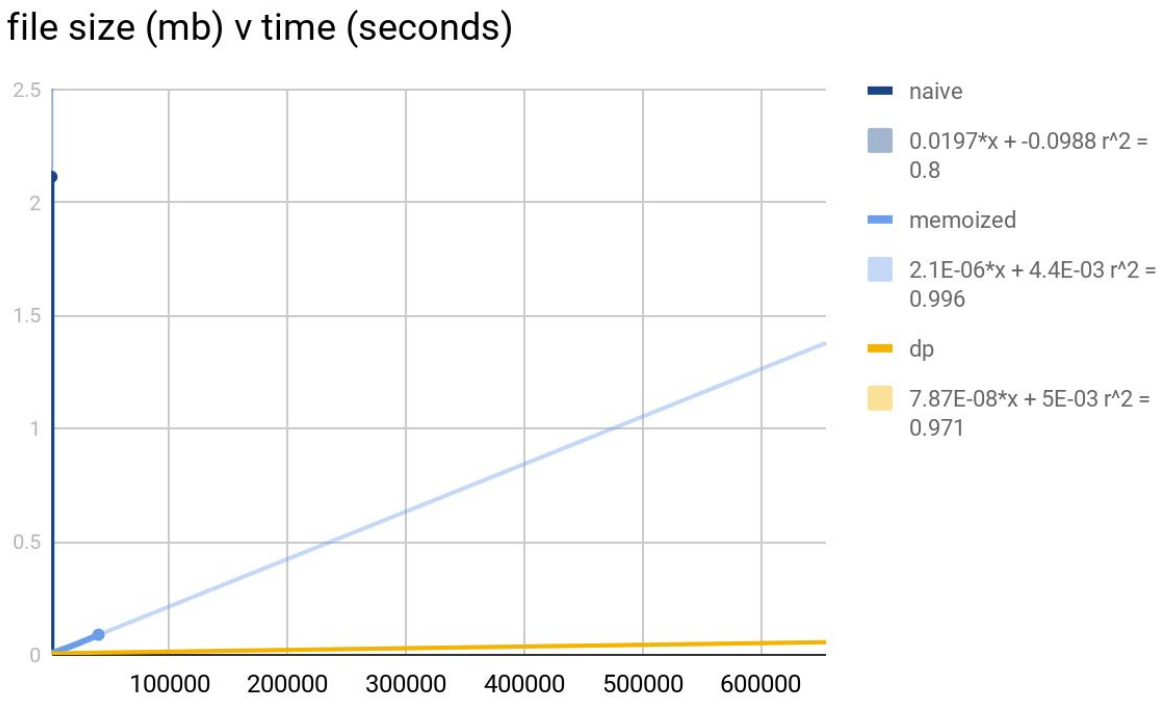
Vincent Cannalla

A13006747

Runtime Comparisons



Naive stopped at 128 mb

Memoized stopped at 32768 mb

Dp got to 150,000,000 mb in a time of 11.579 before I stopped testing it

The difference between the runtimes of the three methods is due to their implementation. The naive implementation has the worst runtime due to its sole reliance on recursion. Unlike the other implementations, naive must recalculate values every time it needs to know them. This leads to a huge runtime, because it has to solve CORRECT THIS many subproblems. Next, the memoization approach is much, much faster than the naive approach. The naive approach hangs at anything larger than 64 mb, whereas memoization has a run time of about 0.06

seconds on size 64 mb. Because the memoization approach stores the values for answers it's already gotten, the runtime becomes much shorter throughout the execution of the method. Lastly, and fastest, is the dynamic programming approach. With a runtime of O(n*m), where n is the amount of file sizes and m is the total file size to fill, the dp approach is by far more efficient than memoization; and while memoization runs quicker than naive, the memoization approach begins to take longer than dp at file sizes larger than 1024 mb. It is able to run so quickly, because it remembers all of the possible m's for each n, which allows it to run linearly. It just has to go through all of