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

1η Προγραμματιστική Εργασία
Υλοποίηση του Locality Sensitive Hashing (LSH) στη γλώσσα C/C++

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

Περιγραφή της άσκησης

Θα υλοποιήσετε τον LSH όπως παρουσιάστηκε στο μάθημα για την εύρεση, μέσα σε σύνολο δεδομένων *P,* που ανήκει σε μετρικό χώρο: α) των γειτόνων εντός ακτίνας R αντικειμένου *q* και β) του κοντινότερου γείτονα αντικειμένου q. Το πρόγραμμα θα υποστηρίζει, κατ' ελάχιστο, αντικείμενα που «ζουν»: α) στον *d*-διάστατο πραγματικό διανυσματικό χώρο βάσει τόσο α1) της ευκλείδειας μετρικής όσο και α2) της μετρικής cosine, β) στον χώρο Hamming, και γ) σε μετρικό χώρο όπου η απόσταση ορίζεται μέσω πίνακα αποστάσεων.

Ο σχεδιασμός του κώδικα θα πρέπει να επιτρέπει την εύκολη επέκτασή του σε διαφορετικούς χώρους όπως διανυσματικούς χώρους με μετρική π.χ., p-norm, ή μετρικούς χώρους όπου η απόσταση υπολογίζεται μέσω κατάλληλης συνάρτησης.

ΕΙΣΟΔΟΣ

1) Ένα αρχείο κειμένου για την είσοδο του συνόλου δεδομένων (αρχείο dataset) διαχωρισμένο με στηλοθέτες (tab-separated), το οποίο θα έχει την ακόλουθη γραμμογράφηση:

```
@metric_space vector
@metric {euclidean, manhattan, cosine} // default: euclidean else define
item_id1    X11     X12     ...     X1d

    .        .       .      ...      .

item_idN    XN1     XN2     ...     XNd
```

όπου **Xij** οι συντεταγμένες double του διανύσματος που αναπαριστά το αντικείμενο *i* όταν τα αντικείμενα «ζουν» στο d-διάστατο Ευκλείδειο χώρο, **ή**

```
@metric_space hamming
item_id1        B1

    ...         ...

item_id         BN
```

όπου **Bi** η δυαδική συμβολοσειρά μήκους <= 64 bits που αναπαριστά το αντικείμενο *i* όταν τα αντικείμενα «ζουν» στο χώρο Hamming, **ή**

```
@metric_space matrix
@items [item_id1, item_id2, ..., item_idN]
0         d12       d13       .         d1N
0         0         d23       .         d2N
0         0         0         .         d3N
.         .         .         .         .
0         0         0         .         0
```

όπου **dij** η απόσταση του αντικειμένου i από το αντικείμενο j (dij = dji, dii = 0), όταν ο μετρικός χώρος προσδιορίζεται από τον πίνακα αποστάσεων του συνόλου των M αντικειμένων **ή**

```
@metric_space function
@items [item_id1, item_id2, ..., item_idN]
@dynamic_library {library_name}  // compiled distance function
@function {function_name}
```

όπου στην τρίτη γραμμή ορίζεται το όνομα της δυναμικής βιβλιοθήκης εντός της οποίας έχει μεταγλωττιστεί η συνάρτηση απόστασης που θα χρησιμοποιηθεί. Τα ονόματα (item_idK) μπορούν να είναι μοναδικοί ακέραιοι ή συμβολοσειρές.

2) Αρχείο κειμένου που περιλαμβάνει το σύνολο των αντικειμένων για τα οποία αναζητούμε τους γείτονες ακτίνας *R* και τον κοντινότερο γείτονα: πρόκειται για το σύνολο αναζήτησης, που περιέχει τουλάχιστον ένα αντικείμενο (αρχείο αναζήτησης). Ο θετικός double *R* δίνεται στην πρώτη γραμμή. Όταν δίνεται R=0 το πρόγραμμα βρίσκει μόνο τον κοντινότερο γείτονα των αντικειμένων στο σύνολο αναζήτησης.

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

```
Radius: <double>
item_idS1     X11     X12      ...      X1d

     .         .       .       ...       .

item_idSQ     XQ1     XQ2      ...      XQd
```

και αντίστοιχα για τον χώρο Hamming, ή

```
Radius: <double>
item_idS1     d(S1)1    d(S1)2    ...    d(S1)N

     .          .         .       ...       .

item_idSQ     d(SQ)1    d(SQ)2    ...    d(SQ)N
```

όταν ο μετρικός χώρος προσδιορίζεται βάσει πίνακα αποστάσεων. Τα ονόματα των αντικειμένων στο σύνολο αναζήτησης item_idSj (1<=*j*<=*Q*) μπορούν να είναι μοναδικοί ακέραιοι ή συμβολοσειρές.

Ως optional παράμετροι μπορεί να δίνονται στη γραμμή εντολών: ο ακέραιος *k* των locality-sensitive συναρτήσεων *hi* που χρησιμοποιούνται για τον ορισμό των συναρτήσεων *g*, καθώς και ο ακέραιος αριθμός

L των πινάκων κατακερματισμού. Αν τα *k, L* δεν δίνονται, το πρόγραμμα χρησιμοποιεί default τιμές: *k*=4, *L*=5.
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>`

ΕΞΟΔΟΣ

Αρχείο κειμένου που περιλαμβάνει για κάθε αντικείμενο του συνόλου αναζήτησης με τη χρήση των κατάλληλων ετικετών: α) τα ονόματα των γειτόνων ακτίνας *R*, β) το όνομα του κοντινότερου γείτονα που βρέθηκε από τον LSH και την απόστασή του από το *q*, γ) την απόσταση του *q* από τον αληθινά πλησιέστερο γείτονα (όπως αυτός προσδιορίζεται μέσω εξαντλητικής αναζήτησης), δ) τον χρόνο εύρεσης του κοντινότερου γείτονα μέσω LSH, και ε) τον χρόνο εύρεσης του αληθινού κοντινότερου γείτονα. Το αρχείο εξόδου ακολουθεί υποχρεωτικά το παρακάτω πρότυπο:

```
Query: itemJ
R-near neighbors:
itemJ
itemK
. . .
itemW
Nearest neighbor: itemY
distanceLSH: <double>
distanceTrue: <double>
tLSH: <double>
tTrue: <double>
```
και ούτω καθεξής.

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. Στην υλοποίηση μπορείτε να χρησιμοποιείτε συναρτήσεις από την πρότυπη βιβλιοθήκη της C, καθώς και τη βιβλιοθήκη που απαιτείται για τη δυναμική φόρτωση βιβλιοθηκών. Αντίθετα δεν επιτρέπεται η χρήση βιβλιοθηκών για τις απαραίτητες δομές δεδομένων (συνδεδεμένες λίστες, vectors, hashtables), για τις γεννήτριες τυχαίων αριθμών (εκτός από τις συναρτήσεις που δηλώνονται στο `stdlib.h` της πρότυπης βιβλιοθήκης της C), για τις συναρτήσεις κατακερματισμού, για τις συναρτήσεις υπολογισμού αποστάσεων ούτε για άλλη λειτουργία του προγράμματος.

3. 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.

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