Maurely Acosta (PID: 3914479)
Tomas Ortega (PID: 5677483)
3/26/2018

# **Project 3 Report**

In project 3 we modified the file **slob.c** to use the best-fit algorithm to allocate a page and a block of memory. We implemented two system calls to compare the degree of fragmentation between first-fit and best-fit algorithms.

Each member contributed to the project. Maurely Acosta tested both the first fit and best fit algorithm, wrote code and fixed errors in slob_page_alloc(), sys_get_slob_amt_claimed, and sys_get_slob_amt_free. She also contributed to writing the report. Tomas Ortega wrote the helper function slob_check_page, contributed to the report, and tested both the first fit and best fit algorithms.

First, we implemented both system calls in the existing **slob.c** file to test the first-fit algorithm and followed the instructions in project 2 to do so. To do this we created two global arrays, one for the amount of memory claimed by the slob allocator and the other one for the amount of memory in the list of free pages. We used a flag to check if we are iterating through the free_slob_small list. To test the system calls we wrote the program **testProject3.c** which allocates different amounts of memory using the function malloc() and then prints the values returned by the system calls.

To implement the best fit algorithm, we first modified the function slob_alloc(). We decided that to find the best page we need to keep the amount of memory that would remain in the page if we decide to choose it for the allocation. To find the amount of free memory remaining after the allocation we decided to implement a helper function slob_check_page().

This function works exactly the same as slob_page_alloc() but instead of allocating the block, it returns the amount of remaining if we decide to choose that page. So, while we iterate through the list we need to call slob_check_page() and compare it to the best page yet and keep the one with the smallest remaining memory.

The first fit algorithm had 695587 bytes free and 174 bytes claimed. We can see that the memory is highly fragmented and when the system tried to claim memory the remaining free bytes are too small to be used. The advantage is that memory allocation will be done fast since we don't need to traverse the entire list.

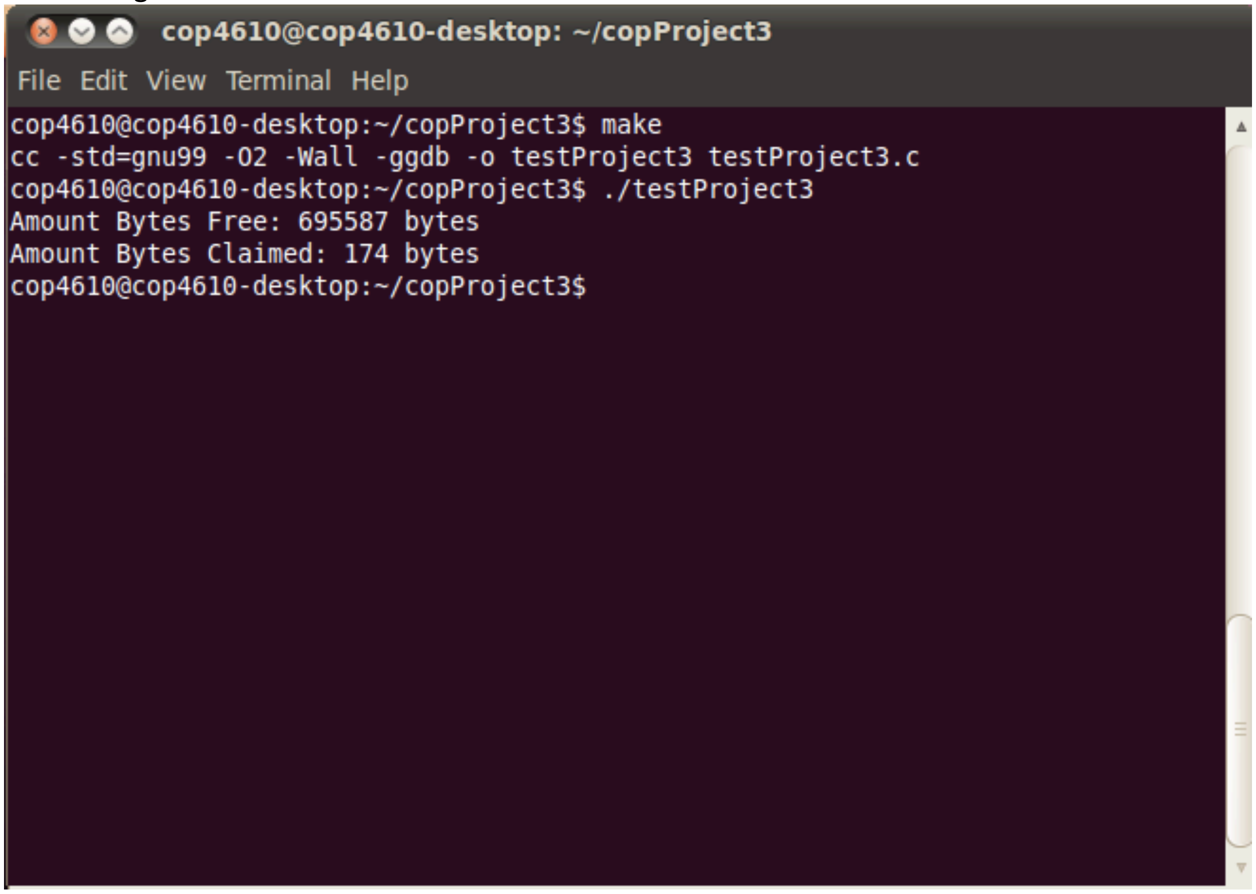
The best fit algorithm had 212190 bytes free and 138 bytes claimed. There is still fragmentation with this algorithm but it is much less fragmentation than the first fit algorithm. The amounts of free bytes still exceed that amount we are trying to allocate. The advantage of this algorithm is we reduce fragmentation by allocation memory in a size that fits just right. The disadvantage is that we must traverse the entire list in order to find the best fit.

