ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
Τμήμα Πληροφορικής και Τηλεπικοινωνιών
1η Εργασία - Τμήμα: Περιττών Αριθμών Μητρώου
K22: Λειτουργικά Συστήματα – Χειμερινό Εξάμηνο '18
Ημερομηνία Ανακοίνωσης: Τρίτη 2 Οκτωβρίου 2018
Ημερομηνία Υποβολής: Τετάρτη 24 Οκτωβρίου 2018 Ώρα 23:59

Εισαγωγή στην Εργασία:

Ο στόχος αυτής της εργασίας είναι να εξοικειωθείτε με το περιβάλλον `Linux` και τα σχετικά βασικά εργαλεία προγραμματισμού και ανάπτυξης λογισμικού.

Θα υλοποιήσετε μια δομή γράφου που απεικονίζει μεταφορές χρηματικών ποσών μεταξύ διαφόρων τραπεζικών λογαριασμών. Ο γράφος παρέχει ένα μηχανισμό που μπορεί να βοηθήσει στην παρακολούθηση διακίνησης χρηματικών ποσών, στην διαλεύκανση πολυ-/τρι-γωνικών μεταφορών, αλλά και την κατανόηση όσον αφορά στην διαμόρφωση ομάδων πελατών. Θα πρέπει να υλοποιήσετε μια δομή στην κυρίως μνήμη στην οποία να μπορείτε να εισάγετε λογαριασμούς και δοσοληψίες, καθώς επίσης και να διαπιστώνετε καταστάσεις που πιθανόν δημιουργούνται από την δυναμική συμπεριφορά χρηστών με την βοήθεια επερωτήσεων.

Μερικές βασικές προϋποθέσεις για την παραπάνω εφαρμογή είναι οι εξής:

1. Δημιουργία δομής κατευθυνόμενου (πολυ-)γράφου (directed multi-graph) που μπορεί να αλλάζει δυναμικά στην κυρίως μνήμη. Δεν υπάρχουν περιορισμοί όσον αφορά στο μέγεθος ΤΟΥ γράφου (δηλ. αριθμό κόμβων και ακμών που διαθέτει). Η κάθε (κατευθυνόμενη) ακμή έχει βάρος που ουσιαστικά υποδηλώνει το συνολικό πόσο δοσοληψίας που έγινε μεταξύ δύο κόμβων.
2. Έλεγχο γράφου για την εύρεση κύκλων, εύρεση της διάχυσης χρημάτων στο δίκτυο, και εκτύπωση τής δομής με ένα κατανοητό στους χρήστες τρόπο.
3. Δυνατότητα προσθαφαίρεσης κόμβων και ακμών στη διάρκεια εκτέλεσης του προγράμματος σας και αναπροσαρμογής των τιμών των (υπαρχουσών) ακμών.
4. Έλεγχο συνδεσιμότητας της/των γράφων και ανάδειξη ομάδων χρηστών.
5. Το πρόγραμμα σας θα πρέπει στο τέλος να ελευθερώνει σταδιακά όλη την μνήμη που έχει δεσμεύσει.

Διαδικαστικά:

Το πρόγραμμά σας θα πρέπει να γραφτεί σε *C* (ή *C++* αν θέλετε αλλά χωρίς την χρήση STL/Templates) και να τρέχει στις μηχανές `Linux workstations` του τμήματος.

Το πρόγραμμα σας (source code) πρέπει να αποτελείται από ***τουλάχιστον δυο*** (και κατά προτίμηση πιο πολλά) διαφορετικά αρχεία. Το πρόγραμμά σας θα πρέπει *απαραιτήτως να κάνει χρήση* separate compilation .

Παρακολουθείτε την ιστοσελίδα του μαθήματος `http://www.di.uoa.gr/~ad/k22/` για επιπρόσθετες ανακοινώσεις αλλά και την ηλεκτρονική-λίστα (η-λίστα) του μαθήματος στο URL `https://piazza.com/uoa.gr/fall2018/k22/home`

- Υπεύθυνοι για την άσκηση αυτή (ερωτήσεις, αξιολόγηση, βαθμολόγηση κλπ.) είναι ο κ. Χρήστος Ζήσης `chrzissis+AT-di`, ο κ. Μάνος Τοράκης `sdi1400201+AT-di`, και ο κ. Παναγιώτης Κολαίτης `m1436+AT-di`.

