

Readme

Σκαρλάτος Αντώνης 1115201400184

Τρουλλινός Ιωάννης 1115201400203

Compile

To compile run **make**

Run

`./run -d <input> -t <type> -f <distance function>`

- -f το input που θα διαβαστεί.
- -t το type του input όπου αν είναι 1 είναι για τη πρωτεΐνες και αν είναι 2 για τους δρόμους και αν είναι 3 δημιουργεί το segment.csv
- -d η συνάρτηση που θα υπολογίζουμε αποστάσεις .

Παραδείγματα χρήσης :

```
./run -t 1 -f input.dat
```

```
./run -t 2 -f segment.csv -d DFT
```

Implementation Notes

Έχουμε υλοποιήσει όλα τα ερωτήματα της άσκησης. Για να τρέξουμε το πρόγραμμα με το αρχείο athens.csv πρέπει να το τρέξουμε πρώτα με `type = 3` (δηλαδή το flag `-t 3`) για να δημιουργηθεί το αρχείο segment.csv και στη συνέχεια τρέχουμε το πρόγραμμα με `type = 2` για να εκτελεστούν οι συναρτήσεις.

Το πρόγραμμα χωρίζετε σε μια `main`, σε αρχεία `.cpp`, που υλοποιούνται οι συναρτήσεις και σε `header files` που γίνονται οι δηλώσεις.

- `main` καλούνται οι συναρτήσεις για τη υλοποίηση του προγράμματος.
- `curve.cpp` υλοποίηση των συναρτήσεων της `curve`
- `distances` υλοποιούνται οι αλγόριθμοι απόστασης.
- `file_functions` συναρτήσεις για `input output`
- `hash_functions` συναρτήσεις για το `hashing` του `grid`
- `hashtable` συναρτήσεις της κλάσης του `hashtable`
- `help_functions` βοηθητικές συναρτήσεις
- `list` συναρτήσεις για την λίστα του `hashtable`

Σαν συμπεράσματα παρατηρούμε στο `lsh` ότι όσο μεγαλώνουμε το `k` έχουμε περισσότερα πλέγματα και δημιουργούνται περισσότερες συστάδες και έτσι έχουμε αποτέλεσμα με μεγαλύτερη ακρίβεια και καλύτερη `silhouette`. Η `silhouette` μας βγήκε πάνω από 0,2.