ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

ΣΧΟΛΗ ΕΠΙΣΤΗΜΩΝ & ΤΕΧΝΟΛΟΓΙΑΣ ΤΗΣ ΠΛΗΡΟΦΟΡΙΑΣ
SCHOOL OF INFORMATION SCIENCES & TECHNOLOGY
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ
DEPARTMENT OF INFORMATICS

ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ

Διδάσκουσα: Κ. Παπακωνσταντινοπούλου

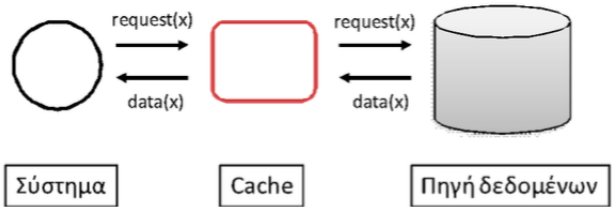
Χειμερινό εξάμηνο 2022

Εργασία 4
Υλοποίηση LRU Cache
Προθεσμία υποβολής: 20/3/2023, 23:59

Στην εργασία αυτή ζητείται να υλοποιήσετε σε Java τις βασικές μεθόδους που απαιτούνται για τη διαχείριση μιας μνήμης Cache γενικού σκοπού που εφαρμόζει την πολιτική αντικατάστασης Least Recently Used – LRU.

Γενικά περί Μνημών Cache

Στην Επιστήμη Υπολογιστών οι μνήμες cache είναι βοηθητικές μνήμες που χρησιμοποιούνται για την προσωρινή αποθήκευση δεδομένων για μελλοντική χρήση. Σε γενικές γραμμές, μια cache χρησιμοποιείται σε περιπτώσεις που κάποιο πρόγραμμα/σύστημα αναζητά δεδομένα από μια πηγή δεδομένων (π.χ. αναζήτηση αρχείων από σκληρό δίσκο ή βάση δεδομένων, πληροφορίες από το Internet – δείτε την παρακάτω εικόνα). Το πρόγραμμα αποθηκεύει την πληροφορία στην cache και την επόμενη φορά που αναζητά δεδομένα από την πηγή δεδομένων, τα αναζητά πρώτα στην cache. Αν τα δεδομένα βρίσκονται στην cache (cache hit), το σύστημα εξοικονομεί χρόνο καθώς δεν απαιτείται να ανατρέξει στην πηγή δεδομένων για να αναζητήσει τα δεδομένα (και είναι πολύ πιο γρήγορο να πάρουμε τα δεδομένα από την cache παρά από την πηγή). Στην περίπτωση που τα δεδομένα δεν βρίσκονται στην cache (cache miss), το σύστημα ανακτά τα δεδομένα από την πηγή και τα αποθηκεύει στην cache για μελλοντικές αναζητήσεις.



Για να είναι αποδοτική η χρήση μιας cache θα πρέπει ο χρόνος αναζήτησης στην περιοχή μνήμης της cache να είναι αρκετά μικρότερος από τον χρόνο που απαιτείται για την ανάκτηση δεδομένων από την πηγή δεδομένων. Για παράδειγμα, μια cache στο σύστημα αρχείων του υπολογιστή χρησιμοποιεί την κύρια μνήμη (RAM) που είναι γρηγορότερη από το σκληρό δίσκο. Επίσης, οι web browsers αποθηκεύουν προσωρινά αρχεία web σελίδων (έγγραφα html, εικόνες, flash objects) στον τοπικό υπολογιστή για να γλιτώσουν χρόνο σε σχέση με το να τα μεταφέρουν από τον server που φιλοξενεί ένα web site. Με αυτό τον τρόπο εξοικονομείται χρόνος για την ανάκτηση δεδομένων, με τίμημα το κόστος της μνήμης που απαιτεί η cache (ως συνήθως, έχουμε time vs. space tradeoff). Η τεχνική του caching χρησιμοποιείται ευρύτατα στα συστήματα υπολογιστών. Μερικά παραδείγματα εφαρμογής caching είναι τα ακόλουθα:

- Στα chip επεξεργαστών (CPU caching [1])
- Στα λειτουργικά συστήματα (Page cache [2])

- Στις βάσεις δεδομένων (Database caching [3])
- Στο World Wide Web (HTTP caching [4])

Μια cache μόνο για ανάγνωση χρησιμοποιείται ως ενδιάμεσος χώρος γρήγορης πρόσβασης. Στην εικόνα που ακολουθεί φαίνεται η δομή της.

