ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΥΠΟΛΟΓΙΣΤΩΝ

ΠΑΡΑΛΛΗΛΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΣΕ ΠΡΟΒΛΗΜΑΤΑ ΜΗΧΑΝΙΚΗΣ ΜΑΘΗΣΗΣ

3η Εργαστηριακή Άσκηση
Το πρόβλημα του περιοδεύοντα πωλητή, τυχαία αναζήτηση και ο αλγόριθμος ant colony

---

ΠΡΟΕΤΟΙΜΑΣΙΑ

Το πρόβλημα του περιοδεύοντα πωλητή (travelling salesman problem, TSP) είναι ένα από πλέον δημοφιλή προβλήματα συνδυαστικής βελτιστοποίησης

https://en.wikipedia.org/wiki/Travelling_salesman_problem

http://nemertes.lis.upatras.gr/jspui/bitstream/10889/6347/1/diplomatiki_Nikolas_Stylianou.pdf

https://www.fourmilab.ch/documents/travelling/anneal/

Το πρόβλημα εύρεσης της κοντινότερης διαδρομής που μπορεί να ακολουθήσει κάποιος ξεκινώντας από την πόλη Α μεταβαίνοντας σε όλες τις πόλεις ενός καταλόγου μόνο μια φορά και καταλήγοντας στην πόλη εκκίνησης είναι ένα NP-hard πρόβλημα. Στο εργαστήριο θέλετε να κατασκευάσετε μια μέθοδο που να εντοπίζει μια "καλή διαδρομή". Ελπίζουμε να κουράσετε όσο το δυνατόν λιγότερο τον πωλητή μας και ο χρόνος του υπολογισμού της διαδρομής πρέπει να ειναι όσο το δυνατόν ταχύτερος.

ΕΚΦΩΝΗΣΗ

Αρχικά θα ορίσετε τις καρτεσιανές συντεταγμένες 10000 πόλεων σε ένα χάρτη που έχει διαστάσεις 1000x1000 χιλιομέτρων, χρησιμοποιώντας την συνάρτηση rand(). Να θεωρήσετε την πρώτη πόλη σαν πόλη εκκίνησης και τερματισμού ταυτόχρονα.

ΤΥΧΑΙΑ ΑΝΑΖΗΤΗΣΗ.
Ο πλέον απλός αλγόριθμος μπορεί να κατασκευαστεί είναι να ορίσετε μια τυχαία διαδρομή και να αναζητήσετε νέες διαδρομές εναλλάσοντας δύο τυχαίες πόλεις στην διαδρομή που έχετε ορίσει. Αν η νέα διαδρομή είναι συντομότερη, τότε διατηρείτε την νέα διαδρομή αν δεν είναι συντομότερη τότε επαναφέρετε τις δύο πόλεις στις αρχικές τους θέσεις και ψάχνετε να βρείτε ένα νέο τυχαίο ζευγάρι πόλεων.

Εργασία 1

Μελετήστε τον αλγόριθμο και αποφασίστε για ποιά είναι οι απαραίτητοι πίνακες που θα πρέπει να χρησιμοποιήσετε έτσι ώστε να αποθηκεύσετε την πληροφορία που χρειάζεται.
Στην συνέχεια αποφασίστε ποιές θα είναι οι βασικές συναρτήσεις που θα εκτελέσουν τους υπολογισμούς που χρειάζονται. Κατασκευάστε το σειριακό πρόγραμμα που υλοποιεί την μέθοδο, επιβεβαιώστε την σωστή λειτουργία του και εντοπίστε τα τμήματα του προγράμματος που καταναλώνουν τον μεγαλύτερο χρόνο υπολογισμού. Υπολογίστε τον χρόνο απόκρισης του προγράμματος στο σύνολο των 10000 πόλεων την συνολική απόσταση που πρέπει να διανύσει ο πωλητής και το μέγεθος της μνήμης που χρησιμοποιείτε.

