S.N.A.P. - SOCIAL NETWORK ANALYSIS PROJECT

Ορέστης Μελκονιάν – 1115201000128

Αθανάσιος Σούλης - 1115200900155

Βασική Λειτουργία

Το σύστημα που έχουμε υλοποιήσει διαβάζει τα στοιχεία ενός κοινωνικού δικτύου από ένα dataset (.csv αρχεία) και, αφού αναπαραστήσει αποδοτικά τους γράφους που το αποτελούν, μπορεί να απαντήσει σε διάφορα ερωτήματα που αφορούν τη δομή του ή συγκεκριμένες ιδιότητες, χρησιμοποιώντας διάφορες γνωστές μεθοδολογίες και γενικές μετρικές γραφημάτων.

Τεχνολογικές Επιλογές

Το σύστημα έχει υλοποιηθεί σε C++ (Eclipse IDE) χωρίς καμία εξωτερική βιβλιοθήκη ή έτοιμη δομή δεδομένων, ακολουθώντας μία αντικειμενοστραφής προσέγγιση. Χρησιμοποιήθηκε επίσης το Subversion(hosted στο Assembla) ως version control system και το Asana για πιο αποδοτική επικοινωνία και γενικότερη οργάνωση.

Δομή Κώδικα

- Στον αρχικό φάκελο βρίσκονται οι πιο βασικές κλάσεις, όπως Κόμβος, Ακμή, Γράφος, Πίνακας Κατακερματισμού, κλπ.
- Στο φάκελο /***list***/ βρίσκεται όποια κλάση αφορά υλοποίηση λιστών ή ουράς και είναι γενικού σκοπού, όπως οι διαφορετικοί τύποι που μπορεί να έχει ως στοιχεία η λίστα.
- Στο φάκελο /***properties***/ βρίσκεται όποια κλάση αφορά τις ιδιότητες μιας ακμής ή ενός κόμβου.
- Στο φάκελο /***search***/ βρίσκεται όποια κλάση αφορά αναζήτηση σε γράφο.
- Στο φάκελο /***graph_metrics***/ βρίσκεται η υλοποίηση όλων των μετρικών που θα χρειαστούν για τα ερωτήματα και στο */**gnuplot**/* το πρόγραμμα gnuplot που χρησιμοποιείται για να αναπαραστηθεί η μετρική «κατανομή βαθμού(degree distribution)».
- Στο φάκελο /***queries***/ βρίσκεται η υλοποίηση όλων των ερωτημάτων που μας έχουν ζητηθεί και στο /***dataset***/ τα στοιχεία του κοινωνικού δικτύου.
- Στο φάκελο /***tests***/ περιέχονται όλα τα προγράμματα που γράψαμε για να τεστάρουμε επιμέρους λειτουργίες (unit testing).
- Στο φάκελο /***project_main***/ βρίσκεται η προσαρμογή του συστήματος μας στα αρχεία που μας έχουν δοθεί από εσάς στο e-class.
- Στο φάκελο /***communities***/ έχουμε οτιδήποτε αφορά το Επίπεδο 3, όπως το preprocessing στάδιο και οι αλγόριθμοι εύρεσης κοινοτήτων.

