας με ένα σύνολο κορυφών n και προσθέτοντας άκρα μεταξύ τους τυχαία. Συχνότερα έχει

μελετηθεί το μοντέλο Erdős - Rényi, στην οποία κάθε δυνατή ακμή εμφανίζεται ανεξάρτητα με πιθανότητα p. Αποτελούν την αφετηρία πολλών μελετητών και είναι από τις πρώτες επιλογές των προγραμματιστών για μελέτη τοπολογιών δικτύων.

Scale-free graphs

Τα δίκτυα αυτά είναι οι κατεξοχήν γράφοι που μελετώνται τα τελευταία χρόνια από πολλούς επιστήμονες της θεωρίας γράφων. Οι γράφοι αυτοί εμφανίζονται σε ένα εκπληκτικό φάσμα ετερογενών συστημάτων, από βιολογικά και κοινωνικά έως και σε καθαρά τεχνολογικά δίκτυα όπως το World Wide Web, καθώς και στο δίκτυο των ανθρώπων που συνδέονται μεταξύ τους με e-mail. Αποτελούν δίκτυα με μεγάλη σημασία και χρηστικότητα μιας και εμφανίζονται παντού.

Οι γράφοι αυτοί είναι συνδεδεμένοι, μη-δενδρικοί, και με κάποιες εγγυήσεις ως προς την δομή τους - αποτελούν ένα δίκτυο που η κατανομή του degree των κόμβων ακολουθεί μια τύπου power-law (ασυμπτωτικά): Η πιθανότητα P(k) των κόμβων του δικτύου που έχουν συνδέσεις *k* σε άλλους κόμβους είναι $P(k) = k^{-\gamma}$, όπου *γ* είναι μια σταθερή αξία συνήθως μεταξύ $2 < \gamma < 3$ (αν και περιστασιακά μπορεί να βρίσκεται εκτός αυτών των ορίων). Όπως συμβαίνει με όλα τα συστήματα που χαρακτηρίζονται από μια κατανομή power-law, οι περισσότεροι κόμβοι θα είναι μικρού βαθμού και λίγοι θα εμφανίζουν πολύ υψηλό βαθμό (το φαινόμενο αυτό είναι γνωστό και ως αναλογία 80-20). Οι κόμβοι υψηλού βαθμού (που συχνά αποκαλούνται κέντρα - "hubs") εξυπηρετούν συγκεκριμένους σκοπούς στα δίκτυά τους (αυτό εξαρτάται σε μεγάλο βαθμό τον τομέα χρήσης).

Η κατανομή του power-law επηρεάζει σε μεγάλο βαθμό την τοπολογία του δικτύου. Αποδεικνύεται ότι τα μεγάλα κέντρα συνδέονται στενά με τα αμέσως μικρότερα (σε πλήθος προσπίπτουσων ακμών). Αυτά τα μικρότερα με τη σειρά τους, ακολουθούνται από άλλους κόμβους με ένα ακόμη μικρότερο βαθμό και ούτω καθεξής. Αυτή η ιεραρχία τα κάνει πιο «ανθεκτικά» απέναντι σε τυχαίες καταστροφές: δεδομένου ότι η συντριπτική πλειοψηφία των κόμβων είναι αυτοί με μικρό βαθμό, η πιθανότητα ένας κεντρικός κόμβος να επηρεαστεί είναι σχεδόν αμελητέα. Ακόμη και αν συμβεί ένα τέτοιο γεγονός, το δίκτυο δεν θα χάσει την συνεκτικότητα του, η οποία διασφαλίζεται από τους υπόλοιπους κεντρικούς κόμβους. Από την άλλη πλευρά, αν μερικοί κεντρικοί κόμβοι απομακρυνθούν από το δίκτυο τότε μετατρέπεται σε ένα σύνολο απομονωμένων γραφημάτων (συνδεδεμένων όμως).

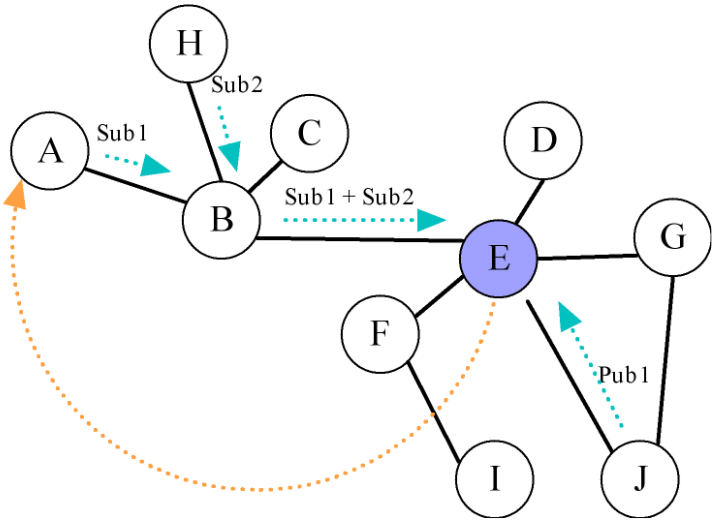
Αλγόριθμος

Θεωρούμε ότι οι peer θα αποτελούν μέλη ενός κατανεμημένου social network. Κάποιοι από αυτούς θα διαθέτουν υλικό ενώ κάποιοι άλλοι θα καταναλώνουν. Ένας αριθμός από αυτούς θα κάνει και τα δύο. Επιδιώκουμε μια topic-based Publish/Subscribe αρχιτεκτονική θεωρώντας ότι είναι ιδανική για ένα τέτοιο κατανεμημένο σύστημα. Τα αντικείμενα που θέλουμε να διαφημίσουμε θα χαρακτηρίζονται από ένα αριθμό για topic που θα λέγεται Rid όπως στο ερευνητικό πρόγραμμα PSIRP [3].

Οι στόχοι που θέτουμε είναι:

1. Να μειώσουμε το αριθμό παρόμοιων αντιτύπων μηνυμάτων που λαμβάνει ένας χρήστης
2. Να χρησιμοποιήσουμε τους κεντρικούς κόμβους, αλλά να μην εξαρτιόμαστε από αυτούς μόνο
3. Να έχουμε τοπικό balancing σε κάθε κόμβο
4. Το σύστημα να είναι επεκτάσιμο

Η βασική μας ιδέα είναι να εκμεταλλευτούμε το ότι υπάρχουν κεντρικοί κόμβοι στους scale-free γράφους, οι οποίοι θα μας βοηθήσουν να στέλνουμε όσο πιο μακριά γίνεται τα pub/sub μηνύματα μας. Επίσης όλοι οι υπόλοιποι που θα λαμβάνουν τα μηνύματα, θα φροντίζουν να τα κάνουν και αυτοί cache, προκειμένου να εξυπηρετούν άλλους και να έχουμε load balancing μεταξύ των peers.



