Κ23γ: Ανάπτυξη Λογισμικού για Αλγοριθμικά Προβλήματα Χειμερινό εξάμηνο 2016-17

2η Προγραμματιστική Εργασία Υλοποίηση του αλγορίθμου συσταδοποίησης K-medoids στη γλώσσα C++

Ομάδα :

- Φελιμέγκα Αγγελική 1115201300192 – sdi1300192@di.uoa.gr
- Βούρου Παγώνα 1115201300254 – sdi1300254@di.uoa.gr

Τίτλος Και Περιγραφή του Προγράμματος:

Το πρόγραμμα ονομάζεται ¨proj – phase2" και έχει υλοποιηθεί σε γλώσσα c++.

Το πρόγραμμα διαβάζει σημεία από ενα αρχείο και τα χωρίζει σε k Clusters, ανάλογα με την απόσταση τους από το κέντρο του κάθε Cluster. Γίνεται επαναπολογισμός των κέντρων για τον καλύτερο διαχωρισμό των σημείων, και έπειτα επαναδιαχώριση των στοιχείων σε κάθε Cluster. Αυτο, πραγματοποιείται μέχρι να έχουν μοιραστεί τα σημεία στα κέντρα με τα οποία έχουν την μικρότερη απόσταση. Τέλος γίνεται έλεγχος για το πόσο καλά έχει γίνει ο διαμοιρασμός των σημείων. Τα αποτελέσματα εκτυπώνονται σε αρχείο εξόδου.

Κατάλογος Αρχείων – Περιγραφή αρχείων :

- main.cpp : Στη main το πρόγραμμα διαβάζει ένα αρχείο και ανάλογα με το είδος των σημείων τα βάζει στους αντίστοιχους πίνακες (hamming, cosine, euclidean, distancematrix). Υπάρχει ένα μενού επιλογής απο τον χρήστη , του αλγορίθμου που θέλει να τρέξει για κάθε μέρος του προγράμματος: Initialize , Assignment και Update, αλλά και επιλογή του ξεχωριστού αλγορίθμου CLARA. Αφού τερματίσουν οι αλγόριθμοι που επιλέχθηκαν, καλείται η Silhouette και τα αποτέλεσματα εκτυπώνονται σε αρχείο.

- DataPoints.cpp : Το αρχείο DataPoints.cpp περιέχει τις υλοποιήσεις των συναρτήσεων : -CreateHammingPoints, που πέρνει τα σημεία απο το αρχείο Hamming και τα βάζει στον πίνακα hamming, -CreateEuclideanPoints, που πέρνει τα σημεία απο το αρχείο Euclidean και τα τοποθετεί στον πίνακα euclidean, -CreateCosinePoints, που πέρνει τα σημεία απο το αρχείο Cosine και τα τοποθετεί στον πίνακα cosine και της - CreateMatrixPoints η οποία πέρνει τα σημεία απο το αρχείο distancematrix και τα τοποθετεί στον πίνακα distancematrix.

- DataPoints.h : Δήλωση των συναρτήσεων CreateHammingPoints, CreateEuclideanPoints, CreateCosinePoints, και CreateMatrixPoints.

- Cluster.cpp : Υλοποίηση κλάσης Cluster για την αποτύπωση και αποθήκευση των κέντρων των Cluster αλλά και των ίδιων των Cluster. Επίσης υλοποίηση συναρτήσεων εισαγωγής σημείων για κάθε είδος στημείου ξεχωριστά, ανανέωσης κέντρου του Cluster για κάθε είδος σημείου ξεχωριστά, ανανέωσης απόστασης σημείου από κέντρο. Σημείου απο σημείο, διαγραφής σημείου και εκτύπωση σημείων για κάθε είδος σημείων ξεχωριστά.

- Cluster.h : Δήλωση των συναρτήσεων που υλοποιούνται στο αρχείο Cluster.cpp. Ορισμός και υλοποίηση συναρτήσεων καταστροφής, καθώς και συνάρτησης λήψης κέντρου από το κάθε Cluster ανάλογα με το είδος των σημείων.

