CS41 Lab 2: Induction and Reductions

September 7, 2021

In typical labs this semester, you'll be working on a number of problems in groups of 3-4 students. You will not be handing in solutions; the primary purpose of these labs is to have a low-pressure space to discuss algorithm design, and to gain experience collaborating with others on algorithm design and analysis. However, it will be common to have some overlap between lab exercises and homework sets.

Reductions. In computer science, a **reduction** is a way of solving one problem using another. Imagine having an algorithm for problem B, and using that algorithm as a subroutine in an algorithm that solves problem A. We say that "problem A reduces to problem B" or that we have a reduction *from* problem A *to* problem B. Reductions are a very deep algorithmic technique that can be used to make connections between problems that might initially look very very different.

Note: There is more than ninety minutes of material on this lab. Consider it a successful lab if you can mostly make it through two problems. If you do not feel fully confident when it comes to proof using induction, I strongly encourage you to focus on the initial problem first. Otherwise, I encourage you to work on whichever problems look the most interesting!

1. **Induction.** Using induction, show that the following summations hold for all $n \geq 0$.

   - $\sum_{k=0}^{n} k = \frac{n(n+1)}{2}$.
   - $\sum_{k=0}^{n} 2^k = 2^{n+1} - 1$.
   - for all positive $c \neq 1$, $\sum_{k=0}^{n} c^k = \frac{c^{n+1}-1}{c-1}$.

2. **Sorting to Half-Sorting**. In the HALF-SORT problem, you're given an array of n integers and must return an array that has the first $\lceil n/2 \rceil$ integers in sorted order. For example, if your array is $A = [5, 9, 1, 2, 6, 3]$, then a valid output of HALF-SORT(A) might be $[1, 2, 3, 9, 5, 6]$ since $1, 2, 3$ are the least elements of A.

   - Reduce the sorting problem to HALF-SORT. i.e., imagine you have an algorithm $\mathcal{A}$ for HALF-SORT, and use it to design a sorting algorithm.
   - Now, suppose that your friend claims to have an algorithm for HALF-SORT that runs in $10n$ time in the worst case. What is the runtime of your sorting algorithm? Is $10n$ a reasonable running time for HALF-SORT?

3. **Driving Reductions**. Driving directions can often be much more complicated locally than regionally. In this problem, you and your lab partners will design algorithms to get from each of your home addresses.

   - First, tell your group what the closest large city is to your home.

- Second, *individually* describe an algorithm for getting from your home to the closest large city.
- Independently, create an algorithm to get from your home to a lab partners' home by *reducing* to their local directions; i.e, by assuming you had an algorithm from their home to their closest large city.
- Finally, combine answers to get overall driving directions between your homes.