1. Representation in Ternary Computer

Nonnegative numbers can be represented in a ternary computer using trits (0, 1, 2) instead of bits (0, 1). For example, the number 5 in decimal can be represented as 12 in ternary (13^1 + 23^0).

We don't use ternary systems because:

- Complexity: Circuit design and logic operations are more complex for ternary systems compared to binary systems.
- Standardization: The binary system is deeply ingrained in current technology, making a switch costly and impractical.
- Efficiency: Binary systems are more efficient for current semiconductor technology.

2. Array Address Calculation in Row-Major Order

Given a 10x20 array RM, with elements taking 4 bytes each, and starting address at 100:

- Address of RM[5][3]:
    - Formula: Address = base_address + ((row * num_columns) + column) * element_size

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

  Address = 100 + (100 + 3) * 4

  Address = 100 + 103 * 4

  Address = 100 + 412

  Address = 512

- Address of RM[9][19]:

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

Address = 100 + (180 + 19) * 4

Address = 100 + 199 * 4

Address = 100 + 796

Address = 896

3. Lower Triangular Matrix

For an nxn lower triangular matrix, the maximum number of non-zero elements is

$n(n + 1)/2$

To store these elements sequentially:

- Use a 1D array to store non-zero elements.
- Formula to find the index k for element

  $a[i][j]\colon\ k = i(i - 1)/2 + j$

4. Tridiagonal Matrix

For an nxn tridiagonal matrix, the maximum number of non-zero elements is $3n-2$

To store these elements sequentially:

- Use a 1D array to store non-zero elements.
- Formula to find the index k for element a[i][j]:
  - If $i=j$: $k=2i$
  - If $i=j+1$: $k=2i-1$
  - If $i=j-1$: $k=2i+1$

5. Intersection of Functions f(n) = an^2 and g(n)=bn log n,

To find the intersection empirically:

1. Choose values for a and b.
2. Graph f(n)=an^2 and g(n)=bn log n
3. Observe the intersection point.

For example:

- a=1, b=1:
  - f(n)=n2 f
  - g(n)=n log n

Graph these functions to find the intersection:

- For n around 43, n2 ≈n log n

Observations:

- The intersection point increases as a increases or b decreases.
- For larger aaa or smaller b, f(n) becomes dominant sooner.

6. Maximum Problem Size for Algorithm Taking lg n Microseconds

Given 1 hour (3,600,000,000 microseconds):

- logn≤3,600,000,000\log n \leq 3,600,000,000logn≤3,600,000,000
- n≤23,600,000,000n \leq 2^{3,600,000,000}n≤23,600,000,000

The maximum size n is extremely large due to the exponential growth of logarithmic functions.

7. Maximum Problem Size for Algorithm Taking n^3 Microseconds

Given 1 hour (3,600,000,000 microseconds):

- n3 ≤3 ,600,000,000n^3 \leq 3,600,000,000n3≤3,600,000,000
- n ≤ (3,600,000,000)1/3≈1532.6n \leq (3,600,000,000)^{1/3} \approx 1532.6n≤(3,600,000,000)1/3≈1532.6

The maximum size n is approximately 1532.