Andrew Taylor

atayl136

EN.605.202.81.SP23

Homework 1: Complexity and ADTs

1) Nonnegative numbers in a ternary computer can be represented using trits, such as the balanced ternary system where each trit can take a value of 0, 1, or -1, representing 0, 1, and -1 respectively. I classic ternary this numbers are 0, 1 and 2. To represent a nonnegative number, a sequence of trits can be used to encode the number.

The first trit digit in ternary is for 1. The second digit is for 3. The third digit is for 9. The fourth is for 27 and so on. Thus, in ternary the number 122 composed of three trits is the decimal number known as 17.

Despite having a unique mathematical structure, trinary computing is considered inferior to binary computing due to several practical challenges. One major challenge is the difficulty in implementing trinary hardware, as most computer hardware and software are designed for binary computing. This makes it more complex and expensive to develop and implement trinary systems. Additionally, a lack of standardization and resources for trinary computing makes it less popular and less widely adopted compared to binary computing.

2) The address of the first element of the array RM, RM[0][0], starts at 100. The address of an element in row-major order can be calculated using the formula:

Address = base address + (row * number of columns + column) * size of each element

For RM[5][3], the address would be:

Address = 100 + (5 * 20 + 3) * 4 = 100 + (100 + 3) * 4 = 512

For RM[9][19], the address would be:

Address = 100 + (9 * 20 + 19) * 4 = 100 + (180 + 19) * 4 = 896

3) The maximum number of non-zero elements in a lower triangular matrix is n(n+1)/2, where n is the number of rows and columns in the nxn matrix. This is because only the elements below the main diagonal, including the main diagonal are non-zero.

The non-zero elements of a lower triangular matrix can be stored in memory sequentially by storing the elements of each row in sequence, starting from the first row. For example, for a 4x4 lower triangular matrix, the elements can be stored as follows:

a[0][0], a[1][0], a[1][1], a[2][0], a[2][1], a[2][2], a[3][0], a[3][1], a[3][2], a[3][3].

This way, all the non-zero elements of the array are stored in a contiguous block of memory. A formula to do so would be to return null when i<j and otherwise k = f(i,j) = (i-1)i/2 + j. Here, k represents the memory location of a[i][j], i is the row number, and j is the column number.

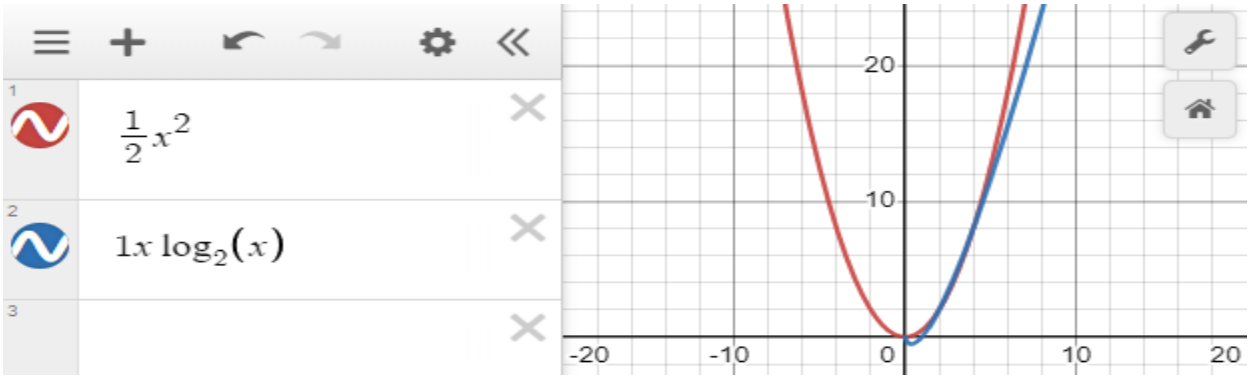
4) The maximum number of non-zero elements in an nxn tridiagonal matrix is 3n-2. They can be stored sequentially by row much the same way we did in the previous question. Four an example 5x5 tridiagonal matrix it would look like:

a[0][0], a[0][1], a[1][0], a[1][1], a[1][2], a[2][1], a[2][2], a[2][3], a[3][2], a[3][3], a[3][4], a[4][3], a[4][4].

Again this would put the nonzero elements into a 1D array. The formula to do so would be to return null whenever $|i - j|$>1 and otherwise k = f(I,j) = 2(i - 1) + j.

5) The prompt for this question is to investigate the relationship between an^2 and bnlgn, by experimenting with different values for a and b. Let's try to observe trends and dominant parts:

Setting a and b equal at 1 they do not intersect, but cutting a in half makes them closely match for a while. Maintaining this proportion pushes the conformance farther and farther up the y axis, however we can clearly see that from the initial parity an^2 is the dominant term and would be associated with algorithms of higher time or space complexity in all cases above some point beta.



6) For problems 6 and 7 I'm not sure I understand correctly, but I assume it is being asked what is the maximum size N of a problem that can be solved in one hour if the algorithm takes lg n microseconds for an operation of a problem size N =1. Given that there are 3.6x10^9 microseconds in an hour, 2^3.6e+9 would be the size N under those assumptions. We find this by dividing the time available by the time taken and solving for n.

7) Following the same logic and rubric for this question, if we ask what is the maximum size problem to solve in one hour using n^3 microseconds per operation N=1, then it is 1532.6.