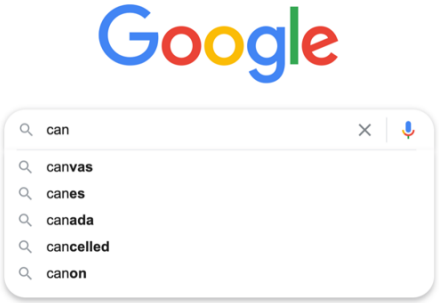
ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Ομαδική Εργασία – Δεντρικές Δομές Δεδομένων

Προθεσμία Παράδοσης: 01/12@10:00

Η εργασία αυτή επικεντρώνεται στη χρήση δεντρικών δομών δεδομένων για την ανάπτυξη εφαρμογής αυτόματης συμπλήρωσης λέξεων και για τη συγκριτική μελέτη της αποδοτικότητάς τους ως προς τις απαιτήσεις μνήμης. Θα υλοποιήσετε δύο εκδοχές της δενδρικής δομής δεδομένων Trie: ένα κλασικό Trie και ένα Compressed Trie με χρήση κατακερματισμού. Στο πρώτο μέρος της εργασίας θα χρησιμοποιήσετε το Compressed Trie για να υλοποιήσετε μια εφαρμογή που φορτώνει ένα λεξικό, ενημερώνει τη σημαντικότητα (συχνότητα) των λέξεων του λεξικού με βάση κάποιο άλλο κείμενο και παρέχει λειτουργίες πρόβλεψης/στατιστικής ανάλυσης πάνω στα προθέματα. Στο δεύτερο μέρος θα αποτιμήσετε πειραματικά τον χώρο που απαιτούν οι δύο δομές για διαφορετικούς τύπους τεχνητά παραγόμενων λεξικών.



Μέρος 1ο: Εφαρμογή αυτόματης συμπλήρωσης λέξεων με χρήση Compressed Trie

1. Φόρτωση λεξικού και πεδίο σημαντικότητας

Αρχικά, η εφαρμογή θα υλοποιήσει ένα λεξικό με τη βοήθεια της δομής **Compressed Trie με χρήση κατακερματισμού**, *όπως αυτή εξηγήθηκε και υλοποιήθηκε στα πλαίσια του εργαστηρίου*. Κάθε κόμβος της δομής Compressed Trie θα πρέπει να διαθέτει και έναν ακέραιο μετρητή σημαντικότητας, έστω importance, ο οποίος αρχικοποιείται σε μηδέν. Η εφαρμογή θα διαβάζει από τη γραμμή εντολών ένα αρχείο λεξικού· κάθε γραμμή του αρχείου αυτού θα περιέχει μία λέξη χωρίς πρόσθετους χαρακτήρες μορφοποίησης ή στίξης. Όλες οι λέξεις του αρχείου θα εισάγονται στο Compressed Trie. Η εισαγωγή των λέξεων δε θα είναι case-sensitive: λέξεις που διαφέρουν σε πεζά/κεφαλαία θεωρούνται ίδιες. Μετατρέψετε όλες τις λέξεις του λεξικού σε πεζά γράμματα πριν από την εισαγωγή τους στο λεξικό.

2. Ενημέρωση συχνοτήτων από αρχείο κειμένου

Η εφαρμογή δέχεται σαν όρισμα από τη γραμμή εντολής και το όνομα δεύτερου αρχείου εισόδου το οποίο θα περιέχει ελεύθερο κείμενο. Καθώς το αρχείο διαβάζεται, κάθε συμβολοσειρά που εξάγεται ως υποψήφια λέξη καθαρίζεται από τυχόν σημεία στίξης στο τέλος ή στην αρχή (ενδεικτικά: τελεία, κόμμα, θαυμαστικό, ερωτηματικό, άνω τελεία, εισαγωγικά) και μετατρέπεται σε πεζά γράμματα. Στη συνέχεια ελέγχεται αν η «καθαρισμένη» λέξη υπάρχει στο Compressed Trie. Αν ναι, αυξάνεται κατά ένα ο μετρητής σημαντικότητας του κόμβου όπου τερματίζει η λέξη. Με την ολοκλήρωση της επεξεργασίας του αρχείου, κάθε λέξη του Compressed Trie (η οποία αντιστοιχεί σε λέξη του λεξικού) θα έχει ως τιμή του μετρητή importance το πλήθος των εμφανίσεών της στο ελεύθερο κείμενο.