Όνομα	Δεδομένα
X	00000000011111100000000111111
Y	00011100011110001001110000000
Z	00001111010101001110101111001
...	

Λειτουργία μιας Read-only Cache

Η cache αποθηκεύει τα δεδομένα της εκάστοτε εφαρμογής μαζί με κάποιο κλειδί που χρησιμοποιείται ως όνομα. Ας θεωρήσουμε στα πλαίσια της εργασίας ότι η cache αποθηκεύει αντικείμενα, τα οποία έχουν ένα πεδίο που χρησιμοποιείται ως κλειδί. Η δομή δεδομένων της cache πρέπει να υποστηρίζει κυρίως 2 λειτουργίες:

LookUp(x): Αναζήτηση στο χώρο μνήμης της cache αντικειμένων με κλειδί x. Αν τα δεδομένα υπάρχουν, τα επιστρέφει στο πρόγραμμα πελάτη, διαφορετικά επιστρέφει την τιμή null.

Store(x, Object): Αποθήκευση στην cache των δεδομένων με κλειδί x.

Ο χώρος μνήμης που καταλαμβάνει μια cache είναι συνήθως αρκετά μικρότερος σε σχέση με το χώρο που καταλαμβάνει η πηγή δεδομένων. Όταν ο χώρος της cache γεμίσει με εγγραφές, κάθε φορά που καλείται η λειτουργία Store, η cache πρέπει να διαγράψει μια εγγραφή προκειμένου να αποθηκεύσει τα νέα δεδομένα. Ο αλγόριθμος που χρησιμοποιείται για την επιλογή της εγγραφής που θα διαγραφεί λέγεται Πολιτική Αντικατάστασης ή Στρατηγική Αντικατάστασης (Replacement Policy ή Replacement Strategy), και παίζει πολύ σημαντικό ρόλο στην επίδοση της cache. Η πιο διαδεδομένη πολιτική αντικατάστασης είναι η πολιτική Least Recently Used (LRU). Σύμφωνα με αυτή την πολιτική, όταν η cache χρειάζεται να διαγράψει μια εγγραφή, διαγράφει την εγγραφή στην οποία η τελευταία προσπέλαση έγινε λιγότερο πρόσφατα. Ενδεικτικά, μια υλοποίηση της πολιτικής LRU μπορεί να γίνει με τη χρήση timestamps: η cache διατηρεί, πέρα από τα δεδομένα, ένα timestamp για κάθε εγγραφή. Στο timestamp αυτό αποθηκεύει τη χρονική στιγμή που έγινε η τελευταία προσπέλαση της εγγραφής. Κάθε φορά που καλείται η λειτουργία LookUp, αν η εγγραφή υπάρχει στη μνήμη, τότε η cache ενημερώνει το αντίστοιχο timestamp με την τρέχουσα ώρα και μετά επιστρέφει τα δεδομένα στο πρόγραμμα πελάτη. Όταν καλείται η λειτουργία Store και η μνήμη της cache είναι γεμάτη, η cache επιλέγει την εγγραφή με το χρονικά παλαιότερο timestamp και τη διαγράφει, ώστε να δημιουργηθεί χώρος για τα νέα δεδομένα (σημειώστε ότι αυτός δεν είναι ο αλγοριθμικά βέλτιστος τρόπος να υλοποιηθεί μια LRU cache).

