2017

Υλοποίηση των αλγορίθμων συσταδοποίησης K-means / K-medoids για πολυγωνικές καμπύλες

Ανάπτυξη Λογισμικού για Αλγοριθμικά Προβλήματα
Μέρος 2ο

Κανελλόπουλος Στέφανος
(Α.Μ. 1115201200050)

Χανιωτάκης-Ψύχος Χαρίδημος
(Α.Μ. 1115201200194)

Περιγραφή προγράμματος

Αρχικά, το πρόγραμμα μας κατασκευάζει την δομή αναζήτησης των hashtables, με την βελτιωμένη έκδοση της διαδικασίας από το πρώτο μέρος του project (δραματική μείωση χρόνου κατασκευής). Διαβάζουμε τα απαραίτητα στοιχεία από το config file (number of clusters κτλ) και καλείται η κεντρική συνάρτηση clustering() για τον έλεγχο της κλήσης των συναρτήσεων για την υλοποίηση του clustering.

Ανάλογα την μετρική συνάρτηση που επιλεγεί ο χρήστης έχουμε λάβει υπόψιν μας στην περίπτωση της Dynamic Time Warping (DTW) να μην χρησιμοποιούμε την Mean Discrete Frechet curve.

Υπάρχουν τρεις ομάδες συναρτήσεων που χρησιμοποιούνται στο clustering. Οι συναρτήσεις για το Initialization (Init_1(), Init_2()) οι όποιες αρχικοποιούν τα κέντρα των clusters. Οι συναρτήσεις για το Assignment (Assignment_1 (), Assignment_2 ()) που αναθέτουν τις υπόλοιπες καμπύλες του dataset στο πιο κοντινό κέντρο των clusters. Οι συναρτήσεις για το Update (PAM (), Mean_Discrete_Frechet ()) που ενημερώνουν τα κέντρα των clusters αφού πλέον περιλαμβάνουν καμπύλες.

Η διαδικασία του clustering τερματίζεται υπό την συνθήκη ότι τα κέντρα είτε δεν μετατοπίστηκαν καθόλου είτε μετατοπίστηκαν ελάχιστα (συμβολική σταθερά THRESHOLD). Αφού τερματιστεί η διαδικασία, υπολογίζεται το silhouette για κάθε cluster ξεχωριστά αλλά και για το σύνολο τους. Τέλος, τα αποτέλεσμα εξάγονται σε ένα αρχείο που έχει επιλέξει ο χρήστης.

Αρχεία κώδικα και σύντομη περιγραφή τους

- makefile: Μεταγλώττιση προγράμματος και παραγωγή εκτελέσιμου αρχείου
- main.c: Δέχεται τα ορίσματα του χρήστη, δημιουργεί την δομή αναζήτησης (από πρώτο μέρος Project) και καλεί την συνάρτηση clustering για την εκτέλεση και των 8 συνδυασμών που ζητήθηκαν
- structs.c: Περιλαμβάνει όλες τις δομές που ήταν απαραίτητες για την υλοποίηση του προγράμματος (hash table, linked list κτλ.)
- functions.c: Περιλαμβάνει τις συναρτήσεις που είναι απαραίτητες για την δημιουργία της δομής αναζήτησης (pre-processing)
- functions.h: Περιλαμβάνει όλες τις standard libraries, όπως και τις συμβολικές σταθερές
- hashtable.c: Περιλαμβάνει τις συναρτήσεις για το hash table (init, insert, destroy) όπως και τις συναρτήσεις για probablistic hashing
- hashtable.h: Αρχείο επικεφαλίδας του hashtable.c
- metric_functions.c: Περιλαμβάνει τις συναρτήσεις για τον υπολογισμό απόστασης μεταξύ δύο καμπυλών.
- metric_functions.h: Αρχείο επικεφαλίδας του metric_functions.c
- output_functions.c: Περιλαμβάνει την συναρτήση για την εξαγωγή των αποτελεσμάτων στο output file
- output_functions.h: Αρχείο επικεφαλίδας του output_functions.c
- preprocessing.c: Περιλαμβάνει τις συναρτήσεις για την δημιουργία της δομής αναζήτησης
- preprocessing.h: Αρχείο επικεφαλίδας του preprocessing.c
- binary_tree.c: Περιλαμβάνει τις συναρτήσεις για την δημιουργία, αρχικοποιηση του δυαδικού δέντρου και την εισαγωγή κόμβων
- binary_tree.h: Αρχείο επικεφαλίδας του binary_tree.c
- clustering_init.c: Περιλαμβάνει τις συναρτήσεις για την υλοποίηση των δύο μεθόδων Initialization
- clustering_init.h: Αρχείο επικεφαλίδας του clustering_init.c
- clustering_assignment.c: Περιλαμβάνει τις συναρτήσεις για την υλοποίηση των δύο μεθόδων Assignment

