Λειτουργικά Συστήματα Υπολογιστών (τμήμα 1)

4η εργαστηριακή αναφορά

Ομάδα oslaba24	
Αναστάσιος Παπαζαφειρόπουλος	el18079
Νικόλαος Παγώνας	el18175

1.1 Κλήσεις συστήματος και βασικοί Μηχανισμοί του ΛΣ για τη διαχείριση της εικονικής μνήμης (Virtual Memory - VM

1. Με τη χρήση της show_maps(), τυπώνουμε τον χάρτη μνήμης της τρέχουσας διεργασίας.

```
Step 1: Print the virtual address space map of this process [14248].


Virtual Memory Map of process [14248]:
00400000-00403000 r-xp 00000000 00:21 20198008                           /home/oslab/oslaba24/4/mmap/mmap
00602000-00603000 rw-p 00002000 00:21 20198008                           /home/oslab/oslaba24/4/mmap/mmap
01dab000-01dcc000 rw-p 00000000 00:00 0                                  [heap]
7f89e3acc000-7f89e3c6d000 r-xp 00000000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3c6d000-7f89e3e6d000 ---p 001a1000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3e6d000-7f89e3e71000 r--p 001a1000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3e71000-7f89e3e73000 rw-p 001a5000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3e73000-7f89e3e77000 rw-p 00000000 00:00 0
7f89e3e77000-7f89e3e98000 r-xp 00000000 08:01 6032224                    /lib/x86_64-linux-gnu/ld-2.19.so
7f89e408a000-7f89e408d000 rw-p 00000000 00:00 0
7f89e4092000-7f89e4097000 rw-p 00000000 00:00 0
7f89e4097000-7f89e4098000 r--p 00020000 08:01 6032224                    /lib/x86_64-linux-gnu/ld-2.19.so
7f89e4098000-7f89e4099000 rw-p 00021000 08:01 6032224                    /lib/x86_64-linux-gnu/ld-2.19.so
7f89e4099000-7f89e409a000 rw-p 00000000 00:00 0
7ffd9f69e000-7ffd9f6bf000 rw-p 00000000 00:00 0                          [stack]
7ffd9f744000-7ffd9f747000 r--p 00000000 00:00 0                          [vvar]
7ffd9f747000-7ffd9f749000 r-xp 00000000 00:00 0                          [vdso]
ffffffffff600000-ffffffffff601000 r-xp 00000000 00:00 0                  [vsyscall]
-----------------------------------------------------------
```

2. Με την κλήση συστήματος mmap() δεσμεύουμε buffer μεγέθους μίας σελίδας και τυπώνουμε ξανά τον χάρτη.

```
Step 2: Use mmap(2) to allocate a private buffer of size equal to 1 page and print the VM map again.


Virtual Memory Map of process [14248]:
00400000-00403000 r-xp 00000000 00:21 20198008                           /home/oslab/oslaba24/4/mmap/mmap
00602000-00603000 rw-p 00002000 00:21 20198008                           /home/oslab/oslaba24/4/mmap/mmap
01dab000-01dcc000 rw-p 00000000 00:00 0                                  [heap]
7f89e3acc000-7f89e3c6d000 r-xp 00000000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3c6d000-7f89e3e6d000 ---p 001a1000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3e6d000-7f89e3e71000 r--p 001a1000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3e71000-7f89e3e73000 rw-p 001a5000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3e73000-7f89e3e77000 rw-p 00000000 00:00 0
7f89e3e77000-7f89e3e98000 r-xp 00000000 08:01 6032224                    /lib/x86_64-linux-gnu/ld-2.19.so
7f89e408a000-7f89e408d000 rw-p 00000000 00:00 0
7f89e4091000-7f89e4097000 rw-p 00000000 00:00 0
7f89e4097000-7f89e4098000 r--p 00020000 08:01 6032224                    /lib/x86_64-linux-gnu/ld-2.19.so
7f89e4098000-7f89e4099000 rw-p 00021000 08:01 6032224                    /lib/x86_64-linux-gnu/ld-2.19.so
7f89e4099000-7f89e409a000 rw-p 00000000 00:00 0
7ffd9f69e000-7ffd9f6bf000 rw-p 00000000 00:00 0                          [stack]
7ffd9f744000-7ffd9f747000 r--p 00000000 00:00 0                          [vvar]
7ffd9f747000-7ffd9f749000 r-xp 00000000 00:00 0                          [vdso]
ffffffffff600000-ffffffffff601000 r-xp 00000000 00:00 0                  [vsyscall]
-----------------------------------------------------------
```

Παρατηρούμε ότι η υπογραμμισμένη γραμμή του χάρτη μνήμης έχει εύρος 7f89e40**91000**-7f89e4097000, ενώ στο προηγούμενο βήμα είχε εύρος 7f89e40**92000**-7f89e4097000, που σημαίνει ότι ο επιπλέον χώρος δεσμεύτηκε επιτυχώς.

