Ανάπτυξη Εφαρμογών για Αλγοριθμικά Προβλήματα

2η εργασία – Χειμερινό εξάμηνο 2018-2019

Καθηγητής: κος Εμίρης Ιωάννης

Στοιχεία

Ονοματεπώνυμο: Μπούσιου Κυριακή

Αριθμός Μητρώου: 1115201400120

Email: sdi1400120@di.uoa.gr

Θέμα

Υλοποίηση αλγορίθμων για τη συσταδοποίηση διανυσμάτων στον χώρο $\mathfrak{R}$d.

Περιεχόμενα

Α) Ανάλυση αρχείων

cluster.cpp	Περιέχει τη main που καλεί όλες τις απαραίτητες συναρτήσεις
functions.cpp – functions.h	Περιέχει συναρτήσεις γενικού περιεχομένου και τα αρχεία επικεφαλίδας
files.cpp – files.h	Συναρτήσεις για διάβασμα και γράψιμο σε αρχεία
initialization.cpp – initialization.h	Υλοποίηση αλγορίθμων αρχικοποίησης των cluster
assignment.cpp – assignment.h	Υλοποίηση αλγορίθμων ανάθεσης στα cluster
update.cpp – update.h	Υλοποίηση αλγορίθμων για ενημέρωση των cluster με νέα
hash.cpp – hash.h	Συναρτήσεις από την $1^{η}$ εργασία για το hashing με συναρτήσεις lsh
distances.cpp – distances.h	Συναρτήσεις που έχουν να κάνουν με τον υπολογισμό αποστάσεων (όπως min απόσταση σημείου από πίνακα σημείων, απόσταση δύο σημείων ανάλογα με τη μετρική κλπ)
classes.h	Περιέχει όλες τις κλάσεις για τις δομές που χρησιμοποιήθηκαν (αναλύονται παρακάτω)
makefile	Για να μεταγλωττίζεται ο κώδικας με την εντολή make

Περιέχονται επίσης τα αρχεία twitter_dataset_small.csv, αρχείο εισόδου με το οποίο δούλευα κυρίως, cluster.conf, από το οποίο διαβάζονται τα δεδομένα για τον αριθμό των clusters, τον αριθμό των hash tables και τον αριθμό των hash functions, όπως περιγράφεται στην εκφώνηση, και το out.txt, αρχείο εξόδου που περιέχει τα αποτελέσματα για όλους τους συνδυασμούς των υλοποιημένων αλγορίθμων για τιμές number_of_clusters: 5, number_of_hash_functions: 4, number_of_hash_tables: 5.

Β) Σύντομη Επεξήγηση Κώδικα

Αρχικά, διαβάζονται τα αρχεία εισόδου και παραμέτρων και τα ορίσματα από τη γραμμή εντολών. Αποθηκεύεται σε κατάλληλη δομή το dataset και στη συνέχεια δημιουργούνται οι συναρτήσεις για hashing lsh ανάλογα με τη μετρική και αποθηκεύονται. Εδώ, χρησιμοποιήθηκαν οι συναρτήσεις από την $1^{η}$ εργασία σαν βιβλιοθήκες με κάποιες αλλαγές (πχ να δέχεται double διανύσματα). Αφού περαστούν όλα τα διανύσματα από τις hash functions και χωριστούν σε buckets, αρχίζει η διαδικασία της συσταδοποίησης.

Ο χρήστης ρωτάται αν θέλει να τρέξει κάποιον συγκεκριμένο συνδυασμό αλγορίθμων, αν θέλει να τρέξουν όλοι με τη σειρά ή να κάνει exit. Μέχρι το τέλος της διαδικασίας, όλα τα αποτελέσματα αποθηκεύονται σ' ένα string, το οποίο στο τέλος γράφεται στο αρχείο εξόδου που έχει δώσει ο χρήστης στην αρχή από την γραμμή εντολών.

1. **Initialization – Random Selection of k points (simplest):** Διαλέγονται με τη σειρά k τυχαία σημεία ως clusters, με έλεγχο (με while(1)) για το αν έχει ήδη επιλεχθεί κάποιο σημείο και με uniform κατανομή.