- clustering_assignment.h: Αρχείο επικεφαλίδας του clustering_assignment.c
- clustering_update.c: Περιλαμβάνει τις συναρτήσεις για την υλοποίηση των δύο μεθόδων Update
- clustering_update.h: Αρχείο επικεφαλίδας του clustering_update.c
- clustering.c: Καλεί κάθε πιθανό συνδυασμό όλων των ζητουμένων μεθόδων και ελέγχει αν ικανοποιείτε η συνθήκη τερματισμού του clustering (μηδενική ή ελάχιστη μετακίνηση των κέντρων). Επίσης περιλαμβάνει την συνάρτηση για τον υπολογισμό του silhouette
- clustering.h: Αρχείο επικεφαλίδας του clustering.c

Μεταγλώττιση και χρήση του προγράμματος

Για την μεταγλώττιση του προγράμματος περιλαμβάνεται αρχείο makefile και πραγματοποιείται με την εντολή "./make" στον τρέχοντα κατάλογο. Με την εντολή "./make clean" γίνεται ο καθαρισμός των αρχείων αντικειμένων.

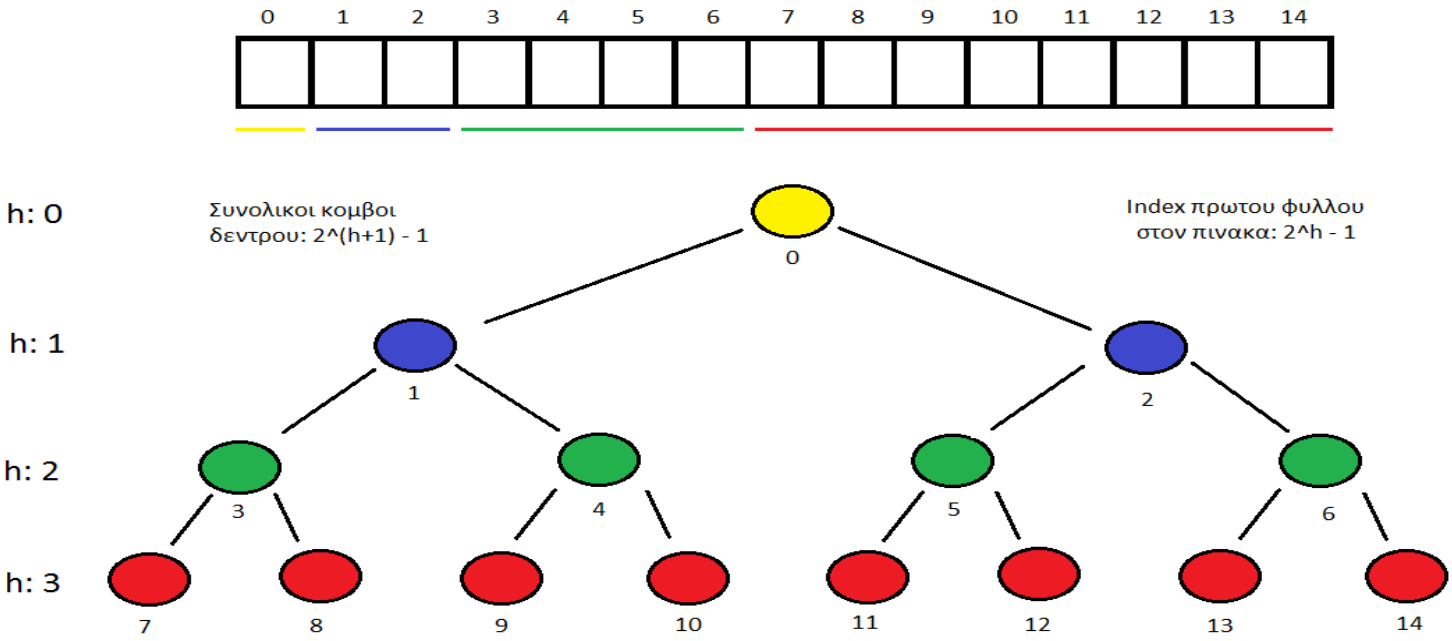
Το πρόγραμμα εκτελείται με την παρακάτω εντολή:

```
./cluster –i <input file> –c <configuration file> -o <output file> -d <metric>
```