- K_medoids.cpp : Υλοποίηση του αλγορίθμου k-medoids++. Δημιουργία και υλοποίηση της συνάρτησης "k_medoidspp" . Ανάλογα με το είδος των σημείων , αρχικοποιείται τυχαία το πρώτο κέντρο και έπειτα σύμφωνα με τον αλγόριθμο και τις αποστάσεις, γίνεται δημιουργία και των υπόλοιπων κέντρων.

- K_medoids.h : Δήλωση της συνάρτησης "k_medoidspp".

- Park_Jun.cpp : Υλοποίηση του αλγορίθμου Park-Jun για την αρχικοποίηση των κέντρων. Δημιουργία και υλοποίηση της συνάρτησης "ParkJun".

- Park_Jun.h : Δήλωση της συνάρτησης ParkJun.

- PAM.cpp : Υλοποίηση του αλγορίθμου PAM για τον διαμερισμό των σημείων στα κέντρα των Clusters. Μέσα στο αρχείο PAM.cpp υπάρχουν δύο συναρτήσεις. Η συνάρτηση PAM για τον δηιαμερισμό των σημείων σε άδεια Clusters ανάλογα με το είδος των σημείων και την απόσταση τους από το κέντρο του κάθε Cluster, και η συνάρτηση PAM_Update , η οποία μοιράζει τα σημεία σε γεμάτα Cluster τα οποία είτε άλλαξαν κέντρο, είτε όχι. Γίνεται έλεγχος για τα σημεία και ανάλογα γίνεται ο διαμερισμός.

- PAM.h : Δήλωση των συναρτήσεων PAM και PAM_Update.

- LSH.cpp : Υλοποίηση του αλγορίθμου LSH για διαμερισμό των σημείων στα στα κέντρα των Clusters. Μέσα στο αρχείο LSH.cpp γίνεται η αρχικοποίηση των Hashtables, η υλοποίηση της συνάρτησης LSH για τον διαμερισμό των σημείων, από τα Hashtables, που τα έχουμε εισάγει στα κέντρα των Clusters και η υλοποίηση της συνάρτησης QuerySearch που βρίσκει τα κκοντινότερα σημεία με βάση την ακτίνα – απόσταση τους απο τα κέντρα. Στη συνάρτηση LSH περνάμε μια μεταβλητή choice , η οποία όταν είναι 4 (στον Clarans μετα την δεύτερη επανάληψη) , απλά κάνει πάλι διαμερισμό τον σημείων στα ήδη γεμάτα Clusters λόγω αλλαγής (ή μη) των κέντρων.

- LSH.h : Δήλωση την συνάρτησης LSH.

- alaLloyds.cpp : Υλοποίηση του αλγορίθμου a La Loyd's για την ανανέωση των κέντρων των Clusters και των σημείων που ανήκει σε κάθε κέντρο. Δημιουργία συνάρτησης alaLloyds. Αφού υπολογιστούν τα καινούρια κέντρα σύμφωνα με τον αλγόριθμο a La Loyd's καλούμε την PAM_Update για να μοιράσουμε τα σημεία στα καίνουρια κέντρα (εάν δεν υπάρχουν ήδη) .

- alaLloyds.h : Δήλωση της συνάρτησης alaLloyds.

- Clarans.cpp : Υλοποίηση του αλγορίθμου Clarans. Δημιουργία συνάρτησης Clarans. Ανάλογα με την επιλογή του χρήστη απο το μενού, η Clarans χρησιμοποιεί για τον διαμερισμό των σημείων, είτε τον PAM στα άδεια αρχικά Clusters και τον PAM_Update έπειτα, είτε τον LSH στα άδεια αρχικά Clusters και τον LSH με τη μεταβλητή choice = 4 έπειτα, για την ανανέωση των σημείων στα καινούρια κέντρα.

- Clarans.h : Δήλωση της συνάρτησης Clarans.h.

- Distances.cpp : Το αρχείο Distances.cpp περιέχει τις συναρτήσεις : DistanceHamming, DistanceCosine, DistanceEuclidean,για τον υπολογισμό της απόστασης κάθε έιδους σημείου έιτε από κέντρο, έιτε από σημείο, και την MinDistance, για τον υπολογισμό της μικρότερης απόστασης μεταξύ αποστάσεων.

