Ανάπτυξη Λογισμικού για Αλγοριθμικά Προβλήματα: Project 2

Σταματόπουλος Βασίλειος 1115201400188

3/12/2018

1 Εισαγωγή

Στην εργασία αυτή υλοποιήθηκαν 12 διαφορετικοί τρόποι clustering, όπως ακριβώς ζητήθηκε και στην εκφώνηση.
Δηλαδή, **2 τρόποι αρχικοποίησης των δεδομένων,** τυχαία και με k-means++ 2 τρόποι update, **Basic Update με μέσα και Partitioning Around Medoids.** Επιπλεόν, το assignment των σημείων γίνεται με τρεις τρόπους. **Normal k-means, Assignment with LSH και assignment with Hypercube.**

2 Αρχέια

Ο φάκελος περιέχει τα κάτωθι αρχεία στα οποία έγινε η υλοποίηση του κώδικα.

1. **cluster.c:** Περιέχει την main function στην οποία με τη σειρά διαβάζουμε τα arguments. ύστερα διαβάζουμε το αρχείο εισόδου και στη συνέχεια αρχικοποιούμε τις μεταβλητές που θα χρειαστούμε. Δημιουργείται ένας πίνακας point* data, ο οποίος περιέχει στοιχεία struct point που έχουν την παρακάτω μορφή

```
typedef struct point{
long int id;
double * coordinates;
long long int ** g_functions;
int centroid_id;
int centroid2_id;
double dist; //distance from centroids
double dist_as_centroid;
struct point * next;
double silhouette;
} *point;
```

Πάνω σε αυτά τα στοιχεία κάνουμε την διαδικασία της συσταδοποίησης.Αρχικοποιούμε τα κέντρα που είναι τύπου struct centroid με την ακόλουθη μορφή

```
typedef struct centroid{
long int id;
int count;
int prev_count;
double dist;
long long int ** g_functions;
double * coordinates;
double silhouette_of_cluster;
point* assigned_points;
} *centroid;
```

Αφού τα αρχικοποιήσουμε με την εκάστοτε μέθοδο, κάλουμε την συνάρτηση **k-means**η οποία είναι αυτή που κάνει την δουλεία καλώντας στην πορεία τις απαραίτητες συναρτήσεις που βρίσκονται στο αρχείο functions.c/functions.h Κατά τη διάρκεια όλης αυτής της διαδικασίας εκτυπώνονται τα κατάλληλα στοιχεία στο αρχείο εξόδου, με τον ίδιο τρόπο που περιγράφεται και στην εκφώνηση.

2. **functions.c:** Στο αρχείο αυτό βρίσκουμε όλες τις συναρτήσεις που χρησιμοποιεί ο k-means,με μια μεγάλη πλειοψηφία εξάυτών να προέρχονται από τον φάκελο της πρώτης εργασίας. Λόγω του μεγάλου όγκου του αρχείου, οι συναρτήσεις δεν περιγράφονται εδώ, αλλά έχουν τα απαραίτητα σχόλια μέσα στο ίδιο το αρχείο.

3. **hashtable.c hashtable.h** Τα αρχεία αυτά περιέχουν τους τύπους των στοιχείων που χρησιμοποιήθηαν καθώς και τις απαραίτητες συναρτήσεις για να λειτουργήσει ένας πίνακας κατακερματισμού, όπως οιinsert, print. Όπως τα δύο προηγούμενα, είναι κατάλληλα σχολιασμένα για τρίτους.

3 Μεταγλώττιση

Το πρόγραμμα χρησιμοποιεί makefile, συνεπώς μπορεί να μεταγλωττιστεί με την εντολή make.

4 Εκτέλεση

Η εκτέλεση του αρχείου ακολουθεί το πρότυπο που δώθηκε στην εκφώνηση

5 Version Control

Για το πρόγραμμα χρησιμοποιήθηκε το github για να γίνει το απαραίτητο version control.[1]

6 Αποτελέσματα

Παρακάτω δίνονται τα αρχεία αποτελεσμάτων για τις default τιμές.

6.1 Euclidean Distance

Algorithm:Initialization:Random Update:Basic Assignment:k-means CLUSTER: 0 SIZE: 4460

CLUSTER: 1 SIZE: 47

CLUSTER: 2 SIZE: 261

CLUSTER: 3 SIZE: 185

CLUSTER: 4 SIZE: 46 Clustering Time:0.702846 Shillouette of cluster 0 is: 0.000000 Shillouette of cluster 1 is: 0.991446 Shillouette of cluster 2 is: 0.946417 Shillouette of cluster 3 is: 0.970529 Shillouette of cluster 4 is: 0.981373 Algorithm:Initialization:Random Update:Basic Assignment:LSH CLUSTER: 0 SIZE: 469

CLUSTER: 1 SIZE: 3824

CLUSTER: 2 SIZE: 205

CLUSTER: 3 SIZE: 349

CLUSTER: 4 SIZE: 152

Clustering Time:7.101877 Shillouette of cluster 0 is: 0.000000 Shillouette of cluster 1 is: -0.848569 Shillouette of cluster 2 is: 0.960562 Shillouette of cluster 3 is: 0.815106 Shillouette of cluster 4 is: 0.919506 Algorithm:Initialization:Random Update:Basic Assignment:Hypercube CLUSTER: 0 SIZE: 2876

