**CSC 380- Group Project: Part II**

Jason Degrace

Ben Michael

Paul-Hugo Dlugy-Hegwer

Part A:

Guessing:

The equation to predict the number of guesses is as follows:

(18750/(155^2)) = c/(N^2) becomes (N^2)\*(18750/(155^2)) = c

Where *N* is the input size, and *c* is the number of guesses.

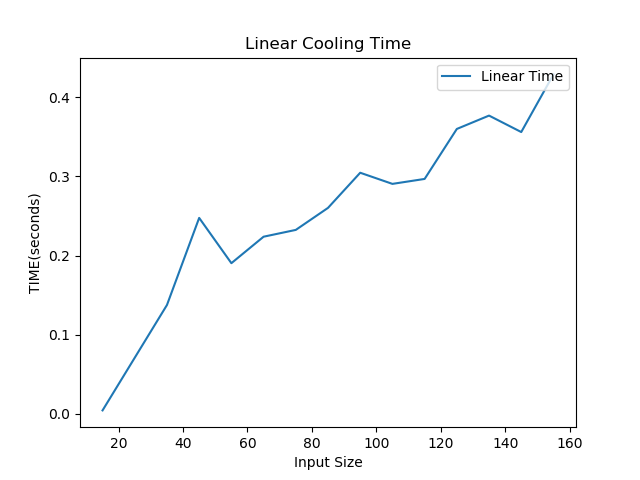
* For input size 200, N = 200, the result c = 31217.48 guesses.
* For input size 250, N = 250, the result c = 48777.32 guesses.
* For input size 300, N = 300, the result c = 70239.33 guesses.

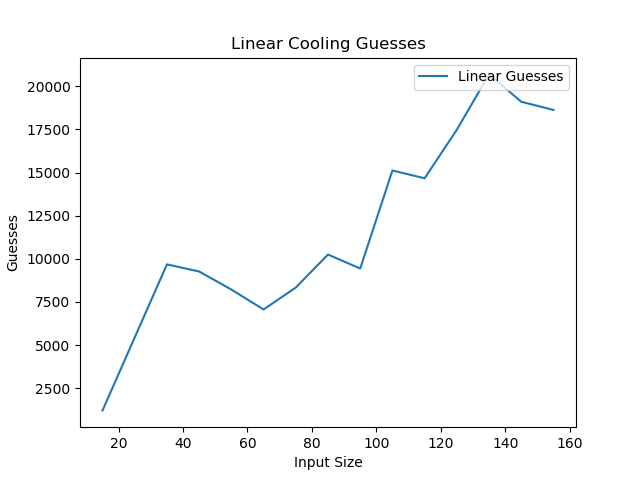
These solutions lineup well with the current trajectory of our graph, this is clearly indicated by the sensical growth in scale compared to where we started

\*note: 18750 was taken as a result of the average between the two final possible guess sizes given the positioning of the line on the graph.

1. Linear Cooling

In this method of cooling, an initial temperature is set proportional to the list size and then decreased by one every time the simulated annealing function is run. This appears method to have a Big-0 of N^2. It demonstrated the tendency to not find the zero more often on when the list size was smaller, with increased time to solve, but also increased success, on the larger lists.