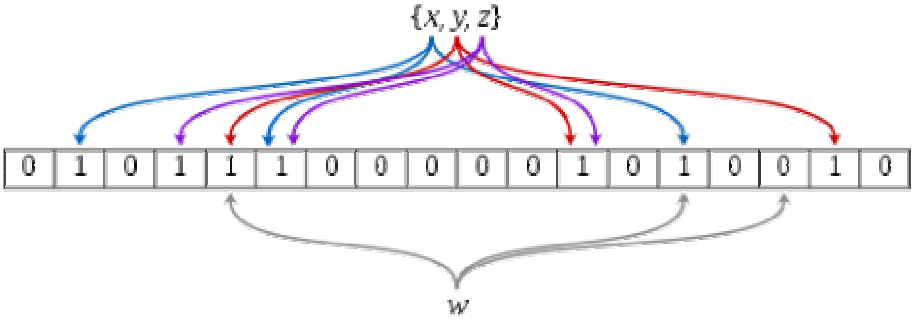
Εικόνα 5: Τρόπος λειτουργίας του πρωτοκόλλου μας

Κάθε peer λοιπόν θα στέλνει κάποια cached pub ή sub μηνύματα σε κάποιους γνωστούς (τα cached μηνύματα είναι δικά του μαζί με άλλων που έχει κάνει προσωρινά αποθήκευση). Όταν κάποιος ενδιάμεσος peer δει ότι κάποια sub και pub μηνύματα κάνουν match (επειδή ανήκουν στο ίδιο topic), τότε φροντίζει να ενημερώσει τους αντίστοιχους subscribers για την παρουσία (δηλαδή την διεύθυνση IP) του συγκεκριμένου publisher.

Παρακάτω ακολουθεί μια περιγραφή των βασικών ιδεών πάνω στις οποίες δομήθηκε ο αλγόριθμος.

Μείωση παρόμοιων λήψεων

Εργασίες παρόμοιες με την δική μας, έχουν αμελήσει την μείωση των διπλότυπων. Στο [9], οι δημιουργοί του χρησιμοποιούν ένα μετρητή για τα hops. Εμείς αποφασίσαμε να χρησιμοποιήσουμε bloom filters [13]. Εν συντομία, είναι μια πιθανοτική δομή δεδομένων (έστω B) που μπορεί να απαντά μόνο σε μια ερώτηση: "δοθέντος ενός στοιχείου x, ανήκει το x στο B;". Το B δεν περιέχει στοιχεία, αλλά αντίθετα διατηρεί ένα πολύ μικρό πιθανοτικό αρχείο του τι έχει κρατήσει. Έτσι, έχει περιθώριο λάθους, τα οποία όμως με κατάλληλες παραμετροποιήσεις μπορεί κάποιος να τα εκμηδενίσει.



Εικόνα 6: Παράδειγμα Bloom filter

Πιο συγκεκριμένα, το bloom filter αρχικά είναι ένας άδειος πίνακας από m bits (αρχικά όλα μηδέν). Επίσης υπάρχει και ένα σύνολο από k hash συναρτήσεις. Όταν θέλουμε να εισάγουμε ένα στοιχείο x σε αυτή την δομή, εκτελούμε και τις k συναρτήσεις, μια φορά με το στοιχείο x. Από κάθε αποτέλεσμα, φροντίζουμε να πάρουμε τους άσσους κάθε θέσης και να κάνουμε άσσους τις αντίστοιχες στον πίνακα με τα m bits. Για να δούμε εάν ένα στοιχείο x ανήκει στο bloom filter, κάνουμε την ίδια διαδικασία για να πάρουμε ένα καινούριο πίνακα m bits. Ελέγχουμε στην συνέχεια εάν αυτός ο πίνακας έχει άσσους εκεί που έχει ο πίνακας του bloom filter. Εάν ναι, τότε σημάνει ότι το έχουμε ήδη μέσα, ειδάλλως όχι. Το bloom filter θα είναι ένας πίνακας λοιπόν που θα διακινείτε μεταξύ των peers και θα περιέχει πληροφορία σχετικά με το ποιο διαφημιστικό έχει περάσει από ποιους χρήστες.

Χρήση περισσότερων κόμβων

Πέρα από τους φίλους που θα έχει κάθε peer, μπορούμε να προσθέσουμε στον καθένα κατά την αρχικοποίηση τους και μερικούς ακόμα, όπως γίνεται και στο [2]. Η επιλογή αυτή θα γίνετε μια φορά σε κάθε peer στην αρχή, επιλέγοντας με ομοιόμορφο τρόπο από όλο το σύνολο peers του πειράματος που τρέχουμε. Αυτό θα έχει ως αποτέλεσμα να αμβλύνει το γράφο με τέτοιο τρόπο, ώστε να βελτιωθεί το πλήθος το μηνυμάτων που θα διέρχονται από αυτούς.

Τοπικό balancing σε κάθε κόμβο

Κάθε κόμβος θα κρατάει πολλά R_{id} 's για που θα προέρχονται από πολλούς κόμβους. Προκειμένου να μην κρατάμε R_{id} 's που δεν χρησιμοποιούνται, θα έχουμε μια δομή που λέγεται ResourceAger. Σε αυτή την

δομή θα υπάρχει μια καταχώρηση για κάθε Rid. Όσο πιο πολλές gossip περίοδοι περνάνε που αυτό δεν χρησιμοποιείται (δηλαδή είναι unmatched) τόσο περισσότερο παλιώνει. Όταν κάποια καταχώρηση ξεπεράσει κάποιο όριο, τότε διαγράφετε από το ResourceAger και το ResourceMemory εκείνου του κόμβου. Πρόκειται για τοπική απόφαση η οποία επηρεάζει την απόφαση άλλων peers.

Επεκτασιμότητα

Για την βελτίωση της επεκτασιμότητας θα χρησιμοποιηθεί ένα gossip πρωτόκολλο σε ένα unstructured δίκτυο. Θα χρησιμοποιήσουμε UDP για μείωση του traffic, και του κόστους της δημιουργίας σύνδεσης. Η επιλογή των bloom filters είναι κάτι που επίσης βοηθά στην καλύτερη κλιμάκωση του συστήματος.

Δομές δεδομένων

Οι δομές που χρησιμοποιούμε είναι δύο: η ResourceAger και η ResourceMemory.

Η ResourceAger όπως προαναφέραμε παρακολουθεί πόσο αχρησιμοποίητα (δηλαδή unmatched) είναι τα Rid's που κρατάει στην δομή ResourceMemory. Όσο πιο πολλές gossip περίοδοι περνάνε που αυτό δεν χρησιμοποιείται (δηλαδή είναι unmatched) τόσο περισσότερο γίνεται παλιό. Όταν κάποια καταχώρηση ξεπεράσει κάποιο όριο, τότε διαγράφεται από το ResourceAger και το ResourceMemory εκείνου του κόμβου. Η υλοποίηση της είναι μια διπλά συνδεδεμένη λίστα.

