Ali Awari

aka87

CS 205: Final Exam Question 2

**1) Formally prove that if and then**

* , when
* , when

**2) True or false - give a mathematical justification:**

a)

* **True**, functions may be bounded by a higher order, though it wouldn’t be too helpful in real world applications.
* **True**
* Bound all values by:
* **False**
* Bound all values:
* **True**
* Multiply both sides by
* **False**
* **True**, assuming that an algorithm must have at least a single step, it is lower bounded by .

g)

* **True**

h)

* **True**
* Prove that:
* For every

i)

* **False**
* Prove that:
* For every

j)

* **True**

Recall the idea of merge sort: to sort a list, divide a list in two, sort the two halves, and merge them to form a sorted whole. In class, we gave an argument that the complexity of merge sort on a list of N elements could therefore be described as M(N) = M(N/2) sort the left half + M(N/2) sort the right half + N merge the two halves = 2M(N/2) + N. (1) Noting that M(1) = 1, since sorting a list of size 1 is easy, this led to an overall complexity of M(N) = O(N ln N). Your good buddy suggests the following: If merge sort gets such good performance dividing the list into two halves and merging them, imagine how fast a merge sort would be that split the list to sort into three parts, sorted them, then merged the result.

**3) Like the recursive relation above, give a rough description of the overall worst case complexity of this tri-merge sort**

* The worst case is when ternary merge-sort must do the maximum number of comparisons on each element of the sub-array while it merges everything back.

**4) In terms of big-O, which approach has the smaller complexity?**

* Binary merge-sort:
* Ternary merge-sort:
* *Ternary* has a smaller big-O

**5) In your opinion, is it worth the additional effort and overhead it would take to implement this approach? Justify.**

* I don’t think it’s worth the overhead, in the worst case, ternary must do 1 more comparison compared to binary while merging. The benefits do not strike me as important enough to code unless given an assumption that ternary will always perform in the best case.