Maurely Acosta (PID: 3914479)
Tomas Ortega (PID: 5677483)
3/26/2018

To emphasize, from the values that we got from the system calls we can arrive at the conclusion that the first-fit algorithm is better than best-fit for small memory allocation. The best-fit algorithm ends up leaving tiny spaces of memory that will never be used causing fragmentation. Best-fit is also slower than first-fit since it has to search through the entire list to find the best candidate.

Maurely Acosta (PID: 3914479)
Tomas Ortega (PID: 5677483)
3/26/2018

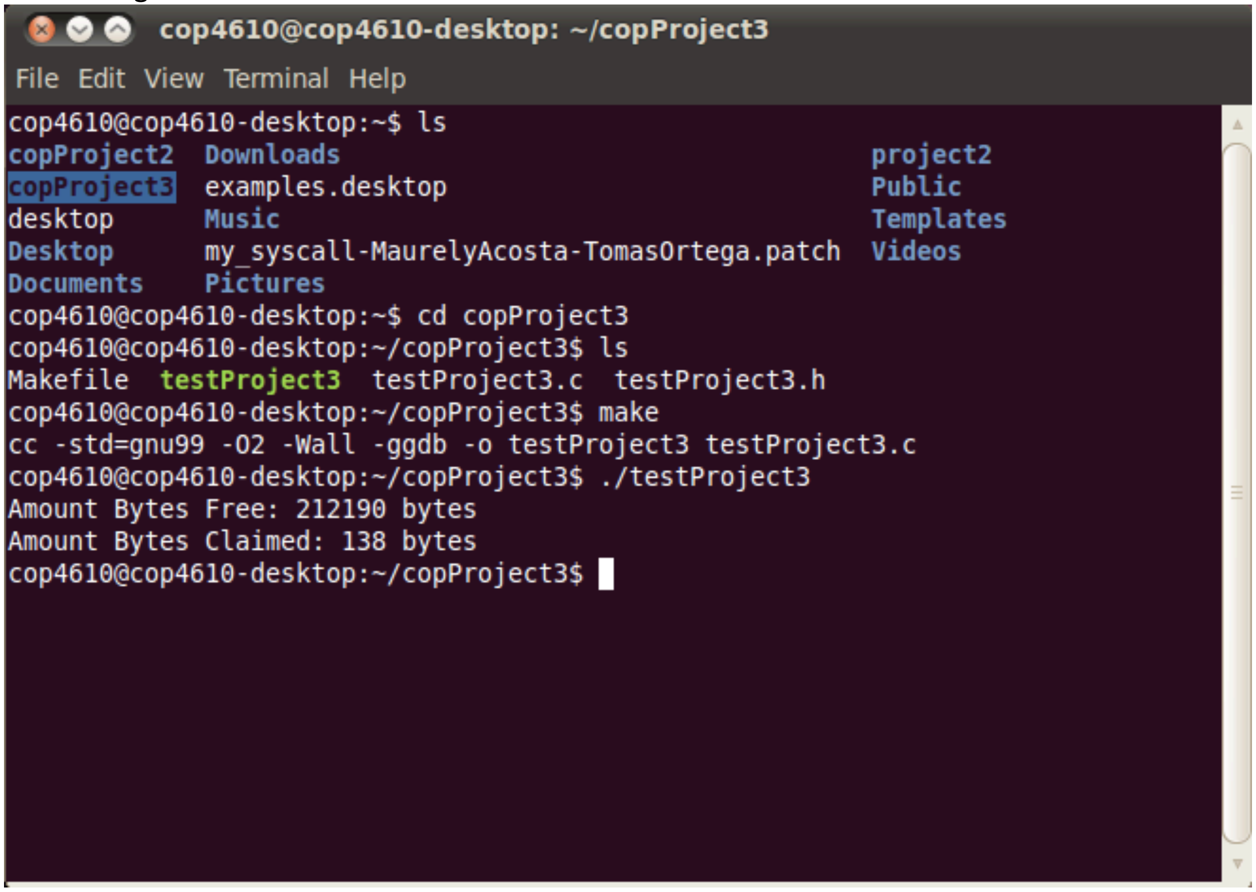
First Fit Algorithm:


cop4610@cop4610-desktop: ~/copProject3
File Edit View Terminal Help

```
cop4610@cop4610-desktop:~/copProject3$ make
cc -std=gnu99 -O2 -Wall -ggdb -o testProject3 testProject3.c
cop4610@cop4610-desktop:~/copProject3$ ./testProject3
Amount Bytes Free: 695587 bytes
Amount Bytes Claimed: 174 bytes
cop4610@cop4610-desktop:~/copProject3$
```

Maurely Acosta (PID: 3914479)
Tomas Ortega (PID: 5677483)
3/26/2018

Best Fit Algorithm:


cop4610@cop4610-desktop: ~/copProject3
File Edit View Terminal Help

```
cop4610@cop4610-desktop:~$ ls
copProject2  Downloads                                  project2
copProject3  examples.desktop                           Public
desktop      Music                                      Templates
Desktop      my_syscall-MaurelyAcosta-TomasOrtega.patch  Videos
Documents    Pictures
cop4610@cop4610-desktop:~$ cd copProject3
cop4610@cop4610-desktop:~/copProject3$ ls
Makefile  testProject3  testProject3.c  testProject3.h
cop4610@cop4610-desktop:~/copProject3$ make
cc -std=gnu99 -O2 -Wall -ggdb -o testProject3 testProject3.c
cop4610@cop4610-desktop:~/copProject3$ ./testProject3
Amount Bytes Free: 212190 bytes
Amount Bytes Claimed: 138 bytes
cop4610@cop4610-desktop:~/copProject3$ 
```

Maurely Acosta (PID: 3914479)
Tomas Ortega (PID: 5677483)
3/26/2018

## References

Mackall, Matt. Linux/Mm/Slob.c - Elixir - Bootlin, elixir.bootlin.com/linux/latest/source/mm/slob.c.