Η δομή ResourceMemory κρατάει για κάθε Rid, ποια μηνύματα sub ή pub υπάρχουν και από ποιους χρήστες. Η υλοποίηση της είναι ένας hash πίνακας.

Πεδία μηνύματος

Το πρωτόκολλο μας θα ανταλλάσσει ένα διαφημιστικό μήνυμα (advertising message). Βάσει όλων αυτών που προαναφέραμε, το μήνυμα πρέπει να περιέχει ένα πίνακα με τα bytes του bloom filter, έναν ακέραιο αριθμό για το μέγεθος του, την διεύθυνση IP που παρήγαγε το μήνυμα, καθώς και το είδος του μηνύματος (Pub ή Sub). Στον πίνακα αναφέρονται όλα τα πεδία του μηνύματος και η σημασία τους.

Origin Address	Originating peer's address
Message Type	Publication or Subscription
Bloom bit array	The character array of the filter
Bit array size	The size of the array (in bytes)
Resource ID	The topic of the message
Object's name	The advertised objects name

Πίνακας 1: Περιεχόμενα μηνύματος advertising

Ψευδοκώδικας

Παρακάτω ακολουθεί ο ψευδοκώδικας του αλγόριθμου. Όπως και οι περισσότεροι gossip αλγόριθμοι βασίζονται σε δύο κύριες συναρτήσεις. Η πρώτη περιοδικά επιλέγει κάτι (GossipTell) και το στέλνει τυχαία σε κάποιους χρήστες.

```
GossipTell()
{
  while(true)
  {
    sleep(gossip interval)
    ResourceAger.increaseAll()
    ResourceAger.discharOldEntries()

    msg := ResourceMemory.selectRandom()
  // favor the unmatched pub/subs

  i = 0
  while(i<maxNodesToSent)
  {
    if(msg.BloomFilter.overloaded()==true)
    msg.BloomFilter.clear();

    BF := new BloomFilter(msg.peopleAlreadySent + msg.Rid +
    msg.origin-peer-name + msg.Name)

   selected_node := selectUserRandomly(friends-list ∪ random-nodes)

   if(BF.check() ==true)
       selectUserRandomly(friends-list ∪ random-nodes) repeat while
   else
   {
    i = i + 1
    // The gossip message is created here
    make package := msg.Rid | users-name | msg.origin-peer-name | msg.Name | BF
   udp_send(selected_node, package)
   }
  }
}
```

Η GossipHear είναι event triggered συνάρτηση. Ενεργοποιείται όταν λαμβάνει ένα πακέτο από το κατώτερο layer (UDP εδώ). Όταν λάβει ένα μήνυμα ελέγχει εάν το έχει ξαναδεί. Έπειτα κοιτάμε στο ResourceMemory εάν χρειάζεται να το στείλουμε σε άλλους λόγω κάποιου matching στο Rid (οπότε και τους ενημερώνουμε), και στην συνέχεια το αποθηκεύουμε.

```
GossipHear()
{
  ResourceMemory.has(msg.Rid , !msg.type))

  unpack package := msg.Rid | users-name | msg.origin-peer-name | msg.Name | BF

  if user-name is blacklisted OR already seen any Name in msg
   discard message and exit function

  if(ResourceMemory.has(msg.Rid , !msg.type))
  {
    intParties := ResourceMemory.getInterestedParties(msg.Rid , !msg.type))

    for each party in intParties
            inform each party and keep statistics
  }
```

```
  else
        ResourceAger.insert(msg.Rid , 0)

 // if already inside ResourceMemory, will not
// be re-inserted but Bloom Filters might changed accordingly

ResourceMemory.insert(msg.Rid , msg.origin-peer-name, BF)

}
```

Υλοποίηση

Για την υλοποίηση του συστήματος, αποφασίσαμε να χρησιμοποιήσουμε ένα πρόγραμμα προσομοίωσης, προκειμένου να αναπτύξουμε και να δοκιμάσουμε το αλγόριθμο μας με μεγαλύτερη ευκολία σε διάφορες συνθήκες. Αν και υπάρχουν πολλοί προσομοιωτές δικτύων από πολλά πανεπιστημιακά ιδρύματα σε όλο το κόσμο, εμάς μας ενδιαφέρει κυρίως η ανάπτυξη gossip αλγόριθμων για p2p συστήματα. Η χρήση ενός framework όπως το Oversim είναι η ενδεικτικότερη μιας και είναι από τους ποιο γρήγορα, σύγχρονα και ευκολότερα στην προγραμματισμό.

Ο προσομοιωτής Oversim στηρίζετε σε δύο άλλα προγράμματα, τον Omnet++ και το INET Framework. Ο πρώτος είναι και ο βασικός προσομοιωτής πάνω στο οποίο στηρίζονται τα υπόλοιπα. Αποτελεί μια μηχανή πεπερασμένων καταστάσεων που εκτελεί ένα απλό αλγόριθμο εκτελέσεις γεγονότων. Επιτρέπει την δημιουργία οντοτήτων ενός δικτύου (που παράγουν διάφορα γεγονότα) με χρήση της γλώσσας. Για την ευκολία ανάπτυξης προγραμμάτων, οι δημιουργοί του παρέχουν και ένα IDE το οποίο θυμίζει αρκετά το περιβάλλον του Eclipse.

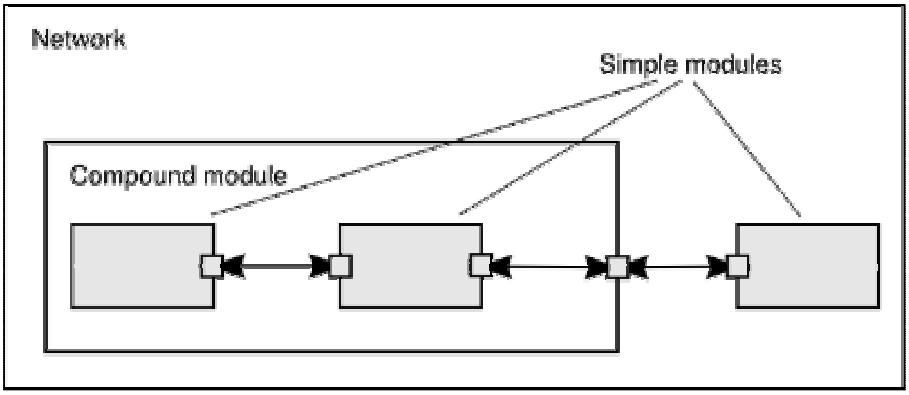
Omnet++

Το Omnet++ είναι ένας προσομοιωτής δικτύων διακριτών γεγονότων. Έχει γραφτεί σε C++, και η κύρια γλώσσα περιγραφής της λειτουργίας των δικτύων είναι επίσης η ίδια. Δημιουργήθηκε το 1992, αλλά παρέμεινε ανενεργό για πολλά χρόνια μέχρι και το 1999, οπότε και άρχισε να παίρνει την μορφή που έχει σήμερα. Διατίθεται δωρεάν για ακαδημαϊκή χρήση, και πλέον έχει αρχίσει να υποστηρίζει και άλλες γλώσσες προγραμματισμού.

