Ανάπτυξη Λογισμικού για Δισεπίλυτα Αλγοριθμικά Προβλήματα

Εργασία 2η - Kmedoids Clustering

Αυτσίδης Σέργιος - sdi0900052
Μπορεκτσίογλου Ιωάννης - sdi1000111
Πατσουράκος Κωνσταντίνος - sdi0900129

Επεξήγηση Εργασίας

1. Γενικά

Το project υλοποιήθηκε κατά τα ζητούμενα. Υλοποιήθηκαν και δοκιμάστηκαν όλοι οι αλγόριθμοι που ζητούνται, καθώς και unit tests για κρίσιμες συναρτήσεις της υλοποίησης. Στις παρακάτω ενότητες παρέχεται μια αναλυτική περιγραφή της υλοποίησης που ακολουθήσαμε, καθώς και μια μελέτη - θεωρητική τεκμηρίωση στα αποτελέσματα των εκτελέσεων.

2. Κατάλογος Αρχείων - Ιεραρχία Φακέλων

- **./Data*.c - ./Data*.h**: Παρέχει τα αρχεία που υλοποιούν την διεπαφή με διάφορα datasets **ΕΚΤΟΣ** από τα **data.h data.c**
- **./hashtable.c-.h - ./list.c-.h:** Παρέχει την υλοποίηση των hashtables και της λίστας που χρησιμοποιήσαμε
- **./LSH.c - ./LSH.h:** Η υλοποίηση του LSH
- **./random.c - ./random.h:** Διεπαφή παραγωγής τυχαίων αριθμών
- **./data.h:** Περιέχει μία δομή που μέσω αυτής καθορίζεται το dataset κάθε φορά, με την εισαγωγή των κατάλληλων function pointers στα data members της. Ακόμη, περιέχει τις συναρτήσεις για την διεπαφή με τα δεδομένα, καθώς και ορισμένες συναρτήσεις που χρησιμοποιούν οι αλγόριθμοι clustering.

- **./data.c:** Οι υλοποιήσεις των παραπάνω, και η δομή που φιλοξενεί τα actual δεδομένα.
- **./initialization/:** Φάκελος που περιέχει τις υλοποιήσεις (*.c) και διεπαφές (*.h) των αλγορίθμων *Initialization*
- **./assignment/:**Φάκελος που περιέχει τις υλοποιήσεις (*.c) και διεπαφές (*.h) των αλγορίθμων *Assignment*
- **./update./:**Φάκελος που περιέχει τις υλοποιήσεις (*.c) και διεπαφές (*.h) των αλγορίθμων *Update*
- **./testRead.c:** Περιέχει τα unit tests που υλοποιήθηκαν.

3. Οδηγίες Μεταγλώττισης - Χρήσης

Για την ευκολότερη μεταγλώττιση του προγράμματος παρέχεται ένα αρχείο `makefile`. Η εντολή μεταγλώττισης συνεπώς είναι

```
% make recomendAPP
```

Για την εκτέλεση του προγράμματος αρκεί μια εντολή της παρακάτω μορφής

```
% ./recomend -d <input> -c <conf> -o <output>
```

,με:

- input: το αρχείο εισόδου, όπως περιγράφεται στην εκφώνηση
- conf: το αρχείο configuration, ομοίως
- output: το αρχείο εξόδου, όπου το πρόγραμμα παρέχει τις ζητούμενες εκτυπώσεις / πληροφορίες

Ακόμη, με την επιλογή --complete, γίνονται και οι εκτυπώσεις όλων των στοιχείων του κάθε cluster.

4. Παραδείγματα Εκτελέσεων - Μελέτη αλγορίθμων

Η απόδοση της συγκεκριμένης εφαρμογής μπορεί να αξιολογηθεί με βάση 2 παράγοντες:

1. **Ποιότητα clustering:** Το πρόγραμμα πρέπει να ομαδοποιεί σωστά τα δεδομένα. Ως μέτρο αξιοποίησης της ποιότητας του clustering υλοποιήθηκε ο δείκτης εσωτερικής ομαδοποίησης clustering **Silhouette**.
2. **Ταχύτητα εκτέλεσης:** Για να ελεγχθεί η ταχύτητα εκτέλεσης του αλγορίθμου, έγιναν μετρήσεις χρόνου στα 3 στάδιά του: *Initialization, Assignment, Update.*

Παρακάτω παρουσιάζονται μερικά ενδεικτικά αποτελέσματα εκτέλεσης του προγράμματως με διάφορους αλγόριθμους, διάφορα μεγέθη / είδη εισόδου και διαφορετικό πλήθος clusters. Τα αποτελέσματα παρουσιάζονται σε πλειάδες της μορφής

Σημείωση: Για το *Assignment* υλοποιήθηκαν και οι 2 αλγόριθμοι. Για καλύτερη ανάλυση όμως τα αποτελέσματα του *LSHAssign* παρουσιάζονται παρακάτω σε ξεχωριστή ενότητα.