Εργασία 2
Τροποποιείστε τον αλγόριθμο της πρώτης εργασίας έτσι ώστε να βελτιώσετε τις δυνατότητες εύρεσης του μονοπατιού ελάχιστης απόστασης με παράλληλο αλγόριθμο. Επιβεβαιώστε την καλή του λειτουργία και χρησιμοποιώντας τα ίδια δεδομένα, συγκρίνατε τον συνολικό χρόνο απόκρισης, την συνολική απόσταση που πρέπει να διανύσει ο πωλητής και το μέγεθος της μνήμης που χρησιμοποιείτε στην εργασία 2.

Αλγόριθμος των Heinritz-Hsiao
Ενας από τους αποτελεσματικότερους αλγόριθμους αναζήτησης της συντομότερης διαδρομής είναι αυτός που προτάιθηκε από τους Heinritz και Hsiao:
Βημα 0. Βρίσκεστε στην πόλη εκκίνησης.
Βημα 1. Βρείτε την κοντινότερη πόλη από την πόλη που βρίσκεστε και δεν την έχετε επισκευθεί. Μεταβείτε σε αυτήν την πόλη και ανανεώστε την συνολική απόσταση που έχετε διανύσει.
Βήμα 2. Αν έχετε επισκευθεί όλες τις πόλεις προσθέστε και την απόσταση της πόλης που βρίσκεστε από την πόλη εκκίνησης στην συνολική απόσταση και ο αλγόριθμος τερματίζει, διαφορετικά επαναλάβατε το βήμα 1.

Εργασία 3
Μελετήστε τον αλγόριθμο και αποφασίστε για ποιά είναι οι απαραίτητοι πίνακες που θα πρέπει να χρησιμοποιήσετε έτσι ώστε να αποθηκεύσετε την πληροφορία που χρειάζεται. Στην συνέχεια αποφασίστε ποιές θα είναι οι βασικές συναρτήσεις που θα εκτελέσουν τους υπολογισμούς που χρειάζονται. Επιβεβαιώστε την ορθή λειτουργία της μεθόδου με μικρό αριθμό πόλεων.
Υπολογίστε τον χρόνο απόκρισης του προγράμματος στο σύνολο των 10000 πόλεων την συνολική απόσταση που πρέπει να διανύσει ο πωλητής και το μέγεθος της μνήμης που χρησιμοποιείτε.

Εργασία 4
Τροποποιείστε τον προηγούμενο αλγόριθμο επιλέγοντας στο βήμα 1 όχι την πλησιέστερη πόλη που δεν έχετε επισκευθεί, αλλά με τυχαίο τρόπο να επιλέγετε την δεύτερη πλησιέστερη πόλη που δεν έχετε επισκευθεί. Η πιθανότητα επιλογής δεν θα είναι ισοπίθανη αλλά η πιθανότητα επιλογής θα πρέπει να ελέγχεται με μια παράμετρο της επιλογής σας.
Υλοποιείστε την σειριακή έκδοση του αλγόριθμου, επιβεβαιώστε την σωστή λειτουργία του και εντοπίστε τα τμήματα του προγράμματος που καταναλώνουν τον μεγαλύτερο χρόνο υπολογισμού. Χρησιμοποιώντας τα ίδια δεδομένα, συγκρίνατε τον συνολικό χρόνο απόκρισης, την συνολική απόσταση που πρέπει να διανύσει ο πωλητής και το μέγεθος της μνήμης που χρησιμοποιείτε στην εργασία 2.

Εργασία 5

Τροποποιείστε τον αλγόριθμο της εργασίας 4 έτσι ώστε τμήματά του να εκτελούνται παράλληλα. Κατασκευάστε το πρόγραμμα που υλοποιεί την μέθοδο, επιβεβαιώστε την σωστή λειτουργία του, εντοπίστε τα τμήματα του προγράμματος που καταναλώνουν τον μεγαλύτερο χρόνο υπολογισμού.
Χρησιμοποιώντας τα ίδια δεδομένα, συγκρίνατε τον συνολικό χρόνο απόκρισης, την συνολική απόσταση που πρέπει να διανύσει ο πωλητής και το μέγεθος της μνήμης που χρησιμοποιείτε στην εργασία 2.

