Πυθαρούλιος Κωνσταντίνος
sdi1600142

Ιγιάμου Περισανίδης Σίμων
sdi1600051

Project 2019-2020

Εργασία 2

Το πρόγραμμα αναπτύχθηκε στα πλαίσια του μαθήματος Ανάπτυξη Λογισμικού για Αλγοριθμικά Προβλήματα 2019-2020 για το Τμήμα Πληροφορικής και Τηλεπικοινωνιών του ΕΚΠΑ.

Αρχεία

Φάκελοι

- **include:** Περιέχει τα αρχεία επικεφαλίδων τα οποία περιέχουν ορισμούς κλάσεων και δηλώσεις συναρτήσεων
- **src:** Περιέχει όλα τα .cpp αρχεία, για την υλοποίηση των κλάσεων και των συναρτήσεων.
- **src/ANN:** Περιέχει της εργασίας 1 που επαναχρησιμοποιήθηκε στην παρούσα εργασία

Αρχεία.

- **Point.hpp, Point.cpp:** Ορισμός και υλοποίηση της κλάσης Point, για την αναπαράσταση των σημείων.
- **Curve.hpp, Curve.cpp:** Ορισμός και υλοποίηση της κλάσης Curve, για την αναπαράσταση των καμπυλών.
- **Clustering.hpp, Clustering.cpp:** Ορισμός και υλοποίηση της κλάσης Clustering και Cluster για την υλοποίηση του αλγορίθμου K-Means.
- **dist.hpp, dist.cpp:** Συναρτήσεις υπολογισμού απόστασης μεταξύ σημείων και μεταξύ καμπυλών.
- **util.hpp, util.cpp:** Βοηθητικές συναρτήσεις.
- **Makefile**

Compilation και Εκτέλεση

Η μεταγλώττιση του προγράμματος μπορεί να πραγματοποιηθεί με την εντολή

`make`,

καθώς υπάρχει ένα Makefile. Να σημειωθεί ότι χρησιμοποιείται το flag –O3 παντού. Το εκτελέσιμο που δημιουργείται ονομάζεται cluster. Τα ορίσματα του είναι τα εξής (τα bold είναι **υποχρεωτικά**):

/cluster –i [inputFile] –o [outputFile] –c [configFile] –complete

Παράδειγμα:
./cluster –i vectors_dataset_small.csv –o outputfile.txt –c cluster.conf

Περιγραφή Κλάσεων

- **Point**
  Αντιπροσωπεύει ένα σημείο.
- **Curve**
  Αντιπροσωπεύει μία καμπύλη. Αποτελείται κυρίως από ένα vector από Points.
- **Cluster**
  Αναπαράσταση των cluster για τον αλγόριθμο συσταδοποίησης.
- **Clustering**
  Ο σκοπός αυτής της κλάσης είναι να αποθηκεύει τα απαραίτητα δεδομένα και συναρτήσεις για την εκτέλεση του αλγορίθμου K-Means. Αποτελείται κυρίως από: το vector dataset στο οποίο αποθηκεύονται τα points/curves που διαβάζονται από το αρχείο εισόδου, το vector από points clusters με τα clusters. Η συνάρτηση KMeans() είναι η κύρια συνάρτηση του αλγορίθμου. Μέσω αυτής καλούνται οι υπόλοιπε(init,assign,update).

Περιγραφή τρόπου προσέγγισης

Υπάρχουν επεξηγηματικά σχόλια σε όλη την έκταση του κώδικα, για να είναι εύκολη η ανάγνωση του. Ως αποτέλεσμα οι λεπτομέρειες την υλοποίησης είναι προφανείς από αυτά. Ωστόσο στη συνέχεια θα περιγράψουμε την γενική ιδέα.

Για την αποφυγή επανάληψης κώδικα για points και curves, έχουμε χρησιμοποιήσει void pointers. Έτσι, οι συναρτήσεις που είναι παρόμοιες και για τους δυο τύπους αντικειμένων έχουν υλοποιηθεί μια φορά, χρησιμοποιώντας casts όπου είναι απαραίτητο.