CLUSTER: 1 SIZE: 454

CLUSTER: 2 SIZE: 435

CLUSTER: 3 SIZE: 246

CLUSTER: 4 SIZE: 988

Clustering Time:1.392212 Shillouette of cluster 0 is: 0.000000 Shillouette of cluster 1 is: 0.848284 Shillouette of cluster 2 is: 0.777253 Shillouette of cluster 3 is: 0.916437 Shillouette of cluster 4 is: 0.588906 Algorithm:Initialization:Random Update:Partitioning Around Medoids Assignment:k-means CLUSTER: 0 SIZE: 152

CLUSTER: 1 SIZE: 202

CLUSTER: 2 SIZE: 4220

CLUSTER: 3 SIZE: 106

CLUSTER: 4 SIZE: 319

Clustering Time:103.451557 Shillouette of cluster 0 is: 0.000000 Shillouette of cluster 1 is: -0.090521 Shillouette of cluster 2 is: -0.962072 Shillouette of clus-

ter 3 is: 0.971822 Shillouette of cluster 4 is: 0.871848 Algorithm:Initialization:Random Update:Partitioning Around Medoids Assignment:LSH CLUSTER: 0 SIZE: 231

CLUSTER: 1 SIZE: 200

CLUSTER: 2 SIZE: 101

CLUSTER: 3 SIZE: 152

CLUSTER: 4 SIZE: 4315

Algorithm:Initialization:Random Update:Basic Assignment:Hypercube CLUSTER: 0 SIZE: 4460

CLUSTER: 1 SIZE: 47

CLUSTER: 2 SIZE: 261

CLUSTER: 3 SIZE: 185

CLUSTER: 4 SIZE: 46 Clustering Time:0.877376 Shillouette of cluster 0 is: 0.000000 Shillouette of cluster 1 is: 0.991446 Shillouette of cluster 2 is: 0.946417 Shillouette of cluster 3 is: 0.970529 Shillouette of cluster 4 is: 0.981373 Algorithm:Initialization:Random Update:Partitioning Around Medoids Assignment:Hypercube CLUSTER: 0 SIZE: 115

CLUSTER: 1 SIZE: 199

CLUSTER: 2 SIZE: 152

CLUSTER: 3 SIZE: 4400

CLUSTER: 4 SIZE: 133

Clustering Time:86.376880 Shillouette of cluster 0 is: 0.000000 Shillouette of cluster 1 is: -0.326268 Shillouette of cluster 2 is: -0.157874 Shillouette of cluster 3 is: -0.964690 Shillouette of cluster 4 is: 0.949213 Algorithm:Initialization:K-Means++ Update:Basic Assignment:Hypercube CLUSTER: 0 SIZE: 940

CLUSTER: 1 SIZE: 2972

CLUSTER: 2 SIZE: 442

CLUSTER: 3 SIZE: 204

CLUSTER: 4 SIZE: 441

Clustering Time:1.463766 Shillouette of cluster 0 is: 0.000000 Shillouette of cluster 1 is: -0.668539 Shillouette of cluster 2 is: 0.812948 Shillouette of cluster 3 is: 0.875574 Shillouette of cluster 4 is: 0.733121 Algorithm:Initialization:K-Means++ Update:Partitioning Around Medoids Assignment:Hypercube CLUSTER: 0 SIZE: 62

CLUSTER: 1 SIZE: 44

CLUSTER: 2 SIZE: 28

CLUSTER: 3 SIZE: 7

CLUSTER: 4 SIZE: 4858

Clustering Time:129.740608 Shillouette of cluster 0 is: 0.000000 Shillouette of cluster 1 is: 0.505922 Shillouette of cluster 2 is: 0.789383 Shillouette of cluster 3 is: 0.920084 Shillouette of cluster 4 is: -0.987679

6.2 Πορίσματα

Για κάποιον λόγο που δεν κατάφερα να εντοπίσω το cluster 0 πάντα είχε μηδενικό shillouette. Παρ΄όλα αυτά, ο αλγόριθμος έδωσε αρκετά καλά και αναμενόμενα αποτελέσματα. Αξιοσημείωτα είναι τα εξής:

1. Αρχικά παρατηρούμε πως όλοι οι αλγόριθμοι συσταδοποίησαν τα δεδομένα παρομοίως, φτίαχνωντας μια πολύ μεγάλη συστάδα και 4 μικρότερες.

2. Σε γενικές γραμμές, οι σιλουέτες των αλγορίθμων είναι αρκετά καλές πλην μερικών περιπτώσεων. Παράξενο είναι πως το πιο απλό version του αλγορίθμου έδωσε τα καλύτερα αποτελέσματα, τόσο σε χρόνο αλλά και σε σιλουέτες.

   Tested on Linux Ubunut 18.03

References

[1] Vasileios Stamatopoulos, Github
`https://github.com/billstam12/Project/tree/master/2`