2. **Initialization – K-means++:** Παίρνουμε το πρώτο cluster τυχαία, όπως πριν, και με τον τύπο που υπάρχει στις διαφάνειες, δηλαδή παίρνοντας για κάθε σημείο την ελάχιστη απόσταση από τα ήδη υπάρχοντα cluster (υπάρχει κατάλληλη συνάρτηση που παίρνει έναν vector από σημεία και ένα μόνο σημείο και επιστρέφει το σημείο αυτό από τον vector που απέχει το λιγότερο από το ένα σημείο που δόθηκε, καθώς και την απόσταση αυτή), κρατάμε το max αυτών των τιμών, γιατί θα το χρησιμοποιήσουμε για normalization (για να μην γίνει overflow διαιρούμε με αυτήν την τιμή στο τέλος), και φτιάχνουμε έναν πίνακα στον οποίο κάθε φορά προσθέτουμε αυτήν την min απόσταση που βρίσκουμε. Μετά, διαλέγουμε με uniform έναν αριθμό από το 0 μέχρι το μέγιστο αυτού του πίνακα, δηλαδή τον αριθμό που βρίσκεται στην τελευταία θέση του, αφού κάθε φορά προσθέτουμε τον αριθμό που βρίσκουμε. Έτσι, με αυτόν τον αριθμό διαλέγουμε και το αντίστοιχο item από το dataset και το κάνουμε cluster και επαναλαμβάνουμε τη διαδικασία.

3. **Assignment – Lloyd's:** Για κάθε σημείο, βρίσκουμε την ελάχιστη απόσταση από τα clusters, με την προαναφερθείσα συνάρτηση, και το αναθέτουμε σε αυτό το cluster.

4. **Assignment – Range Search with LSH:** Για κάθε cluster κάνουμε range search με αρχική ακτίνα όπως στις διαφάνειες, την ελάχιστη απόσταση των cluster μεταξύ τους δια 2, που γίνεται πάλι με κατάλληλη συνάρτηση που βρίσκεται στο distances.cpp, και ξεκινάει το range search με κάθε φορά διπλασιαζόμενη ακτίνα το οποίο σταματάει όταν πια δεν επιστραφεί κανένα σημείο, επομένως τα υπόλοιπα σημεία αναθέτονται σε cluster με lloyd's. Κρατάμε ένα flag για κάθε διάνυσμα το οποίο αρχικοποιείται με 0 και αν αυτό το σημείο ανατεθεί σε cluster με range search lsh, βάζουμε και την τιμή του flag στην τιμή της επανάληψης στην οποία ανατέθηκε. Έτσι, αν ένα σημείο ανατεθεί δύο φορές στην ίδια επανάληψη σε διαφορετικά cluster κρατάμε αυτό με την μικρότερη απόσταση, ενώ αν ανατεθεί σε διαφορετικές επαναλήψεις κρατάμε το cluster της πρώτης επανάληψης.

5. **Update – K-means:** Για κάθε cluster, παίρνουμε όλα τα σημεία που έχουν ανατεθεί σε αυτό και βρίσκουμε τον γεωμετρικό μέσο τους, δηλαδή προσθέτουμε τις τιμές κάθε διάστασης και διαιρούμε με το πλήθος των σημείων, και έτσι δημιουργείται ένα νέο σημείο που δεν υπάρχει στο dataset πιθανώς και γίνεται αυτό το νέο cluster. Επιστρέφεται μία Boolean τιμή true, αν έχει αλλάξει το μέσο, false αν δεν έχει αλλάξει, ώστε να σταματήσει η επανάληψη assignment-update.
6. **Update – PAM improved like Lloyd's:** Για κάθε cluster, για κάθε σημείο μέσα στο cluster, υπολογίζουμε το άθροισμα όλων των αποστάσεων με όλα τα υπόλοιπα σημεία στο cluster και αυτό το σημείο που έχει την ελάχιστη θα γίνει το νέο μέσο. Αν αλλάξει το μέσο επιστρέφεται true, αν όχι false.

Γ) Δομές Δεδομένων

Η κλάση για τα διανύσματα ονομάζεται xclass και περιέχει τα στοιχεία id, δηλαδή έναν int που δείχνει σε ποιο σημείο του vector βρίσκεται, έναν vector με τις τιμές του διανύσματος, έναν int cluster_id, δηλαδή σε ποιο cluster ανήκει, που αρχικοποιείται με -1, και είναι η θέση του cluster στον vector με τα cluster, ένα flag όπως έχει εξηγηθεί παραπάνω που αρχικοποιείται με 0 κι έναν vector από int που λέει σε κάθε hash table σε ποιο bucket ανήκει. Υπάρχει ο constractor και οι αντίστοιχοι setters και getters.

Δομές όπως στην πρώτη εργασία για την αποθήκευση των ιδιοτήτων των hash functions για κάθε μετρική.