Data_Euclidean_10_1000x100.csv:

Init	**Assign**	**Update**	**Sillhouette**	**Time**
KMedoids++	*Pam*	Lloyds	0.989695	2274.476ms / 19
	Pam	Clarans	0.989695	1718.613ms / 11
Concentrate	*Pam*	Lloyds	0.989695	2495.589ms / 18
	Pam	Clarans	0.989696	2146.137ms / 15

Data_Euclidean_15_1000x500.csv:

Init	**Assign**	**Update**	**Sillhouette**	**Time**
KMedoids++	*Pam*	Lloyds	0.999620	11028.430ms / 16
	Pam	Clarans	0.999620	9928.271ms / 14
Concentrate	*Pam*	Lloyds	0.999620	9998.128ms / 14
	Pam	Clarans	0.999620	10596.317ms / 14

Data_Hamming_5_1000x64.csv:

Init	Assign	Update	Sillhouette	Time
KMedoids++	*Pam*	Lloyds	0.596306	40.916ms / 3
	Pam	Clarans	0.596345	106.874ms / 7
Concentrate	*Pam*	Lloyds	0.596306	70.443ms / 3
	Pam	Clarans	0.596412	150.340ms / 8

Data_Hamming_15_10000x64.csv:

Init	Assign	Update	Sillhouette	Time
KMedoids++	*Pam*	Lloyds	0.605929	730.526ms / 5
	Pam	Clarans	0.605961	80.739ms / 4
Concentrate	*Pam*	Lloyds	0.605929	341.504ms / 4
	Pam	Clarans	0.605949	600886ms / 5

*Σχετικά με τον **LSH:***

Παρατηρήσαμε ότι το *LSHAssign* δεν πετυχαίνει τόσο καλή ποιότητα clustering (ο Silhouette συνήθως μας δίνει τιμές κοντά στο 0.6 που δεν συγκρίνονται με τις αντίστοιχες που παίρνουμε με τον *Pam*, οι οποίες σε κάποια datasets αγγίζουν και το 0.98). Αυτό συμβαίνει επειδή ο LSH είναι πιθανοτικός αλγόριθμος και δεν είναι πάντα ο πιο ακριβής. Επιπρόσθετα, οι τιμές των L, k, ακόμα και το πλήθος των buckets του κάθε hashtable του LSH παίζουν πολύ σημαντικό ρόλο στην ποιότητα του clustering που μπορούμε να πετύχουμε με τον LSH, και μικρές διακυμάνσεις μπορεί να οδηγήσουν σε πολύ κακές τιμές (ακόμη και αρνητικές). Επίσης, ο LSH αντιμετωπίζει ορισμένα προβλήματα με το assignment του 2ου καλύτερου medoid, καθώς χρειάζεται να βρει ένα αρκετά μακρινό σημείο για το οποίο δεν υπάρχει καμμία εγγύηση ότι θα βρίσκεται στο ίδιο bucket. Η υλοποίησή μας εκμεταλλεύεται τις αναζητήσεις με πολλαπλό radius για να αποφύγει την γραμμική προσπέλαση και assignment όσο το δυνατόν και περισσότερων σημείων, όμως πάντα υπάρχουν σημεία τα οποία πρέπει να γίνουν assign manually μετά το πέρας του *LSHAssign.*

Data_Hamming_5_1000x64.csv:

Init	Assign	Update	Sillhouette	Time
KMedoids++	*LSHAssign*	Lloyds	0.246395	44.663ms / 3
	LSHAssign	Clarans	0.583713	123.252ms / 7
Concentrate	*LSHAssign*	Lloyds	0.576983	98.107ms / 3
	LSHAssign	Clarans	0.583718	107.921ms / 8