

COMPUTER PROCESSING TIME AND THE $O(N)$ CONCEPT

by Dr. Stephen B. Soffer

Copyright 2004 Stephen B. Soffer and **SDS Publications and Educational Services**

This material may be used within your classroom, school or home schooling parent's home. It is not to be copied or distributed outside of these.

Level: High School Students – precalculus or above. (There is a little calculus, but most of the mini-course does not require it.) College Students – college algebra or higher.

Tools Needed: Minimally, a scientific calculator. A graphing calculator is preferable. A spread sheet or more advanced mathematical software, such as Maple or Mathematica is useful, but not necessary.

INTRODUCTION

True or False? : Any problem involving only basic arithmetical operations (addition, subtraction, multiplication, division, and exponentiation) can be solved on a sufficiently fast computer.

Trying to answer this will be one of the main concerns of this mini-course.

THE TRAVELING SALESMAN PROBLEM

This classic problem asks a simple question. A traveling salesman must visit several cities. He must visit each only one time, but can visit them in any order. The route between any two cities incurs a certain cost. As a first approximation, this can be taken as the cost per mile traveled multiplied by the number of miles between each pair of cities. The problem is: Which order of visits costs the least?

The “brute force” way to solve this problem is to add the costs of each route and then search for the least cost route. The cost of each route is the sum of the costs of the individual segments – the trip from one city to the next one visited. To visit N cities, the salesman must travel $N + 1$ intercity partial routes. The reason for the extra one is to return to his home office from the last (N th) city visited. The number of possible routes is

$${}_{N+1}P_{N+1} = (N + 1)!$$

the number of permutations of all of $N + 1$ things.

How long would it take for a computer to calculate the possible costs for these routes? We will assume that it takes one microsecond (10^{-6} seconds) to calculate the cost of each route. (This is quite conservative for modern supercomputers. We will see more about the effect of faster speeds later.) This time includes the portion of the time to search for the cheapest route. Table 1 summarizes the computing time needed for various numbers of cities (N).

Table 1. Processing Times for Visiting N Cities – Assuming One Microsecond Per Route

N	$(N + 1)!$	Time (In Appropriate Units)
5	720	7.2×10^{-4} seconds (0.72 milliseconds)
10	39,916,800	39.9168 seconds (about 0.66528 minutes)
15	2.092×10^{13}	2.092×10^7 seconds (about 242 days)
20	5.109×10^{19}	1.62×10^6 years (about 1.6 million years)

You can see the trend. The computing time not only grows, but its rate of growth accelerates very rapidly as the number of cities increases. The problem becomes impractically long to process by 15 cities, and is completely intractable by 20 cities.

Well, you say, why not simply use a faster computer? Now we will get more realistic and refine our estimates.

As of 1995, the fastest computers could perform about 281 *floating point operations per second*. *Floating point operations* are basic arithmetic operations. By 1997, this was extended to 1.8 trillion floating point operations per second, called 1.8 *teraflops*, where *flops* is a computing rate measure, giving the number of floating point operations performed per second. (World Book Online Reference Center 1995) For our revised computing time estimates, we

2

will assume that each route segment requires about 10 floating point operations to calculate its cost. For the $N + 1$ segments, this is $10(N + 1)$ operations per route tried. With the $(N + 1)!$ we get

$$\text{Estimated Computing Time} \approx 10(N + 1)(N + 1)! \times 6 \times 10^{-13} \text{ seconds}$$

where $6 \times 10^{-13} \approx 1 / (1.8 \times 10^{12})$ is the approximate time for one floating point operation. Table 2 shows the revised computing times.

Table 2. Processing Times For Visiting N Cities – Assuming 1.8 Teraflops Processing Rate and 10 Flops Per Route Segment