

Ph. D. Qualifying Exam
Algorithmics
Computer Science Department
Indiana University
Fall 2004

This problem requires the design and analysis of two algorithms to add multidigit numbers. Actually, you are given one of the designs. Each algorithm must add n positive integers. The algorithms know that the i th integer uses i words to contain its digits. (Although it has little effect on the problem, you should assume that the numbers are in a large base.)

The first algorithm adds the numbers in order. Thus, the first partial sum is the sum of the first and second input numbers. The i th partial sum is the sum of the $i - 1$ st partial sum and the $i + 1$ st input number. The last partial sum is the final answer.

You design the second algorithm. It should be efficient. If the first algorithm is already the most efficient algorithm, say so, and then give some other algorithm for your second algorithm. If all reasonable algorithms have the essentially the same efficiency, say so and defend your answer instead of giving a second algorithm.

Your analysis should address best-case and worst-case time for each algorithm. The difference in times for the best case and worst case depend on the extent to which the additions create carries from one word to the next. You will need to make some assumptions about the time (as a function of the number of words occupied by each number) needed to add a pair of numbers. Make some reasonable assumptions, briefly explain the assumptions, and briefly explain why you are making them. For the best-case analysis, you should assume that no significant time is spent propagating carries during the additions. For the worst-case analysis you should assume that the numbers are such that a lot of time is spent propagating carries.

For your algorithm design, the best-case time is somewhat more important than the worst-case time.

In summary, your answer should have the following parts:

1. Discussion of the assumptions about the time needed to do addition.
2. Analysis of the time needed for first algorithm with best-case assumptions. If the first algorithm is an optimum method, note that.
3. Analysis of the time needed for first algorithm with worst-case assumptions.
4. Description of your second algorithm (order of adding the numbers).
5. Analysis of the time needed for your second algorithm with best-case assumptions. If all reasonable algorithms take essentially the same amount of time, explain that rather than giving a second algorithm.
6. Analysis of the time needed for your second algorithm with worst-case assumptions.