Δ) Γενικές Παρατηρήσεις

1. Έχουν χρησιμοποιηθεί εξωτερικές μεταβλητές σε κάποιες αναγκαίες περιπτώσεις, όπως για την δομή που έχουμε αποθηκεύσει τα item που χρειάζεται σε όλες σχεδόν τις συναρτήσεις ή άλλες μεταβλητές που πρέπει να περαστούν ως ορίσματα σε μεγάλο βάθος.

2. Έχει γίνει πολύ καλός διαμοιρασμός των συναρτήσεων σε αρχεία ανάλογα με τη λειτουργία τους, ώστε να είναι ευανάγνωστος και ευνόητος ο κώδικας, και υπάρχει επαρκής σχολιασμός σε όλα τα αρχεία κώδικα.

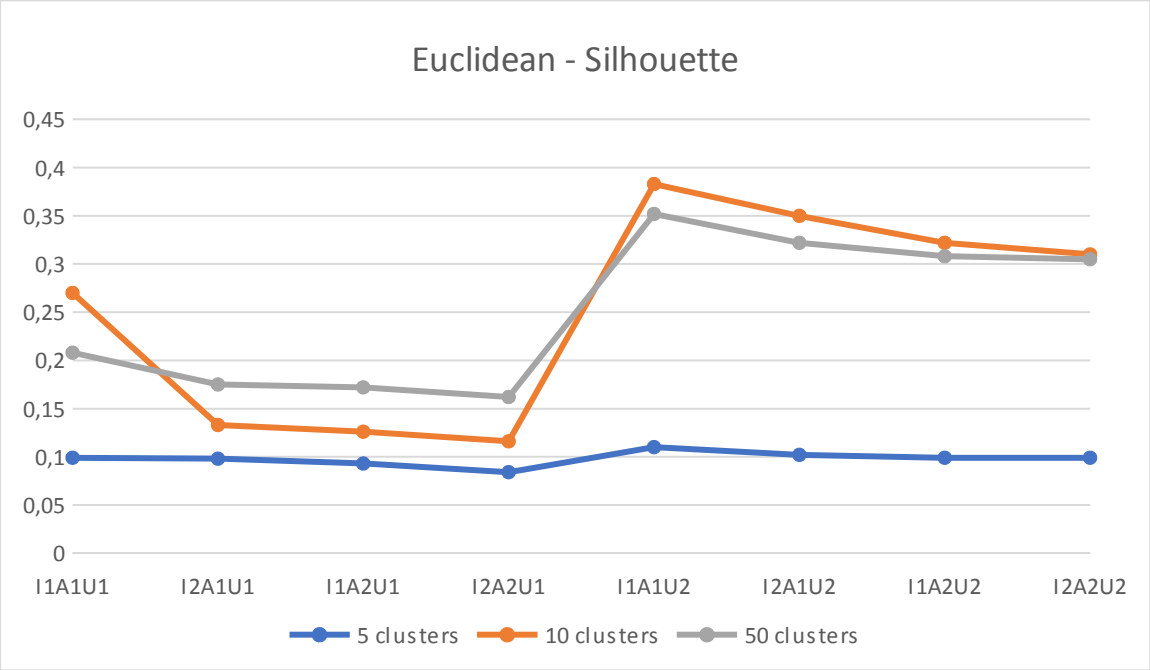
3. Το update, πέρα από τις συνθήκες που ανέλυσα παραπάνω, σταματάει επίσης μετά από 40 επαναλήψεις για τον k-means και μετά από 4 επαναλήψεις για το pam. Αυτές οι τιμές επιλέχθηκαν αφενός πειραματικά, δηλαδή συνήθως δεν ξεπερνιούνται ούτως ή άλλως, και αφετέρου αναλογικά με τον χρόνο που παίρνει η κάθε μέθοδος ώστε να μην είναι πολύ χρονοβόρα η διαδικασία.

