ΕΡΓΑΣΙΑ #2 ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ ΑΡΤΙΟΙ

Α.Μ. : 1115201200106
Όνομα : Βασίλειος
Επίθετο : Μαυρομμάτης

ΠΑΡΟΥΣΙΑΣΗ ΚΩΔΙΚΑ

Η εργασία είναι υλοποιημένη σε C++ (με λίγη C) για 2 λόγους:
1) Την πρώτη την είχα κάνει σε C οπότε ήθελα να έχω ένα project σε C++ με σημαφόρους shared memory και systemcalls .
2) Υπήρχαν ξεχωριστές οντότητες που παρουσίαζαν διαφόρες λειτουργικότητες και συμπεριφορές και εκτιμήσα οτι θα αναπαρίσταναν καλύτερα σαν αντικείμενα κλάσεων.

Η εργασία αποτελείται από τα εξής **10 αρχεία**:
2 δωθέντα αρχεία **gcc.trace** και **bzip.trace** τα οπόια περιέχουν τα ίχνη αναφορών για την κάθε διεργασία, **1** αρχείο **PageReplacement.cpp** το οποίο περιέχει την main του προγράμματος, **2** αρχεία **HashPageTable.cpp** και το αντίστοιχο header **HashPageTable.h** που υλοποιούν τον ζητούμενα κατακερματισμένο πίνακα σελίδων, **2** αρχεία **Frame.cpp** και το αντίστοιχο header **Frame.h** που υλοποιούν τα Frames που διαιρείται η Κύρια Μνήμη, **2** αρχεία **MemoryManagerUnit.cpp** και το αντίστοιχο header **MemoryManagerUnit.h** που υλοποιούν τον ζητούμενα διαχειριστή μνήμης MMU, και τέλος **1** αρχείο **Makefile** για την δημιουργία του εκτελέσιμου αρχείου **project2**.

Υπάρχουν 2 #define μεταβλητές στο PageReplacement.cpp : η MAX_REFS που μπορεί να τροποποιηθεί και ορίζει πόσες αναφορές θα διαβαστούν απ’το κάθε trace file (default = 1.000.000 όλες οι αναφορές), και η PAGE_SIZE η οπόια δηλώνει ποιό θα είναι το μέγεθος της κάθε σελίδας και συνεπώς του αντίστοιχου frame (default = 4096 bytes).
Η main του προγράμματος (PageReplacement.cpp)ελέγχει το input από command line του χρήστη και διαβάζει 3 integers με την σειρά k = το όριο σφαλμάτων σελίδας, frames = ο αριθμός των διαθέσιμων πλαισίων μνήμης τα οπόια μοιράζονται στην μέση στις δύο διεργασίες που παράγουν τα αιτήματα, και q = το πλήθος των blocks αναφορών που θα στείλουν οι διεργασίες και θα διαβάζει εναλλάξ η τρίτη που υλοποιεί το MMU. Αρχικά δημιουργόυνται οι περιοχές διαμοιραζόμενης μνήμης που είναι απαραίτητες καθώς και οι αντίστοιχοι σημαφόροι για τον συντονισμό της όλης προσομοίωσης.
Στην συνέχεια η master process (main) κάνει τρία fork kai δημιουργει τις 3 διεργασίες slaves , οι 2 πρώτες στέλνουν αιτήματα και η τρίτη (MMU) τα επεξεργάζεται.
Μπαίνωντας στην κρίσιμη περιοχή κάθε διεργασία αιτημάτων αρχικά διαβάζει τα data που της έχει επιστρέψει απο προηγόυμενα αιτήματα το MMU και στην συνέχεια στέλνει νέα διαβάζοντας γραμμή γραμμή απτ’ο αντίστοιχο trace file της και μετατρέπει τις λογικές διευθύνσεις σε VPN (virtual page number) , offset(0~PageSize-1) , και mode (Read ή Write). Οι διεθυύνσεις που δίνονται είναι στα 32 bit και επομένως με page size 4096 = 2^12 = 3 bytes τα πρώτα 5 bytes αντιστοιχούν στα entries που μπορεί να αναπαραστήσει ο HPT και είναι 2^20 και τα επόμενα 3 στο offset μέσα στην κάθε σελίδα.
Αφού έχουν σταλεί τα αιτήματα στην κρίσιμη περιοχή μπαίνει η διεργασία MMU η οποία τα διαβάζει και μέσω στης συνάρτησης operate που είναι η καρδία του αλγορίθμου πραγματοποιεί την ζητόυμενη συμπεριφορά και επιστρέφει τα ζητούμενα data στις διεργασίες με βάση το offset.