- Distances.h : Δηλωσή των συναρτήσεων DistanceHamming, DistanceCosine, DistanceEuclidean και MinDistance.

- CLARA.cpp : Υλοποίηση του αλγορίθμου CLARA. Ο αλγόριθμος CLARA έχει υλοποιηθεί ολόκληρος και για τα τρία μέρη του clustering (initialization, assignment, update)με μία συνάρτηση CLARA.

- CLARA.h : Δήλωση της συνάρτησης CLARA.

- Silhouette.cpp : Υλοποίηση του αλγορίθμου Silhouette. Δημιουργία συνάρτησης Silhouette. Η συνάρτηση Silhouette δίνει αριθμούς απο το -1 μεχρι το 1 , για αξιολόγηση των αλγορίθμων που προηγήθηκαν.

- Silhouette.h : Δήλωση της συνάρτησης Silhouette.

- Hashtable.cpp : Σε αυτό το αρχείο εκτός από τους constructors, τον destructor του πίνακα, και την συνάρτηση εκτύπωσης του, printTable(), υλοποιούνται και δύο άλλες συναρτήσεις, η InsertIntoHashtable() , και η SearchBucket(). Η InsertIntoHashtable() ανάλογα με τη μέθοδο και τη g , εισάγει τα στοιχεία στο κατάλληλο bucket. Έχει γίνει διαχωρισμός στον τρόπο εισαγωγής των στοιχείων από Hamming,Cosine και DistanceMatrix από την Euclidean. Η υλοποιήση αυτή έγινε διότι οι τρεις πρώτες επιστρέφουν g, ενώ η Euclidean επιστρέφει την συνάρτηση fi που αποτελείται απο τις g. Η SearchBucket() ανάλογα με τη μέθοδο, Hamming,Cosine και DistanceMatrix, ή Euclidean καλεί τις κατάλληλες συναρτήσεις για τον εντοπισμό των κοντινότερων γειτόνων ακτίνας και τον κοντινότερο γείτονα με την μικρότερη απόσταση.

- Hashtable.h : Αυτό το αρχείο επικεφαλίδας περιέχει την αρχικοποίηση και τον ορισμό του πίνακα κατακερματισμού. Ο πίνακας κατακερματισμού αποτελείται απο αντικείμενα την Συνδεδεμένης Λίστας (Linked List).

- LinkedList.cpp : Σε αυτό το αρχείο εκτός από τους constructors, τον destructor της λίστας και την συνάρτηση εκτύπωσης της ,printList(), υλοποιούνται και δύο άλλες συναρτήσεις, η Search() και η NN_Search(). Η Search ανάλογα με τη μέθοδο που το γράφει το αρχείο, και που της έχει στείλει η SearchBucket του πίνακα κατακερματισμού, ψάχνει και εκτυπώνει τους κοντινότερους γείτονες ακτίνας R. Η κάθε μέθοδος, ανάλογα με τον ορισμό της, έχει και διαφορετικο υπολογισμό των κοντινότερων γειτόνων. Η NN_Search(), αντίστοιχα, ανάλογα με τη μέθοδο που το γράφει το αρχείο, και που της έχει στείλει η SearchBucket του πίνακα κατακερματισμού, ψάχνει και επιστρέφει τον (έναν) κοντινότερο γείτονα, αυτόν με τη μικρότερη απόσταση.Η κάθε μέθοδος, ανάλογα με τον ορισμό της, έχει και διαφορετικο υπολογισμό του κοντινότερου γείτονα.

- LinkedList.h : Αυτό το αρχείο επικεφαλίδας περιέχει την αρχικοποίηση και τον ορισμό της συνδεδεμένης λίστας που χρησιμοποιείται για την υλοποίηση του πίνακα κατακερματισμού.

- CosineSim.cpp : Υλοποίηση της κλάσης CosineSim και των μεθόδων της. Η CosineSim λειτουργεί με βάση την ομοιότητα συνιμητόνου. Περιέχει τους constructors, και τον destructor της κλάσης καθώς και την μέθοδο ConstructGFunctionC(int k), για την δημιουργία της συνάρτησης g. Μέσα σε αυτή τη συνάρτηση υπολογίζεται το εσωτερικό γινόμενο ενός τυχαίου αριθμού με τις συντεταγμένες του αντικειμένου και ανάλογα ,αν το γινόμενο είναι θετικό ή αρνητικό δημιουργούμε μία συμβολοσειρά από παράθεση τιμών 0 και 1, η οποία και επιστρέφεται.

