Adrian deCola
Professor Mwaura
Discrete Structures, Algorithms, and their Applications within Computer Systems
Mar 26, 2023

Lab 8 - Written Exercises

1) Explain what you think the worst-case, Big-O complexity and the best-case, Big-O complexity of merge sort is. Why do you think that?

In the best case scenario, we could say the list is already sorted but we still must split the array into slices of one and then merge the arrays together until they are one. There is log(n) levels of merging each requiring copying over the size of the list, n. Therefore, the Big-Oh complexity in the best case is O(nlog(n)). In the worst case, we still must split the array into slices of one and then merge the arrays together until they are one. There is log(n) levels of merging each requiring copying over the size of the list, n. Therefore, the Big-Oh complexity in the worst case is also O(nlog(n)).

2) Merge sort, as we have implemented it, has a recursive algorithm embedded in the code as this is the easiest way to think about it. It is also possible to implement merge sort iteratively:

```
// Iteratively sort subarray `A[low…high]` using a temporary array
void mergesort(int A[], int temp[], int low, int high)
{
  // divide the array into blocks of size `m`
  // m = [1, 2, 4, 8, 16…]
  for (int m = 1; m <= high - low; m = 2 * m)
  {
    // for m = 1, i = 0, 2, 4, 6, 8…
    // for m = 2, i = 0, 4, 8…
    // for m = 4, i = 0, 8…
    // …
    for (int i = low; i < high; i += 2*m)
    {
      int from = i;
      int mid = i + m - 1;
      int to = min(i + 2*m - 1, high);

      merge(A, temp, from, mid, to);
    }
  }
}
```

Explain what you think the worst-case, Big-O complexity and the best-case, Big-O complexity is for this iterative merge sort. Why do you think that?

The case is the same here. In each scenario, we must loop though the first loop which has a time complexity of O(log(n)) because each time we double our jump size until we get to the end of the list. For each of those loops, in the second nested loop, we loop through, merging each individual list though the whole list requiring O(n) time. Therefore, the best and worst case Big-Oh complexity for this iterative merge sort is still O(nlog(n)).