Ian Miller October 8, 2019

# 6.042 Problem Set 3

## Problem 1 *(Collaborators: Andy Kaspers)*

### Part 1(a)

$\sum F_i^2 = F_i * F_{i+1}$

Proof by induction

Let P represent the following proposition: $\sum F_i^2 = F_i * F_{i+1}$

In this proof by induction, I will prove through the use of induction that P is indeed true.

**Base case:** i = 1

$\sum F_1^2 = 1$

$F_1 * F_2 = 1$

The base case holds.

**Inductive Step:** Assume that $\sum F_i^2 = F_i * F_{i+1}$ for all $i > 0$.

I will now show that P implies P+1.

We know that $\sum F_i^2 = F(1)^2 + .... + F(i)^2 = F(n) * F(n+1)$

Therefore, the n+1 case looks like the following:

$\sum F_i^2 = F(1)^2 + .... + F(i)^2 + F(i+1)^2 = F(n) * F(n+1) + F(n+1)^2$

$= F(n) * F(n+1) + (F(n+1) * (F(n+1)))$

$= F(n+1) * F(n+2)$

This proves our inductive hypothesis and proves that P implies P+1. Therefore P holds. QED

## Part 1(b)

Let $S = \sum F_i/2^i$

$S = F_0 + 1/2 * F_1 + \sum_{i=2}^{\infty} *(F_{i-1} + F_{i-2})(1/2)^i$

$S = 1/2 + \sum_{i=2}^{\infty} F_{i-1}(1/2)^i + \sum_{i=2}^{\infty} F_{i-2}(1/2)^i$

$S = 1/2 + 1/2 * S + 1/4 * S$

$S - 1/2 * S - 1/4 * S = 1/2$

$S * (1 - 1/2 - 1/4) = 1/2$

$1/4 * S = 1/2$

$S = 2$

$\sum_{n=0}^{\infty} F_n/k^n = k/(k^2 - k - 1) = 2/(4 - 2 - 1) = 2$

# Problem 2 *(Collaborators: None)*

## Part 2(a)

| | $\Theta(x^2)$ | $O(2^x)$ | $\omega(log_2 x)$ | $\Omega(e^{\sqrt{x}})$ | $o(x)$ |
|---|---|---|---|---|---|
| ln(x) | NO | YES | YES | NO | YES |
| $1/2(x-1)^2$ | NO | YES | NO | NO | NO |
| 9 | NO | YES | NO | NO | NO |
| $2^{2x}$ | NO | NO | NO | YES | NO |
| $\sqrt{5x+3}$ | NO | YES | NO | NO | YES |
| $\alpha(x)$ | NO | NO | NO | YES | NO |

## Part 2(b)

Let P represent the following proposition: $f(n) = O(g(n))$

Let Q represent the following proposition: $(f(n))^2 = O((g(n))^2)$

In this problem, I will prove that P $->$ Q.

Based on the definition provided in the problem, we can rewrite P as:

$\exists c > 0. \exists n_0 \in \natural. \forall n > n_0. |f(n)| <= c * g(n)$

If you square both sides of the inequality, you get $f(n)^2 <= c^2 * g(n)^2$

which is equivalent to Q. Therefore P implies Q. QED

# Problem 3 *(Collaborators: Textbook, pg 253)*

## Part 3(a)

In this problem, I will prove an estimate of the following sum: $\sum_{i=1}^{\infty} 1/(2*i-1)^3$

Let $S = \sum_{i=1}^{\infty} 1/(2*i-1)^3$

and $I = \int_1^{\infty} 1/(2*i-1)^3$

In this case, since the function is not increasing, $I + 1/(2*i-1)^3 <= S <= I + 1/(2*i-1)^3$

$I = \int_1^{\infty} 1/(2*i-1)^3$

$= -1/(4*(2x-1)^2)$ (from 1 to infinity)

$= 1/4$

QED

# Problem 4 *(Collaborators: Savannah Tynan, Textbook pg 292)*

## Part 4(a)

Proof by Worst Case Scenario

Let P represent the following proposition: *Show that 2n - 3 comparisons are enough to merge three sorted lists each containing n/3 items.*

In this problem, I will prove P by calculating the number of comparisons needed to sort in the **worst-case scenario**.

If I can show that the number of calculations in this worst-case scenario is less than 2n-3, we have proven that there is never a situation where P is false, which means P must be true.

In the worst case:

A) it will take n/3 comparisons to sort each of the three sublists, recursively. That is a total of n comparisons.

B) it will take n - 3 comparisons to sort (n items are emitted in total, and once a sublist becomes empty, it takes at most 3 comparisons to sort those 3 numbers)

The reason it takes at most 3 comparisons to compare 3 numbers (for example, the last element in each of the three sublists) can be described in the following way:

Take x, y, and z as 3 integers that each belong to their own sublists. x must be compared with y and z. That takes 2 comparisons. Then, y and z must be compared. That makes 3 comparisons in total.

If A) and B) are added together, that gets us to n + n -3 comparisons, or $2n-3$ comparisons, proving P. QED.

## Part 4(b)

$T_1 = 0$

$T_2 = 3T_1 + 2 - 3 = -1$

$T_3 = 3T_2 + 4 - 3 = -2$

$T_4 = 3T_2 + 8 - 3 = -1$

$T_5 = 3T_2 + 16 - 3 = -10$

$T(n) = 3T(n/3) + 2n - 3$

$T(n) = 3T(n/3) + 2n - 3$

$= 3(3T(n/9) + 2n/3 - 3) + (n - 3)$

$= 9T(n/9) + 2n - 9) + (n - 3)$

$= 9(3T(n/27) + 2n/27 - 3) + (n - 9) + (n - 3)$

$= 27T(n/27) + 2n - 27) + (n - 27) + (n - 9) + (n - 3)$

$T_n = 3^k T_n/3^k + kn - 3^k + 1$

This is not an asymptotic improvement over MergeSort.