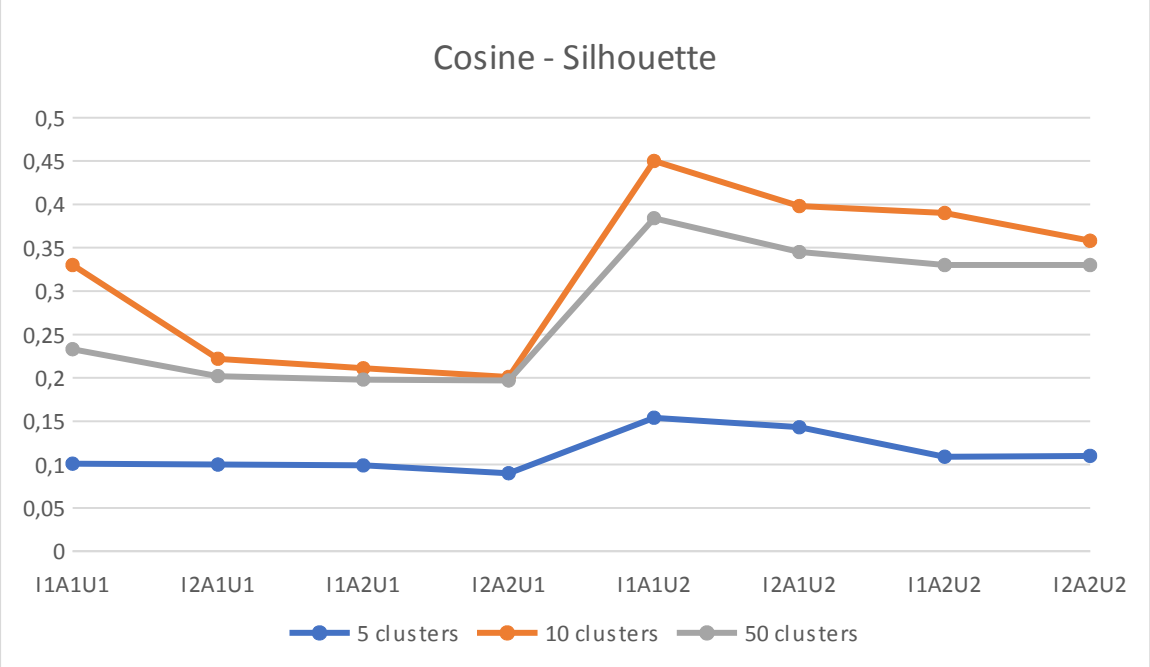
4. Έχει γίνει χρήση github, όλο το project έχει ανέβει στον λογαριασμό μου εκεί και γινόταν τακτικά update κατά τη διάρκεια της προθεσμίας.

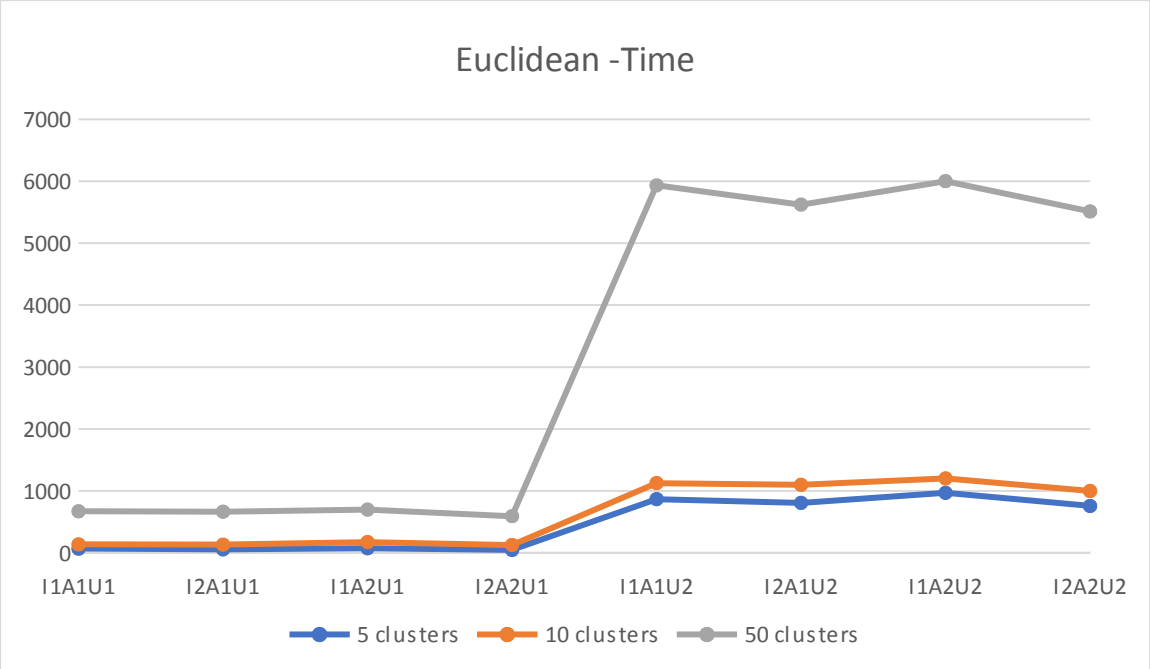
5. Ο κώδικας έχει ελεγχθεί για memory leaks με valgrind και δεν υπάρχουν εμφανή προβλήματα.