Αποικία Μυρμηγκιών (ant colony)
Ο ερευνητής της Τεχνητής Νοημοσύνης Marco Dorigo περιέγραψε το 1993 μια μέθοδο ευρετικής δημιουργίας "καλών λύσεων" στο TSP χρησιμοποιώντας μια προσομοίωση μιας αποικίας μυρμηγκιών που ονομάζεται ACS (σύστημα αποικιών μυρμηγκιών). Υποδηλώνει τη συμπεριφορά που παρατηρείται στα πραγματικά μυρμήγκια για να βρεθούν μικρές διαδρομές μεταξύ των πηγών τροφής και της φωλιάς τους, μια αναδυόμενη συμπεριφορά που προκύπτει από την προτίμηση κάθε μυρμηγκιού να ακολουθήσει τις φερομόνες διαδρομής που έχουν απομείνει στο έδαφος από άλλα μυρμήγκια.
Το ACS στέλνει έναν μεγάλο αριθμό μυρμήγκιών για να διερευνήσει πολλές πιθανές διαδρομές στο χάρτη. Κάθε μυρμήγκι επιλέγει με πιθανοτικό τρόπο την επόμενη πόλη να επισκεφθεί συνδυάζοντας την απόσταση προς την πόλη και την ποσότητα της εικονικής φερομόνης που έχει κατατεθεί στην διαδρομή προς την πόλη. Τα μυρμήγκια διερευνούν, αποθέτουν φερομόνη σε κάθε διαδρομή που διασχίζουν, μέχρι να ολοκληρώσουν μια περιήγηση. Οταν ένα μυρμήγκι ολοκληρώσει όλη την διαδρομή υπολογίζεται η φερομόνη που αφήνει κατά μήκος της πλήρους διαδρομής (global tracking updating). Η ποσότητα της φερομόνης που εναποτίθεται είναι αντιστρόφως ανάλογη με το μήκος της περιοδείας: όσο πιο σύντομη είναι η περιοδεία, τόσο περισσότερη φερορμόνη τοποθετείται στα μονοπάτια της διαδρομής.

Εργασία 6
Αναζητήστε βιβλιογραφία και έτοιμο λογισμικό που θα σας βοηθήσουν να κατανοήσετε τον αλγόριθμο και τις παραλαγές του. Αποφασίστε ποιά παραλλαγή και ενδεχομένως ποιά σειριακή υλοποίηση (με τις κατάλληλες τροποποιήσεις) είναι η καταλληλότερη για το πρόβλημά σας, διαφορετικά υλοποιείστε το δικό σας λογισμικό. Επιβεβαιώστε την σωστή λειτουργία σε μικρό αριθμό πόλεων. Χρησιμοποιώντας τα ίδια δεδομένα των 10000 πόλεων, συγκρίνατε τον συνολικό χρόνο απόκρισης, την συνολική απόσταση που πρέπει να διανύσει ο πωλητής και το μέγεθος της μνήμης που χρησιμοποιείτε στην εργασία 2.

Εργασία 7
Κατασκευάστε την παράλληλη έκδοση του προγράμματος (openMP με simd λειτουργίες όπου αυτό είναι δυνατόν) της εργασίας 6. Επιβεβαιώστε την σωστή λειτουργία σε μικρό αριθμό πόλεων. Χρησιμοποιώντας τα ίδια δεδομένα των 10000 πόλεων, συγκρίνατε τον συνολικό χρόνο απόκρισης, την συνολική απόσταση που πρέπει να διανύσει ο πωλητής και το μέγεθος της μνήμης που χρησιμοποιείτε στην εργασία 2.

ΠΑΡΑΔΟΤΕΑ

Συνοπτική περιγραφή του τρόπου υλοποίησης της κάθε εργασίας, τις υλοποιήσεις σε c ή c++, οι απαντήσεις στα ερωτήματα που θέτει κάθε εργασία και τα αποτελέσματα εκτέλεσης κάθε προγράμματος με

$time myprogram