

Τροποποίηση `unmcopy()`:

Η `unmcopy()` έχει τροποποιηθεί ώστε να υποστηρίζει την διαδικασία του CoW. Έτσι η λειτουργία της συνοψίζεται ως εξής:

- Παίρνει κάθε page table entry που αντιστοιχεί στην virtual address `i`.
- Ανακτά καλώντας την `PTE2PA()` την physical address του pte.
- Αν το παλίο flag του write ήταν 1, γράφει αυτή την πληροφορία στην θέση του `PTE_RSW_W`. Αυτό διότι η σελίδα θα δηλωθεί ως σελίδα CoW. Αυτό σημαίνει ότι όταν προκληθεί σφάλμα γραψίματος θα πρέπει να ξέρουμε αν εξ'αρχής η σελίδα είχε `write flag = 0` ή όχι.
- Θέτω το `PTE_RSW_CoW` flag σε 1 για να δηλώσω την σελίδα ως σελίδα CoW.
- Κάνω `map` στο καινούριο page table, την physical address με τα κατάλληλα flags.
- Καλώ την `referenceCounterIncrease` για να δηλώσω ότι στην `pa` αναφέρεται πλέον και άλλο process. Είναι σημαντικό η αύξηση του counter να γίνει μέσω της συνάρτησης, καθώς έτσι εξασφαλίζεται ότι δεν θα έχουμε πρόβλημα σε συνθήκες ανταγωνισμού στην πρόσβαση στο critical section. Η διαφορά την συνάρτησης δηλαδή είναι ότι καλεί και την `lock`.

Τροποποίηση της `usertrap()`:

Η `usertrap()` έχει τροποποιηθεί για να αναγνωρίζει σφάλματα σελίδας. Η λειτουργία είναι η εξής:

- Αν ο αριθμός του σφάλματος είναι ο 15 ανακτούμε την virtual address που προκάλεσε το σφάλμα μέσω του κατάλληλου register.
- Αν το page table entry που αντιστοιχεί σε αυτή την virtual address δεν είναι valid, επιστρέφουμε κανονικά error.
- Αν το page table entry που αντιστοιχεί σε αυτή την virtual address δεν είναι σελίδα CoW επιστρέφουμε κανονικά error.
- Αν αυτή η σελίδα είναι προϊόν CoW αλλά δεν υπήρχε εξαρχής δικαίωμα για write(αυτό το ελέγχω μέσω του PTE\_RSW\_W bit) επιστρέφω error.
- Ανακτώ το physical address από το page table entry.
- Δημιουργώ μια καινούρια σελίδα στην μνήμη
- Αντιγράφω τα περιεχόμενα της παλιάς physical address στην νέα σελίδα.
- Μέσω της `kfree()` κάνω decrease τον counter που υποδηλώνει πόσες διεργασίες αναφέρονται στην παλιά physical address.
- Τροποποιώ το pte ώστε να δείχνει πλέον στην νέα σελίδα στην μνήμη (Για να το κάνω αυτό εκτελώ το `*pte = PA2PTE(mem) | flags`. Θα μπορούσα να καλούσα και `unmap` και ύστερα `map` αλλά προτίμησα αυτόν τον τρόπο).
- Θέτω πλέον το flag του write σε 1 στην νέα σελίδα και την νέα σελίδα δεν την δηλώνω πλέον ως CoW σελίδα.

Επιπλέον τροποποιήσεις:

- a) Στο αρχείο `kallos.c` έχω προσθέσει έναν πίνακα `referenceArray` που λειτουργεί ως μετρητής αναφορών.
- b) Έχω τροποποιήσει την `kallos` για να αρχικοποιεί τον `referenceArray` της αντίστοιχης θέσης σε 1, κάθε φορά που εκτελείται.
- c) Έχω τροποποιήσει την `kfree()` ώστε να επιστρέφει μια σελίδα στην λίστα με τις διαθέσιμες σελίδες μόνο όταν ο μετρητής αναφορών γίνει 0. Συνεπώς αν ο `counter` της σελίδας ήταν μεγαλύτερος του μηδενός όταν κλήθηκε η `kfree()` μειώνω τον μετρητή.
- d) Αν ο μετρητής μετά την μείωση έγινε 0 ελευθερώνω την σελίδα.
- e) Έχω προσθέσει την συνάρτηση `referenceCounterIncrease()` για να αυξάνει τον μετρητή στην αντίστοιχη θέση του πίνακα με τις σελίδες. Εξασφαλίζει επίσης ότι δεν θα δημιουργηθούν συνθήκες ανταγωνισμού.
- f) Έχω τροποποιήσει το αρχείο `defs.h` προσθέτοντας κάποια ορίσματα συναρτήσεων που μου χρειάστηκαν.
- g) Έχω τροποποιήσει το αρχείο `riscv.h` προσθέτοντας κάποια macros για τα flags `PTE_RSW_CoW` και `PTE_RSW_W`.
- h) Τέλος έχω τροποποιήσει την συνάρτηση `copyout()` ώστε να χρησιμοποιεί τον ίδιο μηχανισμό όπως τα σφάλματα σελίδας όταν συναντά μια σελίδα CoW.