Ο παραπάνω βρόγχος επαναλαμβάνεται μέχρι να έχουν εξυπηρετηθεί 2 * q reference blocks απτις δύο διεργασίες και τέλος εκτυπωνται στο stdout τα μηνύματα με τα στατιστικά λειτουργίας του MMU.
Ολοκληρώνοντας οι σκλάβοι καλούν την exit και σε εκείνο το σημείο βρίσκουν τον master που τους περιμένει. Ο master αναλαμβάνει να κάνει detatch kai να αποδεσμεύσει τις διαμοιραζόμενες μνήμες, καθώς και να αποδεσμεύσει τους σημαφόρους που χρησιμοποιήθηκαν, και επιστρέφει στην main από που το πρόγραμμα τερματίζει.

ΠΑΡΑΤΗΡΗΣΕΙΣ:

1) Η όλη προσομόιωση έχει πραγματοποιηθεί με όσο περισσότερο ρεαλισμό επέτρεπε ο χρόνος μου. Τα frames που δεσμεύονται είναι όλα μεγέθους 4096 bytes και αρχικά γεμίζονται με άχρηστα random data. Υπάρχει η οντότητα disk στην κλάση MMU η οποία αναπαριστά τον δίσκο, τον οποίο για λόγους απλότητας έχει το μέγεθος ενός frame = 4096 bytes μιας και οι μεταφόρες από και προς τον δίσκο όλες πραγματοποιούνται σε blocks δλδ 4096 bytes, οπότε προσεγγίζεται ικανοποιητικά πιστεύω.

2) Για τον HPT μιας και υπάρχουν conflicts απο hashes που αντιστοιχούν στο ίδιο frame απο διαφορετικά VPN, έχω χρησιμοποιήσει collision resolution με Separate Chaining aka open hasing. Εδώ τα conflicting entries τοποθετούνται στην ίδια λίστα , την οπόια την έχω υλοποιήσει ως δυναμικά μονά συνδεδεμένη με έναν δείκτη next για κάθε κόμβο που περιέχει ενα PTE, για οικονομία μνήμης, αλλα είναι υλοποιημένη LIFO έτσι ώστε τουλάχιστον οι καινούριες αναφορές που προκύπτουν και δεν έχουν PTE ακόμα να τοποθετούνται στην κεφαλή ωστέ να κερδίζουμε χρόνο εκμετελλεύοντας την χρονική τοπικότητα με λιγότερα searches για καινούριες αναφορές. Βέβαια άπαξ και ενα PTE υπάρχει η θέση του δεν αλλάζει στην λίστα όταν ξαναανφερόμαστε σε αυτό, δεν υπάρχουν δηλαδή καθόλου ανακατατάξεις των κόμβων.
Επίσης τα heads των entries δεν είναι σκέτα, έχουν και αυτα data δλδ ενα PTE, αυτο το έκανα για να μειώνω τον αριθμό τον searches πάντα κατά 1, βέβαια ο πίνακας θα είναι λιγάκι ποιο μεγάλος σε μέγεθος αφού αδεια chains στον πίνακα θα έχουν μέγεθος σαν να υπήρχε όντως ενα PTE, απλά δεν είναι και τόσο μεγάλο έχει μόνο 3 int kai 2 bool πεδία.

