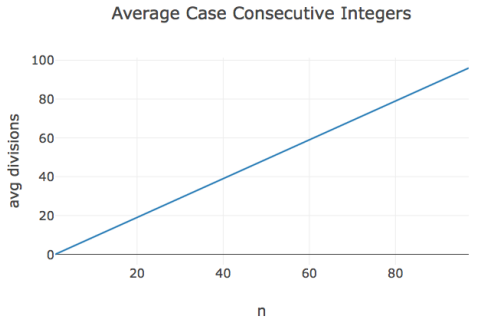
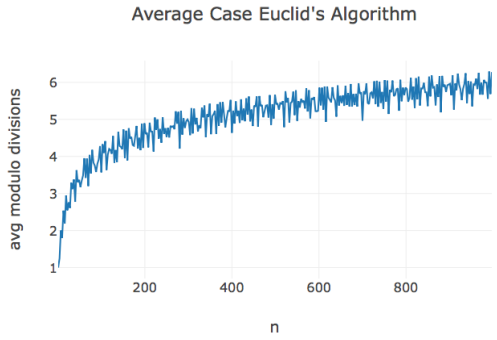
Andrew Langwell
Will Lucic

Project 1 Scatter Plot Analysis

Task 1:

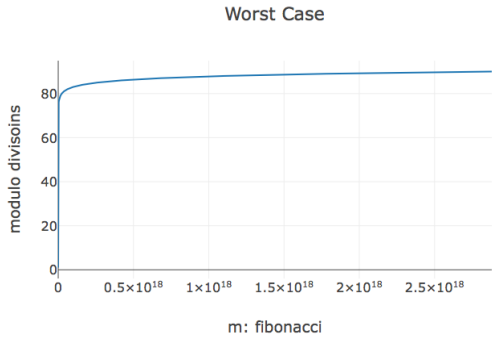
The likely average-case efficiency class for Euclid's algorithm is log n which is shown in the graph below. The consecutive integer checking algorithm (conGCD) has a average-case efficiency class of n. The efficiency class of conGCD makes sense since it's simply iterating through the lesser of its two inputs in a for loop. The efficiency class of Euclid's algorithm also makes sense since the modulo operation reduces its left operand to anywhere between 0 and one less than the original value of the operand, so on average to about half of the operand. Note that this worst case (of the reduction by modulo) can't occur in successive recursive calls in Euclid's algorithm because of the way it swaps its inputs in recursive calls, but the average of Euclid's variable size decrease (by a factor of 2) still holds. For example GCD(99, 100) = GCD(100, 99 mod 100) = GCD(100, 99) = GCD(99, 100 mod 99) = GCD(99, 1) = GCD(1, 99 mod 1) = GCD(1, 0) = 1.





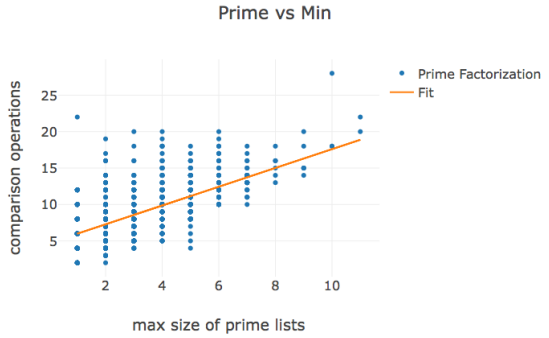
Task 2:

The worst-case efficiency class for number of modulo divisions for Euclid's algorithm is also log n. The efficiency classes are on the same order; log n. The time taken would remain on the order of log n but change by some constant factor because execution time is proportional to the number of basic operations. Our upper bound for k is 91 because the k+1th element of the fibonacci sequence is the largest fibonacci number that can be stored as a 64 bit signed or unsigned integer.



Task 3:

Within the prime factorization GCD algorithm (pfGCD), the number of comparison made when determining the intersection of the two inputs' prime factor lists is directly proportional to the number of prime factors in the longer of those two lists. To demonstrate this we generated 1000 random inputs for pfGCD in the range 1 to 1000. As expected, the variation in the number of comparisons was large (the maximum is on the order of the longer list but the minimum is just 1), but there was a clear trend show by the linear regression we plotted along with the raw data.



[a]"is log n, which is mirrored by the graph below"
[b]why not mention "variable size decrease"?
[c]meant to make above comment here
[d]yeah I'll add that now, thanks
[e]n 46 = 1.83·10e9 vs maxUsize= 2^(63) = 9.223372037E18 === ?
[f](don't think this is true)
[g]Thanks changing it now
[h]nice graph showing regression line
[i]Thanks it took a bit