Υλοποίηση της Δομής Δεδομένων

Στόχος της εργασίας είναι να υλοποιήσετε μια cache γενικού σκοπού σε γλώσσα Java. Η cache θα εφαρμόζει την πολιτική αντικατάστασης LRU. Για τη δομή σας παρέχεται ένα Java interface το οποίο περιγράφει όλες τις συναρτήσεις που θα υποστηρίζει η δομή σας (διεπαφή Cache, διαθέσιμη στην ενότητα 'Εγγραφα/Εργασίες' στο eclass). Το interface θα χρησιμοποιεί τα Java Generics ώστε η δομή να είναι γενικής χρήσης.

Η υλοποίησή σας θα πρέπει να είναι όσο το δυνατόν βέλτιστη ως προς το χρόνο εκτέλεσης των λειτουργιών της cache, δίνοντας έμφαση στο χρόνο που απαιτεί η πράξη LookUp. Για την αποδοτικότερη λειτουργία, η υλοποίηση της cache απαιτεί συνδυασμό κάποιων δομών δεδομένων που καλύφθηκαν στο μάθημα. Για παράδειγμα, αφού δίνουμε έμφαση στο να γίνεται γρήγορα η αναζήτηση, αυτό πρέπει να σας παραπέμψει σε δομές που έχουν τέτοιες ιδιότητες (δέντρα διαφόρων ειδών, κατακερματισμός, και γενικότερα το υλικό που καλύπτεται στα κεφάλαια 12-14 του βιβλίου σας). Παράλληλα, τυχόν χρήση timestamps μπορεί να σας παραπέμψει στη χρήση ουράς προτεραιότητας/σωρού (κεφάλαιο 9). Διάφοροι συνδυασμοί από δομές είναι εφικτοί για την υλοποίηση (δεν είναι όμως όλοι το ίδιο αποδοτικοί). Οι εργασίες θα συγκριθούν μεταξύ τους ως προς το χρόνο εκτέλεσης και η καλύτερη εργασία θα έχει κάποια περαιτέρω επιβράβευση.

Η υλοποίησή σας θα πρέπει στον κατασκευαστή να παίρνει ως όρισμα το μέγιστο πλήθος εγγραφών που θα μπορεί να αποθηκεύει η cache, ώστε το πρόγραμμα πελάτης να καθορίζει το μέγεθος της. Για τη συγγραφή της δομής μπορείτε να επαναχρησιμοποιήσετε κώδικα που έχετε γράψει στη διάρκεια των εργασιών ή έτοιμο κώδικα

από τα εργαστήρια. Σε καμία περίπτωση όμως δε μπορείτε να χρησιμοποιήσετε έτοιμες δομές δεδομένων, είτε από τη βιβλιοθήκη της Java ή από έτοιμες βιβλιοθήκες. Ο εκπαιδευτικός στόχος της εργασίας είναι να υλοποιήσετε τη δομή μόνοι σας. Για την εκπόνηση της εργασίας συστήνεται να ψάξετε σε διάφορες πηγές (βιβλιογραφία, web) και να βρείτε ιδέες για διάφορους τρόπους υλοποίησης της cache (με πίνακες, με λίστες, με δέντρα, με διάφορους συνδυασμούς κλπ) ώστε να αποφασίσετε ποιος είναι ο αποδοτικότερος τρόπος και να τον υλοποιήσετε. Η αναζήτηση πληροφοριών και η επιλογή για το πώς θα υλοποιήσετε τις διάφορες μεθόδους είναι μέρος της εργασίας.

Δοκιμαστικά δεδομένα

Για να επαληθεύσετε την ορθότητα της δομής, καθώς και τον υπολογιστικό χρόνο που χρειάζεται, έχει αναρτηθεί στο eclass έτοιμος κώδικας που προσομοιώνει το πρόγραμμα-πελάτη και την πηγή δεδομένων. Η default τιμή για το μέγεθος της cache στο πρόγραμμα πελάτη είναι 100, αλλά μπορείτε να χρησιμοποιήσετε κι άλλες τιμές. Η πηγή δεδομένων αποτελείται από ένα απλό δομημένο αρχείο κειμένου με πολλές εγγραφές (από 100.000 έως 1.000.000) και ένα πρόγραμμα που διαβάζει το αρχείο αυτό. Το πρόγραμμα πελάτης παίρνει ως είσοδο ένα αρχείο που περιγράφει το φόρτο (workload): μια ακολουθία από κλειδιά για τα δεδομένα που πρέπει να αναζητηθούν. Το σύνολο των αναζητήσεων θα είναι πολλαπλάσιο του συνόλου των εγγραφών.