3) Ο κώδικας είναι πιστεύω αρκετά καθαρός και υπάρχουν σχόλια που ορίζουν τι κάνει η κάθε συνάρτηση στα αντίστοιχα header files όπου δηλώνονται καθώς και σχόλεια μέσα στο σώμα των συναρτησέων. Όταν το έγραφα είχα γεμίσει με print(cout) την κάθε διαδικάσια και στάδιο τις οποίες εντολές τις έχω αφήσει comment out για να μην γεμίζει το std και να εκτυπώνονται μόνο στο τέλος τα stats της προσομοίωσης αλλά και για να μπορεί ο αναγνώστης(εξεταστής :P) να τισ comment in και να δει αν το πρόγραμμα λειτουργεί σωστα.

ΕΝΔΕΙΚΤΙΚΕΣ ΕΚΤΕΛΕΣΕΙΣ

$make
$./project2 (k) (frames) (q)

Με frames = 1000 (4MB RAM) και q = 1000 (1000 blocks x 1000 refs = 1.000.000) για διάφορες τιμές του k = max page faults before flushing frames (FWF) @ k+1

```
ken@KEN-UBUNTU:~/Documents/DI/ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ/project2$ make clean
rm -f project2 project2.o Frame.o HashPageTable.o PageReplacement.o
MemoryManagerUnit.o
ken@KEN-UBUNTU:~/Documents/DI/ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ/project2$ make
g++ -g -Wall -c Frame.cpp -o Frame.o
g++ -g -Wall -c HashPageTable.cpp -o HashPageTable.o
g++ -g -Wall -c PageReplacement.cpp -o PageReplacement.o
g++ -g -Wall -c MemoryManagerUnit.cpp -o MemoryManagerUnit.o
g++ -g -Wall -o project2 Frame.o HashPageTable.o PageReplacement.o
MemoryManagerUnit.o
ken@KEN-UBUNTU:~/Documents/DI/ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ/project2$ ./project2
10 1000 1000
Master: Waiting...

Displaying MMU statistics below!

Disk Read : 304456
Disk Write : 65861
Total Page Faults : 304456
Total Frames still used : 9
Total Searches on all chains: 5536360

Master: All slaves are done...
Master: Detached shared memory...
Master: Destroyed shared memory...
Master: Destroyed the semaphores...
Master: Exiting...
ken@KEN-UBUNTU:~/Documents/DI/ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ/project2$ ./project2
50 1000 1000
Master: Waiting...

Displaying MMU statistics below!

Disk Read : 131696
Disk Write : 19574
Total Page Faults : 131696
Total Frames still used : 53
Total Searches on all chains: 5536360

Master: All slaves are done...
Master: Detached shared memory...
Master: Destroyed shared memory...
Master: Destroyed the semaphores...
Master: Exiting...
ken@KEN-UBUNTU:~/Documents/DI/ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ/project2$ ./project2
250 1000 1000
Master: Waiting...

Displaying MMU statistics below!

Disk Read : 77969
Disk Write : 9438
```

Total Page Faults : 77969
Total Frames still used : 166
Total Searches on all chains: 5536360

Master: All slaves are done...
Master: Detached shared memory...
Master: Destroyed shared memory...
Master: Destroyed the semaphores...
Master: Exiting...
ken@KEN-UBUNTU:~/Documents/DI/ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ/project2$ ./project2 500 1000 1000
Master: Waiting...

Displaying MMU statistics below!

Disk Read : 67638
Disk Write : 7414
Total Page Faults : 67638
Total Frames still used : 162
Total Searches on all chains: 5536360

Master: All slaves are done...
Master: Detached shared memory...
Master: Destroyed shared memory...
Master: Destroyed the semaphores...
Master: Exiting...
ken@KEN-UBUNTU:~/Documents/DI/ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ/project2$ ./project2 1000 1000 1000
Master: Waiting...

Displaying MMU statistics below!

Disk Read : 60653
Disk Write : 5918
Total Page Faults : 60653
Total Frames still used : 119
Total Searches on all chains: 5536360

Master: All slaves are done...
Master: Detached shared memory...
Master: Destroyed shared memory...
Master: Destroyed the semaphores...
Master: Exiting...