ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Τελική Αναφορά: Αλγόριθμος Κοντινότερων Γειτόνων (ΚΝΝ)

ΑΝΑΠΤΥΞΗ ΛΟΓΙΣΜΙΚΟΥ ΓΙΑ ΠΛΗΡΟΦΟΡΙΑΚΑ ΣΥΣΤΗΜΑΤΑ

Ακαδ. Έτος 2023-2024

ΟΜΑΔΑ:
Άννα Γώγουλα - 1115201800305
Βασίλης Μιχαλέας - 1115201900284
Αταλάντη Παπαδάκη - 1115201800148

Επιτεύγματα και Βελτιώσεις

Η εργασία πάνω στον αλγόριθμο KNN αντιπροσωπεύει ένα σημαντικό κομμάτι της πορείας μας και αυτού του εξαμήνου. Μέσα από την υλοποίησή και τη βελτιστοποίηση του αλγορίθμου, αντιμετωπίσαμε πολλές προκλήσεις και αναδείξαμε σημαντικά στοιχεία. Μέσα από την σειρά των τριών παραδοτέων, κατορθώσαμε να εφαρμόσουμε την δομή του γράφου και να αναπτύξουμε μια λειτουργική υλοποίηση του αλγορίθμου. Η συνεργασία μεταξύ των μελών της ομάδας και η συνέπεια προς την εργασία και το ενδιαφέρον οδήγησε σε σημαντικές βελτιώσεις, με την προσθήκη προηγμένων στοιχείων, όπως τα projection trees.

Κατα την εξέλιξη της εργασίας, επικεντρωθήκαμε στην ανάπτυξη εργαλιών για την διαχείριση του κώδικα, με την δημιουργία βελτιώσεων. Η επιτυχής εφαρμογή της λειτουργίας local join και η συνδυαστική χρήση των βελτιστοποιήσεων στον αλγόριθμο αποτέλεσαν ιδιαίτερα σημαντικά οφέλη και συνέβαλαν σημαντικά στην αύξηση της αποδοτικότητας του. Στη συνέχεια, επεκτείναμε τη λειτουργικότητα με την ενσωμάτωση των projection trees και την χρήση παραλληλισμού, καταφέρνοντας να επιτύχουμε ακόμα μεγαλύτερες βελτιώσεις στην απόδοση του αλγορίθμου.

Οι βελτιστοποιήσεις που κάναμε συνέβαλαν στην αποφυγή ανεπιθύμητων υπολογισμών, μειώνοντας σημαντικά τον χρόνο εκτέλεσης. Ενώ μετά από ενδελεχή ανάλυση του κώδικα κατανοήσαμε την αξία της διαχείρισης της πολυπλοκότητας. Η παραλληλοποίηση του κώδικα επέτρεψε την αποτελεσματική εκτέλεση σε συστήματα με πολλούς πυρήνες, εκμεταλλευόμενη πλήρως τους πόρους του συστήματος.

Προκλήσεις και Συμπεράσματα

Αντιμετωπίσαμε πολλές προκλήσεις, όπως η καλή διαχείριση των δεδομένων και τη βελτιστοποίηση της απόδοσης. Αυτές οι προκλήσεις, ωστόσο μας οδήγησαν να κατανοήσουμε βαθύτερα τις λειτουργίες του αλγορίθμου KNN και να μας βάλουν σε σκέψεις για την εύρεση λύσεων.

Η αρχική υλοποίηση του αλγορίθμου αντιμετώπισε προβλήματα σε θέματα απόδοσης, κυρίως όταν χρησιμοποιούσαμε μεγάλα σύνολα δεδομένων.

Στο πλαίσιο της ανάπτυξης του αλγορίθμου, πραγματοποιήσαμε διάφορες βελτιστοποιήσεις προκειμένου να βελτιώσουμε την απόδοση και την αποτελεσματικότητά του. Ακολούθως περιγράφουμε κάποιες από αυτές τις βελτιστοποιήσεις:

- Εφαρμόσαμε την τεχνική Local Join που εξετάζει τις αποστάσεις με τους γείτονες των γειτόνων και επιτρέπει τον υπολογισμό καινούριων κοντινότερων γειτόνων με αποτελεσματικό τρόπο, αποφεύγοντας τη διπλή επεξεργασία και τον τοπικό υπολογισμό για κάθε κόμβο ανεξάρτητα.
- Εφαρμόσαμε την Σταδιακή Αναζήτηση (Incremental Search) χρησιμοποιώντας flags σε κάθε κόμβο, επιτρέποντας τη σύγκριση μόνο με νέους γείτονες, γεγονός που μείωσε τον υπολογιστικό φόρτο καθώς αποφεύχθηκαν περιττοί υπολογισμοί.

