ΚΑΡΑΒΑΓΓΕΛΗ ΕΥΤΥΧΙΑ
ΑΜ: 1115201800062

ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ

ΕΡΓΑΣΙΑ 2

Τα παραδοθέντα αρχεία είναι τα εξής: PageTable.c, Stack_Queue.c, Simulation.c, ReplacementAlgorithms.c και Structures.h. Περιλαμβάνεται και επιπλέον Makefile, οπότε για την μεταγλώττιση του προγράμματος οι εντολές είναι: ''make clean'' και ''make''. Η γλώσσα υλοποίησης είναι C. Το πρόγραμμα δοκιμάστηκε και στον linux04 της σχολής. Δεν παρατηρήθηκε κάποιο πρόβλημα. Το πρόγραμμα δέχεται τα εξής ορίσματα στη **γραμμή εντολών**:

./test a b c d

./test: όνομα προγράμματος (να μην αλλαχθεί)

a: αλγόριθμος αντικατάστασης (1 για LRU και 2 για Second Chance)

b: αριθμούς πλαισίων μνήμης (frames)

c: αριθμός αναφορών που διαβάζονται από κάθε διεργασία (q)

d: maximum αριθμός αναφορών που θα διαβαστούν και από τα δύο αρχεία αναφορών (bzip.trace, gcc.trace). Αν για παράδειγμα δοθούν 10000 αναφορές τότε οι 5000 θα διαβαστούν από το αρχείο bzip.trace, ενώ οι υπόλοιπες 5000 από το αρχείο gcc.trace.

Αν δεν δοθούν σωστά ορίσματα εμφανίζονται μηνύματα λάθους και το πρόγραμμα τερματίζει, ώστε να ξαναδοθούν έπειτα καταλλήλως.

- ## Structures.h

Στο αρχείο αυτό δηλώνονται οι τύποι δεδομένων που αποτελούν τα περιεχόμενα του κατακερματισμένου πίνακα σελίδων και των δομών για τους αλγορίθμους αντικατάστασης. Κάθε κάδος κατακερματισμού του πίνακα σελίδων αποτελεί μια απλά συνδεδεμένη λίστα κόμβων με δείκτη στον πρώτο κόμβο της λίστας αυτής, που ο καθένας τους περιέχει τον αριθμό σελίδας, ο οποίος προκύπτει από τη λογική διεύθυνση (είναι αυτή που διαβάζεται από τα αρχεία αναφορών κάθε φορά) αν αφαιρεθεί το offset, τα τελευταία δηλαδή $log\ 4096 = 12\ bits$ της 32μπιτης αναφοράς. Άρα, για τον αριθμό σελίδας κρατάμε τα πρώτα από αριστερά 20 bits. Ακόμα, ο κόμβος περιέχει τον αριθμό πλαισίου (frame) της κύριας μνήμης (ο οποίος δείχνει σε ποιο πλαίσιο βρίσκεται η σελίδα κάποια δεδομένη χρονική στιγμή), και ένα δείκτη στον επόμενο κόμβο. Τέλος, υπάρχει ένα μέλος που δηλώνει αν η αναφορά από το αρχείο είναι για διάβασμα (R) ή για εγγραφή (W).
Για τα πλαίσια μνήμης υπάρχει μια δομή που αποτελείται από έναν πίνακα ακεραίων και έναν ακέραιο που δηλώνει το μέγεθός του, δηλαδή πρακτικά το πλήθος των πλαισίων. Κάθε κελί του πίνακα ακεραίων, δηλαδή των πλαισίων, περιέχει έναν αριθμό.