- Στην διάρκεια των πρώτων μαθημάτων κυκλοφορούμε hard-copy λίστα στην τάξη στην οποία θα πρέπει να δώσετε το όνομά σας και το `Linux` user-id σας.

  Με αυτό το τρόπο μπορούμε να γνωρίζουμε ποιος/-α προτίθεται να υποβάλλει την πρώτη άσκηση και να προβούμε στις κατάλληλες ενέργειες για την υποβολή της εργασίας σας.
- Με βάση την παραπάνω λίστα, θα σας γράψουμε επίσης στο σύστημα `piazza.com` που βρίσκεται στο `https://piazza.com/uoa.gr/fall2018/k22/home`. Μόλις *αποδεχτείτε* ένα 'κωδικό' που θα σας σταλεί, μπορείτε να κάνετε ερωτήσεις και να δείτε απαντήσεις ή/και διευκρινήσεις που δίνονται σχετικά με την άσκηση.

Χαρακτηριστικά Γράφων σε Συντομία:

Τα βασικά συστατικά της δομής είναι τα εξής:

1. Κόμβοι (nodes) έχουν συγκεκριμένα ονόματα (πχ. alphanumeric strings που αντιπροσωπεύουν όνομά χρηστών ή αριθμούς λογαριασμών).

2. Ακμές/σύνδεσμοι (edges/vertices) έχουν συγκεκριμένη φορά και 'βάρος' που είναι γνωστά.

3. Μπορούν να υπάρχουν πολλαπλές ακμές με ίδια φορά μεταξύ δύο κόμβων οι οποίες μπορούν να έχουν και το ίδιο αριθμητικό βάρος. Κάθε μία από αυτές τις ακμές αντιπροσωπεύει μια δοσοληψία.

4. Ένας κόμβος μπορεί να συνδέεται με ένα η πολλούς άλλους. Το πόσο μεγάλο μπορεί να είναι αυτό το πλήθος δεν είναι γνωστό και φυσικά μπορεί να αυξάνεται/μειώνεται δυναμικά.

5. Νέες ακμές μπορούν να εισαχθούν οποιαδήποτε χρονική στιγμή.

6. Γενικά δεν υπάρχει περιορισμός στον αριθμό των κόμβων και των ακμών. Έτσι οποιαδήποτε μορφή υλοποίησης που αναλίσκεται σε *στατικούς* πίνακες για την απεικόνιση του γράφου *δεν είναι αποδεκτή*. Ακόμα και η χρήση δυναμικών πινάκων δεν θεωρείται η ενδεδειγμένη προσέγγισή στην άσκησή αυτή.

7. Το Σχήμα 1 δείχνει μια αντιπροσωπευτική κατάσταση της δομής που θα δημιουργήσετε. Στο σχήμα αυτό υπάρχει μία μεταφορά $1,340$ χρηματικών μονάδων από τον λογαριασμό kji στον kfc.

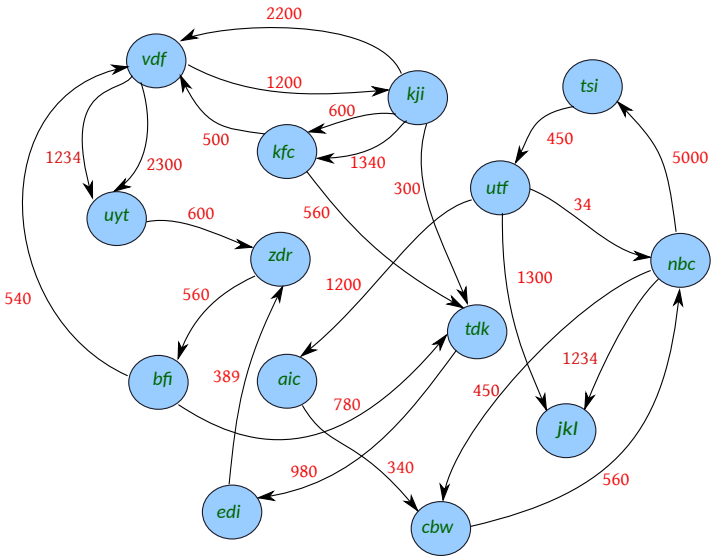
8. Ο αριθμός των αναμενόμενων εισαγωγών/προσαυξήσεων (ή και διαγραφών) κόμβων και ακμών στην δομή δεν είναι γνωστός εξ' αρχής.

