Jianzhi Li
CS444 Project 4 Write-up

1 Design Plan For SLOB Best-Fit Algorithm

Before to design the best-fit algorithm. I was reviewed the knowledge about memory management and study each function of the original slob.c in the kernel. Also I did some research by read a few documents created by others that compared the difference between the first-fit algorithm and the best-fit algorithm. As the result, I should have a function that is can compare the size of memory block and find out a memory block that has the best-fit of the memory request from users. In another word, the best-fit algorithm is an algorithm that is can find the smallest memory block that can fit user's request. For example, a user requests a memory block as 24, and in the list, we have two memory blocks, the size of the first memory block is 48 and the second is 36. The first-fit algorithm will allocate the first memory block to the user since that is the first memory block can feet the user's request. But the best-fit algorithm will allocate the second memory block to the user since that is enough to fit the user's request and does not waste a bigger memory block that does not need. However, to makes my assumptions as above can come true, I need modify a function which is slob_alloc that is using next-fit scheme can find a page that is enough to fit a incoming requests.

Another function I want to modify in my design plan is the slob_page_alloc function. The reason I want to modify it because after I went through each function I found that function is the keyword for this project. This function was implemented by first-fit algorithm that is what I want to use best-fit algorithm to instead of it. For doing that, I plan to using linked-list to traverse the whole list of memory blocks then sort the whole list.

For compare the fragmentation suffered by first-fit algorithm and best-fit algorithm. As a hint to finish this requirement, I need add additional system calls to compare those two different algorithms. For doing that, after I did some research that I need add my new system calls in syscall_64.tbl that is under the kernel directory and allowed me to add new system call lists, and the unistd.h that is under the asm directory and that is allowed me to add the system call numbers.

2 Work Log

1. Tuesday, 11/17/15. Start research on this project. I looked at a few documents on the internet, those documents are compared the first-fit algorithm and the best fit algorithm. Also reviewed the chapter 12 which is memory management especially on the session of SLAB.
2. Wednesday, 11/18/15. Created the project folder on the github. I have going through the original code of slob.c which is used first-fit algorithm to allocate pages.
3. Thursday, 11/19/15. Start modify slob_alloc function and slob_page_alloc function.
4. Friday, 11/20/15. Still working on slob_alloc and slob_page_alloc functions. Also trying to figure out the test method for this project.
5. Sunday, 11/21/15. The slob_alloc and slob_page_alloc can complied by rebuilding the kernel. Enabled the SLOB in General Setup in kernel ny seleted the SLOB to be the allocator.
6. Monday, 11/22/15. Working on testing method to make sure my implementation to match the requirement. Modified
7. Friday, 11/27/15. Test system call, and rebuild kernel by using best_fit slob. The test outcome is not as expect but the whole process is finished.

3 Questions

1. The main point of this assignment is to let me have a better understand of Linux kernel memory management, also the skills to use system calls. For understand the Linux kernel memory management, the comparison of first-fit algorithm and best-fit algorithm can makes me understand the difference performance between those two algorithms, also advantages and disadvantages of those two algorithms. For understand system calls, the way to add system calls in this project that is lead me to have a general idea that is how to finished the process to change some functions or features in the kernel. However, change something in the kernel that is never be just add few lines of code in the source file and complies it. That is about to learn and understand how the whole kernel trees working together.

2. To approach the problems in this project. I personally did a many of researches on the difference and theories for both first-fit algorithm and best-fit algorithm. Also I went through the slob.c file to try to understand how each function work for the SLOB allocator. Of course, I need to know how first-fit algorithm can distribute memory block in SLOB. For using best-fit algorithm to instead of the first-fit algorithm that I used linked-list and to traverse each memory block in the list, I need find the best-bit size memory block to fit users request. And for ideally situation, if in the memory block list there has not have a best-fit block to fit the user request then the function will create a new page that can fits the request. For fragmentations, I think I cannot control it since that is from the original ideas the advantages and disadvantages the best-fit algorithm will have.

   To find out how to add system calls that used me a long time since some documents I read online that is has different file names and kernel version. But that is not that hard to write my system calls in those system files since that is same like last few projects that I just follow the syntax and the way the original file used to create new system calls.

3. As previously projects in this course, I realized myself is not a good kernel tester. My testing skills is lack for the kernel testing. The basically what I did is to rebuild the kernel to see if my source file has any bugs. And to see if system calls can return what I want to see or I can say the result what I think that is correct for this project.

4. I learned the difference between first-fit algorithm and best-fit algorithm especially in their performance. Also, after the pre-research for this project, I have some general ideas for Linux kernel memory allocator and how that works for the system. Otherwise, I learned how to write system calls and which files I should looking for to write system calls.

4 Verison Control Log

commit 81b91d70e780cb271a5840a459f0486c99735ab3
Author: lijian < *lijian@os − class.engr.oregonstate.edu* >
Date: Mon Nov 23 22:25:20 2015 -0800
writeup finised except the version log
commit 408311531f86b66bc115b5da58254c5037a19175
Author: lijian < *lijian@os − class.engr.oregonstate.edu* >
Date: Sun Nov 22 23:49:42 2015 -0800
update writeup and Makefile
commit af7e1de1cbc6df50eac63566c7a980247e49e0d0
Author:lijian < *lijian@os − class.engr.oregonstate.edu* >
Date: Sat Nov 21 22:39:09 2015 -0800
final project create
commit 7cf71f6e071001a06636c513729fd49ab5b26592
Author: lijian < *lijian@os − class.engr.oregonstate.edu* >
Date: Fri Nov 20 00:10:48 2015 -0800
update project 4 writeup
commit b1d0c2fbcd0febe3dab9a037315c827cc7b08ff5
Author: lijian < *lijian@os − class.engr.oregonstate.edu* >
Date: Fri Nov 20 00:06:27 2015 -0800
problem1 has core dump and cannot fix it on time
commit 24a7b6cc488b5e950ff90097b358a4c2bde5c842
Author: lijian < *lijian@os − class.engr.oregonstate.edu* >
Date: Thu Nov 19 22:51:25 2015 -0800
Barber is finished
commit e71aa1c23c938069f4439ff1d9c3916c4ac8c2d0
Author: lijian < *lijian@os − class.engr.oregonstate.edu* >
Date: Thu Nov 19 20:10:16 2015 -0800
unfinied barber
commit 40fb05806f2bdfcfdc60fdbf73bf8dd67e4ab058
Author: lijian < *lijian@os − class.engr.oregonstate.edu* >
Date: Wed Nov 18 20:56:14 2015 -0800
Project 4 created
commit 7958f6fb697f0ae9c1d427c8626df63fb42bc4b0
Author: lijian < *lijian@os − class.engr.oregonstate.edu* >
Date: Tue Nov 17 00:00:31 2015 -0800
con4 files update