0.1 Analysis of Space Complexity Components

Space complexity consists of several components, as shown in Table 1.

Table 1: Analysis of Space Complexity Components

Component	Description	Example	Complexity
Constant part (C)	Memory that is always used and independent of the input.	Program instructions, global variables	$O(1)$
Input space ($Sp(I)$)	Memory occupied by the inputs of the program.	Array of n integers $\rightarrow$ $4n$ bytes	$O(n)$
Recursive stack ($Sc(n)$)	Memory used for each recursive function call.	Recursive calls in sum(A, n)	$O(n)$
Auxiliary space	Temporary memory for computations or auxiliary structures.	Temporary variables in calculations	Problem-dependent

0.1.1 Example: Summing an Array Using Recursion

The following code calculates the sum of an array using recursion:

```
int sum(int A[], int n) {
    if (n == 0)
        return 0;
    return A[n-1] + sum(A, n-1);
}
```

Memory Analysis:

- Constant part (C): Memory for the function code and global variables, about 2000 bytes (fixed amount).
- Input space ($Sp(I)$): Array A of size n. If each element takes 4 bytes, this is $4n$ bytes.
- Recursive stack ($Sc(n)$): Each recursive call takes 16 bytes (return address + parameters), so $16n$ bytes.
- Auxiliary space: Minimal, since no extra structures are used.

Overall formula:

$$S(P) = C + Sp(I) + Sc(n) = 2000 + 4n + 16n$$

Numerical example: If $n = 1000$:

$$S(P) = 2000 + 4000 + 16000 = 22000 \text{ bytes} \approx 22 \text{ KB}$$

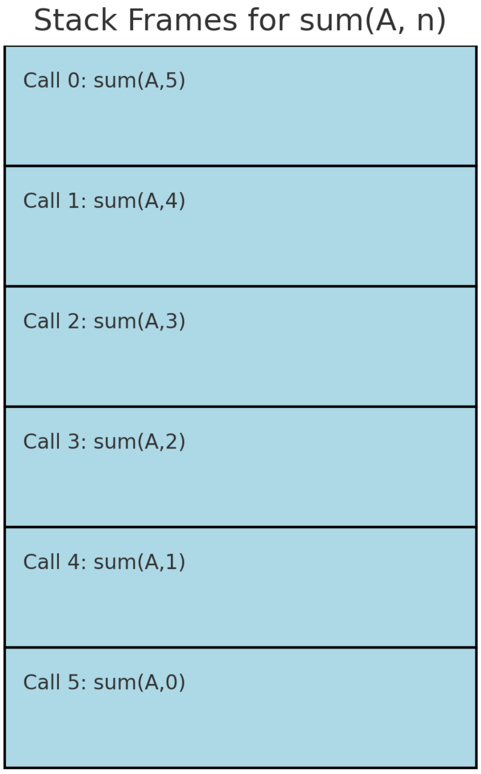

Figure 1: Schematic of stack growth during recursive calls in sum(A, n)

0.2 Methods to Reduce Memory Usage

Several methods can be used to optimize memory usage, as summarized in Table 2.

0.2.1 What is a Sparse Matrix?

A sparse matrix is a matrix in which most of the elements are zero, and only a few elements are non-zero. Instead of storing the entire matrix, only the coordinates and values of the non-zero elements are stored, significantly reducing memory usage. This is widely used in applications like storing large graph matrices or sparse data in machine learning.

Table 2: Methods to Reduce Memory Usage

Method	Description	Example	Space Complexity
In-place algorithms	Data is modified in place without requiring much extra space.	QuickSort	$O(\log n)$
Optimized data structures	Data is stored in compressed or efficient formats.	Sparse matrix (storing only non-zero values), compact trees	Much less than normal
Dynamic programming optimization	Storing only essential states instead of all states.	Storing only the last two values in Fibonacci	$O(1)$
Streaming algorithms	Processing data as a stream without storing the entire dataset.	HyperLogLog, online averaging	Sublinear

0.2.2 Explanation of Streaming Algorithms

Streaming algorithms are used when the dataset is too large to store entirely. These algorithms process data as it arrives and maintain a small summary (sketch) that can be used to make accurate approximations.

- Example: Calculating the average without storing all data (keeping only the sum and count).
- Example: HyperLogLog algorithm for estimating the number of unique values in very large datasets.

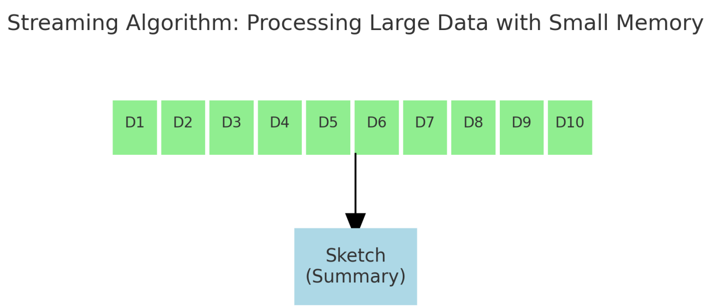

Figure 2: Schematic of a streaming algorithm: data arrives as a stream and a small summary is maintained.