Edward Morkunas

Timothy Indrieri

Brandon Artner

Math 354-Analysis of Algorithms

9/23/16

Extended Euclidean Algorithm Analysis

The main of the python program has two for loops. An outer loop to control the size of the input, and an inner for loop to control the number of trials done on that input size. In order to achieve the intended input, the calculation was as follows: (2(i+1) - 1) where i represented the incrementing of the outer for loop. This allowed us to ensure that our input stays within the number of bits we are testing. We then find a mask that has a range of 1 to our calculated number of bits, and mask our calculated max number of bits. We then pass in the masked number and another number that has been masked with the complement of our mask. Euclid’s Extended Algorithm has one while loop. The algorithm will perform its initial assignments and will enter a while loop. Next the loop will divide the inputs into a number called q. The algorithm then calculates the remainder of the two inputs using the mod operator. A check is then performed, seeing if the remainder is zero. If the remainder is equal to zero then we stop looping. If it is not equal to zero then we continue with the loop. More assignments are performed. The next four lines of the code are as follows:

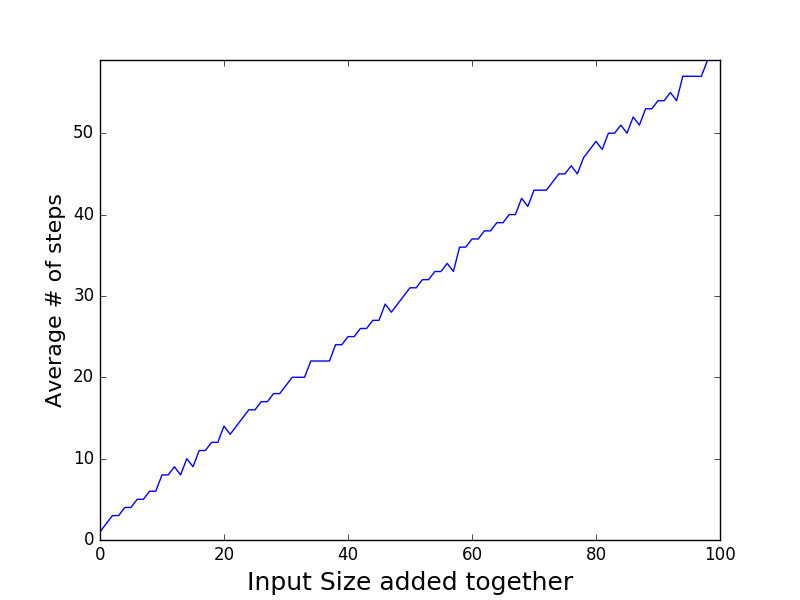
c = d; d = r; t = x; x = a;

a = t - q\*a;

t = y; y = b;

b = t - q\*b;

These assignments are made to be sure that the post-condition of the algorithm has been satisfied. The post-condition is am+bn=gcd(m,n), where m, n are the original inputs and a,b are changed throughout the loop. The previous assignments and calculations are repeated until we find that the remainder is a 0. We know that the loop will terminate because of the statements in line 25. “c=d;” and “d=r;” change the remainder calculation that is performed on line 22, that will eventually reach 0 because the two inputs are changing. After our trials are performed we make use of matplotlib in our program to display a graph of our results.



(Graph of Euclid's Extended Algorithm from the Python Code)

We plotted our results using matplotlib instead of using gnuplot.We found that using matplotlib was simpler for us to get the results we were looking for. Our x-axis is labeled as “Input Size added together”. Which corresponds to the size of the two numbers that was used in the algorithm by binary representation when added together. Our y-axis is labeled as “Average # of steps”. The average number of steps represents the number of times that the algorithm had to perform an iteration of the while loop. For example, the bound of the x-axis is 100, meaning we had tested a number that needed 100 bits to be encoded. Looking at the graph, we noticed that the general trend is linear. When creating a line of best fit we may expect the data to have a line that may look like f(n) = y\*(n) + b. Where n is the number of steps, y is the change in the number of steps per number of bits, and b is the initial number of steps needed for 2 bits. According to our test results, we know that an increase in the number of bits will cause Euclid’s Extended Algorithm to perform more steps. One may expect the trend to be exponential because of the encoding of binary numbers. However, our algorithm shows that this is not the case.