- CosineSim.h : Δήλωση/αρχικοποίηση της κλάσης CosineSim και των μεθόδων της.

- DistanceMatrix.cpp : Υλοποίηση της κλάση DistanceMatrix και των μεθόδων της. Περιέχει τους constructors, και τον destructor της κλάσης καθώς και την μέθοδο ConstructGFunctionC(int item,int k). Στον constructor της κλάσης αυτής διαβάζεται το αρχείο και εισάγωνται τα στοιχεία σε έναν πίνακα. Έπειτα στην συνάρτηση κατασκευής της g, με τις κατάλληλες μαθηματικές

πράξεις δημιουργείται το αποτέλεσμα της g (παράθεση τιμών 0 και1) και επιστρέφεται.

- DistanceMatrix.h : Δήλωση/αρχικοποίηση της κλάσης DistanceMatrix και των μεθόδων της.

- Euclidean.cpp :Υλοποίηση της κλάσης Euclidean και των μεθόδων της. Περιέχει τους constructors, και τον destructor της κλάσης καθώς και την μέθοδο ConstructFiFunctionC(int item,int k), όπου κατασκευάζεται η fi. Μέσα στη συνάρτηση αυτή, αφου αποθηκευτούν οι διαστάσεις του σημείου σε έναν πίνακα, και μετά απο συγκεκριμένες μαθηματικές πράξεις επιστρέφεται το αποτέλεσμα της fi, το οποίο είναι ένας ακέραιος αριθμός.

- Euclidean.h : Δήλωση/αρχικοποίηση της κλάσης Euclidean και των μεθόδων της.

- Hamming.cpp : Υλοποίηση της κλάσης Hamming και των μεθόδων της. Περιέχει τους constructors, και τον destructor της κλάσης καθώς και την μέθοδο ConstructGFunctionC(int item,int k), όπου κατασκευάζεται η g (συμβολοσειρά από παράθεση τιμών 0 και 1) και επιστρέφεται.

- Hamming.h : Δήλωση/αρχικοποίηση της κλάσης Hamming και των μεθόδων της.

- NeighbourSearch.cpp : Υλοποίηση των συναρτήσεων RangeNeighbourSearch και Nearest_Neighbor_Search.
  RangeNeighbourSearch : Ανάλογα με τη μέθοδο, καλείται η κατασκευάστρια της g, της οποίας το αποτέλεσμα, μαζί με την ακτίνα που διαβάζεται απο το αρχείο, στέλνεται σαν όρισμα στη Searchbucket για την έυρεση των κοντινότερων γειτόνων ακτίνας R.
  Nearest_Neighbor_Search : Ανάλογα με τη μέθοδο, καλείται η κατασκευάστρια της g, της οποίας το αποτέλεσμα, στέλνεται σαν όρισμα στη Searchbucket για την έυρεση του κοντινότερου γειτόνα , με τη μικρότερη απόσταση.

- NeighbourSearch.h : Δήλωση/ αρχικοποίηση των συναρτήσεων αναζήτησης του κοντινότερου γείτονα, RangeNeighbourSearch και Nearest_Neighbor_Search.

- - randomfunc.cpp : Υλοποίηση των συναρτήσεων marsagliarandom(int j) και mod (int a, int b).
  marsagliarandom(int j) : Γεννήτρια τυχαίων μεταβλητών με βάση την μέθοδο

marsaglian.
mod (int a, int b): Συνάρτηση υπολογισμού υπολοίπου.

- randomfunc.h : Δήλωση/ αρχικοποίηση των συναρτήσεων marsagliarandom(int j) και mod (int a, int b).

- makefile

Οδηγίες Χρήσης Προγράμματος :

Για την μεταγλώττιση : make all

Για την εκτέλεση : `$./medoids –d<input file> -c<configuration file> -o<output file>`

Έπειτα ζητείται από το χρήστη να επιλέξει ποιόν συνδιασμό αλγορίθμων θέλει.