Τα διακριτά γεγονότα που προαναφέραμε, είναι κυρίως τα μηνύματα που ανταλλάσσουν τα components ενός δικτύου προσομοίωσης. Το Omnet++ παρέχει μια γλώσσα περιγραφής της τοπολογίας των δικτύων που ονομάζετε NED (Network Description). Αποτελεί μια πολύπλοκη και εκφραστική γλώσσα, της οποίας θα χρησιμοποιήσουμε μόνο τα απαραίτητα σε εμάς στοιχεία της.

Το βασικό framework που μας παρέχει το Omnet++, είναι αυτό που φαίνεται στην εικόνα. Το δίκτυο μας, αποτελείται από modules. Ο βασικός λίθος είναι το simple module. Πολλά simple modules μαζί μπορούν να αποτελούν ένα σύνθετο (compound) module.

Η λειτουργία κάθε module περιγράφεται σε C++. Κάθε τέτοιο αρχείο cpp, πρέπει να ενσωματώνει απαραίτητα κάποια headers, ώστε να έχει πρόσβαση σε κλάσεις της μηχανής προσομοίωσης. Μια τέτοια βασική κλάση είναι η cSimpleModule. Η κλάση που θέλουμε να υλοποιεί τα components ενός δικτύου πρέπει να κληρονομεί από αυτήν και να υλοποιεί δύο βασικές μεθόδους, την handleMessage(). Τα μηνύματα που θα ανταλλάζει το δίκτυο μας, θα είναι πάντοτε κληρονόμοι της κλάσης cMessage.



Εικόνα 7: Σχεδιάγραμμα αρχιτεκτονικής στο OMNET++

Ο προσομοιωτής απαιτεί το interface κάθε module (και όχι μόνο το δίκτυο) να περιγραφεί με την γλώσσα NED. Σε αυτή την περίπτωση η γλώσσα NED, δεν περιγράφει συνδέσεις μεταξύ modules, αλλά ποιες παραμέτρους μπορούν να δέχονται τα modules. Τα instances του δικτύου και το modules, όταν θέλουμε να εκτελέσουμε κάποιο πείραμα, γίνεται με χρήση ενός ini αρχείου, που δηλώνει ποια Modules θα πάρουν ποιες παραμέτρους.

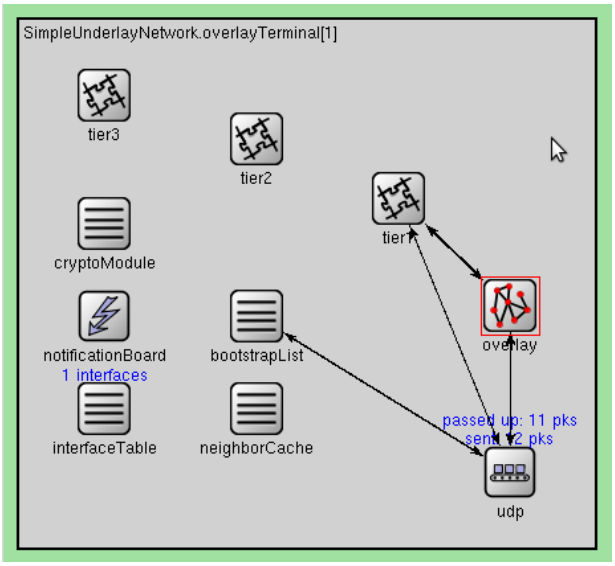
INET Framework

Ο προσομοιωτής του Omnet++, μας παρέχει μόνο την μηχανή εκτέλεσης και την γλώσσα περιγραφής NED. Δεν υποστηρίζετε κανένα πρωτόκολλο επικοινωνίας. Εάν θέλουμε κάτι, πρέπει να το υλοποιούμε μόνοι μας. Παρόλα αυτά, τα τελευταία χρόνια δημιουργήθηκε ένα άλλο framework, που χρησιμοποιεί την μηχανή προσομοίωσης και υλοποιεί μερικά βασικά πρωτόκολλα όπως το TCP, το IP και το UDP.

Oversim

Εξαιτίας το αυξανόμενου ενδιαφέροντος για πειράματα που αφορούν τα δίκτυα p2p, δημιουργήθηκαν πολλά frameworks και προσομοιωτές ακριβώς για αυτήν την θεματολογία. Το Oversim είναι ένα τέτοιο παράδειγμα framework που στηρίζετε πάνω στο INET και τον Omnet++. Παρέχει μερικά modules (που είναι επεκτάσεις του cSimpleModule) σε tier αρχιτεκτονική ώστε να μπορεί κάποιος εύκολα να δημιουργεί προσομοιώσεις εφαρμογών και δικτύων p2p.

Στον Oversim, έχουμε τρία βασικά programming layers, τα tier 1 έως 3. Το tier ένα αφορά την διασύνδεση της p2p εφαρμογής μας με το είδος του πρωτοκόλλου που επιθυμούμε (π.χ. το UDP). Το δεύτερο tier, αποτελεί την υλοποίηση του είδους του overlay πρωτοκόλλου που επιθυμούμε. Το ίδιο το Oversim μας παρέχει πολλά υλοποιημένα πρωτόκολλα όπως το Chord, το Kademlia κ.τ.λ.. Το τελευταίο tier αποτελεί το κομμάτι όπου υλοποιούμε την εφαρμογή που θέλουμε να τρέξουμε πάνω από το συγκεκριμένο overlay πρωτόκολλο του tier 2.



Εικόνα 8: Oversim Framework

Η χρήση των ini αρχείων μας επιτρέπει να τρέχουμε ότι συνδυασμό από τα τρία tiers θέλουμε, δηλαδή, μπορούμε να έχουμε μια εφαρμογή δική μας, και να αλλάζουμε τα overlay δίκτυα για να τρέχουμε διαφορετικά πειράματα.