Τεχνικές Λεπτομέρειες Υλοποίησης

- Η αναπαράσταση του γράφου έχει γίνει ως εξής:
  - Υπάρχει ένα αρχικό hash table(***LinearHashTable***) με βάση το ID της ακμής(***Edge***) ή του κόμβου(***Node***) που θα εισαχθεί. Χρησιμοποίηθηκε γραμμικός κατακερματισμός έναντι επεκτατού για να ελαχιστοποιηθεί η περιττή δέσμευση μνήμης.
  - Σε κάθε κάδο του hash table υπάρχει μία λίστα από buckets(***BucketList***) λόγω της περίπτωσης υπερχείλισης.
  - Κάθε Bucket περιέχει έναν συγκεκριμένο αριθμό από στοιχεία bucket(***BucketItem***).
  - Κάθε BucketItem περιέχει έναν κόμβο και μια λίστα που είναι οι εξερχόμενες ακμές αυτού του κόμβου.
- Για την αναζήτηση στο γράφο χρησιμοποιείται bidirectional breadth-first search(***BBFS***) όταν ψάχνουμε απόσταση μεταξύ 2 κόμβων ή απλή BFS αν ψάχνουμε την απόσταση από έναν κόμβο προς όλους τους άλλους μέσω της δομής ResultSet.
- Η υλοποίηση των μετρικών(***metrics***) είναι προφανής με απλή εφαρμογή των παραπάνω δομών και μεθόδων, εκτός από την περίπτωση του Betweenness Centrality που χρειάστηκε μία επέκταση της BFS(***ExtendedBFS***) έτσι ώστε να κρατάει το μονοπάτι και να υπολογίζει όλα τα συντομότερα μονοπάτια σε περίπτωση που υπάρχουν περισσότερα από ένα.

  Επίσης, για την υλοποίηση του αλγορίθμου εύρεσης κοινοτήτων των Girvan-Newman, χρειάστηκε μία νέα μετρική για τον υπολογισμό της κεντρικότητας μιας ακμής.

- Σε όλα τα ερωτήματα χρησιμοποιήθηκε ο γράφος *person_knows_person* (αρχικός κόμβος).
  - Στο ερώτημα ***MatchSuggestion*** χρειάστηκαν επίσης οι γράφοι *person_isLocatedIn_place, person_workAt_organisation, person_studyAt_organisation, person_hasInterest_tag*. Για να βρεθούν τα ζευγάρια γίνεται μια αναζήτηση BFS από τον αρχικό κόμβο μέχρι βάθος h, και από όσους απόγονους πληρούν τις προϋποθέσεις επιστρέφονται οι limit πρώτοι με βάση την ομοιότητα Jaccard με τον αρχικό.
  - Στο ερώτημα ***TopStalkers*** χρειάστηκαν επίσης οι γράφοι *person_likes_post, post_hascreator_person, forum_hasMember_person*(ουσιαστικά *person_isMemberOf_forum*). Εξετάζονται όλοι οι κόμβοι του αρχικού γράφου αν είναι stalker, σύμφωνα με τις προϋποθέσεις, και επιστρέφονται οι k πιο κεντρικοί.
  - Στο ερώτημα ***FindTrends*** χρειάστηκαν επίσης ο γράφος *person_hasInterest_tag*. Κατασκευάζεται δύο γράφοι για κάθε tag, ένας για τις γυναίκες και ένας για τους άντρες. Υπολογίζεται το μέγεθος του κάθε ενδιαφέροντος ως το πλήθος του μέγιστου συνεκτικού γραφήματος (maxCC) και επιστρέφονται οι περιγραφές των πρώτων k.
  - Στο ερώτημα ***EstimateTrust*** χρειάστηκαν επίσης οι γράφοι *forum_hasMember_person, person_likes_post, post_hasCreator_person, comment_hasCreator_person, comment_replyOf_post, comment_replyOf_comment*. Για να καταστευαστεί ο γράφος εμπιστοσύνης(trust graph) εισάγουμε όλα τα μέλη ενός κόμβου και οι ακμές μεταξύ όσων γνωρίζονται με βάρος το 'trust' μεταξύ τους σύμφωνα με τον τύπο που μας δίνεται. Έπειτα για να υπολογίσουμε το 'trust' μεταξύ δύο ανθρώπων που δεν γνωρίζονται μεταξύ τους, χρησιμοποιούμε μια παραλλαγή του αλγορίθμου Tidal Trust.

