CS 342 Operating Systems

Project Assignment 3

Ufuk Bombar 21703486

Ant Duru 21704108

Introduction

In this assignment, we have implemented the buddy allocation algorithm. This algorithm relies on dividing the memory segment into the powers of two. This happens recursively. In our implementation, the shared segments were only used to store the data. However, as we realised the problem. The book keeping is done by a linked list in our implementation and this linked list is originally allocated in the processes heap segment. This caused some problems. First the allocation bookkeeping happens to be process specific although many processes can use the same segment. Second, since there is bookkeeping info saved in the allocated segment, current bookkeeping info will not be saved. We acknowledge these problems about our implementation, and our implementation can be adjusted to work with multiple processes with small tweaks. Although we have these small issues our core program that allocates the memory works without any problems. Also, our implementation is compatible with the pthread library.

Implementation Analysis

Our allocation algorithm implementation has an O(n) complexity, where n depicts the number of allocated parts by the algorithm. We did this with a linked list which has size and allocated attributes. When we want to allocate a segment, we iterate through the linked list and find the best fit. Then we check for the size of the current node with the request. If their sizes are the same then we simply flip the allocated but to true. If not we divide recursively.

On the other hand we have our deallocation algorithm implementation which also has O(n) complexity. The algorithm simply finds the offset distance between the segment's address and the given address. Then iterates through the linked list to find the node we want to deallocate. When we find it it simply flips the allocated bit to false. Then we run a cleanup function.This function iterates the linked list backwards and merges same sized unallocated nodes.

We also want to discuss the edge cases of this algorithm. In the beginning the linked list has only 1 element and that has the size segsize and allocated equal to false. When we try to allocate elements the size of the linked list grows. The maximum length of the linked list has been given in equation 1.

$$Maximum\ Linked\ List\ Size\ =\ Segment\ Size\ /\ Minimum\ Allocatable\ Size \tag{1}$$

Using this equation, we can calculate the maximum possible length of the linked list. In the assignment the maximum size of the segment is given 256KB and minimum request size 128. These two divides give 2024 elements. Furthermore, this implementation has the node size 17 bytes (8 byte integer as size, 8 byte next address pointer, 1 byte allocated). This means in the worst case there should be a 34KB memory space to bookkeep the memory. This means there is a 13% overhead.

Results

The result of 3 allocation and deallocations are shown in figure 1. In the figure the status of the linked list bookkeeping structure is shown. The first item in each node denotes the size of the list and the second denotes if the node is free or allocated.

```
Allocate 512 bytes:   (512, 1)->(512, 0)->
Allocate 32 bytes:    (512, 1)->(32, 1)->(32, 0)->(64, 0)->(128, 0)->(256, 0)->
Allocate 8 bytes:     (512, 1)->(32, 1)->(8, 1)->(8, 0)->(16, 0)->(64, 0)->(128, 0)->(256, 0)->
Deallocate 512 bytes: (512, 1)->(32, 1)->(32, 0)->(64, 0)->(128, 0)->(256, 0)->
Deallocate 32 bytes:  (512, 1)->(512, 0)->
Deallocate 8 bytes:   (1024, 0)->
```

Figure 1: Status of linked list after 3 alloc and dealloc calls.

In figure 2, the length of their linked list is shown. As can see there is a noise because of the random allocations. However, it is also caused by the algorithm itself. When allocating a smaller segment the algorithm divides the segment thus creates many smaller segments that are power of two. This can be seen in the beginning of the plot. In the first allocation the length of the linked list is 9. However, after deallocating every segment the length drops to 1. Also, as stated in the assignment, we have generated allocation calls with minimum of 128 and maximum of 1024 bytes. Lastly, there are 100 allocation and 100 free calls to our library.