Προϋποθέσεις για συμμετοχή

1. Θα πρέπει να έχετε παραδώσει τις υπόλοιπες 3 εργασίες του μαθήματος. Ασχοληθείτε με αυτή την εργασία μόνο αν έχετε χρόνο και διάθεση. Η εργασία αντιστοιχεί σε 0.5 μονάδες. Δεν θα βαθμολογηθούν όμως εργασίες που δεν έχουν υλοποιήσει όλες τις μεθόδους του interface. Η υλοποίησή σας θα πρέπει να είναι πλήρης, έστω κι αν δεν είναι η πιο αποδοτική σε χρόνο.

2. Μπορείτε να κάνετε την εργασία σε ζευγάρια ή ατομικά. Δε χρειάζεται να την κάνετε με τον ίδιο συνεργάτη που κάνατε τις υπόλοιπες εργασίες.

Παραδοτέα και Οδηγίες Υποβολής

Τα παραδοτέα της εργασίας είναι:

- Ο πηγαίος κώδικας (source code). Τοποθετήστε σε ένα φάκελο με όνομα src τα αρχεία java που έχετε φτιάξει. Φροντίστε να συμπεριλάβετε όποια άλλα αρχεία πηγαίου κώδικα απαιτούνται για να μεταγλωττίζεται η εργασία σας. Χρησιμοποιήστε τα ονόματα των interfaces/κλάσεων/συναρτήσεων όπως ακριβώς δίνονται. Φροντίστε επίσης να προσθέσετε επεξηγηματικά σχόλια όπου κρίνετε απαραίτητο στον κώδικά σας.

- Η αναφορά παράδοσης. Γράψτε μια αναφορά σε pdf αρχείο με όνομα project4-report.pdf, στην οποία θα περιγράφετε τη λειτουργία της δομής σας. Στην αναφορά αυτή θα αναφέρετε:

    1. Πώς λειτουργεί η δομή σας και σε ποιες δομές του μαθήματος βασιστήκατε,
    2. Ποιο είναι το υπολογιστικό κόστος των πράξεων της cache, και θα δικαιολογήσετε πώς καταλήξατε να χρησιμοποιήσετε τη συγκεκριμένη υλοποίηση (την προτιμήσατε σε σχέση με κάποια άλλη και γιατί). Μπορείτε επίσης να συμπεριλάβετε διαγράμματα ή γραφικές παραστάσεις με την επίδοση του προγράμματος σας.
    3. Αναφορές σε πηγές (βιβλία, επιστημονικά άρθρα, ιστοσελίδες) από όπου αντλήσατε πληροφορίες.

΄Ολα τα παραπάνω αρχεία θα πρέπει να μπουν σε ένα αρχείο zip. Το όνομα που θα δώσετε στο αρχείο αυτό θα είναι ο αριθμός μητρώου σας πχ. 3030056_3030066.zip ή 3030056.zip (αν δεν είστε σε ομάδα). Στη συνέχεια, θα υποβάλετε το zip αρχείο σας στην περιοχή του μαθήματος «Εργασίες» στο e-class. Υποβάλετε ένα αρχείο ανά εργασία. Δε χρειάζεται υποβολή και από τους 2 φοιτητές.

Η εργασία σας θα πρέπει να μην έχει συντακτικά λάθη και να μπορεί να μεταγλωττίζεται. Εργασίες που δε μεταγλωττίζονται χάνουν το **50%** της συνολικής αξίας τους.

Πηγές

[1] http://en.wikipedia.org/wiki/CPU_cache

[2] http://en.wikipedia.org/wiki/Page_cache

[3] http://en.wikipedia.org/wiki/Database_cache

[4] http://en.wikipedia.org/wiki/Web_cache