3. Διαδραστικό μενού – λειτουργίες εφαρμογής

Μετά τη φόρτωση των δεδομένων, η εφαρμογή εισέρχεται σε διαδραστικό βρόχο (μενού). Ο χρήστης εισάγει επιλογές έως ότου ζητήσει έξοδο. Ο βρόγχος θα προσφέρει τις ακόλουθες επιλογές/λειτουργίες:

1. Εύρεση των `k` πιο σημαντικών λέξεων που ξεκινούν με το πρόθεμα `prefix`: Δεδομένων ενός ακεραίου k και μιας συμβολοσειράς prefix, η μέθοδος θα εκτυπώνει στην οθόνη τις k πιο σημαντικές λέξεις του λεξικού που ξεκινούν με το πρόθεμα prefix. Οι λέξεις θα πρέπει να εκτυπωθούν στην οθόνη σε φθίνουσα σειρά σημαντικότητας (σε ισοβαθμία θα πρέπει να ταξινομήσετε αλφαβητικά). Για αποδοτικότητα απαιτείται χρήση **σωρού ελαχίστων (min-heap)** μεγέθους το πολύ k.
2. Υπολογισμός της μέσης συχνότητας των λέξεων που ξεκινούν με το πρόθεμα `prefix`. Η μέση συχνότητα, έστω `averageFreq`, ορίζεται ως εξής

$$averageFreq(prefix) = \frac{\sum_{word \in T(prefix)} frequency(word)}{|T(prefix)|}$$

όπου T(prefix) είναι το σύνολο όλων των λέξεων του λεξικού που ξεκινούν με το πρόθεμα prefix, frequency(word) είναι η τιμή σημαντικότητας της λέξης word. Αν δεν υπάρχει λέξη με το δοθέν πρόθεμα, επιστρέφεται μηδέν.

3. Πρόβλεψη του επόμενου γράμματος της λέξης που ξεκινά με το πρόθεμα `prefix`. Η μέθοδος αυτή, δεδομένης μιας συμβολοσειράς prefix, θα εκτυπώνει στην οθόνη τον πιθανότερο **επόμενο χαρακτήρα** που θα πληκτρολογήσει ο χρήστης. Ο χαρακτήρας που θα προταθεί θα αντιστοιχεί σε αυτόν που οδηγεί στη μεγαλύτερη μέση συχνότητα από όλα τα υποδέντρα του προθέματος prefix. Για την υλοποίηση της μεθόδου αυτής θα πρέπει να χρησιμοποιηθεί ο υπολογισμός της μέσης συχνότητας όπως αυτός έχει εξηγηθεί στη 2η επιλογή του μένού. Αν δεν υπάρχουν παιδιά, δεν προτείνεται χαρακτήρας.

Παράδειγμα εξόδου για το λεξικό το οποίο περιέχει τις πιο κάτω λέξεις:

"apple" (50 εμφανίσεις)

"class" (30 εμφανίσεις)

"application" (30 εμφανίσεις)

"ucy" (50 εμφανίσεις)

"apt" (5 εμφανίσεις)

"ape" (10 εμφανίσεις)

topKFrequentWordsWithPrefix(“ap”, 3) → apple application ape

getAverageFrequencyOfPrefix("ap") → (50 + 30 + 5 + 10)/4 = 23.75

predictNextLetter("ap") εξετάζει τα παιδιά του κόμβου "ap": "p" και "t" και "e":

"p" → (50 + 30) → average = 40

"t" → (5) → average = 5

"e" → (10) → average = 10

→ Προβλέπεται ο χαρακτήρας "p"

4. Αποδοτικότητα Υλοποίησης

Η υλοποίηση σας θα πρέπει να αποδοτική από άποψη χρόνου και μνήμης. Επίσης θα πρέπει να μπορεί να χειριστεί διαφορετικά μεγέθη λεξικών και αρχείων σημαντικότητας χωρίς να απαιτείται αλλαγή στον κώδικα ή στις δομές που χρησιμοποιήσατε.