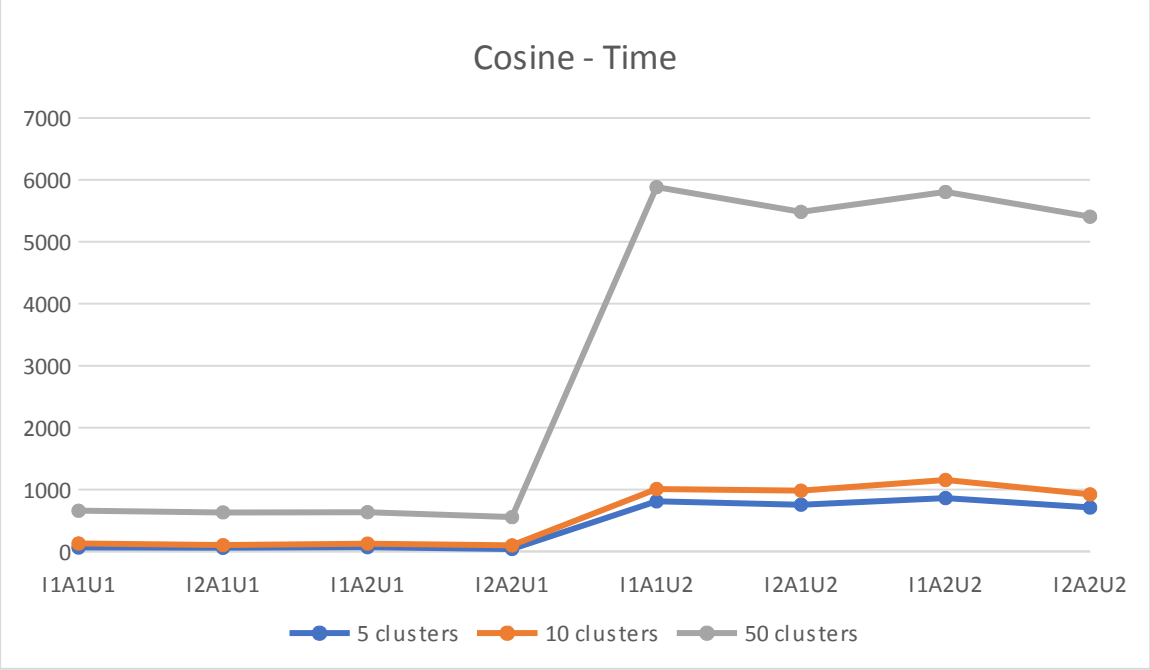
6. Δεν έχει υλοποιηθεί ο υπερκύβος, λόγω του γεγονότος ότι δεν είχε υλοποιηθεί και στην $1^{η}$ εργασία, δεν υπήρχε επαρκής χρόνος και είχα και την ατυχία με την κλοπή του laptop για το οποίο έχω ενημερώσει τον διδάσκοντα.

Ε) Μετρήσεις


Euclidean - Silhouette
0,45
0,4
0,35
0,3
0,25
0,2
0,15
0,1
0,05
0
I1A1U1
I2A1U1
I1A2U1
I2A2U1
I1A1U2
I2A1U2
I1A2U2
I2A2U2
5 clusters
10 clusters
50 clusters


Cosine - Silhouette
0,5
0,45
0,4
0,35
0,3
0,25
0,2
0,15
0,1
0,05
0
I1A1U1
I2A1U1
I1A2U1
I2A2U1
I1A1U2
I2A1U2
I1A2U2
I2A2U2
5 clusters
10 clusters
50 clusters





Παρατηρούμε, ότι μεγάλη διαφορά υπάρχει μεταξύ των αλγορίθμων update. Στον pam, πολλαπλασιάζεται ο χρόνος, αλλά ανεβαίνει αρκετά και το silhouette.

Επίσης, παρατηρούμε ότι με την cosine μετρική έχουμε ελαφρώς καλύτερα αποτελέσματα τόσο από άποψη χρόνου όσο και από άποψη απόδοσης silhouette.

Τέλος, ως επί το πλείστον όσο αυξάνουμε τον αριθμό των cluster αυξάνεται ο χρόνος, αλλά αυξάνεται και το silhouette.