Αν είναι θετικός αντιστοιχεί στη σελίδα που φιλοξενείται σε αυτό το πλαίσιο. Αν είναι αρνητικός, σημαίνει πως το πλαίσιο αυτό είναι άδειο ακόμα.
Επιλέχθηκε σχεδιαστικά το μέγεθος κάθε φορά του πίνακα σελίδων να ισούται με το πλήθος των πλαισίων της κύριας μνήμης.
Η δομή Trace_Node κρατάει τα δεδομένα από το αρχείο αναφορών που διαβάζονται πριν αυτά εισαχθούν στον πίνακα σελίδων, δηλαδή αριθμό σελίδας, offset και mode δηλαδή R ή W.
Η δομή Counters περιέχει 3 ακεραίους έναν για κάθε μετρητή που ζητείται από την εκφώνηση: ένας για να μετράει τον αριθμό των σφαλμάτων σελίδας (page faults), ένας για να μετράει πόσες φορές γίνεται ανάγνωση αναφορών από το δίσκο (reads), και ένας για αν μετράει πόσες φορές χρειάζεται μια αναφορά και εγγραφεί πίσω στο δίσκο (writes).

Τέλος, ανάλογα με το ποιος αλγόριθμος αντικατάστασης ορίζεται, χρησιμοποιείται άλλη δομή. Αν χρησιμοποιείται ο LRU (Least Recently Used), τότε υλοποιούμε στοίβα στην οποία το 1ο στοιχείο είναι το πιο πρόσφατα εισαχθέν, δηλαδή είναι η πιο πρόσφατα σελίδα που έχει εισαχθεί. Άρα όταν τίθεται θέμα αντικατάστασης σελίδας, εξάγεται η σελίδα που βρίσκεται στον πάτο της στοίβας είναι δηλαδή η πιο παλιά στη στοίβα. Η στοίβα υλοποιείται με διπλά συνδεδεμένη λίστα με δείκτες στον 1ο και τελευταίο κόμβο της. Αντίθετα, στον Second Chance υλοποιούμε κυκλική ουρά, με ένα δείκτη να δείχνει στην αρχή της ουράς κάθε φορά και έναν στο τέλος της ουράς. Φυσικά αφού η ουρά είναι κυκλική το επόμενο στοιχείο του τελευταίου στοιχείου της ουράς είναι το 1ο. Τα υπόλοιπα πεδία είναι ο αριθμός σελίδας, ο id της διεργασίας (1 ή 2) και το reference_bit για τον Second Chance.

Και για τις δύο δομές χρησιμοποιείται το ίδιο struct StackNode, απλά η υλοποίηση είναι διαφορετική. Για παράδειγμα στην κυκλική ουρά δεν χρησιμοποιείται ο δείκτης previous και στην στοίβα δεν χρησιμοποιείται το πεδίο reference bit.

- PageTable.c

Περιλαμβάνει υλοποιήσεις συναρτήσεων για τον κατακερματισμένο πίνακα σελίδων. Η συνάρτηση κατακερματισμού που χρησιμοποιείται είναι η mod(size_of_pagetable). Το μέγεθος του πίνακα σελίδων ορίζεται να είναι ίσο με τον αριθμό των πλαισίων που δίνονται ως όρισμα στη γραμμή εντολών. Αν υπάρχει διαθέσιμο πλαίσιο τότε η σελίδα εισάγεται στον κατάλληλο κάδο του πίνακα κατακερματισμού, σχηματίζοντας σε κάθε κάδο απλά συνδεμένες λίστες, όπως ειπώθηκε παραπάνω. Αν δεν υπάρχουν διαθέσιμα πλαίσια καλείται ο κατάλληλος αλγόριθμος αντικατάστασης, ελευθερώνεται πλαίσιο και μετά εισάγεται η νέα σελίδα. Αν μια σελίδα υπάρχει στην κύρια μνήμη λόγω της διεργασίας έστω αυτής με id = 1, και μετά ζητηθεί από τη διεργασία με id = 2 (η σελίδα δεν υπάρχει στον πίνακα σελίδων της διεργασίας με id = 2), θα γίνει αναζήτηση της σελίδας στον πίνακα σελίδων της διεργασίας με id = 2, δεν θα βρεθεί, οπότε θα μπει ξανά στη μνήμη, στο πλαίσιο που θα ελευθερωθεί. Κάθε φορά που εισάγεται μια σελίδα στον πίνακα σελίδων της εκάστοτε διεργασίας, σημαίνει ότι η σελίδα δεν υπήρχε στη κύρια μνήμη, οπότε τότε αυξάνουμε τους μετρητές για pagefaults και read από το δίσκο.