- Εφαρμόσαμε Δειγματοληψία ώστε να μειώσουμε το κόστος του τοπικού join. Επιλέγοντας τυχαία μόνο κόμβους που αναμένεται να συμμετάσχουν στις συγκρίσεις.
- Εφαρμόσαμε Πρόωρο Τερματισμό όπου επηρεάστηκε σε μικρό ποσοστό η ακρίβεια του αλγορίθμου ώστε να μειωθεί σημαντικά το χρονικό κόστος.
- Ανακατασκευάσαμε την ευκλείδεια απόσταση χρησιμοποιώντας την τετραγωνική απόσταση για την εξοικονόμηση υπολογιστικού χρόνου.
- Χρησιμοποιήσαμε Projection Trees για την αρχικοποίηση των ακμών στον γράφο, περιορίζοντας έτσι το χώρο επιλογής τυχαίων γειτόνων. Προσφέροντας έναν αποδοτικότερο τρόπο εύρεσης γειτόνων.

Συμπεράσματα για τις Βελτιστοποιήσεις του Προγράμματος:

Μετά από περίπου 250 εκτελέσεις σε ένα σύστημα με επεξεργαστή 11ης γενιάς Intel Core i7-11700F @ 2.50GHz, βρέθηκαν σημαντικές βελτιστοποιήσεις. Τα χαρακτηριστικά του συστήματος είναι τα εξής:

Επεξεργαστής:
- Μοντέλο: 11th Gen Intel(R) Core(TM) i7-11700F @ 2.50GHz
- Βασική Συχνότητα: 2.50 GHz
- Πυρήνες: 8
- Λογικοί Επεξεργαστές: 16
- Εικονικοποίηση: Ενεργοποιημένη
- L1 cache: 640 KB
- L2 cache: 4.0 MB
- L3 cache: 16.0 MB
- Μέγιστη Συχνότητα: 4.09 GHz

Μνήμη RAM:
- Τύπος: DDR4 @ 2133Hz
(Σημείωση: Υποδεικνύεται ότι θα μπορούσε να είναι και υψηλότερη, ενδεχομένως με καλύτερους χρόνους)
- Διαφορά στους χρόνους μεταξύ 2133Hz και 3100Hz DDR4: Έως και 40-50%

Λειτουργικό Σύστημα:
- Windows 11 WS2 Ubuntu22.2

Σημείωση: Οι δοκιμές πραγματοποιήθηκαν παράλληλα με άλλες διεργασίες, κατά τη διάρκεια των οποίων εκτελούνταν περίπου 10+ προγράμματα ταυτόχρονα.

Αναφορικά με τα Δεδομένα:
- Κόμβοι: 2500
- Διαστάσεις: 400
- Γείτονες: 400

Συμπεράσματα:

1. *Πιο Ακριβής Εκτέλεση:*
   - Ακρίβεια: 84.056500%
   - Χρόνος: 320.568949
   - Δειγματοληψία (p): 0.726551
   - Πρόωρος Τερματισμός (δ): 0.000023
   - Συνθήκη Ολοκλήρωσης Projection Trees (D): 17
   - Threads: 2

