Κ23γ: Ανάπτυξη Λογισμικού για Αλγοριθμικά Προβλήματα
Χειμερινό εξάμηνο 2020-21

1η Προγραμματιστική Εργασία
Αναζήτηση και συσταδοποίηση διανυσμάτων στη C/C++

Η άσκηση θα υλοποιηθεί σε σύστημα Linux και θα υποβληθεί στις Εργασίες του e-class το αργότερο την Παρασκευή 30/10 στις 23.59.

Περιγραφή της εργασίας

Α. Υλοποιήστε τον αλγόριθμο LSH για διανύσματα στον *d*-διάστατο χώρο βάσει της μετρικής Μανχάταν (L1), καθώς και τον αλγόριθμο τυχαίας προβολής στον υπερκύβο για την ίδια μετρική. Το πρόγραμμα θα υλοποιηθεί έτσι ώστε λαμβάνοντας ως είσοδο διάνυσμα q και ακέραιους N και R, να επιστρέφει προσεγγιστικά: α) τον πλησιέστερο γείτονα στο q, β) τους N πλησιέστερους γείτονες στο q και γ) τα διανύσματα εντός ακτίνας R από το q (range search). Ο σχεδιασμός του κώδικα θα πρέπει να επιτρέπει την εύκολη επέκτασή του σε διανυσματικούς χώρους με άλλη μετρική, π.χ., p-norm, ή διαφορετικούς χώρους.

Β. Υλοποίηση αλγορίθμων για τη συσταδοποίηση διανυσμάτων στον χώρο $\Re^d$. Θα χρησιμοποιηθεί η μετρική L1 (Manhattan). Η αρχικοποίηση πραγματοποιείται με την τεχνική k-Means++ και η ενημέρωση με τον υπολογισμό του διάμεσου (median) διανύσματος. Οι αλγόριθμοι διαφοροποιούνται στο βήμα της ανάθεσης όπου χρησιμοποιείται (α) ο ακριβής αλγόριθμος Lloyd's, και η αντίστροφη ανάθεση (reverse) μέσω Range search με (β) LSH, ή (γ) Τυχαία προβολή.

ΕΙΣΟΔΟΣ

Α.

1) Ένα binary αρχείο `input.dat` για την είσοδο του συνόλου δεδομένων (dataset) με την κάτωθι μορφή:

```
[offset] [type]          [value]          [description]
0000     32 bit integer  0x00000803(2051) magic number
0004     32 bit integer  60000            number of images
0008     32 bit integer  28               number of rows
0012     32 bit integer  28               number of columns
0016     unsigned byte   ??               pixel
0017     unsigned byte   ??               pixel
........
xxxx     unsigned byte   ??               pixel
```

Το αρχείο αυτό αντιστοιχεί στο dataset MNIST που περιέχει εικόνες χειρόγραφων αριθμητικών ψηφίων http://yann.lecun.com/exdb/mnist/. Οι εικόνες είναι 28*28 pixels, καθένα από τα οποία παίρνει ακέραια τιμή από το 0 έως το 255. Το διάνυσμα που αντιστοιχεί σε κάθε εικόνα προκύπτει από τη συνένωση των γραμμών της και έχει διάσταση 784.

2) Δυαδικό αρχείο `query.dat` που περιλαμβάνει το σύνολο αναζήτησης και περιέχει τουλάχιστον μία εικόνα MNIST με την ίδια μορφή.

Το πρόγραμμα αρχικά ζητά από τον χρήστη το μονοπάτι του dataset. Μετά τη δημιουργία της δομής αναζήτησης, το πρόγραμμα ζητά από τον χρήστη το μονοπάτι του αρχείου αναζήτησης και του αρχείου εξόδου. Μετά την εκτέλεση του αλγορίθμου και την παραγωγή των αποτελεσμάτων, το πρόγραμμα ζητά από τον χρήστη αν θέλει να τερματίσει το πρόγραμμα ή να επαναλάβει την αναζήτηση για διαφορετικό σύνολο / αρχείο αναζήτησης.

Για το LSH, μπορούν να δίνονται οι εξής παράμετροι προαιρετικά στη γραμμή εντολών: ακέραιο πλήθος k των LSH συναρτήσεων hi που χρησιμοποιούνται για τον ορισμό των g, ο ακέραιος αριθμός L των πινάκων κατακερματισμού, ο ακέραιος αριθμός N των πλησιέστερων γειτόνων και ο δεκαδικός αριθμός R της ακτίνας αναζήτησης. Αν τα k, L δεν δίνονται, το πρόγραμμα χρησιμοποιεί default τιμές k=4, L=5, N=1, R=10000.

Για την τυχαία προβολή στον υπερκύβο, μπορούν να δίνονται οι εξής προαιρετικές παράμετροι στη γραμμή εντολών: η διάσταση στην οποία προβάλλονται τα σημεία *k (=d')*, το μέγιστο επιτρεπόμενο πλήθος υποψήφιων σημείων που θα ελεγχθούν *M*, το μέγιστο επιτρεπόμενο πλήθος κορυφών του υπερκύβου που θα ελεγχθούν (probes), ο ακέραιος αριθμός N των πλησιέστερων γειτόνων και ο δεκαδικός αριθμός R της ακτίνας αναζήτησης. Οι default τιμές ειναι: *k*=14, *M*=10, probes=2, N=1, R=10000.