όταν χρειαστεί να ελευθερωθεί ένα πλαίσιο που περιέχει μια σελίδα η οποία έχει **mode = 'W'**, τότε αυξάνουμε τον μετρητή writes, ώστε η τροποποιημένη σελίδα (όσο βρισκόταν στην κύρια μνήμη σημαίνει ότι τροποποιήθηκε) να γραφεί πίσω στο δίσκο. Κάθε διεργασία, όπως ορίζεται, έχει τον δικό της πίνακα σελίδων, αλλά όταν εισάγεται μια σελίδα στην κύρια μνήμη από μια διεργασία, μπορεί να αντικαταστήσει οποιαδήποτε σελίδα, ανεξάρτητα από τη διεργασία στην οποία ανήκει. Αν η σελίδα υπάρχει στον πίνακα σελίδων άρα και στη μνήμη, τότε ανάλογα με τον αλγόριθμο αντικατάστασης ενημερώνεται η κατάλληλη δομή (στοίβα ή ουρά μαζί με reference bit).

- **Stack_Queue.c**

  Υλοποιούνται συναρτήσεις για τη διαχείριση της στοίβας και της ουράς με τις προαναφερθείσες προδιαγραφές. Χρησιμοποιούν το ίδιο struct ως τύπο δεδομένων, αλλά η υλοποίησή τους διαφέρει.

- **ReplacementAlgorithms.c**

  Για τον LRU οι σελίδες αποθηκεύονται σε μια διπλά συνδεδεμένη λίστα με δείκτες που δείχνουν τον 1ο και τον τελευταίο κόμβο της λίστας και έχει τη λειτουργία στοίβας. Όταν εισέρχεται μια καινούρια σελίδα στον πίνακα σελίδων κάποιας διεργασίας, εισέρχεται και σε αυτή τη δομή, μόνο που αυτή η δομή είναι κοινή και για τις δύο διεργασίες. Κάθε φορά η καινούρια σελίδα εισέρχεται στην κορυφή της στοίβας, είναι δηλαδή πάντα το 1ο στοιχείο της λίστας, αν υπάρχει ήδη κάπου μέσα στη λίστα (αυτό συμβαίνει στην περίπτωση που η σελίδα βρεθεί στον πίνακα σελίδων της διεργασίας που τη ζητάει), τότε μεταφέρεται ώστε να γίνει το 1ο στοιχείο της λίστας. Όταν τίθεται ζήτημα για το ποια σελίδα θα αντικατασταθεί, τότε διαγράφεται το τελευταίο στοιχείο αυτής της λίστας, δηλαδή η πιο παλιά σελίδα, και ελευθερώνεται το πλαίσιό του, ώστε να φιλοξενήσει την καινούρια σελίδα. Η σελίδα που αντικαθίσταται διαγράφεται και από τον πίνακα σελίδων στον οποίο βρέθηκε. Αυξάνεται ο μετρητής για write back στο δίσκο αν είναι απαραίτητο.
  Για τον Second Chance, οι σελίδες αποθηκεύονται σε μια κυκλικά συνδεδεμένη λίστα με δείκτες που δείχνουν τον 1ο και τον τελευταίο κόμβο της λίστας και έχει τη λειτουργία ουράς. Όταν εισέρχεται μια καινούρια σελίδα στον πίνακα σελίδων κάποιας διεργασίας, εισέρχεται και σε αυτή τη δομή, μόνο που αυτή η δομή είναι κοινή και για τις δύο διεργασίες. Κάθε φορά η καινούρια σελίδα εισέρχεται στο τέλος της ουράς, είναι δηλαδή πάντα το τελευταίο στοιχείο της λίστας, επειδή είναι αυτό που έφτασε πιο πρόσφατα. Αν η σελίδα υπάρχει ήδη κάπου μέσα στη ουρά (αυτό συμβαίνει στην περίπτωση που η σελίδα βρεθεί στον πίνακα σελίδων της διεργασίας που τη ζητάει), τότε ψάχνουμε την σελίδα μέσα στην ουρά, ώστε να κάνουμε το reference bit της ίσο με 1, αν δεν είναι ήδη. Όταν τίθεται ζήτημα για το ποια σελίδα θα αντικατασταθεί, τότε ψάχνουμε την ουρά