Πειράματα

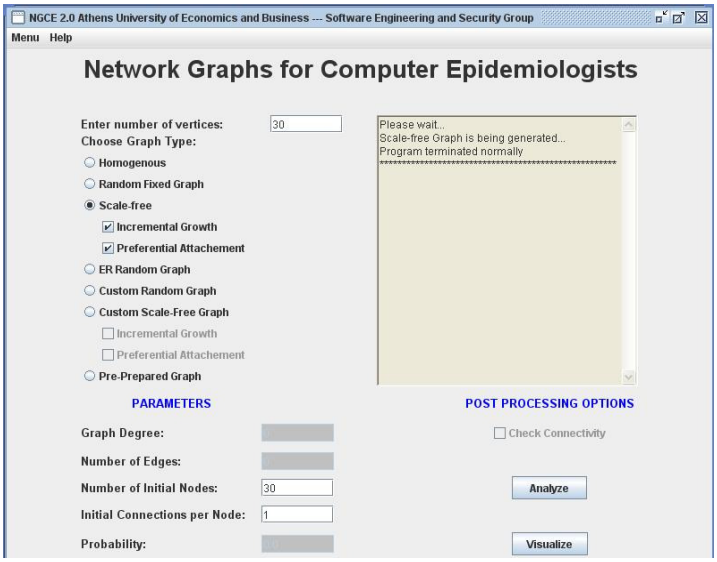
Αυτόματη παραγωγή γράφων

Αφού εντοπίσαμε ποιοι γράφοι αποτελούν τα καλύτερα παραδείγματα για διεξαγωγή πειραμάτων, έμεινε στην συνέχεια να εντοπίσουμε ένα εργαλείο που να επιτρέπει την αυτόματη, τυχαιοκρατική δημιουργία τέτοιων γραφημάτων. Στην ακαδημαϊκή κοινότητα έχουν δημιουργηθεί τα τελευταία χρόνια πολλά τέτοια εργαλεία για να παράγουν τις παραπάνω (και πολλές άλλες) οικογένειες δικτύων (γράφων).

Ένα από τα πρώτα που αναπτύχθηκαν είναι το Stanford GraphBase από τον Donald Knuth. Το συγκεκριμένο εργαλείο όμως δεν ειδικεύεται τόσο στην παραγωγή όσο στην ανάλυση γράφων, και απαιτεί πολύ εκτενή παραμετροποίηση και κατανόηση σε βάθος του συστήματος. Επίσης, δεν λαμβάνει υπόψιν τους γράφους για τους οποίους ενδιαφερόμαστε εμείς.

Το πιο πολυχρησιμοποιημένο εργαλείο από προγραμματιστές και μελετητές επικοινωνιών είναι το Inet. Δημιουργεί γράφους που εμπειρικά έχουν παρατηρηθεί σε τοπολογίες τύπου Internet Service Providers. Παρόλα αυτά, δεν ασχολείται με τύπους γράφων που ενδιαφερόμαστε εμείς και βασίζετε κυρίως σε εμπειρικά και όχι μαθηματικά μοντέλα.

Στην παρούσα εργασία χρησιμοποιήσαμε το εργαλείο NGCE (Network Graphs for Computer Epidemiologists) [29] που κατασκεύασε η ομάδα του ISTLab στο Τμήμα Διοικητικής Επιστήμης του Ο.Π.Α.. Το πρόγραμμα είναι γραμμένο σε Java και με πολύ βασική παραμετροποίηση παράγει σε ASCII αρχείο τα δεδομένα που αφορούν την συνδεσιμότητα σε γράφο.



Εικόνα 9: Το πρόγραμμα παραγωγής γράφων NGCE

Το βασικό εκτελέσιμο του προγράμματος, εμφανίζει ένα παράθυρο με τα βασικά πεδία για την παραμετροποίηση του γράφου που θέλουμε να δημιουργήσουμε. Αρχικά επιλέγουμε τι τύπου γράφο χρειαζόμαστε. Οι βασικές επιλογές που δίνει αυτή η έκδοση του προγράμματος είναι Homogenous, Random fixed Graph, Scale-Free, ER-Random Graph, Pre-Prepared Graph και customizable εκδοχές των παραπάνω. Στην συνέχεια επιλέγουμε το αριθμό των κόμβων που θέλουμε, καθώς και το πόσους θέλουμε να είναι στενά συνδεδεμένοι (ο ίδιος αριθμός εάν θέλουμε να είναι όλοι). Μετά εισάγουμε το πόσα αρχικά connections μπορούμε να έχουμε ανά κόμβο και μια αρχική τιμή για τον seeder του random generator. Η επιλογή «Generate» μας δημιουργεί το γράφο με τις παραμέτρους που επιλέξαμε. Το πρόγραμμα εμφανίζει στο standard output τις πράξεις και τα αποτελέσματα τους, και εγγράφει σε ένα αρχείο ποιοι κόμβοι συνδέονται μεταξύ τους. Εάν έχουμε συνδέσει την εφαρμογή και με κάποιο πρόγραμμα απεικόνισης (π.χ. το Pajek), μπορούμε να επιλέξουμε «Analyze» και να μας παρουσιάσει τον γράφο σε δισδιάστατοι αναπαράσταση.

Node 0 has 1 Edges *	#Nodes:= 30
Node 1 has 1 Edges *	#Initial:= 30
Node 2 has 3 Edges ***	#m:= 1
Node 3 has 1 Edges *	#Class:= FullScaleFreeGraph
Node 4 has 1 Edges *	#Seed:= 2
Node 5 has 4 Edges ****	#Version:= 2.0
Node 6 has 3 Edges ***	0 24
Node 7 has 1 Edges *	1 6
Node 8 has 1 Edges *	2 14
Node 9 has 1 Edges *	2 22
Node 10 has 2 Edges **	2 26
Node 11 has 1 Edges *	3 13
Node 12 has 2 Edges **	4 26
Node 13 has 5 Edges *****	5 11
Node 14 has 1 Edges *	5 21
Node 15 has 3 Edges ***	5 28
....	....

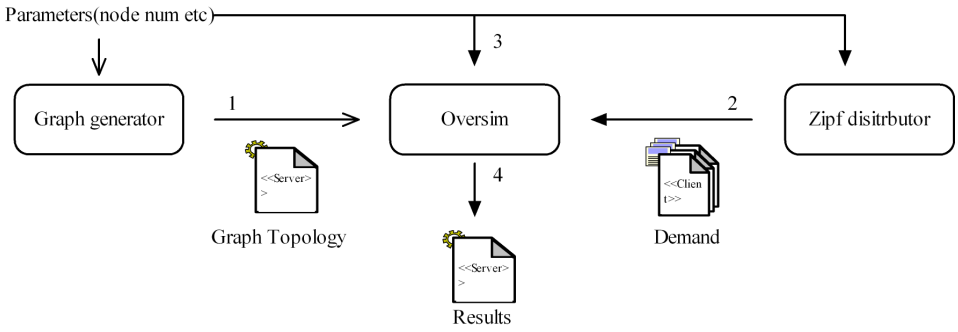
Το πρόγραμμα εμφανίζει τα αποτελέσματα των συνδέσεων στο standard output της γραμμής εντολών όπως φαίνεται στην εικόνα αριστερά. Κάθε

γραμμή μας λέει ότι ο κόμβος x έχει τόσες y συνδέσεις. Παράλληλα, στο αρχείο GraphTopology.txt κάθε γραμμή μας αναφέρει ποιος κόμβος x (με αριθμό) συνδέεται με ποιον (με αριθμό πάλι). Παρατηρούμε ότι για τον κόμβο με αριθμό 2, που έχει 3 συνδέσεις, στο αρχείο GraphTopology.txt έχουμε τρεις γραμμές με το δύο, που μας περιγράφουν ποιοι είναι αυτοί με τους οποίους συνδέεται. Αυτό το αρχείο χρησιμοποιείται σε ένα πρόγραμμα που κατασκευάσαμε και αναθέτει σε κάθε peer τις διευθύνσεις των άλλων peers που είναι φίλοι του, πριν ξεκινήσουμε πειράματα με τον Oversim.