Σε περίπτωση που κάποιο από τα παραπάνω ορίσματα δεν δοθεί στο command line, ο χρήστης έχει την δυνατότητα σε runtime να τα δώσει (θα ζητηθούν όλα όσα λείπουν).

Επιπλέον παρατηρήσεις/παραδοχές

- Αρχικά να αναφέρουμε ότι διορθώσαμε στο πρώτο μέρος του project την μετατροπή της καμπύλης σε grid curve, το οποίο απαιτούσε σύμφωνα με τους υπολογισμούς μας αρκετό χρόνο (ώρες) και πλέον η δομή αναζήτησης δημιουργείτε σε μερικά δευτερόλεπτα.
- Η υλοποίηση του δυαδικού δέντρου που χρειάζεται για την Mean Discrete Frechet έχει γίνει με την χρήση array όπως φαίνεται στο παρακάτω σχήμα



- Στην περίπτωση του Assignment by Range search (LSH) και της Mean Discrete Frechet curve αντιστοιχούμε την νέα καμπύλη με την πιο κοντινή της στο ίδιο cluster. Με αυτό τον τρόπο γνωρίζουμε σε ποιο bucket του hashtable να εφαρμόσουμε το Range search (LSH).
- Στο input file η πρώτη γραμμή πάντα θα περιέχει την πληροφορία για το dimension.
- Για το μέγεθος του table κάθε hashtable έχουμε επιλέξει τον αριθμό των καμπυλών στο input file διά δεκαέξι.
- Στον παραδοτέο φάκελο *src* συμπεριλαμβάνεται ένα medium_dataset και ένα cluster.conf με βάση τα οποία ελέγξαμε τον κώδικα μας, για την ορθή λειτουργία του.

Valgrind – Memory leaks

Έγινε έλεγχος με valgrind για memory leaks και τα αποτελέσματα φαίνονται παρακάτω.

- Για την μετρικη συναρτηση Dynamic Time Warping (DTW) εχουμε:

```
==5321==
==5321== HEAP SUMMARY:
==5321==     in use at exit: 1,269,792 bytes in 50,568 blocks
==5321==   total heap usage: 3,592,139 allocs, 3,541,571 frees, 3,061,809,080 bytes allocated
==5321==
==5321== LEAK SUMMARY:
==5321==    definitely lost: 1,066,588 bytes in 42,453 blocks
==5321==    indirectly lost: 203,204 bytes in 8,115 blocks
==5321==      possibly lost: 0 bytes in 0 blocks
==5321==    still reachable: 0 bytes in 0 blocks
==5321==         suppressed: 0 bytes in 0 blocks
```

Παρατηρούμε ότι το συγκεντρωτικό memory leak του προγράμματος ανέρχεται στα 1.269.792 bytes από τα συνολικά 3.061.809.080 bytes, που αντιστοιχούν στο 4% της συνολικής μνήμης που χρησιμοποιήθηκε.

- Για την μετρικη συναρτηση Frechet εχουμε:

```
==4902==
==4902== HEAP SUMMARY:
==4902==     in use at exit: 13,005,204 bytes in 749,801 blocks
==4902==   total heap usage: 10,387,788 allocs, 9,637,987 frees, 15,798,389,264 bytes allocated
==4902==
==4902== LEAK SUMMARY:
==4902==    definitely lost: 10,891,224 bytes in 655,371 blocks
==4902==    indirectly lost: 2,113,964 bytes in 94,429 blocks
==4902==      possibly lost: 16 bytes in 1 blocks
==4902==    still reachable: 0 bytes in 0 blocks
==4902==         suppressed: 0 bytes in 0 blocks
```

Παρατηρούμε ότι το συγκεντρωτικό memory leak του προγράμματος ανέρχεται στα 13.005.204 bytes από τα συνολικά 15.798.389.264 bytes, που αντιστοιχούν στο 8% της συνολικής μνήμης που χρησιμοποιήθηκε.