3. Καλώντας την get_physical_address(), προσπαθούμε να βρούμε την φυσική διεύθυνση στην οποία απεικονίζεται η εικονική διεύθυνση του buffer. Παρατηρούμε ότι η εικονική διεύθυνση δεν έχει κάποια φυσική απεικόνιση. Αυτό συμβαίνει διότι το λειτουργικό σύστημα ακολουθεί την πολιτική demand paging, οπότε αφού η μνήμη αυτή δεν έχει προσπελαστεί, είναι λογικό να μην έχει (ακόμα) φυσική αναπαράσταση.

```
Step 3: Find and print the physical address of the buffer in main memory. What do you see?

VA[0x7f89e4092000] is not mapped; no physical memory allocated.
```

4. Γεμίζουμε με μηδενικά τον buffer και επαναλαμβάνουμε το βήμα 3.

```
Step 4: Initialize your buffer with zeros and repeat Step 3. What happened?

pa of step 4 = 1066c2000
```

Παρατηρούμε ότι η εικονική διεύθυνση του buffer έχει πλέον αντιστοιχιστεί σε φυσική διεύθυνση. Αυτό είναι λογικό, όπως εξηγήσαμε και στο προηγούμενο βήμα.

5. Χρησιμοποιούμε την mmap() για να απεικονίσουμε το αρχείο file.txt στον χώρο διευθύνσεων της διεργασίας μας και να τυπώσουμε το περιεχόμενό του.

```
Step 5: Use mmap(2) to read and print file.txt. Print the new mapping information that has been created.

Hello everyone!

Virtual Memory Map of process [14248]:
00400000-00403000 r-xp 00000000 00:21 20198008                           /home/oslab/oslaba24/4/mmap/mmap
00602000-00603000 rw-p 00002000 00:21 20198008                           /home/oslab/oslaba24/4/mmap/mmap
01dab000-01dcc000 rw-p 00000000 00:00 0                                  [heap]
7f89e3acc000-7f89e3c6d000 r-xp 00000000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3c6d000-7f89e3e6d000 ---p 001a1000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3e6d000-7f89e3e71000 r--p 001a1000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3e71000-7f89e3e73000 rw-p 001a5000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3e73000-7f89e3e77000 rw-p 00000000 00:00 0
7f89e3e77000-7f89e3e98000 r-xp 00000000 08:01 6032224                    /lib/x86_64-linux-gnu/ld-2.19.so
7f89e408a000-7f89e408d000 rw-p 00000000 00:00 0
7f89e4090000-7f89e4091000 rw-p 00000000 00:00 0
7f89e4091000-7f89e4092000 rw-s 00000000 00:21 20198465                   /home/oslab/oslaba24/4/mmap/file.txt
7f89e4092000-7f89e4097000 rw-p 00000000 00:00 0
7f89e4097000-7f89e4098000 r--p 00020000 08:01 6032224                    /lib/x86_64-linux-gnu/ld-2.19.so
7f89e4098000-7f89e4099000 rw-p 00021000 08:01 6032224                    /lib/x86_64-linux-gnu/ld-2.19.so
7f89e4099000-7f89e409a000 rw-p 00000000 00:00 0
7ffd9f69e000-7ffd9f6bf000 rw-p 00000000 00:00 0                          [stack]
7ffd9f744000-7ffd9f747000 r--p 00000000 00:00 0                          [vvar]
7ffd9f747000-7ffd9f749000 r-xp 00000000 00:00 0                          [vdso]
ffffffffff600000-ffffffffff601000 r-xp 00000000 00:00 0                  [vsyscall]
-----------------------------------------------------------
```

Η απεικόνιση που αντιστοιχεί στην κλήση αυτή της mmap() είναι υπογραμμισμένη στην παραπάνω εικόνα.