Διεξαγωγή πειραμάτων

Για κάθε πείραμα που θα κάνουμε, θα καθορίζουμε ποιες τιμές θέλουμε να έχουμε στις παραμέτρους. Οι παράμετροι που μας ενδιαφέρουν είναι: το πλήθος των χρηστών, το πλήθος των αρχικών pub και sub μηνυμάτων και τις σταθερές στο κατανομή Zipf. Θεωρούμε την πιθανότητα απώλειας πακέτου ομοιόμορφη, κοντά στο 5% σε όλα τα πειράματα μας.

Αφού αποφασίσουμε για αυτές τις τιμές, θα τις εισάγουμε στο πρόγραμμα αυτόματης παραγωγής γράφων και στο πρόγραμμα που παράγει τα αρχεία για τα subscriptions και τα publication κάθε χρήστη. Ο παραγωγός γράφων θα παράγει το κοινωνικό δίκτυο που θέλουμε στο αρχείο GraphTopology.txt . Το πρόγραμμα παραγωγής των αρχείων θα παράγει από ένα αρχείο ASCII με όνομα sX.txt, με τα topics που θα κάνει subscribe ο peer X και ένα pX.txt εάν παράγει και κάποια publications (και στα δύο αρχεία πάντα βάση τις Zipf κατανομής). Να σημειωθεί ότι θεωρούμε όλους τους χρήστες subscribers και ένα ποσοστό (30%) από αυτούς και publishers.



Εικόνα 10: Σχεδιάγραμμα διεξαγωγής των πειραμάτων

Στην συνέχεια, αυτά τα αρχεία τα τροφοδοτούμε στην πρόγραμμα Gossip που φτιάξαμε στον Oversim και ξεκινάμε την εκτέλεση του. Τα αποτελέσματα του κάθε peer σώζονται σε αρχεία. Αυτά το αρχεία χρησιμοποιούμε για επεξεργασία των αποτελεσμάτων μας. Η εικόνα δείχνει εποπτικά την σειρά διεξαγωγής των πειραμάτων.

Πειράματα

Ως αποτελέσματα αποφασίσαμε να δείξουμε δύο βασικές μετρικές:

Η μετρική πρώτη αφορά το πότε ένας peer ολοκληρώνει τα subscriptions του. Θα θεωρήσουμε ότι ένας peer γίνετε πλήρης όταν έχει δεχτεί ένα τουλάχιστον publication για κάθε subscription που έχει. Αυτό θα μας δείχνει

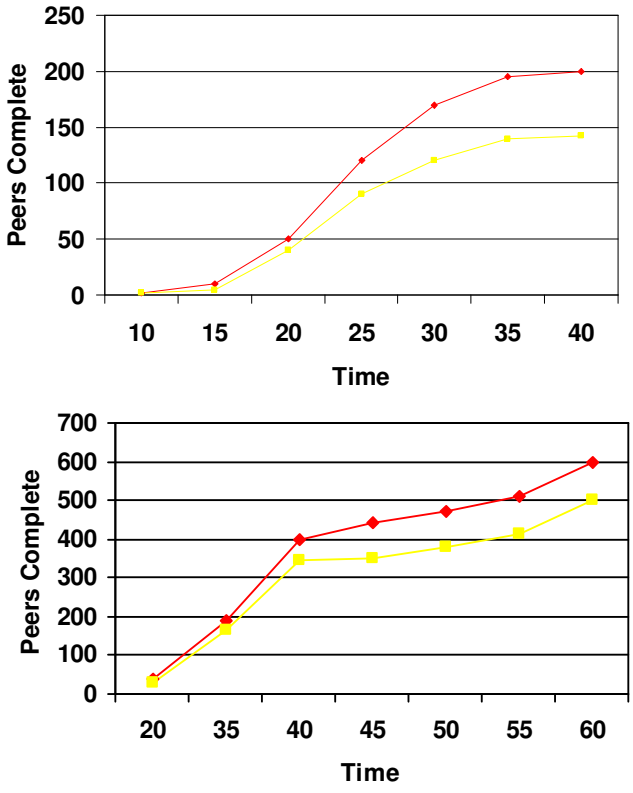
ποσο γρήγορα κινούνται τα «νέα» μέσα στο δίκτυο και πόσο γρήγορα θα ενημερωθούν όλοι για όλα όσα τους ενδιαφέρουν. Για να συγκρίνουμε την πρώτη μετρική με κάποιο άλλο αποτέλεσμα, αποφασίσαμε να εκτελέσουμε σε κάθε πείραμα, και ένα παραπλήσιο αλγόριθμο. Αυτός είναι ένας βασικός gossip αλγόριθμος όπου στέλνονται μόνο τα pub μηνύματα. Αυτά θα διακινόντουσαν στο δίκτυο μέχρι να φτάσουν κάποιους subscribers που θα ενδιαφέρονται. (Δηλ. κανένας ενδιάμεσος peer δεν βοηθάει κάποιον άλλο).

Επίσης, θα μετρήσουμε να δούμε ποια κατηγορία ενδιάμεσων χρηστών (ως προς πλήθος φίλων) κάνει τελικά τις περισσότερες αντιστοιχήσεις.

Εκτελέσαμε διάφορα πειράματα για μεγέθη γράφου με 200 και 600 χρήστες. Κάθε φορά εκτελούσαμε ένα πείραμα, που αφορούσε το αλγόριθμο μας, εκτελούσαμε τον δεύτερο αλγόριθμο με τις ίδιες παραμέτρους.