9. Ο γράφος που θα δημιουργείτε δεν είναι απαραίτητο να έχει όλα τα στοιχεία του συνδεδεμένα (strongly connected). Για παράδειγμα στο Σχήμα 1, οι κόμβοι tsi, nbc, jkl, utf,aic και cbw αποτελούν ένα συνδεδεμένο κομμάτι του γράφου, ενώ οι υπόλοιποι κόμβοι αποτελούν ένα άλλο.

Γραμμή Κλήσης της Εφαρμογής:

Η εφαρμογή μπορεί να κληθεί με τον παρακάτω αυστηρό τρόπο στην γραμμή εντολής (`tty`):
./mygraph -i inputfile -o outfile
όπου

- mygraph είναι το εκτελέσιμο που θα δημιουργήσετε,
- inputfile είναι το αρχείο εισόδου δεδομένων. Σε αυτό κάθε γραμμή εκφράζει μια ακμή μεταξύ δυο κόμβων και παρέχει το 'βάρος' (δηλ. το ποσό της δοσοληψίας).
- outfile είναι το αρχείο εξόδου που την στιγμή της εξόδου της εφαρμογής σας καταγράφει την κατάσταση του γράφου όπως αυτή έχει εξελιχτεί.



Σχήμα 1: Παράδειγμα οργάνωσης γράφου.

Και οι δύο παραπάνω in-line παράμετροι (και αντίστοιχες σημαίες) είναι προαιρετικές όσον αφορά την κλήση τους στην γραμμή εντολής. Ωστόσο η υλοποίησή τους είναι υποχρεωτική. Οι σημαίες μπορούν να εμφανίζονται με οποιαδήποτε σειρά.

Περιγραφή της Διεπαφής του Προγράμματος:

Αφού το πρόγραμμα κληθεί, η/ο χρήστης μπορεί να αλληλεπιδράσει με την βοήθεια ενός prompt μέσα από το οποίο οι επόμενες εντολές μπορούν να κληθούν:

1. `i(nsert) Ni`
   εισήγαγε ένα νέο κόμβο `Ni`.
2. `(i)n(sert) Ni Nj weight`
   εισήγαγε μια ακμή μεταξύ Ni και `Nj` με βάρος `weight`. Οι κόμβοι `Ni` και `Nj` μπορεί να εμφανίζονται για πρώτη φορά στην δομή.
3. `d(elete) Ni`
   διέγραψε από το γράφο το κόμβο `Ni` καθώς επίσης και όλες τις εισερχόμενες/εξερχόμενες ακμές (και βάρη).
4. `(de)l(ete) Ni Nj weight`
   διέγραψε την ακμή μεταξύ `Ni` και `Nj` με βάρος `weight`. Αν υπάρχουν πολλαπλές ακμές με το ίδιο `weight` διέγραψε μια από τις υπάρχουσες. Αν δεν καθοριστεί `weight`, όλες οι υπάρχουσες ακμές μεταξύ `Ni` και `Nj` διαγράφονται.
5. `m(odify) Ni Nj weight nweight`
   άλλαξε το βάρος `weight` στην νέα τιμή `nweight`. Αν υπάρχουν πολλαπλές ακμές με την ίδια τιμή `weight` μετάτρεψε την πρώτη που θα βρεις.
6. `r(eceiving) Ni`
   παρουσίασε όλα τις δοσοληψίες που ο κόμβος `Ni` δέχεται σαν είσοδο.
7. `c(irclefind) Ni`
   βρίσκει αν ο κόμβος `Ni` εμπλέκεται σε απλούς κύκλους (simple cycles) και παρουσιάζει τους εν λόγω

κύκλους.

8. `f(indcircles) Ni k`
   βρίσκει αν ο κόμβος Ni εμπλέκεται σε κυκλικές δοσοληψίες με άλλους. Ωστόσο το ελάχιστο ποσό που κάθε ακμή πρέπει να έχει, είναι k μονάδες (τιμή για minimum `weight`).
9. `t(raceflow) Ni Nj l`
   βρίσκει πιθανές διαχύσεις ποσών που ξεκινούν από τον κόμβο `Ni` και τελειώνουν στον κόμβο `Nj`. Το `l` δίνει το μέγιστο μήκος της διαδρομής σε αριθμό ακμών που θα πρέπει να διατρεχτούν.
10. `e(xit)`
    το πρόγραμμα απλά τερματίζει αφού ελευθερώσει πρώτα με συγκροτημένο τρόπο όλο το χώρο που έχει καταλάβει στην μνήμη.