6. Χρησιμοποιούμε την mmap() για να δεσμεύσουμε έναν νέο buffer, διαμοιραζόμενο αυτή τη φορά μεταξύ διεργασιών με μέγεθος μίας σελίδας (χρησιμοποιούμε το flag **MAP_SHARED**).

```
Step 6: Use mmap(2) to allocate a shared buffer of size equal to 1 page. Initialize the buffer and print the n
ew mapping information that has been created.


Virtual Memory Map of process [14248]:
00400000-00403000 r-xp 00000000 00:21 20198008                           /home/oslab/oslaba24/4/mmap/mmap
00602000-00603000 rw-p 00002000 00:21 20198008                           /home/oslab/oslaba24/4/mmap/mmap
01dab000-01dcc000 rw-p 00000000 00:00 0                                  [heap]
7f89e3acc000-7f89e3c6d000 r-xp 00000000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3c6d000-7f89e3e6d000 ---p 001a1000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3e6d000-7f89e3e71000 r--p 001a1000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3e71000-7f89e3e73000 rw-p 001a5000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3e73000-7f89e3e77000 rw-p 00000000 00:00 0
7f89e3e77000-7f89e3e98000 r-xp 00000000 08:01 6032224                    /lib/x86_64-linux-gnu/ld-2.19.so
7f89e408a000-7f89e408d000 rw-p 00000000 00:00 0
7f89e408f000-7f89e4090000 rw-p 00000000 00:00 0
7f89e4090000-7f89e4091000 rw-s 00000000 00:04 2140428                    /dev/zero (deleted)
7f89e4091000-7f89e4092000 rw-p 00000000 00:21 20198465                   /home/oslab/oslaba24/4/mmap/file.txt
7f89e4092000-7f89e4097000 rw-p 00000000 00:00 0
7f89e4097000-7f89e4098000 r--p 00020000 08:01 6032224                    /lib/x86_64-linux-gnu/ld-2.19.so
7f89e4098000-7f89e4099000 rw-p 00021000 08:01 6032224                    /lib/x86_64-linux-gnu/ld-2.19.so
7f89e4099000-7f89e409a000 rw-p 00000000 00:00 0
7ffd9f69e000-7ffd9f6bf000 rw-p 00000000 00:00 0                          [stack]
7ffd9f744000-7ffd9f747000 r--p 00000000 00:00 0                          [vvar]
7ffd9f747000-7ffd9f749000 r-xp 00000000 00:00 0                          [vdso]
ffffffffff600000-ffffffffff601000 r-xp 00000000 00:00 0                  [vsyscall]
-----------------------------------------------------------
```

Η ζητούμενη απεικόνιση φαίνεται στην παραπάνω εικόνα, υπογραμμισμένη. Βλέπουμε από τα permissions (rw-**s**, δηλαδή read, write και shared), ότι πράγματι ο buffer είναι shared μεταξύ διεργασιών.

Στο σημείο αυτό καλείται η συνάρτηση fork() και δημιουργείται μία νέα διεργασία.