Μετρική 1

Ο κάθετος άξονας δείχνει το πλήθος των peer που ολοκληρώνει ενώ ο οριζόντιος το simulation time. Παρατηρώντας τις κόκκινες γραμμές και στα δύο διαγράμματα, βλέπουμε ότι το σύστημα μπορεί και ολοκληρώνει αρκετά γρήγορα. Η κόκκινη γραμμή δείχνει τα αποτελέσματα του δικού μας αλγόριθμου, ενώ η κίτρινη του δεύτερου απλού gossip αλγόριθμου. Στην περίπτωση μάλιστα του πειράματος με τους 600 χρήστες, βλέπουμε ότι αυξάνει και ποιο γρήγορα – αυτό ερμηνεύεται εύκολα αν αναλογιστούμε ότι έχουμε περισσότερους χρήστες.

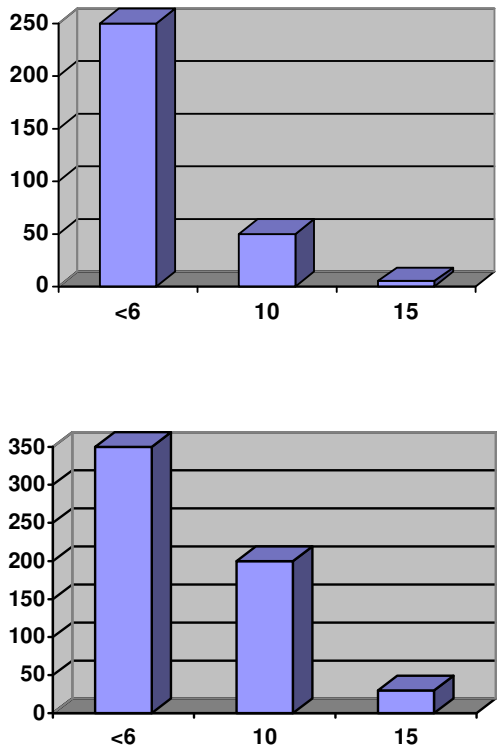


Εικόνα 11: Αποτελέσματα πρώτης μετρικής για 200 και 600 peers

Ενδιαφέρουν είναι ότι συγκρίνοντας και στα δύο πειράματα την κόκκινη με την κίτρινη γραμμή, βλέπουμε ότι εμφανίζουμε συνολικά κάπως καλύτερα αποτελέσματα. Αυτό είναι αναμενόμενο, αφού στον αλγόριθμό μας όλοι οι ενδιάμεσοι χρήστες αποθηκεύουν προσωρινά τα μηνύματα που λαμβάνουν (pub και sub) και βοηθάνε στην αντιστοίχηση, και δεν χρειάζεται ο κάθε peer να περιμένει να τον φτάσει κάτι που μπορεί να τον ενδιαφέρει.

Μετρική 2

Εδώ θα έχουμε τρεις κατηγορίες χρηστών ως προς το πλήθος των φίλων που έχουν: η πρώτη κατηγορία είναι αυτή με τους χρήστες που έχουν έως 6 φίλους. Η δεύτερη είναι αυτή με χρήστες που έχουν από 7-10 φίλους και η τελευταία από 11-15 (οι hubs του κοινωνικού δικτύου, οι οποίοι είναι και λίγοι όπως προαναφέραμε). Τα διαγράμματα δείχνουν πόσες αντιστοιχήσεις έκανε ο μέσος χρήστης κάθε κατηγορίας. Παρατηρούμε ότι οι χρήστες που κάνουν τελικά τις περισσότερες αντιστοιχήσεις είναι αυτοί με το μικρό πλήθος χρηστών και στα δύο πειράματα. Αυτό είναι προφανές ότι συμβαίνει διότι το μεγαλύτερο ποσοστό των χρηστών έχει ένα σχετικά μικρό αριθμό φίλων. Ένα ενδιαφέρον συμπέρασμα προκύπτει όταν αντιπαραβάλαμε τους δύο πίνακες μαζί: βλέπουμε ότι όταν μεγάλωσε το μέγεθος του δικτύου, το ποσοστό αντιστοιχήσεων στην κατηγορίας χρηστών με ενδιάμεσο πλήθος φίλων αυξάνει – αντίστοιχα μειώθηκε αυτών που έχουν πολλούς φίλους. Αυτό είναι σημαντικό διότι αποδεικνύουμε ότι δεν επιβαρύνουμε τους χρήστες με πολλές αντιστοιχήσεις αλλά ότι επιφορτίζονται αρκετά και οι άλλες κατηγορίες. Τα αποτελέσματα είναι κανονικοποιημένα ως προς το μέσο πλήθος χρηστών κάθε ομάδας.

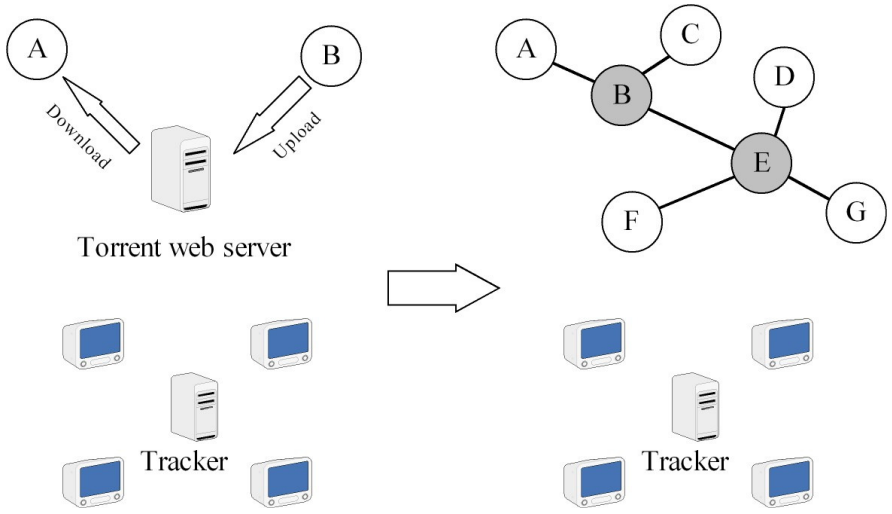


Εικόνα 12: Αποτελέσματα δεύτερης μετρικής για 200 και 600 peers

Σενάρια χρήσης

Το πρωτόκολλο ανταλλαγής αρχείων BitTorrent δημιουργήθηκε το 2001 από τον Bram Cohen [21]. Υπάρχουν χρήστες οι οποίοι διαθέτουν ένα αρχείο ολόκληρο (seeders) και τους ενδιαφέρει να το διαθέσουν σε άλλους που δεν θα το έχουν (leechers). Το ποιο αρχείο είναι αυτό και από ποιο κομμάτια αποτελείτε, περιγράφετε σε ένα μετα-αρχείο που ονομάζετε torrent file. Το αρχείο αυτό αναφέρει ένα αρχικό κόμβο, τον tracker (μέσω του οποίου οι διάφοροι seeders και lechers βρίσκουν ο ένας τον άλλον) και μια περιγραφή των segments του αρχείου που θα ανταλλαχθεί (hash code του segment, μέγεθος του etc). Ένας tracker μπορεί να ασχολείται με περισσότερα του ενός metafiles. Με χρήση του αλγόριθμου tit-for-tat, η μετάδοση των κομματιών γίνετε με τον βέλτιστο τρόπο, έτσι ώστε κάποιος να κάνει download και από lechers και από seeders, αποσυμφορίζοντας έτσι τους τελευταίους. Όταν ένας leecher ολοκληρώσει το downloading ενός αρχείου, τότε γίνεται και αυτός με την σειρά του ένας seeder.