- Στο **preprocessing** στάδιο του επιπέδου 3 τα βήματα είναι τα εξής:
    - Κατασκευάζεται ένας γράφος με όλα τα forum του κοινωνικού δικτύου με τη σχέση *forum_hasMember_person*, .
    - Αρχικοποιούνται #forum threads που το καθένα υπολογίζει το μέγεθος ενός forum και καταγράφει το αποτέλεσμα του σε μια κοινή δομή.
    - Κατασκευάζονται παράλληλα οι γράφοι των N μεγαλύτερων forum που περιέχει τα άτομα του forum και τις γνωριμίες μεταξύ τους.
- Ο αλγόριθμος εύρεσης κοινοτήτων **CPM(Clique Percolation Method)** είναι ο εξής:
    - Εύρεση όλων των k-κλικών του γράφου.
    - Κατασκευή του γράφου υπερ-κόμβων που αντιστοιχούν στις κλίκες που υπολογίστηκαν παραπάνω.
    - Εισαγωγή ακμής στο γράφο εάν 2 κλίκες έχουν k-1 κοινά άτομα.
    - Επιστροφή των συνεκτικών γραφημάτων του παραπάνω γράφου.
- Ο αλγόριθμος εύρεσης κοινοτήτων **Girvan-Newman** είναι ο εξής:
    - Υπολογισμός της κεντρικότητας κάθε ακμής του γράφου.
    - Αφαίρεση της πιο κεντρικής ακμής.
    - Υπολογισμός του modularity του γράφου. Η μετρική αυτή ουσιαστικά επιστρέφει μεγάλη τιμή όταν είναι ισχυρά συνδεδεμένο εσωτερικά το κάθε συνεκτικό υπο-γράφημα όμως είναι ασθενώς συνδεδεμένα μεταξύ τους.
    - Όσο το modularity δεν είναι αρκετά μεγάλο, επανέλαβε.

Testing

Το σύστημα έχει ελεγχθεί εξονυχιστικά για να επιβεβαιωθεί ότι δεν υπάρχουν bugs. Επίσης, η διαχείριση μνήμης έχει έλεγχθει με valgrind και δεν υπάρχουν memory leaks. Υπάρχει επίσης παντού error checking (για open, new, …).

Χρήση – Διεπαφή

Για να γίνει compile το σύστημα ακολουθούμε τα εξής βήματα:

- cd στο φάκελο Debug/
- make clean
- make all
- ./SNAP

Ο χρήστης πρέπει να βάλει το dataset μέσα στο φάκελο **/queries/dataset/** στη μορφή που είναι και το dataset του διαγωνισμού sigmod και να φτιάξει τον αρχικό γράφο από το αρχείο person_knows_person.csv. Επίσης, πρέπει να ορισθεί το μέγεθος του dataset(το πλήθος των ανθρώπων) στο αρχείο queries.h στη σταθερά DATA_SIZE. Έπειτα μπορεί να φτιάξει μία main που να καλεί οποιαδήποτε από τις υλοποιημένες ερωτήσεις ή να υπολογίσει κατευθείαν μετρικές του γράφου.

Επιπλέον, δίνεται η δυνατότητα να τρέξετε την ήδη υπάρχουσα main, η οποία θα τρέξει όποιο test έχει flag == True (όλα τα flags βρίσκονται στην αρχή της main). Αυτά τα test έχουν υλοποιηθεί για το 1k dataset του διαγωνισμού sigmod.

Επίσης, ο χρήστης μπορεί να τρέξει τις main που μας έχουν δοθεί στο e-class αντικαθιστώντας το dataset με αυτό που μας έχει δοθεί (ορίζοντας και τη σταθερά DATA_SIZE) και επιλέγοντας από τα test ποιες από τις main_part1, main_part2, main_part3 θέλει να τρέξει. Σχεδόν όλα τα test περνάνε επιτυχώς, αν και χρειάστηκαν διορθώσεις στις main που είχαν κάποια λάθος αναμενόμενα αποτελέσματα.

Links

https://www.assembla.com/code/social-network-analysis-project/subversion/nodes