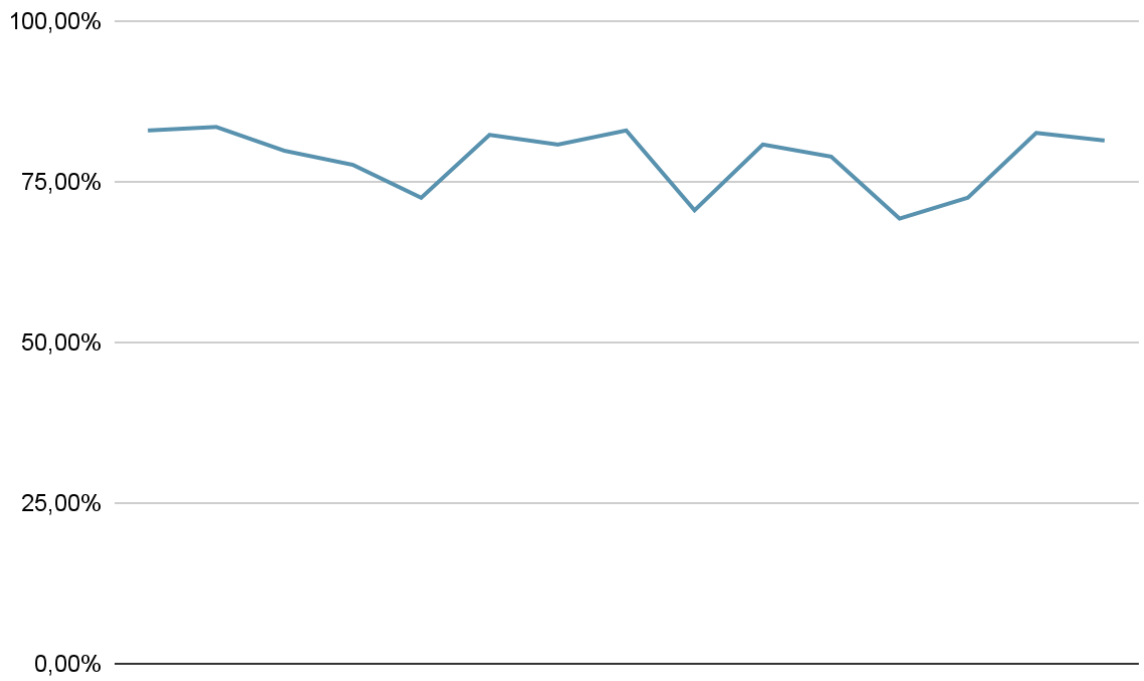
2. *Πιο Γρήγορη Εκτέλεση:*
   - Ακρίβεια: 80.810300
   - Χρόνος: 133.820998
   - Δειγματοληψία (p): 0.717858
   - Πρόωρος Τερματισμός (δ): 0.000048
   - Συνθήκη Ολοκλήρωσης Projection Trees (D): 266
   - Threads: 4

Παρατηρήσεις:

- Αρχικά, παρατηρούμε γενικά ότι η μείωση του χρόνου εκτέλεσης μειώνει το ποσοστό ακρίβειας του αλγόριθμου.
- Η παραλληλοποίηση/πολυνημάτωση του αλγορίθμου βοηθάει δραστικά στην μείωση του χρόνου εκτέλεσης, το οποίο ήταν αναμενόμενο καθώς οι εργασίες χωρίζονται στα διάφορα νήματα που τρέχουν παράλληλα, με κατάλληλο τρόπο.
- Παρατηρούμε πως όταν δειγματοληπτούμε λιγότερους κόμβους, η ακρίβεια αλλά και ο χρόνος μειώνονται. Αυτό συμβαίνει καθώς δεν ελέγχονται όλοι οι πιθανοί κόμβοι, με συνέπεια να παραλειφθεί η εξέταση κάποιων κατάλληλων κόμβων, που οδηγεί και στην μείωση του χρόνου καθώς ελαττώνεται η υπολογιστική πολυπλοκότητα.
- Τέλος, όσο πιο μικρή είναι η συνθήκη ολοκλήρωσης D, ο χρόνος εκτέλεσης είναι μεγαλύτερος καθώς το δέντρο βαθαίνει και ο χρόνος αναζήτησης σε αυτό μεγαλώνει.

Σχεδιαστική παρατήρηση:

- Στην υλοποίηση μας, χρησιμοποιήσαμε συνδεδεμένες λίστες για τις αναπαραστάσεις των δεδομένων, καθώς θεωρήσαμε πιο εύκολη την διαχείρισή τους για το συγκεκριμένο project. Επίσης, η δομή αυτή βοηθάει σε επίπεδο χρόνου γιατί η επεξεργασία των συγκεκριμένων δεδομένων μέσω λιστών είναι πιο αποδοτική.
- Γενικά, δεν περιορίσαμε την κατανάλωση μνήμης στο έπακρο, ώστε να καταφέρουμε να διατηρήσουμε τον χρόνο εκτέλεσής χαμηλό και να γίνει το πρόγραμμά μας πιο αποδοτικό.



Διάγραμμα με (μερικά από) τα ποσοστά ακρίβειας του αλγόριθμου