Ένα βασικό πρόβλημα του πρωτοκόλλου είναι αυτή η διαχείριση των torrent αρχείων. Έως σήμερα, αυτά εναποτίθενται κεντρικά σε web εξυπηρετητής που παρέχουν μηχανές αναζήτησης προς τους χρήστες. Κάθε torrent που αφορά ένα αρχείο εντάσσεται και σε μια κατηγορία π.χ. μια κωμική ταινία με τίτλο «Α» θα εντάσσεται πιθανόν στην κατηγορία «Comedy». Έτσι οι μηχανές αναζήτησης εντάσσουν και κάθε metafile κάτω από κατηγορίες που τους έβαλαν χρήστες για να επιτρέπουν αναζήτηση βάση θεματολογίας και όχι μόνο filename.



Εικόνα 13: Παράδειγμα χρήσης στο πρωτόκολλο BitTorrent

Αυτή ακριβώς την παρατήρηση εκμεταλλευόμαστε και εμείς. Συγκεκριμένα, κάθε αρχικός seeder θα κάνει publish τα στοιχεία του torrent αρχείου (μαζί με μια κατηγορία που θεωρεί ότι ανήκει). Οι χρήστες που ενδιαφέρονται για αυτή την κατηγορία, θα στέλνουν τα subscribe μηνύματα και όταν κάποιος κάνει το match θα τους ενημερώνει. Τα μηνύματα που θα ανταλλάσσουν δηλαδή οι peers, θα περιέχουν τα στοιχεία μεταδεδομένων που αφορούν τα torrent αρχεία.

Μελλοντική εργασία

Ο χώρος των gossip πρωτοκόλλων σε Publish/Subscribe p2p συστήματα είναι ακόμα πολύ καινούριος. Γνώμη του συγγραφέα, είναι ότι έχουν πολλά ακόμα να δώσουν σε διάφορους τομείς οι οποίοι πρέπει να εξεταστούν. Μερικοί από αυτούς είναι:

- Θέματα trust: Πότε πρέπει να εμπιστευόμαστε κάποιον ότι το αρχείο είναι αυτό που ισχυρίζεται; Είναι απαραίτητη η δημιουργία ενός reputation μηχανισμού για τους χρήστες του συστήματος.
- Σύγκριση με άλλους αλγόριθμους: Πρέπει να γίνει εκτενέστερη μελέτη της βιβλιογραφίας με σχετικούς αλγόριθμους να γίνει μια πληρέστερη και ποιοτική σύγκριση.

Βιβλιογραφία

[1]. "Publish/Subscribe in a mobile environment", Hector Garcia-Molina and Huang Yongqiang, Wireless Networks, Springer 2004

[2]. "TRIBLER: A social-based peer-to-peer system", Pouwelse et all, Concurrency and Computation, CiteSeer 2008

[3]. "PSIRP Architecture Deliverable D2.3", Mark Ain et all, http://www.psirp.com/pubs, 2009

[4]. "HERMES: A distributed Event based Middleware Architecture", Bacon et. al, IEEE Journal on Selected Areas in communications, 2002

[5]. "Epidemic-style management of semantic overlays for content based searching", Spyros Voulgaris et all, Computer Science Lectures, Springer, 2005

[6]. "Corona: A high performance Publish-Subscribe System for the WWW", Ramasubramanian et. all, Proceedings of networked design, 2006

[7]. "Measurement and analysis of online social networks", Alan Mislove et all., ACM SIGCOMM Conference on Internet Measurement, ACM 2007

[8]. "Personalization on a peer-to-peer television system", Pouwelse et all, Multimedia Tools Applications, Springer 2007

[9]. "Semi-probabilistic content-based publish-subscribe", P. Costa and P. Picco, 25th IEEE International Conference on Distributed Systems, 2005

[10]. "Six degrees: The science of a connected age", D. Watts, WW Norton and Company, 2004

[11]. "Mirinae: A peer-to-peer overlay network for content-based Publish/Subscribe systems", Choi et all, IEICE TRANS. COMMUNUNICATIONS, VOL. E89–B NO.6, JUNE 2006

[12]. "Meghdoot: Content-based publish/subscribe over P2P networks", Gupta et all, Lecture notes in computer science, Springer, 2004

[13]. "Network applications of Bloom Filters: A survey", Broder at al., Internet Mathematics, 2004

[14]. "The art of computer programming", Volume 3, Donald Knuth, Addison-Wesley, 1979

[15]. "Chord: a scalable peer-to-peer lookup protocol for Internet applications", Stoica I., Liben-Nowell, Karger, D. R. Kaashoek, M. F. Dabek, F. and Balakrishnan, IEEE JNET, 2003

[16]. "OverSim: A Flexible Overlay Network Simulation Framework", Baumgart, I., Heep, B. and Krause, Proc. IEEE Global Internet Symposium, p.79-84, 2007

[17]. "Graph Mining", Faloutsos et. all., ACM Computing Surveys, 2006

[18]."Zipf's law and the Internet", LA Adamic and BA Huberman, Glottometrics, Citeseer, 2002

[19]. "The Many Faces of Publish/Subscribe", Eugster et. all., ACM Computing, 2003

[20]. StumbleUpon, http://www.stumbleupon.com

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

[22]. OMNET++ documentation, http://www.omnetpp.org/documentation

[23]. "Epidemic algorithms for database maintenance", Demers et. all., Proceedings of the sixth annual ACM Symposium, 1987

[24]. "Epidemic algorithms", Tim Hollerung and Peter Bleckmann, University of Paderborn, 2004

[25]. "A Gossip-based Distributed News Service for Wireless Mesh Networks", Spyros Voulgaris et. all., Computer Science Lectures, 2006

[26]. "Random Graphs", B.Bollobas. Cambridge University Press, Cambridge, 2001

[27]. "Randomized Gossip Algorithms", S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, IEEE Transactions of Information Theory, June 2006

[28]. "Exploiting virtual synchronization in distributed systems", K Birman, T Joseph - ACM SIGOPS Operating Systems Review, 1987

[29]. "NGCE-network graphs for computer epidemiologists", V.Vlachos, V.Vouzi, D.Chatziantoniou, D.Spinellis, Advances in Informatics, Springer, 2003