7. Τυπώνουμε τον χάρτη εικονικής μνήμης της διεργασίας πατέρα και της διεργασίας παιδιού αντίστοιχα.

```
Step 7: Print parent's and child's map.

Parent show_maps():

Virtual Memory Map of process [14248]:
00400000-00403000 r-xp 00000000 00:21 20198008                           /home/oslab/oslaba24/4/mmap/mmap
00602000-00603000 rw-p 00002000 00:21 20198008                           /home/oslab/oslaba24/4/mmap/mmap
01dab000-01dcc000 rw-p 00000000 00:00 0                                  [heap]
7f89e3acc000-7f89e3c6d000 r-xp 00000000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3c6d000-7f89e3e6d000 ---p 001a1000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3e6d000-7f89e3e71000 r--p 001a1000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3e71000-7f89e3e73000 rw-p 001a5000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3e73000-7f89e3e77000 rw-p 00000000 00:00 0
7f89e3e77000-7f89e3e98000 r-xp 00000000 08:01 6032224                    /lib/x86_64-linux-gnu/ld-2.19.so
7f89e408a000-7f89e408d000 rw-p 00000000 00:00 0
7f89e408f000-7f89e4090000 rw-p 00000000 00:00 0
7f89e4090000-7f89e4091000 rw-s 00000000 00:04 2140428                    /dev/zero (deleted)
7f89e4091000-7f89e4092000 rw-p 00000000 00:21 20198465                   /home/oslab/oslaba24/4/mmap/file.txt
7f89e4092000-7f89e4097000 rw-p 00000000 00:00 0
7f89e4097000-7f89e4098000 r--p 00020000 08:01 6032224                    /lib/x86_64-linux-gnu/ld-2.19.so
7f89e4098000-7f89e4099000 rw-p 00021000 08:01 6032224                    /lib/x86_64-linux-gnu/ld-2.19.so
7f89e4099000-7f89e409a000 rw-p 00000000 00:00 0
7ffd9f69e000-7ffd9f6bf000 rw-p 00000000 00:00 0                          [stack]
7ffd9f744000-7ffd9f747000 r--p 00000000 00:00 0                          [vvar]
7ffd9f747000-7ffd9f749000 r-xp 00000000 00:00 0                          [vdso]
ffffffffff600000-ffffffffff601000 r-xp 00000000 00:00 0                  [vsyscall]
-----------------------------------------------------------

Child show_maps():

Virtual Memory Map of process [14249]:
00400000-00403000 r-xp 00000000 00:21 20198008                           /home/oslab/oslaba24/4/mmap/mmap
00602000-00603000 rw-p 00002000 00:21 20198008                           /home/oslab/oslaba24/4/mmap/mmap
01dab000-01dcc000 rw-p 00000000 00:00 0                                  [heap]
7f89e3acc000-7f89e3c6d000 r-xp 00000000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3c6d000-7f89e3e6d000 ---p 001a1000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3e6d000-7f89e3e71000 r--p 001a1000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3e71000-7f89e3e73000 rw-p 001a5000 08:01 6032227                    /lib/x86_64-linux-gnu/libc-2.19.so
7f89e3e73000-7f89e3e77000 rw-p 00000000 00:00 0
7f89e3e77000-7f89e3e98000 r-xp 00000000 08:01 6032224                    /lib/x86_64-linux-gnu/ld-2.19.so
7f89e408a000-7f89e408d000 rw-p 00000000 00:00 0
7f89e408f000-7f89e4090000 rw-p 00000000 00:00 0
7f89e4090000-7f89e4091000 rw-s 00000000 00:04 2140428                    /dev/zero (deleted)
7f89e4091000-7f89e4092000 rw-p 00000000 00:21 20198465                   /home/oslab/oslaba24/4/mmap/file.txt
7f89e4092000-7f89e4097000 rw-p 00000000 00:00 0
7f89e4097000-7f89e4098000 r--p 00020000 08:01 6032224                    /lib/x86_64-linux-gnu/ld-2.19.so
7f89e4098000-7f89e4099000 rw-p 00021000 08:01 6032224                    /lib/x86_64-linux-gnu/ld-2.19.so
7f89e4099000-7f89e409a000 rw-p 00000000 00:00 0
7ffd9f69e000-7ffd9f6bf000 rw-p 00000000 00:00 0                          [stack]
7ffd9f744000-7ffd9f747000 r--p 00000000 00:00 0                          [vvar]
7ffd9f747000-7ffd9f749000 r-xp 00000000 00:00 0                          [vdso]
ffffffffff600000-ffffffffff601000 r-xp 00000000 00:00 0                  [vsyscall]
-----------------------------------------------------------
```

Παρατηρούμε ότι οι δύο χάρτες μνήμης είναι ίδιοι. Αυτό είναι λογικό αφού, όπως έχουμε ξαναπεί, η fork() δημιουργεί ένα ακριβές αντίγραφο της διεργασίας πατέρα. Εξάλλου, η

διεργασία παιδί δεν έχει κάνει ακόμα κάποιου είδους δέσμευση μνήμης, οπότε δεν θα έπρεπε να παρατηρούμε κάποια διαφοροποίηση στον χάρτη εικονικής μνήμης.

8. Με χρήση της get_physical_address() τυπώνουμε την φυσική διεύθυνση του private buffer (βήμα 3) για τις διεργασίες πατέρα και παιδί.

```
Step 8: Find the physical address of the private heap buffer (main) for both the parent and the child.

Parent = c3ffd000
Child = c3ffd000
```

Παρατηρούμε ότι οι δύο φυσικές αντιστοιχίσεις είναι ίδιες. Αυτό συμβαίνει διότι το λειτουργικό σύστημα ακολουθεί πολιτική copy-on-write, οπότε αφού το παιδί δεν έχει γράψει στον private buffer, δεν έχει γίνει ακόμη δέσμευση φυσικής μνήμης για το αντίγραφο του buffer που ανήκει στο παιδί.

9. Γράφουμε στον private buffer από την διεργασία παιδί και επαναλαμβάνουμε το βήμα 8.

```
Step 9: Write to the private buffer from the child and repeat step 8. What happened?

Parent = c3ffd000
Child = b3f29000
```

Παρατηρούμε ότι τώρα η φυσική απεικόνιση του buffer της διεργασίας παιδιού άλλαξε. Αυτό είναι αναμενόμενο, για τον λόγο που εξηγήσαμε παραπάνω (πολιτική copy-on-write).