Συγκρίσεις

Τρέχουμε το πρόγραμμα μας για Points και για Curves, χρησιμοποιώντας ένα μικρό και ένα μεγάλο dataset και δοκιμάζοντας δυο αλγορίθμους για να συγκρίνουμε τις επιδόσεις τους. Οι αλγόριθμοι που δοκιμάζουμε είναι ο απλός Lloyd, και αυτός τις βελτιωμένες μεθόδους(K-means++, Reverse assignment, Mean centroid). Οι μετρήσεις έχουν γίνει με το εξής conf file:

Για Points:

Απλός Lloyd's, με μικρή είσοδο (DataVectors_5_500x100.csv)

```
Algorithm:  Init:random Assignment:lloyd Update:pam

CLUSTER-0 {size: 95, centroid: item261}

CLUSTER-1 {size: 237, centroid: item15}

CLUSTER-2 {size: 124, centroid: item393}

clustering_time: 0.052423 seconds

Silhouette: si:[0.910525, 0.263331, 0.515986] stotal: 0.56328
```

Απλός Lloyd's, με μεγάλη είσοδο (DataVectors_5_10000x100.csv)

```
Algorithm:  Init:random Assignment:lloyd Update:pam

CLUSTER-0 {size: 1987, centroid: item7673}

CLUSTER-1 {size: 5962, centroid: item1014}

CLUSTER-2 {size: 1984, centroid: item4815}

clustering_time: 31.5666 seconds

Silhouette: si:[0.91366, 0.344782, 0.914346] stotal: 0.724262
```

Βελτιωμένος, με μικρή είσοδο (DataVectors_5_500x100.csv)

```
Algorithm:  Init:k-means++ Assignment:reverse Update:mean

CLUSTER-0 {size: 91, centroid: …}

CLUSTER-1 {size: 273, centroid…}

CLUSTER-2 {size: 92, centroid: …}

clustering_time: 0.040219 seconds

Silhouette: si:[0.914266, 0.325889, 0.90811] stotal: 0.716089
```

Βελτιωμένος, με μεγάλη είσοδο (DataVectors_5_10000x100.csv)

```
Algorithm:  Init:k-means++ Assignment:reverse Update:mean

CLUSTER-0 {size: 1984, centroid: …}

CLUSTER-1 {size: 3974, centroid: …}

CLUSTER-2 {size: 3975, centroid: …}

clustering_time: 0.887571 seconds

Silhouette: si:[0.912918, 0.396323, 0.456672] stotal: 0.588638
```

Για Curves:

Απλός Lloyd's, με μικρή είσοδο (trajectories_dataset_small.csv)

```
Algorithm:  Init:random Assignment:lloyd Update:pam

CLUSTER-0 {size: 10, centroid: 7}

CLUSTER-1 {size: 45, centroid: 71}

CLUSTER-2 {size: 45, centroid: 43}

clustering_time: 1.34384 seconds

Silhouette: si:[0.921739, 0.127443, 0.781479] stotal: 0.61022
```

Βελτιωμένος, με μικρή είσοδο (trajectories_dataset_small.csv)

```
Algorithm:  Init:k-means++ Assignment:reverse Update:mean

CLUSTER-0 {size: 1984, centroid: …}

CLUSTER-1 {size: 3974, centroid: …}

CLUSTER-2 {size: 3975, centroid: …}

clustering_time: 0.887571 seconds

Silhouette: si:[0.912918, 0.396323, 0.456672] stotal: 0.588638
```

Παρατηρούμε ότι και για Points και για Curves, ο βελτιωμένος αλγόριθμος έχει δραστικά καλύτερο χρόνο. Βλέπουμε ότι στο μεγάλο dataset, ο απλός αλγόριθμος για Points χρειάστηκε 31.567 δευτερόλεπτα, ενώ ο βελτιωμένος μόλις 0.887 δευτερόλεπτα . Ωστόσο, στο ιδιο παράδειγμα, η αποτελεσματικότητα του βελτιωμένου αλγορίθμου είναι μειωμένη(stotal=0.588) σε σύγκριση με τον απλό Lloyd’s (stotal=0.724).