Η *μορφή εξόδου* της κάθε μία από τις παραπάνω εντολές δίνεται με *λεπτομέρεια* στην σελίδα του μαθήματος.

Χαρακτηριστικά του Προγράμματος που Πρέπει να Γράψετε:

1. Δεν μπορείτε να κάνετε pre-allocate οποιοσδήποτε χώρο αφού η δομή(-ές) θα πρέπει να μπορεί(-ουν) να μεγαλώσει(-ουν) χωρίς ουσιαστικά κανέναν περιορισμό όσον αφορά στον αριθμό των εγγραφών που μπορούν να αποθηκεύσουν. Η χρήση στατικών πινάκων/δομών που δεσμεύονται στην διάρκεια της συμβολομετάφρασης του προγράμματος σας δεν είναι αποδεκτές επιλογές.
2. Για την δομή που θα υιοθετήσετε θα πρέπει να μπορείτε να εξηγήσετε (και να δικαιολογήσετε στην αναφορά σας) την πολυπλοκότητα που οι τεχνικές που υιοθετείτε παρουσιάζουν.
3. Πριν να τερματίσει η εκτέλεση της εφαρμογής, το περιεχόμενο των δομών απελευθερώνεται με σταδιακό και ελεγχόμενο τρόπο.
4. Αν πιθανώς κάποια κομμάτια του κωδικά σας προέλθουν από κάποια δημόσια πηγή, θα πρέπει *να δώσετε αναφορά στη εν λόγω πηγή* είτε αυτή είναι βιβλίο, σημειώσεις, Internet URL κλπ. και να εξηγήσετε πως ακριβώς χρησιμοποιήσατε την εν λόγω αναφορά.
5. Έχετε πλήρη ελευθερία να επιλέξετε το τρόπο με τον οποίο τελικά θα υλοποιήσετε βοηθητικές δομές.

Τι πρέπει να Παραδοθεί:

1. Μια σύντομη και περιεκτική εξήγηση για τις επιλογές που έχετε κάνει στο σχεδιασμό του προγράμματος σας (2-3 σελίδες σε ASCII κειμένου είναι αρκετές).
2. Οπωσδήποτε ένα `Makefile` (που να μπορεί να χρησιμοποιηθεί για να γίνει αυτόματα το compile του προγράμματος σας). Πιο πολλές λεπτομέρειες για το (`Makefile`) και πως αυτό δημιουργείται δίνονται στην ιστοσελίδα του μαθήματος.
3. Ένα `tar-file` με όλη σας την δουλειά σε έναν κατάλογο που πιθανώς να φέρει το όνομα σας και θα περιέχει όλη σας την δουλειά δηλ. `source files, header files, output files` (αν υπάρχουν) και οτιδήποτε άλλο χρειάζεται.

Άλλες Σημαντικές Παρατηρήσεις:

1. Οι εργασίες είναι **ατομικές**.
2. Το πρόγραμμα σας θα πρέπει να τρέχει στα `Linux` συστήματα του τμήματος αλλιώς **δεν μπορεί να βαθμολογηθεί**.
3. Αν και αναμένεται να συζητήσετε με φίλους και συνεργάτες το πως θα επιχειρήσετε να δώσετε λύση στο πρόβλημα, αντιγραφή κώδικα (οποιαδήποτε μορφής) είναι κάτι που **δεν επιτρέπεται** και δεν πρέπει να γίνει. Οποιοσδήποτε βρεθεί αναμεμειγμένος σε αντιγραφή κώδικά **απλά παίρνει μηδέν** στο μάθημα. Αυτό ισχύει για **όσους εμπλέκονται ανεξάρτητα** από το ποιος έδωσε/πήρε κλπ.

4. Το παραπάνω ισχύει αν διαπιστωθεί *έστω και μερική άγνοια* του κώδικα που έχετε υποβάλει ή *άπλα υπάρχει υποψία* ότι ο κώδικας είναι προϊόν συναλλαγής με τρίτο/-α άτομο/α.
5. Προγράμματα που **δεν χρησιμοποιούν** separate compilation χάνουν αυτόματα 5% του βαθμού.
6. Σε καμιά περίπτωση τα Windows *δεν είναι επιλέξιμη* πλατφόρμα για την παρουσίαση αυτής της άσκησης.