10. Γράφουμε στον shared buffer από τη διεργασία παιδί και τυπώνουμε τη φυσική του διεύθυνση για τις διεργασίες πατέρα και παιδί.

```
Step 10: Write to the shared heap buffer (main) from child and get the physical address for both the parent an
d the child. What happened?

Parent = 17a41000
Child = 17a41000
```

Παρατηρούμε ότι η φυσική απεικόνιση του shared buffer παραμένει η ίδια και για τις δύο διεργασίες. Αυτό είναι λογικό, αφού ο buffer είναι shared αυτή τη φορά.

11. Με χρήση της mprotect() απαγορεύουμε τις εγγραφές στον shared buffer για την διεργασία παιδί. Ύστερα με την show_va_info() τυπώνουμε την απεικόνιση του shared buffer στον χάρτη μνήμης των δύο διεργασιών.

```
Step 11: Disable writing on the shared buffer for the child. Verify through the maps for the parent and the ch
ild.

Parent:

7f89e4090000-7f89e4091000 rw-s 00000000 00:04 2140428                    /dev/zero (deleted)

Child:

7f89e4090000-7f89e4091000 r--s 00000000 00:04 2140428                    /dev/zero (deleted)
```

Η απαγόρευση έγινε επιτυχώς, αφού πλέον τα permissions του παιδιού άλλαξαν (r--s, δεν υπάρχει το write).

12. Τέλος, με χρήση της munmap() αποδεσμεύουμε όλους τους buffers (heap_private_buf, heap_shared_buf, file_shared_buf) και για τις δύο διεργασίες.

1.2 Παράλληλος υπολογισμός Mandelbrot με διεργασίες αντί για νήματα

1.2.1 Semaphores πάνω από διαμοιραζόμενη μνήμη

Ερωτήσεις:

1. Η υλοποίηση που περιμένουμε να έχει καλύτερη επίδοση είναι με threads. Γενικά, τα threads είναι πιο “ελαφρύς” μηχανισμός σε σχέση με τα processes. Πιο συγκεκριμένα, όταν δημιουργούμε μία νέα διεργασία, χρειάζεται να αντιγράψουμε όλα τα περιεχόμενα του πατέρα (μάλιστα ο διαμοιρασμός μνήμης στην πραγματικότητα γίνεται μέσω αρχείων, που δημιουργεί ένα παραπάνω overhead). Αντίθετα, τα threads έχουν εξαρχής διαμοιραζόμενη μνήμη και δεν χρειάζεται κάποια παραπάνω διαδικασία, όπως στα processes. Έτσι, γενικά τα threads είναι προτιμότερα για παράλληλη επεξεργασία.

Στην συγκεκριμένη περίπτωση, βέβαια, ο φόρτος του προγράμματος είναι πολύ μικρός για να μπορούμε να δούμε κάποια ουσιαστική διαφορά στον χρόνο.

1.2.2 Υλοποίηση χωρίς Semaphores

Ερωτήσεις:

1. Στην παρούσα υλοποίηση, η i-οστή διεργασία γράφει στις γραμμές i, i+N, i+2N, … του buffer μεγέθους y_chars x x_chars, όπου N ο συνολικός αριθμός διεργασιών (NPROCS). Έτσι, δεν απαιτείται κάποιου είδους συγχρονισμός μεταξύ των διεργασιών-παιδιών, αφού κάθε διεργασία γράφει σε διαφορετικό κομμάτι μνήμης. Το μόνο που χρειάζεται, είναι η αρχική διεργασία να κάνει wait για όλα της τα παιδιά, έτσι ώστε ο buffer να έχει τις σωστές τιμές, προτού τυπωθεί.

Αν ο buffer είχε διαστάσεις NPROCS x x_chars:

1. Στην τετριμμένη περίπτωση που NPROCS ≥ y_chars, έχουμε ακόμη περισσότερο χώρο από πριν, οπότε η υλοποίηση μπορεί να παραμείνει ως έχει.
2. Στην περίπτωση που NPROCS < y_chars, έχουμε πιο περιορισμένο χώρο, και άρα πρέπει να τυπώνουμε τον buffer όταν αυτός γεμίζει, προκειμένου να μην γίνεται overwrite από τις νέες τιμές που έρχονται. Εδώ ο συγχρονισμός είναι αναπόφευκτος: Θα πρέπει η i-οστή διεργασία να περιμένει να τυπωθεί ο buffer, προτού γράψει σε αυτόν την i+N γραμμή του συνόλου Mandelbrot. Αντίστοιχα, αυτό ισχύει και για τις γραμμές i+2N, i+3N, ... Όπως φαίνεται, υπάρχει ένα tradeoff μεταξύ αποθηκευτικού χώρου και χρόνου εκτέλεσης.