Μέρος 2ο: Πειραματική Αξιολόγηση χρήσης μνήμης – Trie έναντι Compressed Trie

Στο δεύτερο μέρος της εργασίας θα συγκρίνετε ποσοτικά τον χώρο μνήμης που απαιτούν οι δύο δομές: (i) κλασικό Trie όπου κάθε κόμβος αντιστοιχεί σε έναν χαρακτήρα και φέρει δείκτες προς τα παιδιά του, καθώς και πεδίο σημαντικότητας για τερματιζόμενες λέξεις· (ii) Compressed Trie, όπως αυτό της εφαρμογής του Μέρους 1. Η σύγκριση επικεντρώνεται αποκλειστικά στη μνήμη που χρειάζεται η αποθήκευση του λεξικού· οι λειτουργίες συμπλήρωσης δεν εμπλέκονται εδώ.

1. Συνθετικά λεξικά και πειραματικές παράμετροι

Θα δημιουργήσετε τεχνητά λεξικά (δεν απαιτείται πραγματικό λεξικό φυσικής γλώσσας) ώστε να ελέγξετε συστηματικά δύο παράγοντες. Για την παραγωγή των λεξικών, θα χρειαστείτε το συνολικό πλήθος λέξεων n του λεξικού, την κατανομή του μήκους των λέξεων και την κατανομή των εμφανίσεων των γραμμάτων στις λέξεις. Επιλέξτε τιμές n που σας επιτρέπουν να διακρίνετε τάσεις (ενδεικτικά: 10000, 100000, 1000000· μπορείτε να παρεμβάλετε ενδιάμεσες ή να ξεκινήσετε με μεγαλύτερα λεξικά). Εξετάστε δύο σενάρια μήκους: (α) όλες οι λέξεις έχουν ίδιο μήκος· (β) τα μήκη ποικίλουν. Στο σενάρια δεν πρέπει να χρησιμοποιήσετε απλή ομοιόμορφη τυχαιοποίηση· η εισαγωγή των γραμμάτων και του μήκους των λέξεων (για το δεύτερο σενάριο) θα πρέπει να αντανακλά ρεαλιστικότερα λεξικά (συνήθως μεγαλύτερη πυκνότητα σε μεσαία μήκη, πολύ λιγότερες λέξεις στα άκρα κτλ). Τεκμηριώστε τη μέθοδο παραγωγής (π.χ. γεωμετρική, κανονικοποιημένη εμπειρική κατανομή, προσεγγιστικά δεδομένα από γνωστό λεξικό) και παραθέστε βιβλιογραφικές ή διαδικτυακές πηγές από τις οποίες αντλήσατε τις πληροφορίες.

2. Διαδικασία Πειράματος και μέτρηση μνήμης:

Αντί να στηριχθείτε σε profiler του λειτουργικού συστήματος, υπολογίστε τη θεωρητική μνήμη από τις ίδιες τις δομές δεδομένων, κάνοντας σαφείς υποθέσεις μεγέθους μεταβλητών (bytes ανά δείκτη, ανά χαρακτήρα, ανά ακέραιο). Εφαρμόστε το ίδιο μοντέλο κόστους σε όλες τις εκτελέσεις ώστε τα αποτελέσματα να είναι συγκρίσιμα.

Για κάθε n και κάθε τύπο κατανομής μήκους δημιουργήστε πολλαπλά τυχαία δείγματα λεξικού και αναφέρετε τον μέσο όρο της απαιτούμενης μνήμης (και, αν θέλετε, τυπική απόκλιση).

3. Παρουσίαση Αποτελεσμάτων:

Τα αποτελέσματα θα παρουσιαστούν γραφικά: στον οριζόντιο άξονα το μέγεθος του λεξικού n, στον κατακόρυφο η μέση μνήμη για κάθε n (σε bytes, KB ή MB—εξηγήστε τη μονάδα). Να δημιουργήσετε ένα γράφημα ανά σενάριο μήκους (ίδιο / διαφορετικό), όπου εμφανίζονται και οι δύο δομές (Trie και Compressed Trie) για άμεση σύγκριση. Για κάθε διάγραμμα, γράψτε μία σύντομη ανάλυση (μέχρι 8 γραμμές) των παρατηρήσεών σας (πότε υπερέχει η κάθε δομή, πότε συγκλίνουν οι δομές, τι ρόλο παίζει το κοινό πρόθεμα κτλ)

Κριτήρια αξιολόγησης

Η προγραμματιστική αυτή άσκηση είναι ομαδική και η μέγιστη δυνατή βαθμολογία είναι 100. Για την αξιολόγηση της άσκησης, θα ληφθούν οι πιο κάτω παράμετροι:

