Πανεπιστήμιο Πατρών

Τμήμα Μηχανικών Η/Υ και Πληροφορικής

Παράλληλη Επεξεργασία

Υπεύθυνος Καθηγητής: Βενέτης Ιωάννης

Φοιτητές: Παπαχρονόπουλος Γεράσιμος 1059629
Καραβοκύρης Μιχαήλ 1059636
Κουτσοχέρας Ιωάννης 1059638

Ομάδα Χρηστών: 34

Αρχικά, το συνημμένο αρχείο που σας στέλνουμε αποτελείται από 3 υπό-αρχεία. Συγκεκριμένα τα:

- init_matrix_specs.h
- list.h
- main.c

Το init_matrix_specs.h είναι υπεύθυνο για το καθορισμό των αρχικών προδιαγραφών του αρχείου dag.txt και αποτελείται από δύο συναρτήσεις, την get_matrix_size() που επιστρέφει το μέγεθος του μητρώου (αριθμό σειρών και στηλών) και την get_edges_number() που επιστρέφει τον αριθμό των ακμών που υπάρχουν στο γράφημα.

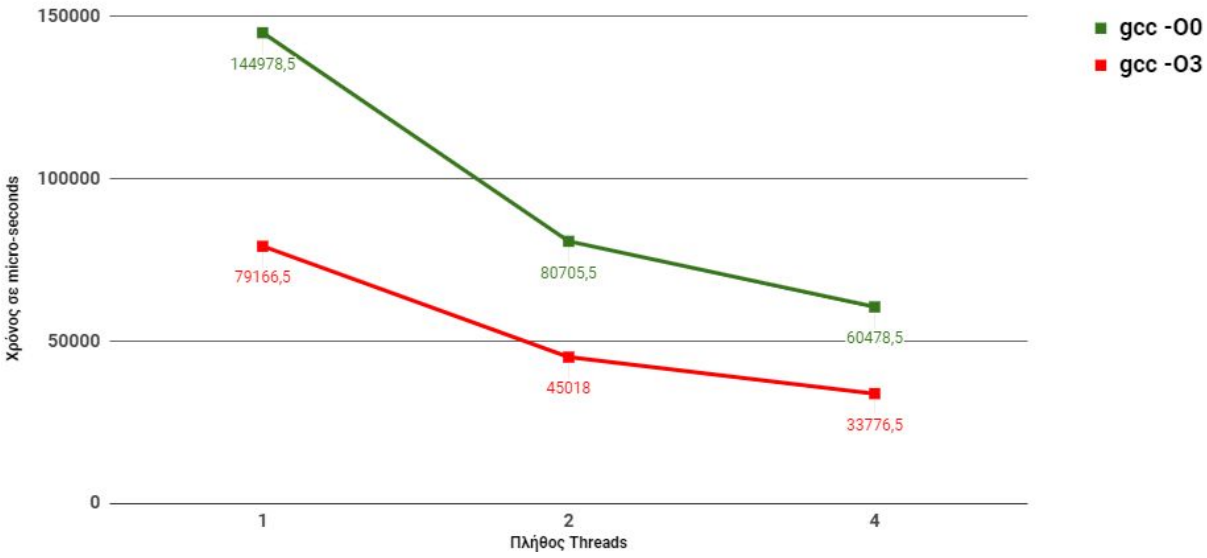
Στο αρχείο list.h υπάρχουν οι συναρτήσεις διαχείρισης λιστας, καθώς και η δήλωση της δομής διασυνδεδεμένης λίστας. Η χρήση των λιστών είναι η δημιουργία ουρών που περιέχουν κόμβους. Οι συναρτήσεις που χρησιμοποιούνται είναι οι εξής: push() για την εκχώρηση τον κόμβων στην ουρά, dequeue() για την διαγραφή του πρώτου κόμβου της ουράς και printList() για την εκτύπωση της ουράς.

Στη main.c αρχικά διαβάζουμε το αρχείο txt με το dag και εν συνεχεία δημιουργούμε τον πίνακα γειτνίασης των κόμβων. Αφού δημιουργηθεί ο πίνακας ορίζουμε τις λίστες S, L και καλούμε συνάρτηση για την καταγραφή του χρόνου έναρξης του παράλληλου τμήματος. Κατά την υλοποίηση του αλγορίθμου του Kahn διαπιστώσαμε ότι θα μπορούσαμε να υλοποιήσουμε το L με πίνακα αφού γνωρίζουμε το πλήθος τον κόμβων. Ωστόσο, σε περαιτέρω ανάλυση διαπιστώσαμε ότι για μεγάλα dataset η δέσμευση μνήμης για τους πίνακες θα δημιουργούσε πρόβλημα σε υπολογιστικά συστήματα με μικρή μνήμη. Το αποτέλεσμα αυτού θα ήταν το πρόγραμμα να μην τρέχει καθόλου λόγω segmentation fault. Με τη χρήση λιστών όμως το πρόγραμμα τρέχει έως ότου γεμίσει η διαθέσιμη μνήμη με αποτέλεσμα να εκτυπώνεται ένα μέρος της τοπολογικής διάταξης. Ως προς την παραλληλοποίηση, τα κρίσιμα σημεία του κώδικα είναι η εκχώρηση και διαγραφή στις λίστες καθώς και

η αλλαγή τιμών στον πίνακα γειτνίασης. Για αυτό το λόγο στα κομμάτια του κώδικα όπου γίνονται οι παραπάνω ενέργειες, χρησιμοποιείται το critical. Στο τέλος του κώδικα καλείται πάλι η συνάρτηση καταγραφής χρόνου για την λήξη, οι οποίοι χρόνοι μετριούνται σε microseconds, αλλά εκτυπώνεται και η λίστα L με το χρόνο εκτέλεσης του παράλληλου τμήματος.

Έκτος από τα παραπάνω επισημαίνουμε ότι όλοι οι κώδικες έχουν αναλυτικά σχόλια που εξηγούν τα βήματα και τον τρόπο σκέψης.

Παρακάτω παρουσιάζεται το γράφημα με τις ζητούμενες μετρήσεις:



Κάποιες παρατηρήσεις:

- Οι δοκιμές έγιναν σε virtual environment με 5GB RAM και 4 πυρήνες.
- Το dataset αποτελούνταν από ένα μητρώο 1435x1435 και συνολικά υπήρχαν 102715 ακμές στο γράφημα.
- Χρόνοι που δεν ανταποκρίνονταν στην πραγματικότητα και σε φυσιολογικά πλαίσια δεν λήφθηκαν υπ όψιν (π.χ αρνητικοί χρόνοι ή χρόνοι με τεράστια απόκλιση από τα υπόλοιπα αποτελέσματα).
- Για την εξαγωγή των αποτελεσμάτων πάρθηκαν οι μέσοι όροι 8 δοκιμών.

Τα κύρια συμπεράσματα από τις μετρήσεις είναι 2.

- Η σημαντική διαφορά στους χρόνους εκτέλεσης του προγράμματος με μεταγλώττιση χωρίς βελτιώσεις (gcc -O0) και με τις μέγιστες βελτιώσεις (gcc -O3)

- Η μείωση του χρόνου μέτρησης της παράλληλης περιοχής όσο αυξάνεται το πλήθος των threads. Να σημειώσουμε ότι η βελτίωση που παρατηρούμε προσεγγίζει το νόμο του Amdahl για s=0 (καθώς η μέτρηση χρόνου γίνεται μόνο στην παράλληλη περιοχή)