Ανάπτυξη Λογισμικού για Πληροφοριακά Συστήματα 2016 - 2017

Μιχάλης Κατερίνης – Λινάρδος

Ενδεικτικές εντολές μεταγλώτισης και εκτέλεσης
make maint
g++ -o maint maint.cpp structs.cpp strongly.cpp grail.cpp bidbfs.cpp thread.cpp -lpthread
./maint mediumGraph.txt mediumWorkload_FINAL.txt

Εισαγωγικά

Βασικός στόχος της προσέγγισής μας στο δοθέν πρόβλημα είναι η ελαχιστοποίηση του χρόνου εκτέλεσης. Οι προδιαγραφές που ορίζονται από τις εκφωνήσεις ήδη κάνουν βήματα προς αυτή την κατεύθυνση, παρόλα αυτά υπάρχουν μία σειρά επιλογών υλοποίησης κατά την σχεδίαση του προγράμματος προκειμένου να οδηγηθούμε σε καλύτερους χρόνους υλοποίησης.

Εισαγωγικά αναφέρουμε ότι παρόλο που το κύριο βάρος του χρόνου εργασίας του προγράμματος αφορά τη διαδικασία αναζήτησης, ήδη η διαδικασία αρχικοποίησης των γράφων μπορεί να είναι αρκετά χρονοβόρα στο βαθμό που γίνονται κακές σχεδιαστικές επιλογές. Η διαδικασία αρχικοποίησης μπορεί να βελτιωθεί με δύο βασικές αρχές:

Αφενός, ελαχιστοποιώντας τις επαναλαμβανόμενες διαδικασίες δέσμευσης μνήμης οι οποίες είναι αρκετά χρονοβόρες. Συγκεκριμένα επιλέγοντας διπλασιασμούς όλων των πινάκων κατά τη χρήση realloc και όχι προσθήκη μνήμης και επαναχρησιμοποιόντας δομές όπως οι ουρές που αξιοποιούνται στην bidirectional bfs, αντί για τη δημιουργία ουρών σε κάθε κλήση της συνάρτησης.

Αφεταίρου, χρησιμοποιόντας σε όλες τις δομές (buffer, ουρές και στοίβες) δείκτες στην επόμενη θέση εγγραφής προκειμένου καμία διαδικασία εγγραφής να μην απαιτεί προσπέλαση μνήμης και διαδικασία αναζήτησης.

εισαγωγή και αναζήτηση

Βασική παραδοχή για την υλοποίησή μας είναι η χρήση μίας διπλής λειτουργίας hashing στην βασική δομή index – buffer που χρησιμοποιείται τόσο για τους αρχικούς γράφους όσο και για τον hypergraph που προκύπτει από την tarjan. Στόχος αυτής της επιλογής είναι η μεγαλύτερη δυνατή ταχύτητα της ανάκτησης ακμής και προκύπτει αξιοποιώντας ένα hashtable για κάθε node όπου με βάση την τιμή H δημιουργεί μία σειρά αλυσίδων list_node μέσα στον buffer για τις διαφορετικές τιμές του (node->id % H). Όσο μεγαλώνει η τιμή του H τόσο μειώνεται ο χρόνος αναζήτησης ωστόσο, για μικρούς γράφους δεσμεύονται περισσότερη αχρησιμοποιήτη μνήμη και αυξάνεται ο ρυθμός των επαναδεσμεύσεων, κάνοντας τις επιλογές μεγάλων τιμών του H αποδοτικές για μεγαλύτερους γράφους και προβληματικές για μικρούς γράφους.

Στην χρήση του bidbfs πέρα από την προφανή επιλογή αποφυγής περιττών αναζητήσεων, επιλέγεται ένα ενδιάμεσο βήμα αναζήτησης ακμής μεταξύ των δύο ουρών το οποίο μας επιτρέπει να αποφύγουμε μία σειρά αναζητήσεων κόμβων η οποία στην χειρότερη περίπτωση είναι ίση με το άθροισμα των κόμβων σε κάθε ουρά για το συγκεκριμένο βάθος (όταν συνδέονται οι δύο τελευταίοι κόμβοι σε σειρά προτεραιότητας από τις δύο ουρές).

Ευρετήρια σύνδεσης

Το πως η δημιουργία ευρετηρίου grail ή ευρετηρίου connected component βελτιστοποιεί την απόδοση του προγράμματος περιγράφεται από την εκφώνηση. Κατά την υλοποίηση η βασικές σχεδιαστικές επιλογές που καλούμαστε να αντιμετωπίσουμε είναι ο ορισμός της σταθεράς metric για την επαναδημιουργία του ευρετηρίου connected component στον δυναμικό γράφο και ο αριθμός των ευρετηρίων grail.

Λόγο του μικρού χρόνου που απαιτεί η δημιουργία των ευρετηρίων και του σχετικά μεγαλύτερου χρόνου αναζήτησης ακμών μεταξύ components στην περίπτωση δυναμικού γράφου ορίσαμε αρχικά την μεταβλητή metric σε χαμηλές τιμές, διαπιστώνοντας ότι η optimal τιμή του ήταν κοντά στο 0.2

Ο αριθμός των ευρετηρίων grail λειτουργεί αντιστρόφος ανάλογα με τον χρόνο εκτέλεσης, οπότε το ερώτημα ξεφεύγει από τις δικές μας επιλογές υλοποίησης και αφορά το πόσο χώρο μνήμης θέλουμε να αξιοποιήσουμε.

Multi-threading

Η χρήση πολυνηματικού προγραμματισμού όπως περιγράφεται στην εκφώνηση μας επιτρέπει, ανά ριπή να αξιοποιήσουμε τον χρόνο που μπορεί να καταλαμβάνει μία “δύσκολη” αναζήτηση για να γίνουν μία σειρά άλλων αναζητήσεων. Για να αξιοποιηθεί το πλεονέκτημα της παραλληλίας και την ίδια στιγμή τα αποτελέσματα να εκτυπώνονται με τη σειρά που πρέπει. Τα αποτελέσματα αποθηκεύονται με βάση έναν κωδικό κάθε εντολής αναζήτησης και εκτυπώνονται αφού πραγματοποιηθούν όλες οι αναζητήσεις. Για τον medium και large γράφο ο χρόνος εκτέλεσης είναι αντιστρόφως ανάλογος του αριθμού των νημάτων που χρησιμοποιούνται. Για τους μικρούς γράφους οι χρόνοι αναμονής και αρχικοποίησης των νημάτων επιβαρύνουν των ήδη μικρό χρόνο εκτέλεσης.