1. **Κατανοητά Σχόλια:** Γράφετε κατανοητά σχόλια που να εξηγούν την λειτουργία της κάθε κλάσης/πεδίου/μεθόδου.
2. **Σχολιασμός και Δομή Προγράμματος**: Σωστή οργάνωση, σχεδίαση και τεκμηρίωση των κλάσεων, μεθόδων και συναρτήσεων. Θα πρέπει να υπάρχει η κατάλληλη στοίχιση του κώδικα και κατάλληλη ονοματολογία σε κλάσεις του προγράμματος.
3. Οι δομές που θα χρησιμοποιήσετε στο πρόγραμμα σας ***θα πρέπει*** να έχουν διδαχθεί στο μάθημα και οι κλάσεις θα πρέπει να υλοποιηθούν από εσάς. Απαγορεύεται η χρήση java κλάσεων που υλοποιούν δομές όπως κατακερματισμού, συνόλων, λεξικών, σωρών κτλ. Χρήση τέτοιων δομών θα οδηγήσει σε μηδενισμό της εργασίας σας.
4. **Ορθότητα Λειτουργίας (50%):** Η εφαρμογή πρέπει να καλύπτει πλήρως τις προδιαγραφές όπως αυτές περιγράφονται πιο πάνω και να επιστρέφει τη σωστή απάντηση για όλα τα στιγμιότυπα του

πεδίου ορισμού του προβλήματος που επιλύει. Βεβαιωθείτε ότι έχετε ελέγξει την ορθότητα του προγράμματός σας πριν το παραδώσετε.

5. **Πειραματική Αξιολόγηση Δομών Trie (50%):** Ποιοτική συλλογή και ανάλυση των αποτελεσμάτων της πειραματικής ανάλυσης.
6. Σε περίπτωση που το πρόγραμμά σας δεν επιστρέφει αποτέλεσμα ή παρουσιάζει σφάλματα, η μέγιστη βαθμολογία της εργασίας σας θα είναι 20. Σε περίπτωση που παραδώσετε μόνο πρόγραμμα χωρίς αναφορά, η μέγιστη βαθμολογία της εργασίας σας θα είναι 40.
7. Η εργασία θα τύχει αξιολόγησης κατά τη διάρκεια των εργαστηρίων στις 01/12. Στην περίπτωση αυτή, θα πρέπει όχι μόνο να είστε σε θέση να απαντήσετε σε ερωτήσεις που αφορούν την υλοποίησή σας, αλλά και να αλλάξετε τον κώδικά σας έτσι ώστε να ικανοποιεί τις νέες απαιτήσεις που θα σας δοθούν κατά τη διάρκεια της εξέτασης, εάν αυτό ζητηθεί. Αποτυχία στην απάντηση των ερωτήσεων θα οδηγήσει στον μηδενισμό της εργασίας. Σε περίπτωση μη παρουσίασης στην αξιολόγηση δηλώνει ότι η εργασία σας έχει κάποιο από τα προβλήματα του σημείου 6 και η μέγιστη βαθμολογία της εργασίας σας θα είναι 20.

Χρήση εργαλείων Τεχνητής Νοημοσύνης (ΤΝ)

Επιτρέπεται η υποβοηθητική χρήση εργαλείων ΤΝ υπό την προϋπόθεση ότι δηλώνεται ρητά στην αναφορά ποιο εργαλείο χρησιμοποιήθηκε, σε ποια στάδια (σχεδιασμός, debug, τεκμηρίωση, δημιουργία δεδομένων) και σε ποιο βαθμό το παραχθέν υλικό τροποποιήθηκε από εσάς. Η αυτούσια αντιγραφή κώδικα ή κειμένου απαγορεύεται.

Οδηγίες Υποβολής

Η εργασία θα πρέπει να παραδοθεί ηλεκτρονικά στο BlackBoard ως συμπιεσμένο αρχείο .zip ή .rar, το οποίο θα περιλαμβάνει τον πηγαίο κώδικα (*src*), την αναφορά σε αρχείο τύπου PDF και τον κώδικα παραγωγής συνθετικών λεξικών και αρχεία οδηγιών εκτέλεσης. Δώστε στο αρχείο σας το όνομα των φοιτητικών ταυτοτήτων σας και ακολουθήστε τη μορφοποίηση package ID1#.ID2# (πχ **ID768213. ID128412**).

ΚΑΛΗ ΕΠΙΤΥΧΙΑ