Τα αρχεία εισόδου και αναζήτησης θα μπορούν να δίνονται και μέσω παραμέτρων στη γραμμή εντολών. Οπότε η εκτέλεση θα γίνεται μέσω της εντολής:

```
$./lsh –d <input file> –q <query file> –k <int> -L <int> -o <output file> -N <number of nearest> -R <radius>
$./cube –d <input file> –q <query file> –k <int> -M <int> -probes <int> -o <output file> -N <number of nearest> -R <radius>
```

B.

1) Το αρχείο της περίπτωσης Α. 1

2) Ένα αρχείο ρύθμισης παραμέτρων `cluster.conf` με την ακόλουθη μορφή (γραμμές όπου υπάρχει default τιμή μπορούν να μην δίνονται οπότε χρησιμοποιείται η default τιμή):

```
number_of_clusters: <int>                 // K of K-medians
number_of_vector_hash_tables: <int>       // default: L=3
number_of_vector_hash_functions: <int>    // k of LSH for vectors, default: 4
max_number_M_hypercube: <int>             // M of Hypercube, default: 10
number_of_hypercube_dimensions: <int>     // k of Hypercube, default: 3
number_of_probes: <int>                   // probes of Hypercube, default: 2
```

Τα αρχεία `input.dat, cluster.conf` δίνονται μέσω παραμέτρων στη γραμμή εντολών. Η εκτέλεση θα γίνεται μέσω της εντολής:

```
$./cluster –i <input file> –c <configuration file> -o <output file> -complete <optional> -m <method: Classic OR LSH or Hypercube>
```

ΕΞΟΔΟΣ

Α. Αρχείο κειμένου που περιλαμβάνει για κάθε εικόνα του συνόλου αναζήτησης με την χρήση των κατάλληλων ετικετών: α) τον αριθμό του N-οστού προσεγγιστικά πλησιέστερου γείτονα που βρέθηκε και την απόστασή του από το *q*, β) την απόσταση του *q* από τον αληθινά N-οστό πλησιέστερο γείτονα (μέσω

εξαντλητικής αναζήτησης), γ) τον χρόνο εύρεσης των (α), δ) τον χρόνο εύρεσης των (β) και ε) τους γείτονες εντός ακτίνας R. Το αρχείο εξόδου ακολουθεί υποχρεωτικά το εξής πρότυπο:

```
Query: image_number_in_query_set
Nearest neighbor-1: image_number_in_data_set
distanceLSH: <double> [ή distanceHypercube αντίστοιχα]
distanceTrue: <double>
...
Nearest neighbor-N: image_number_in_data_set
distanceLSH: <double> [ή distanceHypercube αντίστοιχα]
distanceTrue: <double>
tLSH: <double>
tTrue: <double>
R-near neighbors:
image_number_A
image_number_B
. . .
image_number_Z
```

και ούτω καθεξής. Το image_number προσδιορίζεται βάσει της σειράς με την οποία εμφανίζεται η εικόνα στα εκάστοτε αρχεία.

Β. Ένα αρχείο κειμένου το οποίο περιλαμβάνει τις συστάδες των εικόνων που παρήχθησαν από κάθε παραλλαγή του αλγορίθμου, τον χρόνο εκτέλεσης σε κάθε περίπτωση καθώς και τον δείκτη εσωτερικής αξιολόγησης της συσταδοποίησης **Silhouette**.

Το αρχείο εξόδου ακολουθεί υποχρεωτικά το παρακάτω πρότυπο, το οποίο επαναλαμβάνεται για κάθε παραλλαγή:

```
Algorithm: Lloyds OR Range Search LSH OR Range Search Hypercube
CLUSTER-1 {size: <int>, centroid: πίνακας με τις συντεταγμένες του centroid}
.     .     .     .     .     .     .
CLUSTER-K {size: <int>, centroid: πίνακας με τις συντεταγμένες του centroid}
clustering_time: <double> //in seconds
Silhouette: [s1,...,si,...,sK, stotal]
/* si=average s(p) of points in cluster i, stotal=average s(p) of points in
dataset */

/* Optionally with command line parameter –complete */
CLUSTER-1 {centroid, image_numberA, ..., image_numberX}
.     .     .     .     .     .     .

CLUSTER-K {centroid, image_numberR, ..., image_numberZ}
```

Επιπρόσθετες απαιτήσεις

1) Το πρόγραμμα πρέπει να είναι καλά οργανωμένο με χωρισμό των δηλώσεων / ορισμών των συναρτήσεων, των δομών και των τύπων δεδομένων σε λογικές ομάδες που αντιστοιχούν σε ξεχωριστά αρχεία επικεφαλίδων και πηγαίου κώδικα. Βαθμολογείται και η ποιότητα του κώδικα (π.χ. αποφυγή memory leaks). Η μεταγλώττιση του προγράμματος πρέπει να γίνεται με τη χρήση του εργαλείου make και την ύπαρξη κατάλληλου Makefile.

2) Το παραδοτέο πρέπει να είναι επαρκώς τεκμηριωμένο με πλήρη σχολιασμό του κώδικα και την ύπαρξη αρχείου readme το οποίο περιλαμβάνει κατ' ελάχιστο: α) τίτλο και περιγραφή του προγράμματος, β) κατάλογο των αρχείων κώδικα / επικεφαλίδων και περιγραφή τους, γ) οδηγίες μεταγλώττισης του προγράμματος, δ) οδηγίες χρήσης του προγράμματος και ε) πλήρη στοιχεία των φοιτητών που το ανέπτυξαν.

3) Η υλοποίηση του προγράμματος θα πρέπει να γίνει με τη χρήση συστήματος διαχείρισης εκδόσεων λογισμικού και συνεργασίας (Git).