από την αρχή (όποια είναι η αρχή κάθε φορά, γιατί αυτή μεταβάλλεται) και διαγράφουμε το πρώτο στοιχείο αυτής της ουράς που θα έχει reference bit = 0, ενώ όσες σελίδες – στοιχεία που εξετάστηκαν πιο πριν reference bit = 1 αυτό τίθεται στο 0. Έτσι, ελευθερώνεται το πλαίσιό του, ώστε να φιλοξενήσει την καινούρια σελίδα. Η σελίδα που αντικαθίσταται διαγράφεται και από τον πίνακα σελίδων στον οποίο βρέθηκε. Αυξάνεται ο μετρητής για write back στο δίσκο αν είναι απαραίτητο.

- ## Simulation.c

Περιλαμβάνει τη main του προγράμματος καθώς και κάποιες βοηθητικές συναρτήσεις για άνοιγμα και διάβασμα των αρχείων αναφορών, open_file, read_traces και συναρτήσεις για αρχικοποίηση κάποιων δομών (frames, counters). Η λειτουργία έχει ως εξής:

Αφού λαμβάνονται τα κατάλληλα ορίσματα και αρχικοποιηθούν οι κατάλληλες δομές, ανοίγονται τα δύο αρχεία με τις αναφορές (traces) και το bzip.trace προσομοιώνει τη διεργασία με id = 1 και το gcc.trace τη διεργασία με id = 2. Το διάβασμα των αναφορών ξεκινάει από το bzip αρχείο. Οπότε μέχρι να διαβαστούν max αναφορές ορίζει το όρισμα για τις μέγιστες αναφορές, διαβάζονται q αναφορές από το ένα αρχείο και q αναφορές από το άλλο αρχείο, (q = argv[3]). Οπότε, διαβάζεται μια αναφορά, διαχωρίζονται τα μέρη της (page number, offset, mode, το οποίο είναι βασικά το λεγόμενο dirty bit) και στη συνέχεια γίνεται το hashing για να καθοριστεί η θέση του πίνακα σελίδων στον οποίο θα εισαχθεί η σελίδα. Οι σελίδες για την κάθε διεργασία μπαίνουν στο αντίστοιχο PageTable της διεργασίας. Αν εισαχθεί η σελίδα, έχει καλώς, αλλιώς καλείται ο αλγόριθμος αντικατάστασης και μόλις ελευθερωθεί ένα πλαίσιο, τότε επιχειρείται ξανά η εισαγωγή της σελίδας στην κύρια μνήμη. Όταν από την εισαγωγή σελίδας στον πίνακα σελίδων επιστρέφεται αριθμός >= 0, αυτό σημαίνει ότι κάποια νέα σελίδα μπήκε σε πλαίσιο που έχει αριθμό αυτόν που επιστράφηκε, οπότε υπολογίζεται η φυσική διεύθυνση. Όταν γίνει αυτή η δουλειά για τον αριθμό όλων των αναφορών που απαιτήθηκε, τότε, εκτυπώνονται οι μετρητές που ζητήθηκαν, αποδεσμεύεται η μνήμη για τις διάφορες δομές και τερματίζει το πρόγραμμα.