Operating Systems II HW4 Writeup

June 8, 2018

Prepared for

Dr. Kevin McGrath

Prepared by

Max Moulds
Monica Sek
Nicholas Wong

Abstract

This document is the writeup for homework 4 of Operating Systems II Spring term 2018, written by Group 20.

Contents

1 Design Plan

The current algorithm for the slob.c is a first-fit algorithm, which simply allocates the first spot that is open. To implement a best-fit algorithm, we will simply need to add a checking loop before the memory is allocated, to test if that allocation is truly the best fit or not. In order to accomplish this, we can create a temporary variable which will hold our best allocation size. We can then iterate through all available spots, while testing if that spot has the smallest difference or not. Once we find it, we will continue with the first-fit algorithm and allow it to allocate that memory.

2 Version Control Log

Author	Date	Message
nickywongdong	2018-06-07	adding the initial design plan, and additional files/directories
nickywongdong	2018-06-07	initial attempt at best fit slob
nickywongdong	2018-06-07	adding program to test for output
nickywongdong	2018-06-08	forgot to select Slob.c in config. Trying to run tests on original slob.c for Fragmentation results using syscalls
nickywongdong	2018-06-08	adding necessary syscalls and backing up related files
nickywongdong	2018-06-08	adding linkage for syscalls in slob.c
nickywongdong	2018-06-08	attempting to loop through to find best fit using a minimum value and continually updating it when we find a better delta
nickywongdong	2018-06-08	need to make best a pointer
nickywongdong	2018-06-08	maybe need to define an entrypoint for for loop
nickywongdong	2018-06-08	stuck on boot, attempting to run original slob.c with test.c to see output
nickywongdong	2018-06-08	more attempts at implementing best fit algorithm
nickywongdong	2018-06-08	finished running best-fit algorithm test.c
nickywongdong	2018-06-08	updating with best-fit slob.c
nickywongdong	2018-06-08	add patch file

3 Work Log

June 7	Began assignment, researching from the description, and pulling examples from online sources
June 7	Began design, logged with design.md
June 8	Began implementation of best fit algorithm, and testing programs for Fragmentation
June 8	Write up and review

4 Additional Questions

1) What do you think the main point of the assignment is?

   This assignment further familiarize oneself with kernel development specifically memory management. It also introduces memory management algorithms and explores the strengths and weaknesses of each in terms of Fragmentation.

2) How did you personally approach the problem? Design the decisions, algorithm, etc.

   First we began a design plan for how to implement the best-fit algorithm within slob.c. After looking at examples of the differences between the original first-fit algorithm, and the best-fit algorithm, we began to implement the design plan onto slob.c. Diving right into the slob.c file was the easiest way to start. Then adding some syscalls to test was the next step. After that hurdle the project seemed to get easier.

3) How did you ensure your solution was correct? Testing details, for instance.

   We used a calculation for fragmentation by creating syscalls which would allow us to determine the amount of memory claimed, and freed. We ran this testing program with the kernel while it was using the first-fit slob.c, and while it was running the best-fit slob.c. We noticed a significant difference in Fragmentation.
   The following test program used to determine Fragmentation:

```
for (i = 0; i < 5; i++)
  Fragmentation = (float)freeMemory / (float)claimedMemory;
  printf("Claimed Memory: %lu\n", claimedMemory);
  printf("Free Memory: %lu\n", freeMemory);
  printf("Fragmentation: %f\n", Fragmentation);
  printf("-----\n");
  sleep(1);
```

   Testing output (first fit - original slob.c):

```
Running 5 tests:
Claimed Memory: 5543936
Free Memory: 395614
Fragmentation: 0.071406
-----
Claimed Memory: 5543936
Free Memory: 396168
Fragmentation: 0.071460
-----
Claimed Memory: 5543936
Free Memory: 396168
Fragmentation: 0.071460
-----
Claimed Memory: 5543936
Free Memory: 396500
Fragmentation: 0.071520
-----
Claimed Memory: 5543936
Free Memory: 396500
Fragmentation: 0.071520
-----
```

Testing output (best fit - updated slob.c)

```
Running 5 tests:
Claimed Memory: 5396480
Free Memory: 241057
Fragmentation: 0.044717
-----
Claimed Memory: 5396480
Free Memory: 241443
Fragmentation: 0.044741
-----
Claimed Memory: 5396480
Free Memory: 241443
Fragmentation: 0.044741
-----
Claimed Memory: 5396480
Free Memory: 241443
Fragmentation: 0.044741
-----
Claimed Memory: 5396480
Free Memory: 241539
Fragmentation: 0.044759
-----
```

4) What did you learn?

In addition to learning more about kernel development and memory management tools that helped produce usable information between git commits helped identify issues with syscalls and struct name changes over time. It also helped create a more understandable representation of the actual changes needed to implement a best fit algorithm over the existing slob code.

5) How should the TA evaluate your work? Provide detailed steps to prove correctness

First, gather the required file(s) and a vanilla kernel of version 3.19.2. Apply the patch file which is located at:

```
https://raw.githubusercontent.com/nickywongdong/CS444_GROUP20/master/hw4/hw4.patch
```

Once the patch is applied and the configuration files have been loaded. (Enable slob by going into General setup and then Configure standard kernel features and finally enable slob.) Compile the kernel. Run the vm. Then compile the testing program named test.c. Running the compiled executable should give the test results for the update slob.c file containing the best fit algorithm. To test the standard algorithm repeat all processes excluding applying the patch.

And that concludes our writeup for HW4. Thank you.