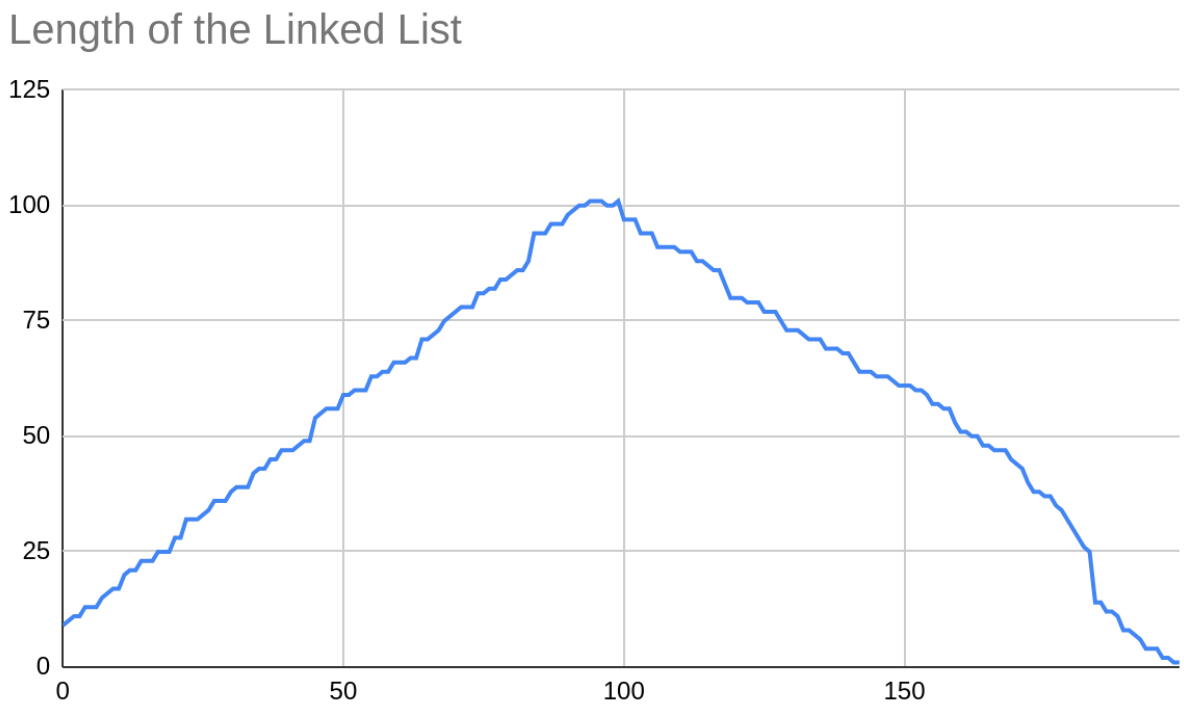

Figure 2:Length of the linked list.

In the last figure, we have displayed the chunk sizes of randomly generated spaces. This can be seen in figure 3.

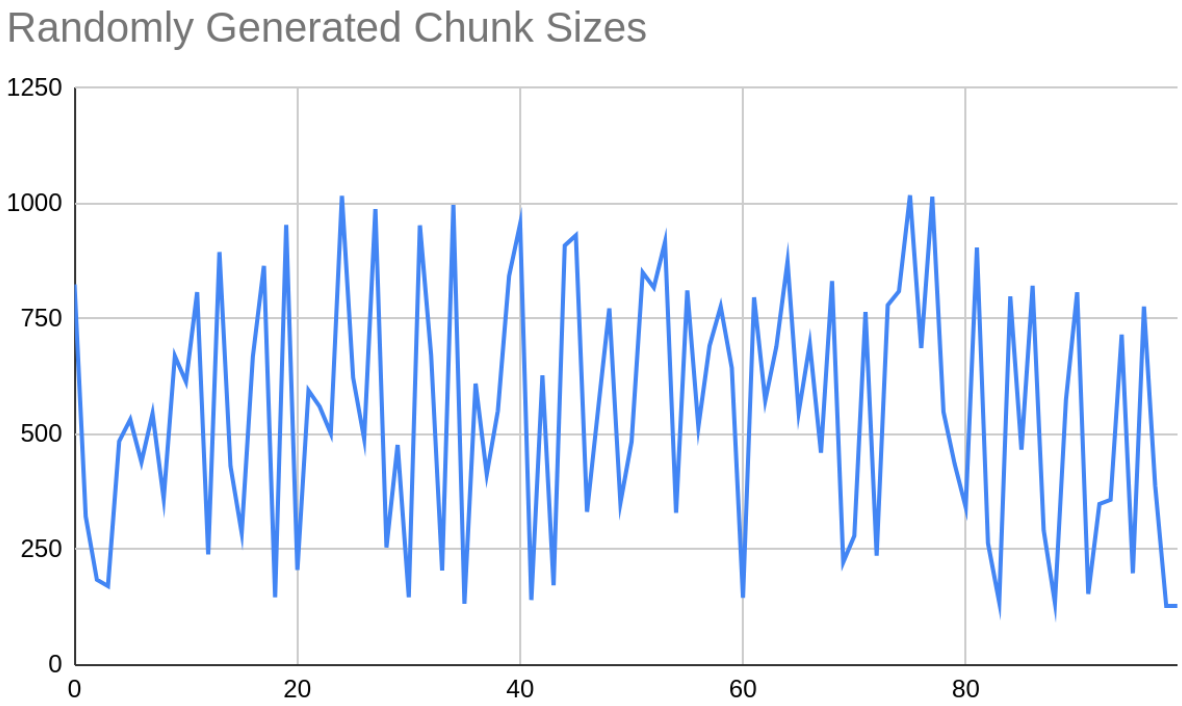

Figure 3: Randomly generated chunk sizes.

Conclusion

In conclusion, there are many allocation algorithms. Buddy allocation algorithm is one of them and is relatively easy to implement. Also, with an overhead percentage of 13 this implementation can be considered for real life use.

References

1. Knuth, Donald (1997). *Fundamental Algorithms*. The Art of Computer Programming. **1** (Second ed.). Reading, Massachusetts: Addison-Wesley. pp. 435–455. ISBN 0-201-89683-4.