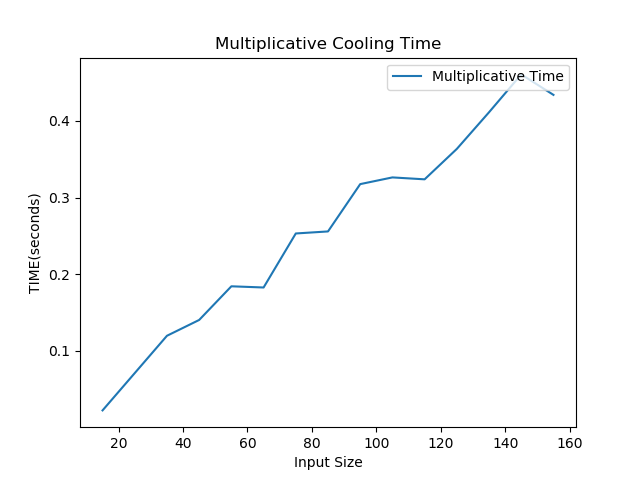


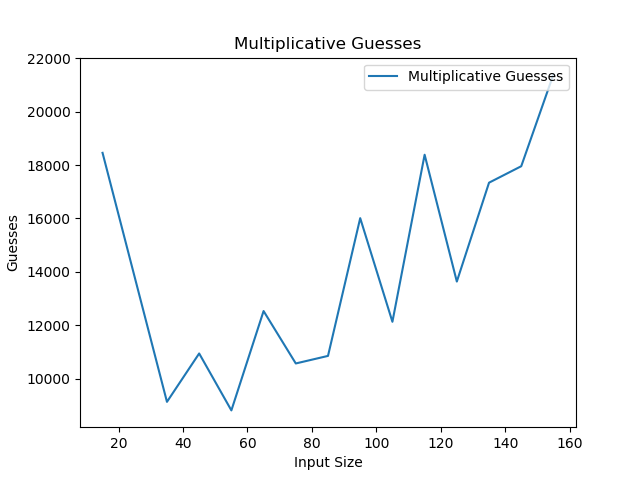


Part B:

Alternative method 1) Multiplicative Cooling

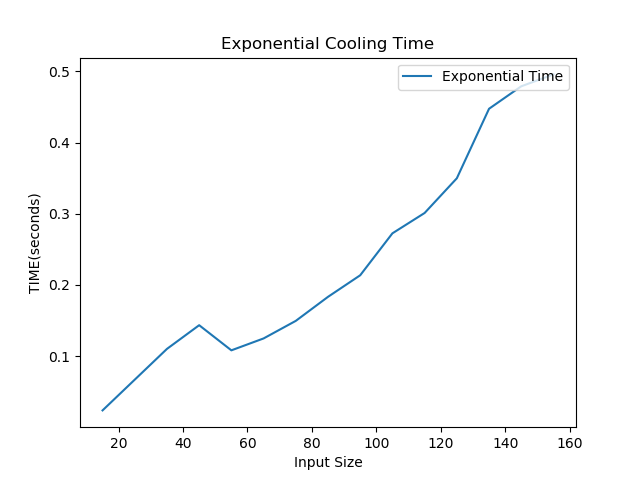
In this method of cooling, the temperature decreases by multiplying the current temperature by a set amount (in this case 0.999). This caused faster cooling than Linear, especially in the beginning when the temperature is very large. The quick cooling of the Multiplicative method aided it in finding the solution faster than Linear, but it also meant that it failed more often than linear on the smaller lists.

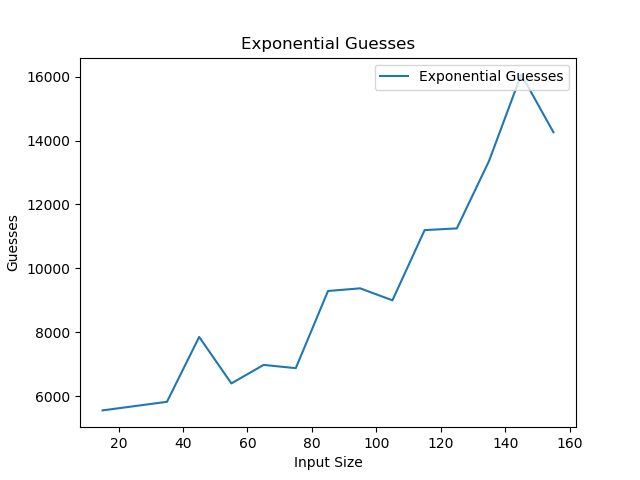




Alternative Method 2) Exponential Root Cooling

With exponential root cooling, the temperature is cooled by subtracting the product of the current temperature and the current temperature raised to one fourth ( T^ (1/4)). This method, much like the multiplicative method, cooled faster in the beginning, due to the fact that the value being subtracted is directly dependent upon how large the temperature is. It tended to fail more often on the larger list sizes, with the program failing to find zero around a fourth of the time with list sizes over 100.





Alternative Method 3) Variable Cooling dependent upon List Length

In this method, the cooling of the temperature depended upon the length of the list being cooled. This was an attempt to match a better method of cooling to each respective list size. For lists below 25, a multiplicative cooling was applied (multiplier was 0.98). For lists between 25 and 50, Linear cooling was used; for lists between 50 and 100, a modified multiplicative was used (multiplier was list length \* .1), and finally for lists above 100, a straight multiplicative was used. This attempt was met with mixed success. While it did sometimes deal far more quickly with its respective lists, other times it would do poorly. It most likely could have been further modified to optimize the results, based off more experimentation.

